ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 5, Issue 10, October 2017

Domain-Independent Text Mining Framework based on Open Information Extraction

Mohamed Lotfy*, Mohamed H Haggag, Ensaf H Mohamed

Department of Computers and Information, Helwan University, Cairo Governorate, Egypt

E-mail: mlotfy82@gmail.com*

Abstract: We propose a robust hybrid text mining solution that combines Open Information Extraction (OIE), Knowledge Discovery from Databases (KDD) and Association Rules Mining (ARM) methods to perform domain independent text mining task. For OIE, we used ClausIE system with extra added features like "Co-Reference Resolution" and "Acronyms Detection" that allow the system to extract more information and can help later in the mining process. The added features in the proposed solution will enhance both the quality and the quantity of the extracted propositions regarding both precision and recall.

Keywords: Text mining; Data mining; Knowledge discovery from databases; Association rules mining; Information extraction; Open information extraction; Natural language processing

I. INTRODUCTION

Text is a common way for the exchange of information. Extracting useful information from texts is not an easy task, so, there is a need in our life to have automated tools which are able to extract useful information in an intelligent way and at a low cost [1]. Text Mining enables computer to discover new, previously unknown information from different text resources [2,3]. Text mining tools enable to answer difficult questions and perform searches on text with some sort of intelligence.

II. TECHNICAL BACKGROUND

Text mining process is similar to data mining with a major difference between them; this difference is in the type of input data [4]. The input to data mining tools is structured data, but text mining can deal with unstructured or semi-structured data such as text documents, emails and HTML documents etc. [5-8].

2.1 Text Mining Techniques

There are various methods for applying text mining; like natural language processing (NLP), information extraction (IE) and association rules mining (ARM).

2.1.1 Information extraction (IE):

Information Extraction (IE) is about deriving structured information from unstructured or semi-structured text sources. Given the following sample text as an example:

Nielsen Co. said George Garrick, 40 years old, president of Information Resources Inc.’s London-based European Information Services operation, will become president of Nielsen Marketing Research USA, a unit of Dun and Bradstreet Corp. He succeeds John I. Costello, who resigned in March. Who resigned in March?

The IE system should be able to recognize and create the result existing in Table 1.

New manager	George Garrick
Old manager	John I. Costello
Post	president
Org	Nielsen Marketing Research

Table 1: IE system result.

2.1.2 Association rule mining:

Association Rule Mining (ARM) is a technique by which important relationships are extracted from large databases. It has been widely used in decision-making process in business. E.g., these relationships are currently implemented in various supermarkets where items are placed based on purchasing habits of customers i.e. those items are placed at minimum distance which are purchased frequently [5].

2.2 Open Information Extraction

The paradigm of Open Information Extraction (Open IE) was introduced in that aim to facilitate domain-independent discovery of relations extracted from texts and to scale to heterogeneous and large-size corpora such as the Web. An Open IE system takes as input only a corpus of texts without any prior knowledge or specification of the relations of interest and outputs a set of all extracted relations [9]. The key goals of Open IE are domain independence, unsupervised extraction and scalability to large amounts of text.

III. RELATED WORK

Many text mining systems that built using IE method, have been proposed and implemented using both IE and data mining techniques. Some of these frameworks are listed below [10].

3.1 Text Mining Systems Based on Information Extraction

3.1.1 DiscoTEX:

(Discovery from Text Extraction) uses an IE system that is learned to extract structured data from text and this data is then mined for interesting relationships. Rules mined from an extracted database from texts are used to extract additional information from future documents [11,12].

3.1.2 RAPIER:

(Robust Automated Production of Information Extraction Rules) uses together sample documents and filled templates to produce pattern-match rules that directly reproduce fillers for the slots in the template. The learned patterns use limited syntactic and semantic information to recognize potential slot fillers and their encirclement context. RAPIER is bottom-up learning framework that combines techniques from many programming systems and permits patterns to have constraints on the words, parts-of-speech tags, and semantic classes exist in the filler and the encirclement text [13].

3.1.3 BWI:

(Boosted Wrapper Induction) is a process to make a trainable IE system. It learns comparatively simple contextual patterns detecting the beginning and end of pertinent text fields. BWI employs AdaBoost algorithm in repeated manner for learning boundaries. BWI executes in repeated manner so that patterns missed by previous rules can be extracted [3]. Since the mentioned frameworks are based on traditional IE methods, then, they are domain specific text mining techniques. We aimed in the proposed solution at making it domain independent, so, it is based on Open Information Extraction (OIE) approach.

3.2 Open Information Extraction Systems

There are two main categories of OIE systems: Approaches that use only shallow syntactic parsing, and others that apply heavier NLP technology. Approaches such as TextRunner [9], Reverb [10] belongs to the first category that focuses 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 OLLIE [11] and ClausIE [12] use dependency parsing and belong to the second category. These extractors are generally more costly than the extractors above; they trade efficiency for improved precision and recall.

3.2.1 TextRunner:

TextRunner's input is a bulk of text and its output is a collection of extractions that are efficiently indexed to help searching via user queries. It consists of three key modules:

1. Self-Supervised Learner: Given a sample text as input, the Learner outputs a classifier that labels nominee extractions as "reliable" or not. The Learner needs no hand-tagged data.
2. Single-Pass Extractor: The Extractor passes single time over the input text to extract tuples for all potential relations. It does not use a parser. The Extractor produces one or more nominee tuples from each sentence, sends each nominee to the classifier, and keeps the ones labelled as reliable.
3. Redundancy-Based Assessor: The Assessor allocates a probability to each saved tuple based on a probabilistic model of redundancy in text.

Shortcomings in TextRunner [12]:

1. Extracts incoherent extractions.
2. Extracts uninformative extractions.

3.2.2 ReVerb:

ReVerb is an Open IE system, its input is a POS-tagged and NP-chunked sentence and results in a collection of (x; r; y) extraction triples. For an input sentence s, ReVerb applies the following extraction process:

1. Relation Extraction: For each verb v in s, extract the longest series of words rv such that (1) rv begins at v, (2) rv applies the syntactic constraint, and (3) rv applies the lexical constraint. If any two matches are adjacent or overlap in s, combine them into one match.
2. Argument Extraction: For each relation phrase r recognized in Step 1, locate the nearest noun phrase x to the left of r in s such that x is not a relative pronoun, WH-term, or existential "there". Locate the nearest noun phrase y to the right of r in s. If such an (x; y) couple could be located, return (x; r; y) as an extraction.

To solve the problems exist in TextRunner, ReVerb applied two simple syntactic and lexical constraints on binary relations expressed by verbs [12].

3.2.3 OLLIE:

OLLIE is an improved Open IE system that solves both the problems found in ReVerb. First, OLLIE obtains high output by producing relations mediated by nouns, adjectives, and more. Second, a context-analysis step enhances precision by providing contextual information from the sentence in the results. OLLIE has architecture (Figure 1) for learning and applying binary extraction patterns. First, it employs a collection of high precision tuples from ReVerb to bootstrap a large training collection. Second, it learns open pattern templates over this training collection. Next, OLLIE uses these pattern templates at time of extraction.

3.2.4 ClausIE:

ClausIE (for clause-based open information extraction), applies dependency parsing (DP). ClausIE basically differs from prior approaches in that it detaches (i) the extraction of helpful pieces of information expressed in a sentence from (ii) its representation in the form of one or more propositions. The central reason for this detachment is that (i) can be processed carefully and in a principled way by employing properties of the English language. Particularly, the collection of clauses of each sentence is detected and, for each clause, the relative clause type is extracted according to the grammatical function of its constituent (e.g., subject-verb object, SVO) [13].

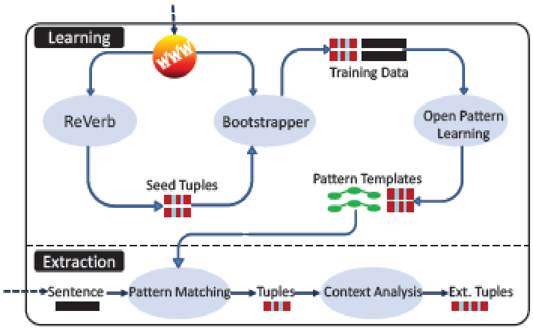


Figure 1: OLLIE system.

IV. PROPOSED SYSTEM

The proposed solution can be figured as shown in Figure 2, where input text/documents are processed in a pre-processing step, then the results are fed to Open information Extraction step to produce tuples of information, then these tuples are processed in post-processing step to produce a structured data, then we apply the traditional data mining approaches to get the target patterns [14-17].

Processing steps of the system are:

1. Input text documents (Input Text).
2. Annotation
    2.1. Text annotation: using StanfordCoreNLP.
    2.2. Part Of Speech (POS) tagging.
    2.3. Named Entity Recognition (NER).
    2.4. Sentences Extraction from text.
3. Information Extraction
    3.1. Computing the DP of the sentence.
    3.2. Determining the set of clauses using the DP.
    3.3. For each clause, determine the set of coherent derived clauses based on the DP and small, domain-independent lexica.
    3.4. Co-Reference resolution: Using "Co-ref Processor" component.
    3.5. Acronyms detection: Detecting acronyms in text.
    3.6. Generating propositions from (a subset of) the coherent clauses.
    3.7. Storing extracted sentences and tokens in a database for further processing.
4. Tokenization
    4.1. Initial tokens extraction: Extracting tokens from sentences for mining process.
    4.2. Smart tokenization which depending on co-reference resolution and acronyms detection.
5. Data mining for result data stored in database.

4.1 System Components

The main modules composing our system are discussed in the following subsections:

4.1.1 Annotator:

This component extracts all the existing sentences in the input text in addition to other objects as Part Of Speech (POS) tagging, Named Entity Recognition (NER) and Co-References.

POS: The POS Tagger assigns parts of speech to each word, such as noun, verb, adjective, etc. [18]. We used Stanford Log-linear Part-Of-Speech Tagger in our implementation.

NER: NER Labels sequences of words in a text, which are the names of things, such as person and company names. We used Stanford NER in our implementation.

Co-Reference: Co-Reference occurs when two or more expressions in a text refer to the same thing.

4.1.2 Information extractor:

We have used ClausIE OIE system as a seed system and provided some enhancements to it to obtain better precision and recall. Enhancements include Co-reference resolution and Acronyms detection.

Co-reference resolution: This process relies on the dependency tree generated using the Stanford Dependency Parser in the annotation module, where the POS tags {NNP (Proper noun, singular), NNPS (Proper noun, plural), NN (Noun, singular or mass) and NNS (Noun, plural)} are used to detect the referenced objects that will be referred later by any related POS tag PRP (Pronoun).

For example, if an input text is "Mohamed Lotfy is working as a software engineer. He studied at faculty of computers and information." the result propositions from both regular ClausIE and the proposed solution are as following:

1) ClausIE:
- ([Mohamed Lotfy] [is working] [as a software engineer])
- ([He] [studied] [at faculty of computers and information])

2) Proposed System:
- ([Mohamed Lotfy] [is working] [as a software engineer])
- ([Mohamed Lotfy] [studied] [at faculty of computers and information])

Acronyms detection: In this step, depending on the annotations resulted from the pre-processing step, all the acronyms mentioned in the input text are extracted and every acronym is added to the extracted propositions as a new proposition. The proposed solution can find different forms of acronym phrases, for example, if an input text is "The International Business Machines Corporation (IBM) is an American multinational technology and consulting corporation”. The following propositions will be added to the list of propositions resulted from the input text:

([IBM] [is acronym for] [International Business Machines])

These propositions will result from the original sentence in addition to the following proposition that is originally resulting from ClausIE system.

([The International-Business-Machines Corporation IBM] [is] [an American multinational technology and consulting corporation])

Our acronym detection engine can also process a phrase as the one "The International Center for Research on Women (ICRW)". It will extract the following proposition:

([ICRW] [is acronym for] [International Center for Research on Women])

The proposed solution enhanced both the quality and quantity of extracted propositions in terms of precision and recall. Precision and Recall defined by the following equations:

$$precision = \frac{\textbf{correct extractions count}}{\textbf{total extractions count}}$$

$$recall = \frac{\textbf{correct extractions count}}{\textbf{all correct propositions count}}$$

4.1.3 Tokenizer:

A tokenizer accomplishes the task of splitting the input text into named entities (or terms) which roughly correspond to "words". Stanford Tokenizer is used, which supply a class able to tokenize English text, called PTBTokenizer. It was at first designed to mimic Penn Treebank 3(PTB) tokenization, hence its name, after that the tokenizer has added a few options and a fair amount of Unicode compatibility [17,18].

Tokenization process uses "WordNet"[19], which is a large lexical database of English. Nouns, verbs, adverbs and adjectives are assembled into collections of close synonyms (synsets), each expressing a different concept. Synsets are interlinked through conceptual-semantic and lexical relations. Each word from the WordNet is assigned a code that is used later in the mining process.

Tokenization module generates a Comma Separated Values (CSV) of all tokens from the input sentence and stores it in a transactional database to be ready for mining. Indeed not the tokens itself that is stored in the CSV, but the IDs of the tokens that are stored, where each token is stored in database with ID that is used in the CSV for better performance. For the same sample text mentioned in section 4.4.3.1, the output CSV is: [3858, 196767, 3505, 197294, 72387] that is a comma separated list of IDs of the words (Tennis, Olympic, sport, played, society) respectively existing in the dictionary of words taken from WordNet. If a new token is found which not existing in the dictionary, we add this token as a new record in the dictionary.

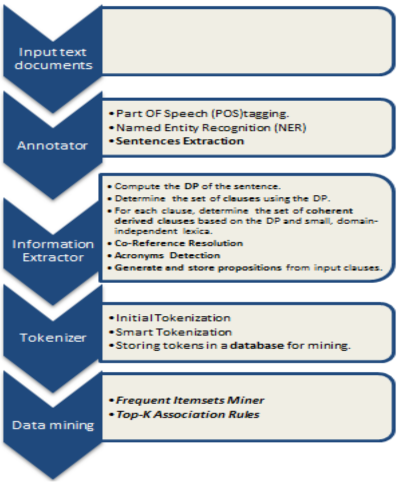

Input text documents
Annotator
•Part OF Speech (POS)tagging.
•Named Entity Recognition (NER)
•Sentences Extraction
Information Extractor
• Compute the DP of the sentence.
• Determine the set of clauses using the DP.
• For each clause, determine the set of coherent derived clauses based on the DP and small, domain-independent lexica.
• Co-Reference Resolution
• Acronyms Detection
• Generate and store propositions from input clauses.
Tokenizer
•Initial Tokenization
•Smart Tokenization
•Storing tokens in a database for mining.
Data mining
•Frequent Itemsets Miner
•Top-K Association Rules

Figure 2: Proposed system.

Our Tokenizer is smart enough to use the previous processes of co-reference resolution and acronyms detection in the tokenization. Every extracted acronym is tokenized as one token and recorded as one word in the dictionary. For example "The International Business Machines Corporation (IBM) is an American multinational technology and consulting corporation." when tokenized in our system, the phrase "International Business Machines" is considered one token. Resolved co-references are also handled by the tokenizer so that the term "He" that refers to a person mention is detected and tokenized with the same token of the mentioned person. For example, in the sample text "Alex Kalinovsky was born in Ukraine. He has been in the IT industry for more than 10 years.", both "Alex Kalinovsky" and "He" are considered the same token.

4.1.4 Data miner:

This component is one of the most important components in the proposed system, as it is the result of the natural evolution of information technology and is the value of doing information extraction. There are a number of data mining functionalities. These include characterization and discrimination, the mining of frequent patterns, associations, and correlations, classification and regression, clustering analysis, and outlier analysis. Data mining algorithms are used to allocate the kinds of patterns to be found in data mining tasks. In the proposed solution, frequent patterns and associations are that of our interest. Frequent patterns are patterns that occur frequently in data. There exist many sorts of frequent patterns, including frequent item-sets, frequent subsequence's (also known as sequential patterns), and frequent substructures. A frequent item-set points to a collection of items that often appear together in a transactional data set-for example, cheese and bread, which are frequently bought together in retail stores by many customers. A frequently occurring subsequence, like the pattern that customers, first purchase a laptop, then a digital camera, and then a memory card, is a (frequent) sequential pattern. A substructure can point to different structural forms (e.g., graphs, trees, or lattices) that may be merged with item-sets or subsequences. If a substructure happens frequently, it is called a (frequent) structured pattern. Mining frequent patterns leads to the detection of helpful associations and correlations inside data.

Frequent item-sets miner (FIN): FIN is very fast algorithm for detecting frequent item-sets in transaction databases, submitted by Deng et al.

- **Input of the FIN algorithm:**

Transaction database (aka binary context) and a threshold named minsup (a value between 0 and 100%) [10]. A transaction database is a set of transactions. Each transaction is a collection of items. For example, consider the following transaction database in Table 2. It contains 5 transactions (t1, t2... t5) and 5 items (1, 2, 3, 4, 5). For example, the first transaction represents the set of items 1, 3 and 4. An item is not permitted to appear more than once in the same transaction and those items are assumed to be sorted by lexicographical order in a transaction [20].

Transaction id	Items
t1	{1, 3, 4}
t2	{2, 3, 5}
t3	{1, 2, 3, 5}
t4	{2, 5}
t5	{1, 2, 3, 5}

Table 2: Example for transaction database for FIN algorithm.

- **Output of the FIN algorithm:**

FIN is an algorithm for detecting item-sets (group of items) occurring frequently in a transaction database (frequent item-sets). A frequent item-set is an item-set appearing in at least minsup transactions from the transaction database, where minsup is a parameter provided by the user. For example, if FIN is run on the prior transaction database with a minsup of 40% (2 transactions), FIN results in Table 3 [10].

item-sets	support
{1}	3
{2}	4
{3}	4
{5}	4
{1, 2}	2
{1, 3}	3
{1, 5}	2
{2, 3}	3
{2, 5}	4
{3, 5}	3
{1, 2, 3}	2
{1, 2, 5}	2
{1, 3, 5}	2
{2, 3, 5}	3
{1, 2, 3, 5}	2

Table 3: The output of the FIN algorithm.

In the results, each item-set is annotated with its support. The support of an item-set is how many times the item-set appears in the transaction database. For example, the item-set {2, 3 5} has a support of 3 because it appears in transactions t2, t3 and t5. It is a frequent item-set because its support is higher or equal to the minsup parameter [21].

Mining Top-K Association Rules: TopKRules is an algorithm for detecting the top-k association rules happening in a transaction database. Why is it useful to detect top-k association rules? Because other association rules mining algorithms requires setting a minimum support (minsup) parameter that is hard to set (usually users set it by trial and error, which is time consuming). TopKRules proposed a solution to this problem by allowing users to indicate k, the number of rules to be detected instead of using minsup.

- **Input of TopK Rules:**

TopKRules takes three parameters as input:

1) A transaction database,
2) A parameter k that represent the number of association rules to be detected (a positive integer),
3) A parameter minconf that represent the minimum confidence that the association rules should have (a value in [0, 1] representing a percentage).

A transaction database is a collection of transactions. Each transaction is composed a set of items. For example, consider the following transaction database (Table 4). It contains 6 transactions (t1, t2... t5, t6) and 5 items (1, 2, 3, 4, 5). For example, the first transaction represents the set of items 1, 2, 4 and 5. An item is not permitted to appear twice in the same transaction and those items are assumed to be sorted by lexicographical order in a transaction.

Transaction id	Items
t1	{1, 2, 4, 5}
t2	{2, 3, 5}
t3	{1, 2, 4, 5}
t4	{1, 2, 3, 5}
t5	{1, 2, 3, 4, 5}
t6	{2, 3, 4}

Table 4: Example for transaction database for Top-K association rules.

- **Output of TopK Rules:**

TopKRules outputs the top-k association rules. To explain what top-k association rules are, it is necessary to review some definitions. An item-set is a set of different items. The support of an item-set is the number of times that it happens in the database divided by the whole number of transactions in the database. For example, the item-set {1 3} has a support of 33% because it appears in 2 out of 6 transactions from the database. An association rule X→Y is an association between two item-sets X and Y that are disjoint. The support of an association rule is the number of transactions that contains X and Y divided by the whole number of transactions. The confidence of an association rule is the number of transactions that contains X and Y divided by the number of transactions that contains X. The top-k association rules are the k most frequent association rules in the database that have a confidence more than or equal to minconf.

For example, if we run TopKRules with k = 2 and minconf = 0.8, we obtain the top-2 rules in the database having a confidence higher or equals to 80%.

2 → 5, which have a support of 5 (it appears in 5 sequences) and a confidence of 83%
5 → 2, which have a support of 5 (it appears in 5 sequences) and a confidence of 100%

For instance, the rule 2 →5 means that if item 2 appears, it is likely to be associated with item 5 with a confidence of 83% in a transaction. Moreover, this rule has a support of 83% because it appears in five transactions (S1, S2 and S3) out of the six transactions contained in this database. The mentioned mining patterns are applied to the result CSV values of tokenization module.

V. RESULTS AND EVALUATION

The evaluation of the proposed solution will include two phases; the first phase is evaluating the Enhanced OIE Extractor compared to the ClausIE extractor to show the enhancements added in the new proposed extractor in the system. The second phase is evaluating the mining process over the extracted information from the first phase. Before the evaluation process, we will first describe the datasets and the methodology used to evaluate the proposed system.

5.1 System Environment Setup

The proposed solution is developed using Java programming language version 8. The database used is MySQL version 5. We used the Stanford DP (version 2.0.4) and configured the proposed system to generate triple propositions for each input sentence.

5.2 Dataset Setup

We used three different datasets to evaluate the proposed solution. First, about 200 random sentence extracted from Wikipedia pages; these sentences are both short and simple, but these sentences are somehow noisy, because some articles that exist in Wikipedia are written by non-native speakers, however, the Wikipedia sentences do contain some incorrect grammatical constructions. Second, we extracted 199 random sentences from the New York Times collection (NYT); these sentences are generally very clean but tend to be long and complex. These two datasets are taken from the datasets used to evaluate ClausIE compared to other extractors like TextRunner, Reverb, OLLIE, and WOE. The final dataset used, complex paragraphs, is more long and more complex than the first two datasets, it should be like this to enable me to test the added features to the proposed solution, like coreference resolution and acronym detection; It's a set of paragraphs where each one is a set of many sentences that are related to one topic.

5.3 Evaluation Methodology

The proposed system is configured to generate triple propositions in the form of ("subject", "relation" and "arguments").We manually labeled the extractions obtained from both ClausIE and our proposed extractor; each extraction is labeled by one of three values, "0" if it's not correct, "1" if it's correct but not informative and "2" if the extraction is correct and more informative, for example, if we have a phrase like:
"Gandhi was 24 when he arrived in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria; he spent 21 years in South Africa".

- If the extraction is "[he] [spent] [21 years] ", then it will be labeled with value "0".
- If the extraction is "[he] [spent] [21 years in South Africa]", then it will be labeled with value "1" because we don't know what "he" refers to except we look for the context of the whole phrase, it will be labeled by the value "1" if the extraction is correct and is the same in both ClausIE and Our proposed extractor.
- If the extraction is "[Gandhi] [spent] [21 years in South Africa] ", then it will be labeled with value "2" because we don't need to know the context of the whole phrase and can detect that "Gandhi" is the person of interest.

Redundancy in extractions are considered, For example, ClausIE extracts from sentence "AE remained in Princeton until his death" propositions ("AE", "remained", "in Princeton") and ("AE", "remained", "in Princeton until his death"); the former extraction is marked redundant and hence is considered not correct. After that, all the extractions that are labelled to be correct are used to rank the precision of each extractor over all the datasets used in the evaluation process. We will use the precision as a measure for evaluating the proposed instead of recall since it is infeasible to obtain the set of "all correct" propositions. To rank the precision, we need to count the whole number of extractions. We assume that each extraction will increase the whole count by one if it evaluated to have the value "0" or "1". If it's evaluated by "2", then it will increase the whole count by "2", then the precision calculated as following:

$$\boldsymbol{precision} = \frac{\textbf{summation of evaluation of all extractions}(\textbf{correct trnsactions})}{\textbf{total extractions count}}$$

5.4 Example Extractions

We selected samples from our datasets to show the result of applying the proposed solution compared to ClausIE. Samples and result of extraction presented in the following subsections including Tables 5-9.

5.4.1 On the Wikipedia dataset:

Example 1: Gandhi was 24 when he arrived in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria; he spent 21 years in South Africa.		
Extractor	**Extractions**	**Value**
ClausIE	[Gandhi] [was] [24 when he arrived in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria]	1
	[he] [arrived] [to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria when]	0
	[he] [arrived] [to work as a legal representative for the Muslim Indian Traders when]	0
	[he] [arrived] [in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria]	1
	[he] [arrived] [in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders]	1
	[he] [arrived] [to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria]	1
	[he] [arrived] [to work as a legal representative for the Muslim Indian Traders]	1
	[he] [spent] [21 years in South Africa]	1
	[he] [spent] [21 years]	0
Precision		**0.66**
Proposed System	[Gandhi] [was] [24 when Gandhi arrived in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria]	1
	[Gandhi] [arrived] [to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria when]	0
	[Gandhi] [arrived] [to work as a legal representative for the Muslim Indian Traders when]	0
	[Gandhi] [arrived] [in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria]	2
	[Gandhi] [arrived] [in South Africa in 1893 to work as a legal representative for the Muslim Indian Traders]	2
	[Gandhi] [arrived] [to work as a legal representative for the Muslim Indian Traders based in the city of Pretoria]	2
	[Gandhi] [arrived] [to work as a legal representative for the Muslim Indian Traders]	2
	[Gandhi] [spent] [21 years in South Africa]	2
	[Gandhi] [spent] [21 years]	0
Precision		**0.78**

Table 5: Propositions extraction list from Wikipedia example 1.

Example 2: The Islamic Jihad Union (IJU), also known as Islamic Jihad Group (IJG), is a terrorist organization which splintered from the Islamic Movement of Uzbekistan (IMU), and has conducted attacks in Uzbekistan and attempted attacks in Germany.		
Extractor	**Extractions**	**Value**
ClausIE	[The Islamic Jihad Union IJU] [be known] [also as Islamic Jihad Group IJG]	1
	[The Islamic Jihad Union IJU] [be known] [also]	0
	[The Islamic Jihad Union IJU also known as Islamic Jihad Group IJG] [is] [a terrorist organization]	1
	[a terrorist organization] [splintered] [from the Islamic Movement of Uzbekistan IMU]	1
	[a terrorist organization] [has conducted] [attacks in Uzbekistan and attempted attacks in Germany]	1
Precision		**0.80**
Proposed System	[The Islamic Jihad Union IJU] [be known] [also as Islamic Jihad Group IJG]	1
	[The Islamic Jihad Union IJU] [be known] [also]	0
	[The Islamic Jihad Union IJU also known as Islamic Jihad Group IJG] [is] [a terrorist organization]	1
	a terrorist organization] [splintered] [from the Islamic Movement of Uzbekistan IMU]	1
	[a terrorist organization] [has conducted] [attacks in Uzbekistan and attempted attacks in Germany]	1
	[IJU] [is acronym for] [Islamic Jihad Union]	1
	[IJG] [is acronym for] [Islamic Jihad Group]	1
	[IMU] [is acronym for] [Islamic Movement of	1

	Uzbekistan]	
Precision		**0.87**

Table 6: Propositions extraction list from Wikipedia example 2.

5.4.2 On the NYT dataset:

NYT Example: Barack Obama held a similar series of meetings two years ago after tensions boiled over following a police shooting in Ferguson, Mo., that sparked protests and rioting. Those meetings prompted the president to provide federal funding for community policing and anti-bias efforts, including for the purchase of body-worn cameras. He also created a policing task force that released recommendations last year on how to build trust between law enforcement and the communities they protect.		
Extractor	**Extractions**	**Value**
ClausIE	[Barack Obama] [held] [a similar series of meetings two years ago after tensions boiled over following a police shooting in Ferguson Mo.]	1
	[Barack Obama] [held] [a similar series of meetings]	1
	[tensions] [be boiled] [Following a police shooting in Ferguson Mo.]	1
	[a police] [be shooting] [in Ferguson Mo.]	1
	[Ferguson Mo.] [sparked] [protests and rioting]	0
	[Those meetings] [prompted] [the president to provide federal funding for community policing and anti-bias efforts including for the purchase of body-worn cameras]	1
	[Those meetings] [prompted] [the president to provide federal funding for community policing and anti-bias efforts]	1
	[He] [created] [a policing task force also]	1
	[He] [created] [a policing task force]	1
	[a policing task force] [released] [recommendations last year]	1
	[a policing task force] [released] [recommendations on how to build trust between law enforcement and the communities they protect]	1
	[a policing task force] [released] [recommendations]	1
	[they] [protect] [null]	0
Precision		**0.84**
Proposed System	[Barack Obama] [held] [a similar series of meetings two years ago after tensions boiled over following a police shooting in Ferguson Mo.]	1
	[Barack Obama] [held] [a similar series of meetings]	1
	[tensions] [be boiled] [following a police shooting in Ferguson Mo.]	1
	[a police] [be shooting] [in Ferguson Mo.]	1
	[Ferguson Mo.] [sparked] [protests and rioting]	0
	[Those meetings] [prompted] [the president to provide federal funding for community policing and anti-bias efforts including for the purchase of body-worn cameras]	1
	[Those meetings] [prompted] [the president to provide federal funding for community policing and anti-bias efforts]	1
	[Barack Obama] [created] [a policing task force also]	2
	[Barack Obama] [created] [a policing task force]	2
	[a policing task force] [released] [recommendations last year]	1
	[a policing task force] [released] [recommendations on how to build trust between law enforcement and the communities they protect]	1
	[a policing task force] [released] [recommendations]	1
	[they] [protect] [null]	0
Precision		**0.86**

Table 7: Propositions extraction list from NYT example.

5.4.3 On the complex paragraphs dataset:

The third dataset is more complex and long compared to the previous datasets items. We selected the following example. "Alex Kalinovsky was born in Ukraine in 1974 and moved to the United States in 1997. He has been in the IT industry for more than 10 years, with experience that ranges from writing C and C++ applications to developing enterprise Java solutions. Since 1997, Alex has worked solely with Java and is proud to be one of its original evangelists. He has taught more than 15 classes on Enterprise Java technologies and worked as a mentor for many teams. Alex has written for various publications, including JavaWorld, Sun JavaSoft, Information Week, and the Washington Post. He is a Certified Enterprise Java Architect consulting for leading companies that use Java and J2EE. He is also a lead architect for WebCream, a revolutionary Java product that bridges Swing and HTML. In his spare time, Alex enjoys traveling, reading, windsurfing, snowboarding, and bodybuilding". The result of extraction listed in Table 8.

Extractor	Extractions	Value
ClausIE	[Alex Kalinovsky] [was born] [in Ukraine in 1974]	1
	[Alex Kalinovsky] [was born] [in Ukraine]	0
	[Alex Kalinovsky] [was moved] [to the United States in 1997]	1
	[Alex Kalinovsky] [was moved] [to the United States]	0
	[He] [has been] [in the IT industry for more than 10 years]	1
	[He] [has been] [in the IT industry with experience]	1
	[He] [has been] [in the IT industry]	0
	[experience] [ranges] [from writing C and C + + applications to developing enterprise Java solutions]	1
	[Alex] [has worked] [solely Since 1997]	1
	[Alex] [has worked] [solely with Java]	1
	[Alex] [has worked] [solely]	0
	[Alex] [is] [proud to be one of its original evangelists Since 1997]	1
	[Alex] [is] [proud to be one Since 1997]	0
	[Alex] [is] [proud to be one of its original evangelists]	0
	[Alex] [is] [proud to be one]	0
	[its] [has] [original evangelists]	0
	[He] [has taught] [more than 15 classes on Enterprise Java technologies]	1
	[He] [has taught] [more than 15 classes]	0
	[He] [has worked] [as a mentor for many teams]	1
	[Alex] [has written] [for various publications including JavaWorld Sun JavaSoft Information Week and the Washington Post]	1
	[Alex] [has written] [for various publications]	0
	[He] [is] [a Certified Enterprise Java Architect consulting for leading companies]	1
	[He] [is] [a Certified Enterprise Java Architect consulting]	0
	[leading companies] [use] [Java and J2EE]	1
	[He] [is] [a lead architect a revolutionary Java product that bridges Swing and HTML also]	1
	[He] [is] [a lead architect a revolutionary Java product that bridges Swing and HTML for WebCream]	1
	[He] [is] [a lead architect a revolutionary Java product that bridges Swing and HTML]	0
	[his] [has] [spare time]	0
	[Alex] [enjoys] [traveling reading windsurfing snowboarding and bodybuilding In his spare time]	1
	[Alex] [enjoys] [traveling reading windsurfing snowboarding and bodybuilding]	0
Precision		**0.53**
Proposed System	[Alex Kalinovsky] [was born] [in Ukraine in 1974]	1
	[Alex Kalinovsky] [was born] [in Ukraine]	0
	[Alex Kalinovsky] [was moved] [to the United States in 1997]	1
	[Alex Kalinovsky] [was moved] [to the United States]	0
	[Alex Kalinovsky] [has been] [in the IT industry for more than 10 years]	2
	[Alex Kalinovsky] [has been] [in the IT industry with experience]	2
	[Alex Kalinovsky] [has been] [in the IT industry]	0
	[experience] [ranges] [from writing C and C + + applications to developing enterprise Java solutions]	1
	[Alex] [has worked] [solely Since 1997]	1
	[Alex] [has worked] [solely with Java]	1
	[Alex] [has worked] [solely]	0
	[Alex] [is] [proud to be one of its original evangelists Since 1997]	1
	[Alex] [is] [proud to be one Since 1997]	0
	[Alex] [is] [proud to be one of its original evangelists]	0
	[Alex] [is] [proud to be one]	0
	[its] [has] [original evangelists]	0
	[Alex Kalinovsky] [has taught] [more than 15 classes on Enterprise Java technologies]	2
	[Alex Kalinovsky] [has taught] [more than 15 classes]	0

	[Alex Kalinovsky] [has worked] [as a mentor for many teams]	2
	[Alex] [has written] [for various publications including JavaWorld Sun JavaSoft Information Week and the Washington Post]	1
	[Alex] [has written] [for various publications]	0
	[Alex Kalinovsky] [is] [a Certified Enterprise Java Architect consulting for leading companies]	2
	[Alex Kalinovsky] [is] [a Certified Enterprise Java Architect consulting]	0
	[leading companies] [use] [Java and J2EE]	1
	[Alex Kalinovsky] [is] [a lead architect a revolutionary Java product that bridges Swing and HTML also]	2
	[Alex Kalinovsky] [is] [a lead architect a revolutionary Java product that bridges Swing and HTML for WebCream]	2
	[Alex Kalinovsky] [is] [a lead architect a revolutionary Java product that bridges Swing and HTML]	0
	[his] [has] [spare time]	0
	[Alex] [enjoys] [traveling reading windsurfing snowboarding and bodybuilding In his spare time]	1
	[Alex] [enjoys] [traveling reading windsurfing snowboarding and bodybuilding]	0
Precision		0.62

Table 8: Propositions extraction list from our complex sample paragraph.

5.5 Extractions Number and Evaluation

We have tested both ClausIE and the proposed solution with three datasets as we mentioned before, Wikipedia with 200 sentences, NYT dataset with 199 sentences and the third dataset with 5 complex paragraphs, which result in 37 sentences. We manually labelled all the extractions from all datasets and evaluated each extraction according to the methodology mentioned in section 5.3. For the Wikipedia dataset, when using the proposed solution, the whole number of extracted propositions are 830, 565 of them are correct, when using ClausIE, the whole number of extractions are 805, 543 of them are correct. For the NYT dataset, when using the proposed solution, the whole number of extracted propositions are 982, 644 of them are correct, when using ClausIE, the whole number of extractions are 953, 617 of them are correct. For the third dataset, when using the proposed solution, the whole number of extracted propositions are 193, 126 of them are correct, when using ClausIE, the whole number of extractions are 160, 93 of them are correct.
Our results are summarized in Tables 9 and 10 and Figure 3. Table 9 shows the whole number of correct extractions as well as the whole number of extractions for each extractor and dataset. Table 10 shows the precision of each extractor using the same dataset. Figure 3 plots the precision of each OIE extractor as a function of the number of extractions.

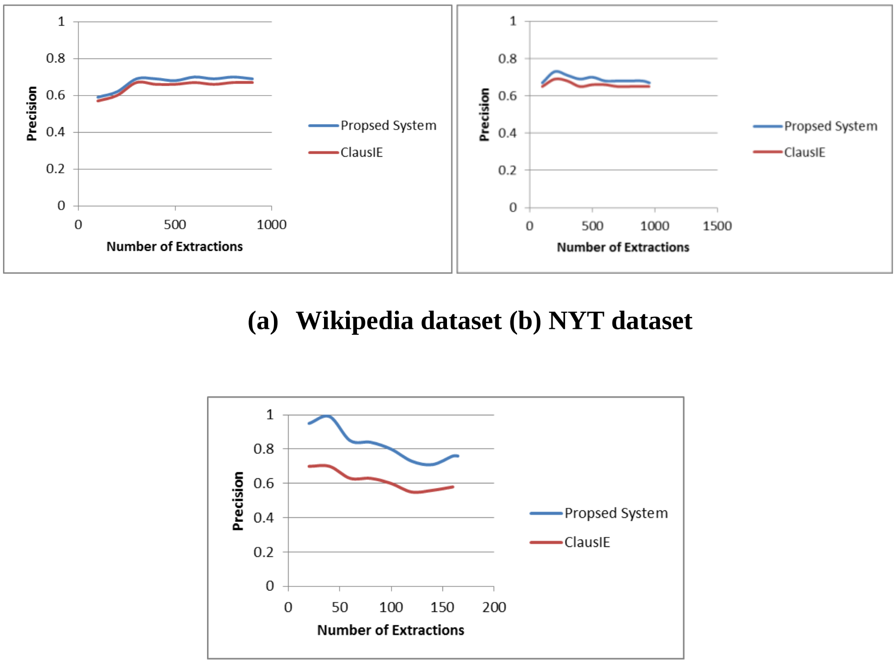
	Proposed System	ClausIE
Wikipedia dataset	565/830	543/805
NYT dataset	644/982	617/953
Complex Paragraphs dataset	126/193	93/160

Table 9: Number of correct extractions and total number of extractions.

	Proposed System	ClausIE
Wikipedia dataset	0.68	0.67
NYT dataset	0.65	0.64
Complex Paragraphs dataset	0.65	0.58

Table 10: Precision of extractors.

The result of extractions from the first two datasets shows that the precision of extractions are nearly similar for both extractors, but the number of extractions using the proposed solution is more than that of ClausIE, but the difference is still not so much because of the nature of input sentences, which are simple and have no much co-reference occurrences. The power of the proposed solution, which represented in added features of coreference resolution and acronyms detection, appear in the third dataset where sentences are more complex and have many references to each other. The high recall and consistently good precision of the proposed solution observed in our experiments indicates that our enhancements over ClausIE are a viable approach enrich the process of Open Information Extraction (OIE) and hence the process of Text Mining in open domain.



(a) Wikipedia dataset (b) NYT dataset

(c) Custom dataset

Figure 3: Plots the precision.

5.6 Mining in Extractions

We have applied the data mining algorithms mentioned in sections 4.1.4.1 and 4.1.4.2 for Frequent Item Sets and Top-K Association Rules respectively on the extractions from a sample of our third dataset, where complex paragraphs dataset are used, to evaluate the proposed solution.

Our sample dataset is: "Oracle Corporation is an American multinational computer technology corporation, headquartered in Redwood City, California. The company primarily specializes in developing and marketing database software and technology, cloud engineered systems and enterprise software products—particularly its own brands of database management systems. In 2011 Oracle was the second-largest software maker by revenue, after Microsoft.

The company also develops and builds tools for database development and systems of middle-tier software, Enterprise Resource Planning (ERP) software, Customer Relationship Management (CRM) software and Supply Chain Management (SCM) software.

Larry Ellison, a co-founder of Oracle, served as Oracle's CEO from its founding until September 18, 2014, when it was announced that he would be stepping down, with Mark Hurd and Safra Catz to become CEOs. Ellison became executive chairman and CTO. He also previously served as chairman until his replacement by Jeff Henley in 2004, before becoming executive chairman in 2014. On August 22, 2008, the Associated Press ranked Ellison as the top-paid chief executive in the world".

We have run the proposed solution over this sample text, it produces the following list of sentences where each sentence is presented by its corresponding list of CSV tokens as shown in Table 11.

Sentence	CSV Tokens
Oracle Corporation is an American multinational computer technology corporation, headquartered in Redwood City, California.	36676, 51887, 63036, 181994, 29602, 7814, 51887, 105548, 74374, 80390

Larry Ellison, a co-founder of Oracle , served as Oracle 's CEO from its founding until September 18 , 2014 , when it was announced that Larry Ellison would be stepping down , with Mark Hurd and Safra Catz to become CEOs.	97787, 80429, 89182, 36676, 87278, 1899, 140881, 199531, 173158,135715, 3709, 338, 143045
The company also develops and builds tools for database development and systems of middle-tier software , Enterprise Resource Planning -LRB- ERP -RRB- software , Customer Relationship Management -LRB- CRM -RRB- software and Supply Chain Management -LRB- ...	73031, 197856, 60227, 1970, 51876, 41930, 59564, 203149, 203148, 203151, 203150, 203153, 203152
Larry Ellison also previously served as chairman until his replacement by Jeff Henley in 2004, before becoming executive chairman in 2014.	197856, 198040, 92608, 1525, 175178, 73919
In 2011 Oracle was the second-largest software maker by revenue, after Microsoft.	36676, 5819, 180129, 59564, 73044, 123622
The company primarily specializes in developing and marketing database software and technology, cloud engineered systems and enterprise software products -- particularly its own brands of database management systems.	73031, 198200, 125406, 496, 60227, 59564, 7814, 72583, 6479, 198344, 161289, 9339
Ellison became executive chairman and CTO.	97787, 73919, 92608
On August 22, 2008, the Associated Press ranked Ellison as the top-paid chief executive in the world.	140878, 185943, 780, 178274, 97787, 29162, 183298, 89152, 73919, 52995

Table 11: Sentences and their corresponding CSV tokens.

The CSV tokens are the IDs of the words to be mined in the dictionary database of our system. The mining process is performed in these CSV tokens. When using FIN algorithm for mining and discovering item-sets (group of items) occurring frequently in the extracted propositions from the proposed solution with a minsup of 30%, FIN produces the result in Table 12.

Token	Word	Support
97787	Ellison	3
36676	oracle	3
73919	executive	3
59564	software	3

Table 12: Frequent item-set mining result on sample extractions.

When mining with Top-K Rules algorithm for discovering the top-k association rules appearing in our transaction database for the extracted propositions, with k = 1 and minconf = 0.8, we obtain the top-1 rules in the database having a confidence higher or equals to 80%. We found that the top association exists between the two items ["software company" and "database"] which means that if item "**software company**" appears, it is likely to be associated with item "**database**" with a confidence of 100% in a transaction.

VI. SOCIAL MEDIA USE CASE

We applied our proposed approach on a selected use case of a social media like Facebook. We mined the comments of a public discussion. The mining process includes a pre-processing step, which first clusters the comments in a set of clusters. Each cluster contains a sub set of similar comments. The clustering is performed as following [10]:

1. Calculate (tf*idf) value for each comment in the input file.
2. Calculate similarity matrix by using the (tf*idf) values of the records.
3. Take most similar records per each comment and make them as clusters initially.
4. Use the transitive rule A,B are most similar and B,C are most similar; A and C are likely to be similar. This imply that A, B, C are in the same cluster.
5. Merge the clusters based on the above rule for all the records.

- **tf:** Term Frequency, defines how frequently a term occur in a document.
- **Idf**: Inverse document frequency, is the frequency of a word in the whole set of documents.
- **Similarity Matrix**: 2 dimensional matrix representation of a record's similarity with all other records.

After clustering the comments, we apply the proposed solution on each cluster separately and mine it to extract the important information and association rules in it.
Sample Facebook discussion with comments, its processing and mining result is presented in Appendix I.

VII. CONCLUSION AND FUTURE WORK

We presented in this thesis an approach for text-mining using open information extraction and data mining techniques. We presented an enhanced implementation to ClausIE [12], as an approach for open information extraction, through the addition of two features, co-reference resolution and acronyms detection, and this, in turn, helps in improving both the quality and the quantity of the extracted information in terms of precision and recall. For data mining, we used association rules mining, but other methods can be used successfully. The proposed solution aims to obtain a shallow and a lot of semantic representation of large amounts of natural language text in the form of verbs (or verbal phrases) and their arguments, this extracted information is stored in a database for farther mining and processing. Due to the huge extracted information that could be produced as a result of applying the proposed solution on a large amount of natural language text, we belief that the proposed solution could help and support other fields of research like, Question Answering (QA) systems, Ontology Building in open domain and Semantic Web technologies.

VIII. REFERENCES

1. J Shaidah, MA Hejab, Techniques, Applications and Challenging Issue in Text Mining. International Journal of Computer Science Issues 2012; 9: 431-436.
2. H Marti, What Is Text Mining? 2003.
3. F Dayne, K Nicholas, Boosted Wrapper Induction. 17th National Conference on Artificial Intelligence 2000; 577-583.
4. G Vishal, LS Gurpreet, A Survey of Text Mining Techniques and Applications. Journal of Emerging Technologies in Web Intelligence 2009; 1: 60-76.
5. H Jiawei, K Michelin, et al. Data Mining Concepts and Techniques. Morgan Kaufmann publishers 2001: 1-740.
6. WB Michael, Survey of Text Mining: Clustering, Classification and Retrieval. Springer Verlag, New York 2004: 24-43.
7. NB Shamkant, E Ramez, Fundamentals of Database Systems. Pearson Education pvt Inc, Singapore 2000 841-872.
8. F Weiguo, W Linda, et al. Tapping into the Power of Text Mining. Communications of ACM, Blacksburg 2005; 49: 76-82.
9. B Michele, CJ Michael, et al. Open information extraction from the web. Proceedings of the 20th International Joint Conference on Artifical intelligence, Hyderabad 2007: 2670-2676.
10. D Zhi-Hong, L Sheng-Long, Fast mining frequent item-sets using Nodesets. Expert System. Application 2014; 41: 4505-4512.
11. Mausam, S Michael, et al. Open language learning for information extraction. In Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2012.
12. CD Luciano, G Rainer, ClausIE: Clause-Based Open Information Extraction. International World Wide Web Conference 2013.
13. CE Mary, MJ Raymond, Relational Learning of Pattern-Match Rules for Information Extraction. Proceedings of the 16th National Conference on Artificial Intelligence.1999: 9-15.
14. T Haidong, Y Jia, A Survey for Information Extraction Method 2011.
15. P Thierry, S Horacio, et al. Multi-source, multilingual information extraction and summarization. Theory and Applications of Natural Language Processing 2013.
16. VF Philippe, S Vincent, Mining Top-K Non-Redundant Association Rules. International Symposium on Methodologies for Intelligent Systems 2012: 31- 40.
17. http://nlp.stanford.edu/software/tokenizer.shtml,
18. http://nlp.stanford.edu/software/tagger.shtml,
19. https://wordnet.princeton.edu/

20. R Prakhyath, MT Vijaya, Survey on Existing Text Mining Frameworks and A Proposed Idealistic Framework for Text Mining by Integrating IE and KDD. International Journal of Computational Engineering Research 2014; 4: 2250-3005.
21. MJ Raymond, N Yong UN, Text Mining with Information Extraction. Multilingualism and Electronic Language Management 2005: 141-160.