Computing and Informatics, Vol. 38, 2019, 986–1008, doi: 10.31577/cai_2019_4_986

A NEW OPEN INFORMATION EXTRACTION SYSTEM USING SENTENCE DIFFICULTY ESTIMATION

Vahideh Reshadat

Miyaneh Technical and Engineering Faculty
University of Tabriz, Tabriz, Iran
e-mail: v.reshadat@gmail.com

Heshaam Faili

School of Electrical and Computer Engineering
College of Engineering
University of Tehran, Tehran, Iran
e-mail: hfaili@ut.ac.ir

Abstract. The World Wide Web has a considerable amount of information expressed using natural language. While unstructured text is often difficult for machines to understand, Open Information Extraction (OIE) is a relation-independent extraction paradigm designed to extract assertions directly from massive and heterogeneous corpora. Allocation of low-cost computational resources is a main demand for Open Relation Extraction (ORE) systems. A large number of ORE methods have been proposed recently, covering a wide range of NLP tools, from "shallow" (e.g., part-of-speech tagging) to "deep" (e.g., semantic role labeling). There is a trade-off between NLP tools depth versus efficiency (computational cost) of ORE systems. This paper describes a novel approach called Sentence Difficulty Estimator for Open Information Extraction (SDE-OIE) for automatic estimation of relation extraction difficulty by developing some difficulty classifiers. These classifiers dedicate the input sentence to an appropriate OIE extractor in order to decrease the overall computational cost. Our evaluations show that an intelligent selection of a proper depth of ORE systems has a significant improvement on the effectiveness and scalability of SDE-OIE. It avoids wasting resources and achieves almost the same performance as its constituent deep extractor in a more reasonable time.

Keywords: Information extraction, open information extraction, relation extraction, knowledge discovery, fact extraction

1 INTRODUCTION

Information Extraction is the task of automatically extracting structured data from unstructured text. One of the core information extraction tasks is the relation extraction, which aims at extracting semantic relations among entities from natural language text. Relation extraction can potentially benefit a wide range of NLP tasks such as: Web search, question answering, ontology learning, summarization, building knowledge bases, etc. [1].

The huge and fast-growing scale, a mixed genre of documents and infinite types of relations are challenges of the Web-scale relation extraction [2]. The traditional approaches to information extraction assume a fixed set of predefined target relations and usually do not scale to corpora where the number of target relations is very large [3]. An alternative paradigm OIE aims to scale information extraction methods to the size and diversity of the Web corpus. OIE systems extract relational tuples from texts, without requiring a pre-specified vocabulary [4].

The key goals of OIE are: 1. domain independence, 2. unsupervised extraction, and 3. scalability to large amounts of text [5]. Scalability of OIE systems relies on the different sophistication levels of the NLP tools they use. Shallow extractors try to improve performance by limiting extraction procedure to shallow linguistic analysis. Although the ORE approaches in this category (such as TextRunner [6], WOEpos [7], ReVerb [8], R2A2 [9] and SONEX [10]) are fast and more scalable, they are unable to deal with complicated structures such as long distance relations. In addition, due to usage only shallow syntactic features, high performance is not guaranteed, thus resulting in a significant drop of effectiveness.

In contrast to shallow extractors, some approaches (such as Wanderlust [11], WOEparse [7], KrakeN [12], OLLIE [4], ZORE [13], DepOE [14], SRL-IE-Lund [15], SRL-IE-UIUC [15], the methods proposed in [16] and [17]) use deep syntactic or semantic analysis tools such as dependency parsing. These extractors are generally more expensive than the previous extractors; they trade efficiency for improved precision and recall [5]. The former extractors are rapid, guarantee scalability and require less effort due to usage shallow syntactic analysis, while the latter extractors are efficient for precision and recall but time consuming and require considerable effort due to usage deep syntactic analysis in the extraction process [18].

Given the pros and cons of shallow and deep extractors, we proposed an approach for automatic estimation of ORE difficulty. We developed different classifiers that recognize sentences that are hard for ORE task and pass them to a deep extractor. Thus, it attempts to categorize them with the aim of reducing computational cost. The proposed approach is a combination of two types of OIE systems and we employed ReVerb [8] and EXEMPLAR [19] as shallow and deep OIE extractors, respectively.

According to the results in [19], shallow methods handle ten times more sentences than deep ones. We examined the trade-off between effectiveness (F-measure) and efficiency (computational cost) and found that using a deep extractor on the intelligent subset of input sentences can yield a substantial improvement in F-measure. We present a novel approach for predicting ORE difficulty using different classifiers with light-weight features. The classifiers recognize sentences that are hard for ORE task and pass input sentences to a deep extractor only if needed. Therefore, our difficulty classifiers prioritize the sentences likelihood of improving performance and lead to better allocation of computational resources. Sentence difficulty is used in many applications of natural language processing such as measuring translation difficulty [21], evaluating the reliability of parses [22], measuring text difficulty [23] and text readability [24], etc. The idea of this work can be ported into other tasks in natural language processing. Application systems such as Speech Processing, Question Answering and Search Engines can benefit from automatic detection of difficult subtasks.

The rest of this paper is organized as follows. Section 2 introduces previous works in the areas of OIE systems. Our proposed approach is described in Section 3. We present results of our experiments in Section 4 and end with the conclusion in Section 5.

2 RELATED WORKS

In this section we review some related works on OIE, in particular works on ORE. OIE has received much attention recently. It covers a wide range of NLP tools, from shallow (e.g., part-of-speech tagging (POS)) to deep (e.g., semantic role labeling (SRL)). These systems can be divided into two main categories based on the linguistic analysis which is applied for relation extraction task [18, 5, 14]. In the following two subsections, we examine these two categories.

2.1 Deep Open Information Extraction Systems

ORE approaches which use parsing-based or SRL-based tools are grouped in deep OIE systems. Most deep OIE systems apply dependency tree paths to learn extraction patterns. Wanderlust [11] uses hand-labeled training data to learn extraction patterns on the dependency tree. The authors of this system annotated 10 000 sentences parsed with LinkGrammar. This system learns 46 general link paths as patterns for relation extraction. WOEparse [7] is a pattern classifier learned from dependency path patterns which uses typed dependencies as features [18]. PATTY [25] extracts textual patterns from sentences based on paths in the dependency tree between the two named entities. It finds the shortest path in the dependency tree that connects the two named entities. The TreeKernel approach [26] first inspects whether there is a relation between a pair of entities in a sentence and then whether there are explicit relation words for this pair. A set of syntactic patterns is used for generating candidate relations. One of the main drawbacks of dependency-based deep OIE systems is restricting extraction to the paths of dependency tree.

Some approaches use bootstrapping to learn patterns. OLLIE [4] is a hybrid approach based on bootstrapping which learns pattern templates automatically from a training set that is bootstrapped from relations extracted by ReVerb. OLLIE produces n-ary extractions by merging binary relations and has 1.9 to 2.7 times more area under precision-yield curves [1] compared to ReVerb and WOE. BONIE [27] is an open numerical relation extractor, for extracting OIE tuples where one of the arguments is a number or a quantity-unit phrase. BONIE also uses bootstrapping to learn the specific dependency patterns that express numerical relations in a sentence. Bootstrapping methods have some limitations because extraction samples can vary considerably depending on initial seed selection.

Some OIE methods are designed for languages other than English. Similarly, most of them are based on rules or patterns. ZORE [13] is a syntax-based Chinese ORE system that extracts relations and semantic patterns from Chinese texts. The approach proposed in [17] also focuses on Chinese ORE. This system can be considered as a pipeline of word segmentation, POS tagging and parsing [18]. An OIE system for German language was proposed in [28]. It is a straightforward approach for adapting PropS, a rule-based predicate-argument analysis for English, to a new language, German. DepOE [14] is a multilingual OIE system based on fast dependency parsing. It uses DepPattern [29], a multilingual dependency-based parser, to analyze sentences and obtain fine-grained information. Then, a small set of extraction rules is applied and the target verb-based triples are generated. There is a more recent version of DepOE system, called ArgOE [30]. ArgOE is a multilingual rule-based OIE method that obtains as input dependency parses in the CoNLL-X format, recognizes argument structures within the dependency parses, and extracts a set of basic propositions from each argument structure. Since most of the OIE systems designed in languages other than English are based on rules or patterns, they have the same problems as rule-based and pattern-based methods.

Most OIE approaches usually extract binary facts and are not designed to capture n-ary relations. KrakeN [12] addresses this limitation by capturing unary, binary and higher order n-ary facts. It has been built specifically for capturing complete facts from sentences and can extract more facts per sentence with high precision. EXEMPLAR [19] addresses the problem of extracting n-ary relations by using handcrafted rules over dependency trees. These rules are applied to each candidate argument individually by inspecting the path between an entity and a relational word. OIE approaches which deal with n-ary relations can increase the number of correct and informative extractions and achieve high precision and recall.

[1] Receiver Operating Characteristic (ROC)

Some deep OIE methods separate the detection of "useful" pieces of information expressed in a sentence from their representation in terms of extractions. ClauseIE [5] is a clause-based approach which uses linguistic knowledge about the grammar of the English language to first detect clauses in an input sentences and to subsequently identify the type of each clause according to the grammatical function of its constituents [18]. CSD-IE [31] is a method that uses contextual sentence decomposition for OIE. A sentence is decomposed into the parts that are semantically dependent and then the (implicit or explicit) verb in each part is identified and the facts are obtained [8]. The performance of decomposition-based OIE systems is highly dependent on the detection of effective pieces which produce the facts.

Although the majority of deep OIE systems are parser-based, there is a limited quantity of approaches that exploit semantic role labelers. A deep OIE system based on SRL has been proposed in [15]. This system has been developed based on two SRL systems: UIUC [32] and LUND [33]. It produces the extractions by applying some rules on the outputs of these SRL systems. The authors proposed two hybrid methods that employ SRL only on a specific subset of TextRunner outputs. This work is similar to our approach in terms of combining two OIE systems, but there are some differences. Applying TextRunner to all input sentences and using SRL via some restrictions rules on TextRunner outputs are the main differences. Another version of SRL-IE was implemented in [20] by relying on the output of two SRL systems: LUND [34] and SwiRL [35]. Efficiens [20] has a module for each NLP tool. The Efficiens[POS] module relies on POS tagging, while the Efficiens[DEP] and Efficiens[SRL] rely on dependency parsing and SRL, respectively. Since SRL-based deep methods need more computational time than parse-based deep methods, they are computationally expensive, even though they are robust to noisy text. The related surveys are summarized in Table 2.

Although deep OIE systems have a high performance, the cost of leveraging deep NLP tools and scarcity of them in other languages are the main challenges of deep OIE methods. In this paper, we present an approach to alleviate these critical challenges. We developed a strategy which mitigates these challenges by intelligent use of different methods.

2.2 Shallow Open Information Extraction Systems

ORE methods, which are based on shallow NLP tools (such as POS taggers), are grouped in shallow OIE systems. Some shallow OIE systems use classifiers with some lightweight features to recognize the relation between name entities in a sentence. TextRunner [36] is the first OIE system. It applies a Naive Bayes classifier which determines whether the context between a pair of noun phrases in a sentence describes a relation instance or not. WOEpos [7] is also inspired by TextRunner and limited to shallow features like POS tags. WOEpos exploits the relations in Wikipedia Infoboxes to match corresponding sentences in an unlabelled corpus that mention these relations. It uses these examples as relation-independent training data to learn an unlexicalized extractor. R2A2 [8] uses an argument learning com-

	N	D	S	E	P	R
Wanderlust [11]	×	✓	×	✓	✓	×
$\text{WOE}_{\text{parse, 7}}$	×	✓	×	✓	✓	×
PATTY [25]	×	✓	×	✓	✓	✓
TreeKernel [26]	×	✓	×	✓	✓	×
OLLIE [4]	✓	✓	×	✓	✓	×
BONIE [27]	×	✓	×	✓	✓	×
ZORE [13]	×	✓	×	×	✓	×
Chinese OIE [17]	×	✓	×	×	×	✓
German OIE [28]	✓	✓	×	×	×	✓
DepOE [14]	×	✓	×	✓	×	✓
ArgOE [30]	×	✓	×	✓	×	✓
KrakeN [12]	✓	✓	×	✓	×	✓
EXEMPLAR [19]	✓	✓	×	✓	×	✓
ClauseIE [5]	✓	✓	×	✓	×	✓
CSD-IE [31]	×	✓	×	✓	×	✓
SRL-IE [15]	✓	×	✓	✓	×	✓
Efficiens [20]	✓	✓	✓	✓	×	×

Table 1. Comparison of different deep OIE methods. N: extracts N-ary relations? D: extracts relations based on dependency parse tree? S: extracts relations based on SRL? E: extracts relations in English language? P: extracts relations based on patterns? R: extracts relations based on rules?

ponent. It makes use of a number of classifiers to identify the arguments of a verb phrase (based on hand-labeled training data). Two classifiers identify the left and right bounds for the first argument and one classifier identifies the right bound of the second argument.

Some shallow OIE systems are based on patterns. ReVerb [8] is a strong and successful pattern-based shallow OIE system. It makes use of a simple POS tag sequence as a syntactic constraint in order to extract relation phrases and eliminate incoherent extractions and also reduce uninformative extractions. ReVerb exploits a lexical constraint that aims to alleviate the amount of over-specified extractions. Experiments show ReVerb outperforms TextRunner and its performance is more than twice as much as that of TextRunner [7, 18, 37]. SONEX [10] extends ReVerb by detecting patterns with appositions and possessives [19]. It identifies every entity pair (e.g., "Google", "Apple Inc.") and all sentences where this pair is mentioned together. From these sentences, SONEX extracts a context (e.g., a list of surrounding words) for the pair and applies clustering techniques to group together pairs with similar contexts. SONEX sees each cluster of entity pairs as a relation. LSOE [38] is also a pattern-based system which exploits two kinds of patterns: 1. generic patterns, 2. rules from Cimiano and Wenderoth proposal [39]. The performance of LSOE was compared with two other OIE systems: ReVerb and DepOE. The results show that LSOE extracts relations that are not learned by other extractors and also achieves compatible precision.

There are some shallow OIE methods such as R-OpenIE [40] which are based on rules. R-OpenIE defines some text-based rules to generate relation extraction templates. It applies the cascaded finite-state transducer model to match the satisfied relational tuples.

The main drawback of all the above shallow OIE approaches is that they are inefficient for high performance. With the fast growth of the Internet and the emerging problem of information overload, the computational cost of processing a large volume of information is becoming an increasingly important issue of artificial intelligence researches. Based on the methods discussed above, despite the high performance of pure deep OIE systems, applying them is time consuming. Unlike deep OIE systems, shallow ones are fast and do not achieve high performance measures. Therefore, each one of these categories has their own pros and cons and raises the question of what is the trade-off between NLP depth (and associated computational cost) versus effectiveness. In this paper we develop some probabilistic classifiers that apply different combination parameters as features, for different classes of extractors. This approach is not limited to specific types of system. It divides the input sentences to proper extractors.

3 SENTENCE DIFFICULTY ESTIMATION FOR OPEN INFORMATION EXTRACTION SYSTEMS

Various levels of linguistic analysis tools from shallow (e.g. POS tagging) to deep (e.g. SRL) were used to develop OIE systems. Applying expensive NLP tools for extracting facts from huge and heterogeneous corpora in reasonable time is time-consuming and costly. This problem worsens when such methods are applied on World Wide Web documents. In addition, tools for automatic deep analysis are available only for a limited number of natural languages, and produce imperfect results. Manual deep analysis, on the other hand, is time consuming and expensive [41]. Automatic tools for approaches that rely only on a shallow linguistic analysis are available for many languages and sufficiently reliable [41]. These extractors are usually fast, but the restriction to shallow syntactic analysis reduces maximum recall and/or may lead to a significant drop of precision at higher points of recall [5]. Indeed, there is a need to have a system that enables effective use of available time and offers a reasonable balance of precision and recall. The advantages of these two kinds of extractors motivate us to focus on developing a method that gets the best of both worlds. A hybrid OIE paradigm by incorporating strengths of ReVerb and EXEMPLAR is suggested. Figure 1 presents the general framework of our proposed approach.

Preprocessor. The preprocessor converts raw web pages into a sequence of sentences. It takes web pages as input and transforms them to plain sentences using pre-processing tools. The pages were then segmented into sentences, tokenized, tagged with POS and chunked using the OpenNLP[2] package.

[2] Downloadable at `http://opennlp.sourceforge.net`.

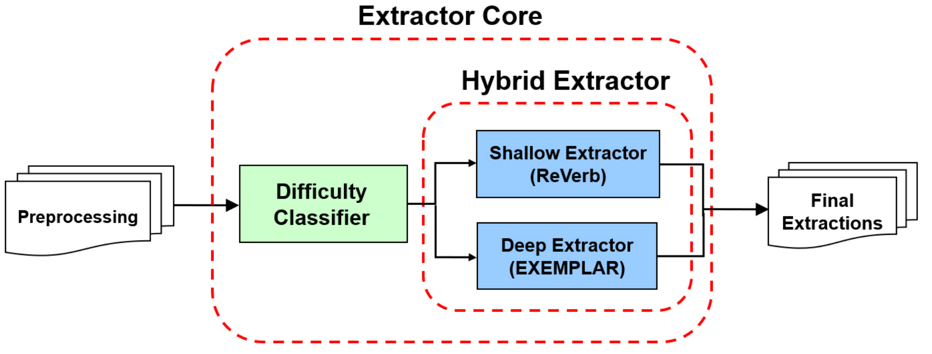


Figure 1. SDE-OIE's framework: Difficulty classifier exploits the best of both the shallow and the deep OIE extractors

Extractor Core. This component takes each sentence and assigns it to a proper extractor. Extractor Core consists of two main subcomponents: Sentence Difficulty Estimator and Hybrid Extractor.

Difficulty Classifier. The difficulty classifier and the hybrid extractor are the main parts of SDE-OIE. SDE-OIE reads each sentence sequentially. Given a sentence, the difficulty classifier extracts a set of features and predicts whether it is difficult or not. In other words, for each sentence, the difficulty classifier finds the most appropriate system for processing it. In regards to binary classification of sentences, we use different classifiers; a Logistic Regression, a Naive Bayes and a Decision Tree. Due to strong classification results of these classifiers, they have been used for many classification problems in computational linguistics. In addition to classification, we need to find difficult sentences, where we care about the severity of the extraction difficulty. For this purpose, we benefit from the classification score as a difficulty measure.

SDE-OIE focuses on the difficulty estimation of a relation extraction task for input sentences of OIE systems. We formulate this problem as a classification problem, where the goal is to assign a class label of *easy* or *difficult* to a candidate sentence s based on a classifier c and then pass it to an appropriate extractor. $c{:}s \rightarrow \{easy, difficult\}$

For this purpose, different probabilistic classifiers were used. We used Naive Bayes, Logistic Regression and Decision Tree to automatically assign a difficult/easy class to each input instance. Naive Bayes is a simple and common generative classifier that chooses the most probable extractor class out of a set of possible classes given a feature vector [42]. The features of data samples are independent. Naive Bayes employs the normal distribution to model numeric attributes.

Logistic regression belongs to the family of classifiers known as the exponential or log-linear classifiers. Like Naive Bayes, it works by extracting some set of weighted features from the input, taking logs, and combining them linearly [43]. In order to train a model to classify with the least amount of error possible, the cost function should be minimized. Gradient descent is our learning algorithm that finds values for the parameters that result in the best parameter values and a smaller minimum error.

Binary Decision Tree consists of terminal vertices and nonterminal vertices. Compared to Naive Bayes, decision tree is a somewhat more transparent approach that lends itself to inspection [42]. Our decision tree was built by C4.5. For implementation of these classifiers we used the Weka package. A variety of features have been used to train the classifiers. These features are discussed with more detail in Section 3.1 and the appendix.

Hybrid Extractor. The hybrid extractor also includes two main subcomponents, a shallow (ReVerb) and a deep (EXEMPLAR) OIE system. A complete and fair experimental comparison of 10 approaches have been presented in [19] and [20]. According to that research, ReVerb is the fastest method based on matching patterns over POS tags. SONEX is a shallow ORE system which produces results comparable to ReVerb. It focuses on overcoming the challenges in deploying ORE systems in the blogosphere and uses a clustering algorithm to group pairs with similar context together in a large scale. Beside the challenges with large-scale clustering (time and space), it recognizes instances at corpus-level. Since our system is sentence-based (meaning the extraction process can take individual sentences as input), we employ ReVerb as the shallow constituent extractor. Through these adaptations, we gain a range of extraction quality improvements at the sentence level. EXEMPLAR is based on using rules over dependency trees. It outperforms ReVerb and differs greatly in efficiency. It achieves the best effectiveness and is faster than the deeper methods such as SRL-based OIE extractors. Thus, EXEMPLAR's processing time is much less. More details about these OIE systems were presented in Section 2. As illustrated in Figure 1, final extractions are obtained by taking the union of these two extractors' outputs.

The proposed approach has some advantages in the following aspects:

- While in the structure of previous similar approaches, a pure shallow or deep linguistic analysis tool is applied to all input sentences at least once; to our knowledge, we are the first to propose an approach to partition the input to an appropriate extractor in order to achieve higher performance.
- The constituent systems of the extractor core are based on shallow and deep linguistic analysis tools and neither ReVerb nor EXEMPLAR needs training data. Therefore, SDE-OIE's performance will be independent from training parameters.

- The proposed approach is independent of its constituent systems and can be designed by other shallow or deep systems. In other words, it is a general framework and it is not designed for certain OIE systems. Hence, it can be designed by incorporating different systems with different depths.
- As will be discussed in Section 4, our experiments indicate that extraction difficulty can be modeled and automatically predicated with decent accuracy. Detecting difficult sentences has significant influence on the extraction time and quality. It prevents wasting resources and helps to achieve approximately the same performance as the deep constituent extractor. SDE-OIE is particularly effective when there is a large dataset and the processing time is limited. In this case, our hybrid extractor makes effective use of available time and runs the best algorithm given the available computation time.

In case all input sentences are difficult, using a difficulty classifier would be an overhead operation rather than applying a deep extractor individually. Additionally, sometimes only a few sentences in the whole dataset produce better instances with deep NLP tools. In this case, a classifier that applies deep extractor for most sentences will be wasting computational resources for the rest of the sentences in that dataset. SDE-OIE will prefer shallow extractor when both extractors produce correct extraction and therefore efficiency improves.

3.1 Feature Set

Deep features could improve precision and recall over shallow syntactic features, but at the cost of extraction speed. For instance, parser-based features can help to handle complicated and long distance relations in sentences. Such cases usually cannot be detected by shallow features. Regarding the computational cost associated with rich syntactic features, we used 61 light-weight features. All features are independent of applied classifiers, scalable, domain independent, and can be evaluated at extraction time without the use of expensive tools.

These features allow the difficulty classifier to estimate the challenge that the system faces in extracting instances from a sentence. Although these features can be extracted from the underlying systems, they are collected from the syntactic and semantic structure of the sentence. Since our difficulty modeling is system-independent, we particularly do not incorporate knowledge (features) from the underlying OIE systems into the difficulty classifier. Additionally, we use source-language features which bring deeper linguistic knowledge into our modeling and classification.We list below some important features which the difficulty classifier uses to recognize the class of an input sentence s:

- F1: s contains at least two name entities where the context between them has a verb phrase.
- F2: Number of capital words in s is greater than 6.

- F3: s contains communication verbs [1].
- F4: Number of stop words in s is equal or greater than 10.
- F5: s contains 'if'.
- F6: s contains at least one coordinating conjunction (and, but, for, nor, or, so, yet).
- F7: s contains 'say'.
- F8: s contains at least one pronoun (PRP, PRP$, WP, WP$).
- F9: s contains 'that' or 'whether'.
- F10: s contains at least one relative pronoun.
- F11: s contains 'there'.
- F12: s contains feature1 (F1) and the first name entity is a pronoun.
- F13: s contains F1 and the second name entity is a pronoun.
- F14: s contains F1 and there is a preposition ('to' or 'in') in s.
- F15: s contains F1 and there is a verb before the first name entity.
- F16: s contains F1 and there is a verb after the second name entity.
- F17: s contains F1 and the first name entity is a proper noun.
- F18: s contains at least one entity pair where there is a verb after the second name entity.
- F19: Length(s) is greater than 10.
- F20: s contains cognition verbs [2].

Examples of other features include presence of punctuation, capitalization, WH-words, comma, quotation, parentheses, specific POS tag sequences, a verb with a specific tag (such as vbz, vbg, vbd, vbn, vbp, vb) in the sentence and a specific preposition at the end of the sentence (such as to, in, for, of, on). Following feature extraction, this set of automatically labeled feature vectors is used for training the classifier; then each sentence is passed to an extractor based on the classifier output.

4 EXPERIMENTAL RESULTS

In this section, we first describe the benchmark dataset and performance metrics, and then give the evaluation results obtained by our approach, baseline methods and state-of-the-art approaches.

4.1 Dataset

A gold standard data and a set of features are required to train the difficulty classifier. The lack of standard dataset is one of the main challenges of the OIE systems [20]. The current evaluation approaches rely on manual evaluation (e.g., [7, 8, 4, 12, 14, 5, 31, 44, 26]), whose main limitation is that it is not scalable. Combining available datasets to make a large one has some difficulties. Differences in annotation and evaluation methodology are some of these challenges. Manual creation of a large dataset, on the other hand, is time consuming and expensive.

Based on available resources, we used two state-of-the-art datasets [19] to validate and compare our approach with other methods developed for extracting open relations from the Web. These datasets contain relatively more data than the others (e.g., [7, 8, 4, 12, 14, 5, 30, 26, 44]). They are standard datasets which have been used in several recent studies such as [45, 46, 47].

This datasets try to alleviate the problems related to the lack of ground truth and differences in evaluation methodologies by providing reusable annotations that are flexible and can be used to evaluate a wide range of methods [19, 20]. They cover sentences from the New York Times (NYT-500), the Penn Treebank (PENN-100), a popular Web corpus (WEB-500) and a much larger dataset from the New York Times which has been annotated automatically. WEB-500 is a commonly used dataset, developed for the TextRunner experiments [3]. This dataset contains 500 sentences extracted from search engine snippets. These sentences are often incomplete and grammatically unsound, representing the challenges of dealing with web text. NYT-500 represents the other end of the spectrum with individual sentences from formal, well written new stories from the New York Times corpus [48]. PENN-100 contains sentences from the Penn Treebank recently used in an evaluation of the TreeKernel method [26]. The NYT-500 and the WEB-500 are used as training data and the PENN-100 is used as test data. We also randomly selected 300 sentences from the data source which was built automatically from Freebase and WordNet [19] as our test set.

The gold standard data is a set of sentences which have easy or difficult labels. Given a corpus, SDE-OIE should select sentences for the shallow/deep extractor. We manually annotated these datasets. We label a sentence as easy if the shallow extractor generates a correct result. In cases where the shallow extractor generates an incorrect result, it is labeled as difficult, except for cases where the deep extractor also generates an incorrect result. A sentence is also labeled as easy if the shallow extractor has no output for that sentence, but the deep extractor generates an incorrect result. In this case, if the deep extractor generates a correct result, the sentence will be labeled as difficult.

4.2 Performance Measures

Our evaluation focuses on the extraction of relation instances at sentence level. The metrics used in the evaluations are: Precision (P), Recall (R) and F-measure (F). Precision is the ratio of the number of correctly extracted instances to the total number of extracted instances. Recall is the ratio of the number of correctly extracted instances to the total number of correct instances in the dataset. The F-measure is the harmonic mean of precision and recall [18].

$$P = \frac{\text{number of correctly identified relation instances}}{\text{total number of identified relation instances}},$$
$$R = \frac{\text{number of correctly identified relation instances}}{\text{total number of correct relation instances}},$$
$$F = \frac{2 \times P \times R}{P + R}.$$

4.3 Numerical Results and Discussion

The effect of applying the difficulty classifier to the input sentences was evaluated and the behavior of the shallow and the deep extractors was explored. In our experiments, we used the datasets previously described in Section 4.1. We trained three different classifiers which read a sentence and decide if the sentence is easy or difficult for the extraction of relation.

To collect syntactic features, we need to perform POS tagging and chunking. Therefore, we use the OpenNLP package. We modeled extraction difficulty for sentences. Our modeling of sentence difficulty was binary: sentences are easy or difficult to extract for a system. Given a corpus, SDE-OIE should select sentences for the shallow/deep extractor so as to maximize the number of correctly extracted instances. In other words, it selects the extractor which produces a correct instance when the other extractor generates an incorrect result.

After testing, relation instances with score values equal to or higher than a specific threshold are considered to belong to the class 1. The instances with score values lower than this threshold are considered to belong to class 0. Different values of the classifiers scores were examined. It was observed that the threshold of 0.6 for both Logistic Regression and Decision Tree, and 0.7 for Naive Bayes yields the highest performance.

We ran different OIE systems on these datasets[3]. Table 3 shows the precision and recall of each system on two different datasets. NB, DT and LR subscripts are used for Naive Bayes, Decision Tree and Logistic Regression, respectively. There is an insignificant difference between the precision of SDE-OIE and its constituent systems. ReVerb and EXEMPLAR have relatively high precision due to designing good patterns for relation extraction; thus this can lead to a higher rate of precision in SDE-OIE. The best result for the precision of SDE-OIE was obtained by the Logistic Regression classifier. High precision of SDE-OIE is caused by its primary elements. This is also the case for recall. As a result, selecting the main components of the proposed approach has a direct effect on the overall precision and recall. In terms of precision, SONEX outperforms all other approaches

[3] We used the source codes of these OIE systems for implementation. The source code of SDE-OIE is available at `https://github.com/VahidehRt/SDE-OIE`.

since its pattern-based design is able to detect predicates triggered by a noun properly.

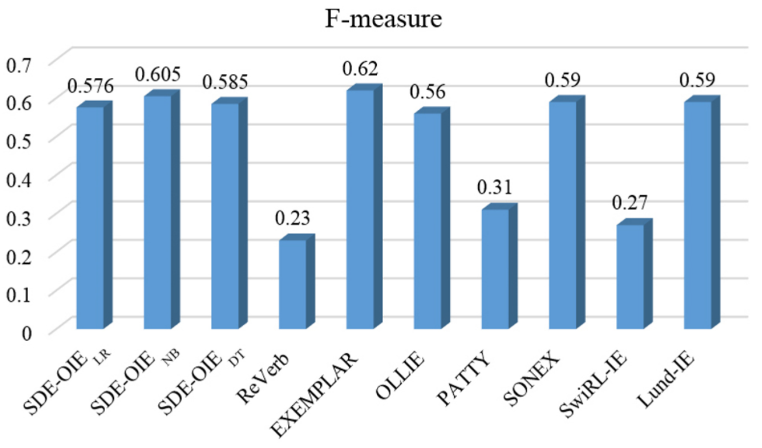
SDE-OIE's recall is higher than that of ReVerb. Moreover, it is lower than that for EXEMPLAR. EXEMPLAR has the highest precision among all the systems. It is superior mainly because it can recognize more correct instances, particularly those with verb + noun predicates [20]. The higher precision and the lower recall in comparison to EXEMPLAR reflect that our approach finds less relation instances than EXEMPLAR but most of the retrieved instances are accurate. Since both ReVerb and EXEMPLAR have no output for some sentences, the number of missing instances may be increased in comparison with EXEMPLAR individually. The best result for SDE-OIE's recall was achieved by the Naive Bayes classifier.

	Penn Treebank		New York Times	
Method	Precision	Recall	Precision	Recall
SDE-OIE$_{\mathrm{NB}}$	0.78	0.49	0.8	0.3
SDE-OIE$_{\mathrm{DT}}$	0.77	0.49	0.8	0.3
SDE-OIE$_{\mathrm{LR}}$	0.79	0.43	0.81	0.26
ReVerb	0.78	0.14	0.8	0.13
EXEMPLAR	0.79	0.51	0.82	0.31
SONEX	0.92	0.43	0.84	0.22
OLLIE	0.81	0.43	0.81	0.25
PATTY	0.46	0.24	0.82	0.21
SwiRL-IE	0.89	0.16	0.84	0.2
Lund-IE	0.86	0.35	0.83	0.24

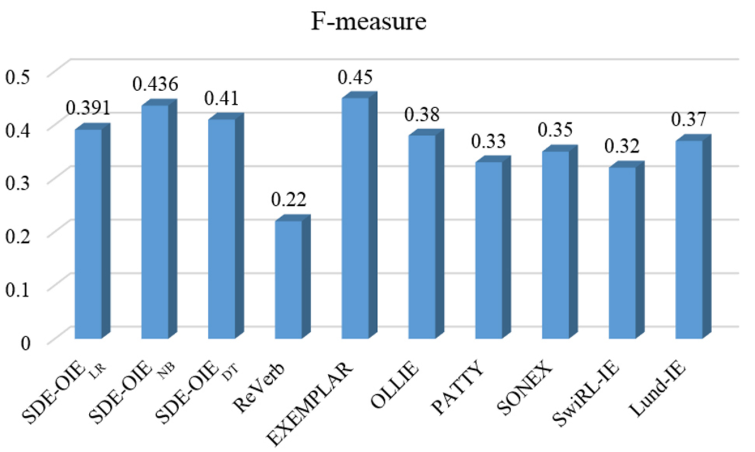
Table 2. Results for the task of extracting relations

Figure 2 shows the F-measure of each system. EXEMPLAR outperforms all methods. This is mainly because of its relatively higher recall in comparison with other methods. SDE-OIE$_{\mathrm{NB}}$ has the best F-measure among the other SDE-OIE methods. SDE-OIE$_{\mathrm{NB}}$ and EXEMPLAR are both at a very close level of F-measure. Based on the description given above, this can be interpreted in terms of precision and recall. ReVerb and EXEMPLAR have relatively high precision, therefore, SDE-OIE$_{\mathrm{NB}}$'s precision is also high. SDE-OIE$_{\mathrm{NB}}$ and EXEMPLAR have the same precision. SDE-OIE$_{\mathrm{NB}}$'s recall is significantly higher than ReVerb, but slightly lower than EXEMPLAR. Thus, SDE-OIE$_{\mathrm{NB}}$'s F-measure is slightly lower than that of EXEMPLAR, as it is defined as a harmonic mean of precision and recall.

SDE-OIE achieves an F-measure that is almost triple that of ReVerb. ReVerb has a lower recall than other approaches because of the intrinsic weakness of shallow tools in detecting relation instances. This leads to a significant drop in its F-measure. The experiment results demonstrate that proper incorporation of shallow and deep extractors decreases the number of incorrect extractions and increases the correct ones, resulting in higher performance. On the other hand, our hybrid method is able



a)



b)

Figure 2. The F-measure of our method in comparison with other methods drawn from a) Penn Treebank b) New York Times. SDE-OIE$_{\mathrm{NB}}$ and EXEMPLAR have almost the same F-measure. Their F-measure is better than that of the others.

to cover the limitations of the shallow OIE system and provides significant boost in its performance. The satisfactory level of F-measure indicates that our approach is at least as good as its deep constituent system.

The total computing time of each method was measured. We excluded the time for initializing or loading any libraries or models into memory. To ensure a fair comparison, we make sure each method runs in a single-threaded mode, thus utilizing a single computing core at all times. The results are reported in Table 4. The best results of our approach are highlighted in the table.

Method	Penn Treebank	New York Times
SDE-OIE$_{\mathrm{NB}}$	0.41	0.73
SDE-OIE$_{\mathrm{DT}}$	0.4	0.71
SDE-OIE$_{\mathrm{LR}}$	**0.38**	**0.68**
ReVerb	0.02	0.01
EXEMPLAR	0.62	1.19
SONEX	0.04	0.03
OLLIE	0.14	0.23
PATTY	0.66	1.27
SwiRL-IE	2.17	3.49
Lund-IE	5.21	11.20

Table 3. Computing time (per second) for each method

The processing times vary with different types of extractors. There is an explicit differentiation of almost one order of magnitude among approaches based on semantic parsing (SwiRL-IE and Lund-IE), dependency parsing (EXEMPLAR, OLLIE and PATTY) and shallow parsing (ReVerb and SONEX). ReVerb is the fastest method since it uses shallow patterns and does not rely on any deep tool.

As the results show, deep extractors usually have a high computational cost. There is always a trade-off between performance and speed when selecting a deep extractor. Deep extractors usually have high computational cost. In general, the deeper the extractor, the higher is the incurred computational cost. In the same period of time, shallow extractors process several times more sentences than dependency parsing extractors, which in turn process several times more sentences than semantic parsing extractors. Since our main purpose is to achieve a high-performance system both in time and performance, we have leveraged the best of shallow and deep OIE systems. Selecting shallow and deep components was made according to a fair and objective experimental comparison of 10 state-of-the-art approaches which is presented in [19] and [20]. Thus, SDE-OIE makes effective use of available time and achieves a reasonable balance of precision and recall. As experiment results show, the efficiency of the whole system has been affected by optimizing the processing time which has been reduced more than 33 % for all classifiers. The processing time for SDE-OIE$_{\mathrm{NB}}$, SDE-OIE$_{\mathrm{DT}}$ and SDE-OIE$_{\mathrm{LR}}$ was reduced by 33 %, 35 % and 38 %, respectively, in the Penn Treebank dataset and 38 %, 40 % and 42 %, respectively, in the New York Times corpus.

SDE-OIE$_{\mathrm{LR}}$ is the fastest model among the other SDE-OIE models. Generally, SDE-OIE has approximately the same F-measure as EXEMPLAR, but at a much lower processing time. This becomes important in large inputs such as Web-scale data. When the number of sentences processed by ReVerb is high, the total time reduces to the processing time of ReVerb. An interesting result is that despite achieving high accuracy, the methods based on semantic parsing (SwiRL-IE and Lund-IE) have lower F-measure than SDE-OIE and also need too much computational time.

4.4 Evaluation of Sentence Difficulty Estimator

The distribution of easy and difficult sentences is 39 % and 61 %, respectively (difficult being the majority class). Table 5 shows the distribution of the values in the confusion matrix for all classifiers (TN, FP, FN, TP). In general, there are two types of errors in our hybrid extractor. The first type is related to intrinsic weakness of the constituent systems and amending these errors depends on the improvement of the main extractors of the hybrid method. This kind of error is not caused by difficulty classifier; it can be generated even if the proper extractor is selected. The second type of error occurs with the incorrect selection of the extractor by the difficulty classifier. This type of error almost always occurs as a direct result of the first type of error. For example, in this case, the input sentence should be processed by the shallow extractor but it is processed by the deep extractor. Thus, the optimal result is not gained. Although this error may not affect the performance of the hybrid method, it is not beneficial for the extraction speed.

Table 5 shows the accuracy and error rate for each classifier on the two datasets. The accuracy for SDE-OIE$_{\mathrm{NB}}$, SDE-OIE$_{\mathrm{DT}}$ and SDE-OIE$_{\mathrm{LR}}$ is 72 %, 68 % and 73 %, respectively, in the Penn Treebank dataset and 72 %, 67 % and 74 %, respectively, in the New York Times corpus. The results show that SDE-OIE$_{\mathrm{LR}}$ is the most accurate classifier among the three classifiers. We observe that the dominant error in SDE-OIE$_{\mathrm{LR}}$ is classifying a difficult sentence as easy. In general, a sentence difficulty classifier with a high accuracy results in a reasonable trade-off between time and performance, because selecting the proper OIE system leads to a significant reduction on the computational time of the whole system.

		SDE-OIE$_{\mathrm{NB}}$		SDE-OIE$_{\mathrm{DT}}$		SDE-OIE$_{\mathrm{LR}}$	
		Difficult	Easy	Difficult	Easy	Difficult	Easy
Penn Treebank	Difficult	81.8 %	18.1 %	72.7 %	27.2 %	68.1 %	31.8 %
	Easy	37.7 %	62.2 %	35.7 %	64.2 %	21.4 %	78.5 %
New York Times	Difficult	82.7 %	17.3 %	71.6 %	28.3 %	69.3 %	30.6 %
	Easy	37.3 %	62.6 %	36.5 %	63.4 %	20.3%	79.6 %

Table 4. The confusion matrix for the performance of the sentence difficulty classifiers

	Penn Treebank		New York Times	
	Accuracy	Error rate	Accuracy	Error rate
SDE-OIE$_{\mathrm{DT}}$	68.5 %	27.3 %	67.5 %	32.4 %
SDE-OIE$_{\mathrm{NB}}$	72 %	28 %	72.6 %	27.3 %
SDE-OIE$_{\mathrm{LR}}$	73.3 %	26.6 %	74.5 %	25.4 %

Table 5. The accuracy and error rate of the difficulty classifiers

5 CONCLUSIONS

We presented a new method for the automatic estimation of sentence difficulty in ORE systems. In other words, this work explores the notion of relation extraction difficulty and the ways how the difficulty information can be used to enhance extraction quality in shallow extractors. We applied different classifiers with a set of efficient sentence-based features to incorporate strengths of a shallow OIE system with a deep one. Our sentence difficulty classifier detects difficult sentences for processing by the deep extractor. We detected the best trade-off between efficiency (computational cost) and effectiveness (F-measure). Experiment results demonstrate that the proposed approach achieves significantly better F-measure than its shallow extractor. SDE-OIE also has approximately the same level of F-measure as its deep constituent extractor, but at a much lower processing time. This shows that isolation of difficult sentences from the rest of the sentences creates flexibility for applying different types of system customizations.

The aim of OIE is to scale information extraction methods to the size and diversity of the Web domain. SDE-OIE passes an input sentence to a deep extractor only if it is needed. SDE-OIE is able to better allocate computational resources and avoid wasting them, and thus it is suitable in cases where the computing time is limited and high performance is desired.

We believe that an extended work on difficulty modeling should incorporate different sophistication levels of NLP tools; thus this method can be extended to a multi-class problem as well. A SRL-based approach can be applied as the deepest underlying extractor. The proposed features are very fast to compute, which is an important property from a practical implementation perspective. In addition to our proposed features, some features from the underlying systems can be incorporated into the difficulty classifier. Using semantic features can also bring deeper linguistic knowledge into our model, but at the cost of time.

REFERENCES

[1] Piskorski, J.—Yangarber, R.: Information Extraction: Past, Present and Future. In: Poibeau, T., Saggion, H., Piskorski, J., Yangarber, R. (Eds.): Multi-Source, Multilingual Information Extraction and Summarization, Springer, 2013, pp. 23–49, doi: 10.1007/978-3-642-28569-1_2.

[2] Min, B.—Shi, S.—Grishman, R.—Lin, C.-Y.: Towards Large-Scale Unsupervised Relation Extraction from the Web. International Journal on Semantic Web and Information Systems (IJSWIS), Vol. 8, 2012, No. 3, pp. 1–23, doi: 10.4018/jswis.2012070101.

[3] Banko, M.—Etzioni, O.: The Tradeoffs Between Open and Traditional Relation Extraction. Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, ACL, 2008, pp. 28–36.

[4] Mausam—Schmitz, M.—Bart, R.—Soderland, S.—Etzioni, O.: Open Language Learning for Information Extraction. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12), 2012, pp. 523–534.

[5] Del Corro, L.—Gemulla, R.: ClausIE: Clause-Based Open Information Extraction. Proceedings of the 22nd International Conference on World Wide Web (WWW '13), 2013, pp. 355–366, doi: 10.1145/2488388.2488420.

[6] Etzioni, O.—Banko, M.—Soderland, S.—Weld, D. S.: Open Information Extraction from the Web. Communications of the ACM, Vol. 51, 2008, No. 12, pp. 68–74, doi: 10.1145/1409360.1409378.

[7] Wu, F.—Weld, D. S.: Open Information Extraction Using Wikipedia. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10), 2010, pp. 118–127.

[8] Fader, A.—Soderland, S.—Etzioni, O.: Identifying Relations for Open Information Extraction. Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11), 2011, pp. 1535–1545.

[9] Etzioni, O.—Fader, A.—Christensen, J.—Soderland, S.—Mausam, M.: Open Information Extraction: The Second Generation. Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence – Volume One (IJCAI '11), 2011, pp. 3–10.

[10] Merhav, Y.—Mesquita, F.—Barbosa, D.—Yee, W. G.—Frieder, O.: Extracting Information Networks from the Blogosphere. ACM Transactions on the Web (TWEB), Vol. 6, 2012, Art. No. 11, doi: 10.1145/2344416.2344418.

[11] Akbik, A.—Bross, J.: Wanderlust: Extracting Semantic Relations from Natural Language Text Using Dependency Grammar Patterns. Proceedings of the 2009 Semantic Search Workshop at the 18th International World Wide Web Conference, 2009, pp. 6–15.

[12] Akbik, A.—Löser, A.: Kraken: N-ary Facts in Open Information Extraction. Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-Scale Knowledge Extraction (AKBC-WEKEX), 2012, pp. 52–56.

[13] Qiu, L.—Zhang, Y.: ZORE: A Syntax-Based System for Chinese Open Relation Extraction. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1870–1880, doi: 10.3115/v1/d14-1201.

[14] Gamallo, P.—Garcia, M.—Fernández-Lanza, S.: Dependency-Based Open Information Extraction. Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP, 2012, pp. 10–18.

[15] Christensen, J.—Mausam—Soderland, S.—Etzioni, O.: An Analysis of Open Information Extraction Based on Semantic Role Labeling. Proceedings of the Sixth International Conference on Knowledge Capture (K-CAP '11), 2011, pp. 113–120, doi: 10.1145/1999676.1999697.

[16] Kim, M. H.—Compton, P.: Improving Open Information Extraction for Informal Web Documents with Ripple-Down Rules. In: Richards, D., Kang, B. H. (Eds.): Knowledge Management and Acquisition for Intelligent Systems (PKAW 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7457, 2012, pp. 160–174, doi: 10.1007/978-3-642-32541-0_14.

[17] Tseng, Y.-H.—Lee, L.-H.—Lin, S.-Y.—Liao, B.-S.—Liu, M.-J.—Chen, H.-H.—Etzioni, O.—Fader, A.: Chinese Open Relation Extraction for Knowledge Acquisition. Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014), 2014, pp. 12–16.

[18] Reshadat, V.—Hoorali, M.—Faili, H.: A Hybrid Method for Open Information Extraction Based on Shallow and Deep Linguistic Analysis. Interdisciplinary Information Sciences, Vol. 22, 2016, pp. 87–100, doi: 10.4036/iis.2016.r.03.

[19] Mesquita, F.—Schmidek, J.—Barbosa, D.: Effectiveness and Efficiency of Open Relation Extraction. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013, pp. 447–457.

[20] de Sá Mesquita, F.: Extracting Information Networks from Text. Ph.D. Thesis, University of Alberta, 2015.

[21] Mohit, B.—Liberato, F.—Hwa, R.: Language Model Adaptation for Difficult to Translate Phrases. Proceedings of the 13th Annual Conference of the EAMT, 2009, pp. 160–167.

[22] Kawahara, D.—Uchimoto, K.: Learning Reliability of Parses for Domain Adaptation of Dependency Parsing. Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II (IJCNLP), 2008, pp. 709–714.

[23] Kauchak, D.—Leroy, G.—Hogue, A.: Measuring Text Difficulty Using Parse-Tree Frequency. Journal of the Association for Information Science and Technology, Vol. 68, 2017, No. 9, pp. 2088–2100, doi: 10.1002/asi.23855.

[24] Hale, J. T.: Heuristic Search in a Cognitive Model of Human Parsing. Proceedings of the 11th International Conference on Parsing Technologies (IWPT '09), 2009, pp. 230–233, doi: 10.3115/1697236.1697283.

[25] Nakashole, N.—Weikum, G.—Suchanek, F.: PATTY: A Taxonomy of Relational Patterns with Semantic Types. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12), 2012, pp. 1135–1145.

[26] Xu, Y.—Kim, M.-Y.—Quinn, K.—Goebel, R.—Barbosa, D.: Open Information Extraction with Tree Kernels. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL), 2013, pp. 868–877.

[27] Saha, S.—Pal, H.—Mausam: Bootstrapping for Numerical Open IE. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2017, pp. 317–323, doi: 10.18653/v1/p17-2050.

[28] Falke, T.—Stanovsky, G.—Gurevych, I.—Dagan, I.: Porting an Open Information Extraction System from English to German. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016, pp. 892–898, doi: 10.18653/v1/d16-1086.

[29] Otero, P. G.—López, I. G.: A Grammatical Formalism Based on Patterns of Part of Speech Tags. International Journal of Corpus Linguistics, Vol. 16, 2011, No. 1, pp. 45–71, doi: 10.1075/ijcl.16.1.03gam.

[30] Gamallo, P.—Garcia, M.: Multilingual Open Information Extraction. In: Pereira, F., Machado, P., Costa, E., Cardoso, A. (Eds.): Progress in Artificial Intelligence (EPIA 2015). Springer, Cham, Lecture Notes in Computer Science, Vol. 9273, 2015, pp. 711–722, doi: 10.1007/978-3-319-23485-4_72.

[31] Bast, H.—Haussmann, E.: Open Information Extraction via Contextual Sentence Decomposition. 2013 IEEE Seventh International Conference on Semantic Computing (ICSC), 2013, pp. 154–159, doi: 10.1109/icsc.2013.36.

[32] Punyakanok, V.—Roth, D.—Yih, W.-T.: The Importance of Syntactic Parsing and Inference in Semantic Role Labeling. Computational Linguistics, Vol. 34, 2008, No. 2, pp. 257–287, doi: 10.1162/coli.2008.34.2.257.

[33] Johansson, R.—Nugues, P.: The Effect of Syntactic Representation on Semantic Role Labeling. Proceedings of the 22nd International Conference on Computational Linguistics (COLING '08), Vol. 1, 2008, pp. 393–400, doi: 10.3115/1599081.1599131.

[34] Johansson, R.—Nugues, P.: Dependency-Based Semantic Role Labeling of PropBank. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '08), 2008, pp. 69–78, doi: 10.3115/1613715.1613726.

[35] Surdeanu, M.—Harabagiu, S.—Williams, J.—Aarseth, P.: Using Predicate-Argument Structures for Information Extraction. Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL '03), Vol. 1, 2003, pp. 8–15, doi: 10.3115/1075096.1075098.

[36] Banko, M.—Cafarella, M. J.—Soderland, S.—Broadhead, M.—Etzioni, O.: Open Information Extraction for the Web. Proceedings of the 20th International Joint Conference on Artifical Intelligence (IJCAI '07), 2007, pp. 2670–2676.

[37] Mesquita, F.: Clustering Techniques for Open Relation Extraction. Proceedings of the SIGMOD/PODS 2012 Ph.D. Symposium, 2012, pp. 27–32, doi: 10.1145/2213598.2213607.

[38] Castellã Xavier, C.—Strube de Lima, V. L.—Souza, M.: Open Information Extraction Based on Lexical-Syntactic Patterns. 2013 Brazilian Conference on Intelligent Systems (BRACIS), 2013, pp. 189–194, doi: 10.1109/bracis.2013.39.

[39] Cimiano, P.—Wenderoth, J.: Automatically Learning Qualia Structures from the Web. Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition (DeepLA '05), 2005, pp. 28–37, doi: 10.3115/1631850.1631854.

[40] Lin, H.—Wang, Y.—Zhang, P.—Wang, W.—Yue, Y.—Lin, Z.: A Rule Based Open Information Extraction Method Using Cascaded Finite-State Transducer. In: Bailey, J., Khan, L., Washio, T., Dobbie, G., Huang, J., Wang, R. (Eds.): Advances

in Knowledge Discovery and Data Mining (PAKDD 2016). Springer, Cham, Lecture Notes in Computer Science, Vol. 9652, 2016, pp. 325–337, doi: 10.1007/978-3-319-31750-2_26.

[41] Ebadat, A.-R.—Claveau, V.—Sébillot, P.: Using Shallow Linguistic Features for Relation Extraction in Bio-Medical Texts. Traitement Automatique des Langues Naturelles (TALN 2011), 2011, pp. 125–130.

[42] Jurafsky, D.—Martin, J. H.: Speech and Language Processing. Pearson London, 2014.

[43] Ng, A. Y.—Jordan, M. I.: On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. Advances in Neural Information Processing Systems 14 (NIPS 2001), 2002, pp. 841–848.

[44] Castellã Xavier, C.—Strube de Lima, V. L.—Souza, M.: Open Information Extraction Based on Lexical Semantics. Journal of the Brazilian Computer Society, Vol. 21, 2015, Art. No. 4, 14 pp., doi: 10.1186/s13173-015-0023-2.

[45] Léchelle, W.—Langlais, P.: An Informativeness Approach to Open IE Evaluation. In: Gelbukh, A. (Ed.): Computational Linguistics and Intelligent Text Processing (CICLing 2016). Springer, Cham, Lecture Notes in Computer Science, Vol. 9624, 2018, pp. 523–534, doi: 10.1007/978-3-319-75487-1_40.

[46] Xu, Y.: Relation Extraction and Its Application to Question Answering. Ph.D. Thesis, University of Alberta, 2017.

[47] Schmidek, J.—Barbosa, D.: Improving Open Relation Extraction via Sentence Re-Structuring. Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), 2014, pp. 3720–3723.

[48] Sandhaus, E.: The New York Times Annotated Corpus. Philadelphia, PA: Linguistic Data Consortium. `https://catalog.ldc.upenn.edu/LDC2008T19`, 2008.

Vahideh Reshadat received her B.Sc. and M.Sc. both in computer engineering in 2008 and 2012, respectively. She has been teaching as Lecturer in different universities of Iran since 2012. Her main interests are natural language processing, machine learning, deep neural networks and text mining.

Heshaam Faili received his B.Sc. and M.Sc. degrees in computer engineering and his Ph.D. in artificial intelligence from Sharif University of Technology in 1997, 1999 and 2006, respectively. He joined the Artificial Intelligence and Robotic Group of the School of Electrical and Computer Engineering, University of Tehran in 2008. He is now Associate Professor and his main research interests include AI, statistical approaches, text processing, and mining methods.