

Extracting Relations from Italian Wikipedia using Unsupervised Information Extraction

Pierluigi Cassotti¹, Lucia Siciliani¹, Pierpaolo Basile¹, Marco de Gemmis¹ and Pasquale Lops¹

¹Department of Computer Science, University of Bari Aldo Moro, Via E. Orabona, 4 - 70125 Bari, Italy

Abstract

In this paper, we describe WikiOIE, a framework for extracting relations from Wikipedia. The framework is based on UDPipe and the Universal Dependencies project for text processing. It easily allows customizing the information extraction (IE) approach to automatically extract triples (subject, predicate, object). In this work, we propose two unsupervised IE methods to extract triples from the Italian version of Wikipedia. The former is based only on PoS-tag patterns; the latter also uses syntactic dependencies. The extraction process is provided in JSON format and is used to build a simple web interface for searching and browsing the extracted triples. A preliminary evaluation conducted on a dataset sample shows promising results, although the approach is completely unsupervised.

Keywords

Open Information Extraction, Natural Language Processing, Information Retrieval, Wikipedia, Universal Dependencies

1. Introduction

Open Information Extraction has the aim of extracting relations within a huge amount of text available on the Web. Relations take the form of relational tuples denoted as $\{(arg1; rel; arg2)\}$. Given a relation, $arg1$ and $arg2$ can be nouns or phrases, while rel is a phrase denoting the semantic relation between them.

The importance of Open IE is further enhanced by its role in several NLP applications like Question Answering, Knowledge Graph Acquisition, Knowledge Graph Completion, and Text Summarization.

Given the nature of the task, approaches for Open IE are deeply intertwined with the language of the corpora that has to be analyzed. Due to the availability of English corpora, the majority of the works available at the state of the art are specific for that language. For what concerns the Italian language, there is the approach proposed by Guarasci et al. [1], which is founded

IIR 2021: The Italian Information Retrieval (IIR) Workshop, September 13–15, 2021, Bari, IT

✉ pierluigi.cassotti@uniba.it (P. Cassotti); lucia.siciliani@uniba.it (L. Siciliani); pierpaolo.basile@uniba.it (P. Basile); marco.degemmis@uniba.it (M. d. Gemmis); pasquale.lops@uniba.it (P. Lops)

🌐 <http://www.di.uniba.it/~swap/cassotti.html> (P. Cassotti); <http://www.di.uniba.it/~swap/siciliani.html> (L. Siciliani); <http://www.di.uniba.it/~basilepp/> (P. Basile); <http://www.di.uniba.it/~swap/degemmis.html> (M. d. Gemmis); <http://www.di.uniba.it/~swap/lops.html> (P. Lops)

🆔 0000-0001-7824-7167 (P. Cassotti); 0000-0002-1438-280X (L. Siciliani); 0000-0002-0545-1105 (P. Basile); 0000-0002-2007-9559 (M. d. Gemmis); 0000-0002-6866-9451 (P. Lops)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

on the use of verbal behavior patterns created based upon Lexicon-Grammar features. The authors have also produced a dataset built on the itWaC corpus, which is not currently available. WikiOIE proposes a framework in which Open IE approaches can be easily developed with the focus on the Italian¹ language with the aim to encourage researchers to develop approaches for under-represented languages.

The paper is structured as follows: 3 provides details about WikiOIE and its methodology, while Section 4 describes the dataset developed from the Italian version of Wikipedia. A preliminary evaluation is reported in Section 5.

2. Related Work

At the beginning, the task of IE was focused on the extraction of relations pre-specified by the user from small corpora. Even though several works like DIPRE [2], Snowball [3], and [4] tried to reduce the amount of annotated data necessary in the training phase, the systems still required a substantial human intervention to obtain relevant results, and changing domain was difficult.

The shift to Open IE system was proposed by [5], along with their system called TextRunner. In this case, the goal is to apply IE algorithms to extract relations from a large collection of documents that can cover any topic. For this reason, limiting the extraction process to a set of relations defined a priori is infeasible.

TextRunner applies an approach that is composed of three main modules. The first one is a *learner* that exploits a parser to label the training data as trustworthy or not and then uses the extracted information to train a Naive Bayes classifier. Next, the *extractor* uses POS-tag features to obtain a set of candidate tuples from the corpus, which are then sent to the classifier, and only those labeled as trustworthy are kept. Finally, a module denominated *assessor* assigns a probability score to the tuples extracted at the previous step based on the number of occurrences of each tuple in the corpus.

The learning-based approach used in TextRunner has also been applied by several other systems like WOE [6], OLLIE [7], and ReNoun [8]. In particular, WOE exploits Wikipedia-based bootstrapping: the system extracts the sentences matching the attribute-value pairs available within the info-boxes of Wikipedia articles. This data is then used to build two versions of the system: the first one based on PoS-tags, regular expressions, and other shallow features of the sentence, the latter based on features of dependency-parse trees, thus obtaining better results than the other one but with a lack of performance in terms of speed.

OLLIE is also capable of identifying not only verb-based relations but also those that involve the use of nouns and adjectives. The approach first builds a training set by selecting the sentences that contain one of the 110,000 seed tuples taken from ReVerb. Then, using a dependency parser, the training set is used to learn a set of extraction pattern templates. OLLIE also tries to take into account the context of the sentence in the extraction process, extending the tuples with additional fields if necessary and assigning a confidence score to each tuple.

Finally, ReNoun, as the name suggests, is a system that mainly focuses on extracting noun-based relations. It exploits a set of manually created lexical patterns used as seed facts to learn

¹The framework can be extended to other languages since it is based on universal dependencies.

a more extensive set of patterns based on the dependency parse feature of the sentences in the corpus and distant supervision. Next, these patterns are used to generate the final set of extractions, each one having an associated score that represents its correctness.

Another common approach to tackle the Open IE task is represented by rule-based systems like ReVerb. ReVerb takes in input a sentence PoS-tagged and NP-chunked using OpenNLP. The sentence is analyzed to check if it is compliant concerning different syntactic and lexical constraints and extract its relations. Given each relation phrase, the system then derives its arguments by checking the nearest noun phrases at the left and the right of the relation.

Using a set of rules allows this kind of systems to obtain relations of better quality compared to other approaches, but their variety is limited when compared with other approaches.

Finally, there are approaches based on exploiting the presence of specific clauses within the text. This is the case of ClausIE [9] which makes use of the information derived by the dependency parse of the input sentence to determine the set of clauses that compose it. ClausIE then generates a proposition for each selected clause determining which part represents the subject, the relation, and the arguments.

Another system that uses this kind of approach is Stanford Open IE [10], where the dependency tree of the sentence is traversed to identify the subtrees that represent independent clauses. These clauses are then reduced to shorter entailed sentences through the use of natural logic inference which can be easily transformed into triples.

3. Methodology

In this section, we describe our information extraction system called WikiOIE². The main WikiOIE features are:

- completely unsupervised approach for extracting triples from Wikipedia dumps
- text processing based on Universal Dependencies³ to make easy the integration of other languages in the future
- customizable pipeline for supporting both new processing and extraction algorithms
- integration of an indexing and search engine for browsing the extracted facts.

The overall architecture of WikiOIE is presented in Figure 1.

The input of the pipeline is the textual format of the Wikipedia dump obtained by the WikiExtractor tool [11], which is the only external component of our framework. It is possible to use other tools for extracting text from a Wikipedia dump, but our import module can read the simple XML format provided by WikiExtractor⁴.

The text is extracted from the dump and processed by the UDPipe tool [12]. We use version 1 of UDPipe with version 2.5 of the ISDT-Italian model. UDPipe can be trained by using Universal Dependencies data for over 100 languages. UDPipe allows to potentially use our system on different Wikipedia dumps in several languages. WikiOIE directly calls the REST API provided by UDPipe. In this way, it is easy to change the endpoint and the model/language.

²The code is available on GitHub: <https://github.com/pippokill/WikiOIE>.

³<https://universaldependencies.org/>

⁴<https://github.com/attardi/wikiextractor/wiki/File-Format>

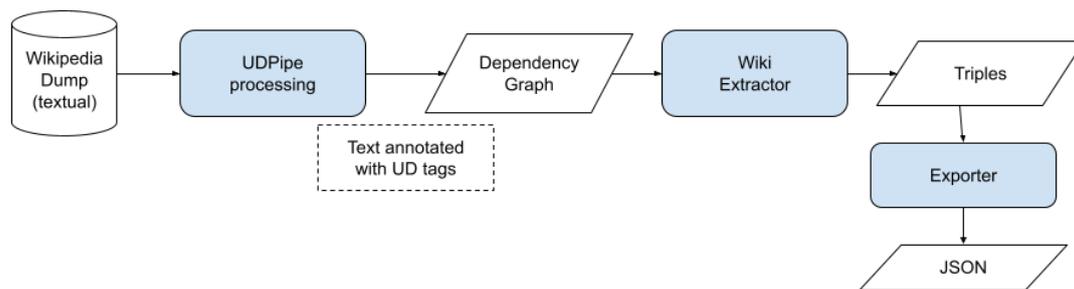


Figure 1: The WikiOIE architecture of the information extraction process.

Another advantage of using Universal Dependencies is the common tag-set for all the languages. PoS-tag⁵ and syntactic dependencies⁶ are annotated with shared set of tags.

The Wikipedia dump is read line-by-line. Each line contains a fragment (passage) of text that is processed by UDPipe. The output of this process is a set of sentences, and each sentence is annotated with syntactic dependencies. The sentence is transformed into a dependency graph that is the input of the Wiki Extractor module. This module extract facts from the sentence in the form of triples (*subject, predicate, object*) and assigns a score that take into account the number of nouns involved in the subject/object.

Triples are stored in a JSON format that reports not only the extracted triple but also the original text of the sentence, the output provided by UDPipe, the reference to the Wikipedia page, the title of the Wikipedia page, and the positions of the subject, predicate and object within the sentence.

All this information is useful for indexing and searching and can be exploited by further post-processing steps during the indexing process. The details about both the JSON format and indexing are provided in Section 4.

3.1. The Information Extraction process

The information extraction process is sketched in Figure 2. The input text is annotated by UDPipe that provides the CoNLL-U format⁷ of each sentence occurring in the text. Each token into the sentence is denoted by an index (first column) corresponding to the token position into the sentence (starting from 1). As depicted in Figure 2, the other columns are the features extracted by UDPipe, such as the token, the lemma, the universal PoS-tag, the head of the current word, and the universal dependency relation to the HEAD (root if the head is equal to 0). Figure 2 also reports the dependency graph of the sentence that is used by the Wiki Extractor module for extracting triples.

The extraction process that we implement in this first version of the pipeline is an entirely

⁵<https://universaldependencies.org/u/pos/>

⁶<https://universaldependencies.org/u/dep/>

⁷<https://universaldependencies.org/format.html>

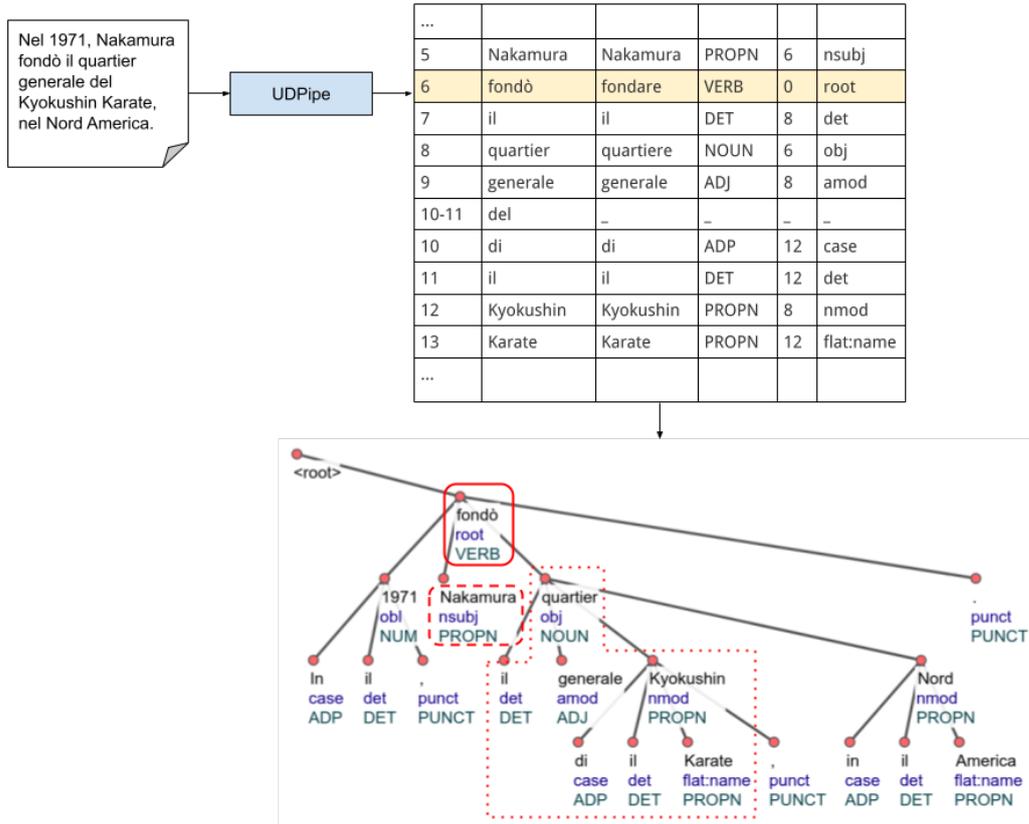


Figure 2: An example of UDpipe processing.

unsupervised approach based on both PoS-tag and graph structure. The first step consists of identifying sequences of PoS-tags that match verbs as reported in Table 1.

PoS-tag Pattern	Example
AUX VERB ADP	... è nato nel ...
AUX VERB	... è nato ...
AUX=(essere, to be)	... è ...
VERB ADP	... nacque nel ...
VERB	... acquisì ...

Table 1
Patterns of valid predicates.

In Table 1, the first column reports the PoS-tag patterns, while the second one reports an example of pattern usage. The sentence showed in Figure 2 matches the last pattern (VERB, *fondò*).

When the information extraction algorithm finds a valid predicate pattern, it checks for a candidate subject and object for the predicate. A valid subject/object candidate must match the

following constraints:

1. a sequence of tokens belonging to the following PoS-tags: noun, adjective, number, determiner, adposition, proper noun;
2. the sequence can contain only one determiner and/or one adposition.

The candidate subject must precede the verb, while the candidate object must follow the predicate pattern. For the sentence in Figure 2 the candidate subject is *Nakamura*, while the candidate object is *il quartier generale di il Kyokushin Karate*⁸.

After the identification of the candidate subject and object, we apply two strategies:

simple: the algorithm accepts the subject, predicate, and object as a valid triple and assigns a score. Two distinct scores are calculated for the subject and object. For each noun occurring in the subject/object the score is incremented by 1, while for each proper noun by 2. Finally, the score is multiplied by $1/l$ where l is the number of tokens occurring in the subject/object. The idea is that a subject/object containing only nouns or proper nouns is more relevant. The final triple score is the sum of the scores assigned to both subject and object.

simpledep: in this case, the triple is accepted only if both the subject and the object have a syntactic relation with the verb. In particular, one of the tokens belonging to the subject/object must have a dependent relation with a token of the verb pattern. Taking into account the sentence in Figure 2 the subject token “Nakamura” is connected with the verb “fondò”, while the object token “quartier” is linked with the same verb, which makes the triple valid. The triple score is computed with the same strategy adopted for the *simple* approach. The idea behind this approach is to validate the triple by verifying the existence of a syntactic relation between the verb and the subject/object.

4. Dataset

We provide three datasets⁹ in JSON format as in Listing 1. The first dataset is the output of the UDpipe process. This dataset contains all the text passages extracted from WikiExtractor and processed by UDpipe. In particular, the JSON reports: the *id* of the Wikipedia page, the *title* of the Wikipedia article, the *text* and the UDpipe output in *conll* format. In this dataset, the JSON object array is empty since triples are not yet extracted. This dataset contains 23,602,140 text passages.

```
{"id": "3859480",  
"title": "Anthony Joshua",  
"text": "Lo spettacolo e drammatico..."  
"conll": "1 Lo lo DET..."  
"triples": [  
{"subject": {"span": "drammatico incontro", "start": 3, "end": 5, "score": 1.0},
```

⁸It is important to note that UDpipe splits the articulated preposition “del” in “di:ADP” and “il:DET”.

⁹A sample of the dataset is available here: <https://github.com/pippokill/WikiOIE/tree/master/sample>. We will upload the full dataset on Zenodo.

```

"predicate": {"span": "vide", "start": 5, "end": 6, "score": 1.0},
"object": {"span": "Joshua", "start": 6, "end": 7, "score": 3.0}, "score": 4.0},
{"subject": {"span": "Joshua", "start": 6, "end": 7, "score": 3.0},
"predicate": {"span": "subire", "start": 7, "end": 8, "score": 1.0},
"object": {"span": "il primo knockdown in carriera", "start": 8, "end": 13, "
score": 0.6}, "score": 3.6}
]]

```

Listing 1: JSON structure of the dataset.

This dataset can be used by the WikiOIE tool for extracting triples by using several strategies. In this version, we include the approaches described in Section 3: **simple** and **simpledep**. The system can be easily extended by implementing other strategies.

The other two datasets that we provide contain triples extracted respectively by **simple** and **simpledep**. In these datasets the *conll* value is empty, while the field *triples* stores the JSON array of extracted triples.

Table 2 reports information about the number of triples extracted and the number of distinct subjects, objects, and predicates for each dataset.

Dataset	#triples	#dist. subj	#dist pred	#dist obj
simple	5,739,830	2,184,493	387,706	2,981,188
simpledep	3,562,803	1,298,481	269,551	2,030,742

Table 2

Dataset statistics.

We observe that the number of triples extracted by *simpledep* is less than the ones extracted by *simple* since *simpledep* removes triples in which the subject or the object is not linked to the verb.

Moreover, WikiOIE provides two valuable tools for indexing the triples dataset using Apache Lucene¹⁰, and for exporting the only triples in TSV format. During the indexing, it is possible to apply a further processing step for post-processing or filtering triples. For example, during the indexing or the export, it is possible to remove triples in which the predicate occurs less than a specified value. This last feature is important for removing not-relevant triples.

Table 3 shows the ten most frequent predicates extracted by *simpledep*.

5. Evaluation

For the evaluation, we randomly select a subset of 200 triples from both the *simple* and *simpledep* datasets. The triples are selected for the ones in which the predicate occurs at least 20 times. Then, each triple is annotated by two experts as relevant (valid) or not-relevant. We used the Cohen’s Kappa coefficient (K) to measure the pairwise agreement between the two experts. K is a more robust measure than simple percent agreement calculation, since it takes into

¹⁰<https://lucene.apache.org/>

Predicate	Occurrences
ha	55,443
divenne	24,361
ebbe	23,363
raccoglie	19,650
fu prodotto da	18,520
fa	16,664
aveva	16,078
presenta	14,445
comprende	13,629
ha battuto in	11,599

Table 3

Most frequent predicates extracted by *simpledep*.

account the agreement occurring by chance. Higher values of K corresponds to higher inter-rater reliability. Open IE task lacks a formal definition of triple relevance. A first attempt to define triple relevance is reported in [13]. The core aspects of this definition are assertiveness, minimalism, and completeness. Namely, they state that the extracted triple should be asserted by the original sentence, the triples are compact and self-contained and that all the relations asserted in the input text are extracted. In our evaluation, we decide to give less weight to minimalism and focus more on the extraction completeness. After the annotation, we compute the ratio of relevant triples for each dataset and expert. Cohen’s kappa coefficient for each dataset is provided in the last column of Table 4.

Dataset	#valid (exp 1)	#ratio (exp 1)	#valid (exp 2)	#ratio (exp 2)	Kappa C.
simple	96	0.48	110	0.55	0.42
simpledep	115	0.64	161	0.81	0.24

Table 4

Results of the annotation process.

Results in Table 4 show that the *simpledep* approach provides better results since it can filter out triples in which the subject or the object have not a syntactic link with the verb. However, Cohen’s kappa coefficient shows a low agreement between the experts, and in the case of the *simpledep* method, the agreement is inferior. A possible solution would be to refine the annotation by introducing a scale of relevance (e.g., from 1 to 5). Moreover, the annotation results may be affected by the small set of triples considered. We plan to extend the annotation to a wider range of extracted triples and employ more annotators.

Further, we perform a detailed analysis of triples considered non-relevant by both annotators, classifying errors in four types:

1. **Generic subject:** the triple subject cannot be linked to a real-world entity;
2. **Missing subject:** the extracted subject is not a real-world entity;
3. **Incomplete object:** the extracted object features incomplete information;
4. **Generic relation:** the whole triple cannot be self-explained.

In Table 5, we report examples for each type of error. A frequent error is the extraction of generic subjects, incomplete objects, or whole generic triples. This type of triples cannot be self-contained since they require contextual information (e.g., the feeding Wikipedia page, the feeding sentence). For instance, often generic triples involve demonstrative adjectives that require contextual information uncaught in the extracted triple.

Error type	Subject	Predicate	Object
Generic subject	<i>Quattro di queste specie</i>	<i>beneficiano di</i>	<i>una protezione speciale</i>
Incomplete object	<i>Questa voce</i>	<i>incorpora</i>	<i>informazioni</i>
Generic relation	<i>il montaggio</i>	<i>è ridotto a</i>	<i>il minimo</i>
Missing subject	<i>In quel momento</i>	<i>arriva</i>	<i>Pitch</i>
Missing subject	<i>L'anno seguente</i>	<i>passò a</i>	<i>lo Start</i>

Table 5
Instances of error types.

Another common error is missing subjects. Analyzing this type of error, we find that most of them occur when complements of time or place are recognized as the triple subject, e.g. *In quel momento - arriva - Pitch*. To overcome this issue, it is possible to recognize and filter complements of time and place when they appear in the subject.

6. WikiOIE search engine

To facilitate the search and browsing of extracted triples, we develop a web interface based on the index built by WikiOIE. In particular, using the Lucene search engine it is possible to search triples containing specific tokens occurring in selected triple's fields (e.g., subject, predicate, object, or a combination of them).

Figure 3 shows the query result about all triples containing the word 'Amiga' in the subject. The user interface reports, for each triple, the score provided by the search engine and the score assigned to the triple.

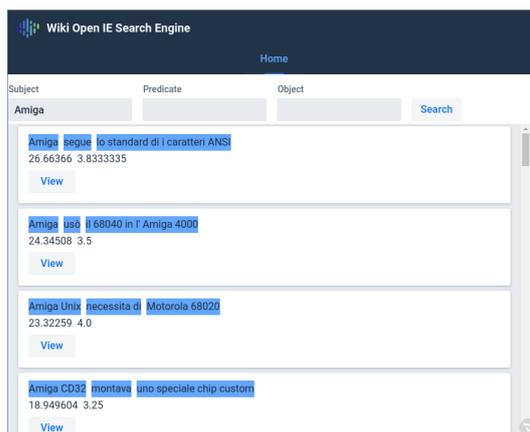


Figure 3: Search engine results.

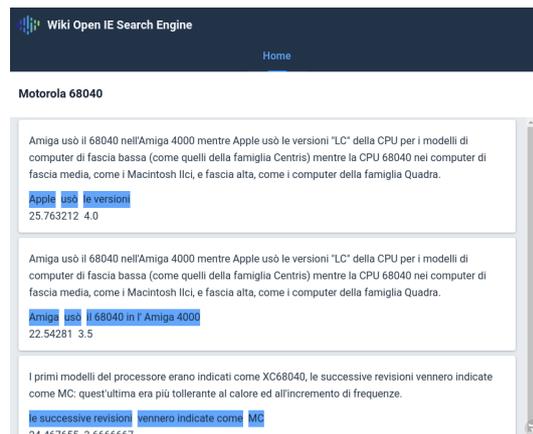


Figure 4: Triples details.

For each triple, it is possible to visualize details by clicking on the *View* button. The details page (Figure 4) shows the triple and the sentence from which it was extracted and the title of the Wikipedia page. Moreover, also all the other triples extracted from the same page are visualized.

7. Conclusions and Future Work

In this paper, we propose WikiOIE, a framework for open information extraction on Wikipedia dumps. The tool exploits UDpipe for processing and annotating the text and can be extended by adding several information extraction approaches. It is possible to add new extraction algorithms by extending a simple software interface. In this work, we propose two simple unsupervised methods for the Italian version of Wikipedia. Both the approaches are based on heuristics based on both PoS-tags and syntactic dependencies. We apply these methods to the Italian dump of Wikipedia by extracting millions of triples. A small subset of extracted triples is evaluated by two experts obtaining promising results. However, the Kappa coefficient shows a low agreement between annotators. To improve the annotators' agreement and get a more reliable overview of the system performance, we plan to extend the evaluation to a larger scale study.

Moreover, as future work, we plan to improve the extraction accuracy by implementing supervised approaches. In particular, the idea is to collect users' feedback by the web interface for building a training set.

References

- [1] R. Guarasci, E. Damiano, A. Minutolo, M. Esposito, G. De Pietro, Lexicon-grammar based open information extraction from natural language sentences in Italian, *Expert Systems with Applications* 143 (2020) 112954.
- [2] S. Brin, Extracting patterns and relations from the world wide web, in: *International workshop on the world wide web and databases*, Springer, 1998, pp. 172–183.
- [3] E. Agichtein, L. Gravano, Snowball: Extracting relations from large plain-text collections, in: *Proceedings of the fifth ACM conference on Digital libraries*, 2000, pp. 85–94.
- [4] E. Riloff, R. Jones, et al., Learning dictionaries for information extraction by multi-level bootstrapping, in: *AAAI/IAAI*, 1999, pp. 474–479.
- [5] O. Etzioni, M. Banko, S. Soderland, D. S. Weld, Open information extraction from the web, *Communications of the ACM* 51 (2008) 68–74.
- [6] F. Wu, D. S. Weld, Open information extraction using wikipedia, in: *Proceedings of the 48th annual meeting of the association for computational linguistics*, 2010, pp. 118–127.
- [7] M. Schmitz, S. Soderland, R. Bart, O. Etzioni, et al., Open language learning for information extraction, in: *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 2012, pp. 523–534.
- [8] M. Yahya, S. Whang, R. Gupta, A. Halevy, Renoun: Fact extraction for nominal attributes, in: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 325–335.

- [9] L. Del Corro, R. Gemulla, Clausie: clause-based open information extraction, in: Proceedings of the 22nd international conference on World Wide Web, 2013, pp. 355–366.
- [10] G. Angeli, M. J. J. Premkumar, C. D. Manning, Leveraging linguistic structure for open domain information extraction, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2015, pp. 344–354.
- [11] G. Attardi, Wikiextractor, <https://github.com/attardi/wikiextractor>, 2015.
- [12] M. Straka, J. Straková, Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipes, in: Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 88–99. URL: <http://www.aclweb.org/anthology/K/K17/K17-3009.pdf>.
- [13] G. Stanovsky, I. Dagan, Creating a large benchmark for open information extraction, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 2016, pp. 2300–2305.