LOD Construction Through Supervised Web Relation Extraction and Crowd Validation

Goran Rumin[1] and Igor Mekterović[2]

[1]*Infobip, Zagreb, Croatia*
[2]*University of Zagreb Faculty of Electrical Engineering and Computing, Croatia*
E-mail: goran.rumin@outlook.com; igor.mekterovic@fer.hr

Received 26 January 2019;
Accepted 23 March 2019

Abstract

Free, unstructured text is the dominant format in which information is stored and published. To interpret such vast amount of data one must employ a programmatic approach. In this paper, we describe a novel approach – a pipeline in which interesting relations are extracted from web portals news texts, stored as RDF triplets, and finally validated by end user via browser extension. In the process, different machine learning algorithms were tested on relation extraction, enhanced with our own set of features and thoroughly evaluated, with excellent precision and recall results compared to models used for semantic knowledge expansion. Building on those results, we implement and describe the component to resolve discovered entities to existing semantic entities from three major online repositories. Finally, we implement and describe the validation process in which RDF triplets are presented to the web portal reader for validation via Chrome extension.

Journal of Web Engineering, Vol. 18_1-3, 229–256.
doi: 10.13052/jwe1540-9589.18137

Keywords: Relation Extraction, Machine Learning, RDF, Linked Open Data, Crowd validation, Semantic Web, Web Application.

1 Introduction

Regardless the recent advent of LOD repositories, semantic data lags behind unstructured data in terms of volume. For instance, if we consider only publicly available web pages, Web Data Commons statistics from November 2017 show that less than half HTML pages (38.9% of 3.2 billion pages) [3] uses structured data. Producing high-quality semantic repositories is expensive. It is resource consuming, requires expert knowledge and, regardless of the progress in the AI/NLP domain, still requires human validation. On the other hand, development and standardization of new semantic web formats and technologies have contributed to the expanse of publicly available semantic web repositories. However, research shows that they widely vary in data quality ranging from extensively curated datasets to crowdsourced and extracted data of relatively low quality [20]. In this paper we pursue a hybrid approach to the problem of semantic web repository construction from free text where we employ state of the art NLP and machine learning techniques to acquire RDF triplets with minimally engaging human validation. Figure 1 shows the entire data processing pipeline. News web portal's texts are extracted from five popular English language news portals and submitted for NLP processing. Following that, the system uses machine learning techniques to recognize and extract a set of relations between named entities from text.

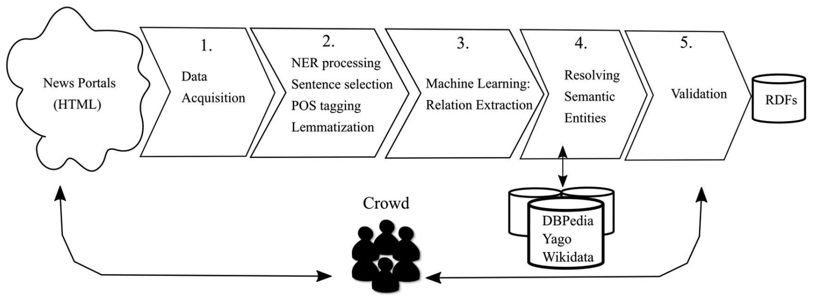

News Portals (HTML)
1. Data Acquisition
2. NER processing Sentence selection POS tagging Lemmatization
3. Machine Learning: Relation Extraction
4. Resolving Semantic Entities
5. Validation
RDFs
Crowd
DBPedia Yago Wikidata

Figure 1 Data processing pipeline.

In general, relation could be defined between any two nominals in a sentence. For our use case, and in similar papers which target semantic web, relation is defined only between two named entities (person, location and organization), since they are usually nodes in semantic graphs, and extraction of this type of relation allows graph expansion.

Although there are quite few models for relation extraction within our scope, many of them use distant supervision on various training algorithms and with it, target larger set of relations (usually 50–100). Their research shows that, although they manage to extract target relation number with good precision, recall of extraction remains low [12, 22]. This was our motivation, since we wanted to extract relatively large set of most used relations, while maintaining good precision and recall. With that two opposite requirements in mind, we chose supervised approach, since it gives us full control over the learning process.

Our NLP model outperforms the related models found in literature in terms of combined precision and recall (reflected in F1 score). Although we extract less relations when compared to the state of the art, we believe that larger extraction count, seen through higher recall, of smaller number of relation types could bring more benefit to the semantic web. To achieve that we have evaluated multiple learning algorithms and devised our own set of features that closely match natural usage of relations. After that, rule based semantic processing is used to resolve subject and object from extracted relations to existing semantic entities in three large and popular semantic repositories (DBPedia [1], Wikidata [19] and YAGO [17]). To this end, a component was constructed that combines multiple text similarity rules and elects the best semantic entity.

Finally, resolved triples are shown in form of a simple binary answer question through Chrome extension to a regular user browsing those portals with a goal of truthfulness assessment. At this stage benefits of this approach are visible – user does not have to be familiar with semantic web theory nor be acquainted with various ontologies to validate the proposed triple.

Relations that were marked true by five people are added to knowledge graph and exported in RDF/XML and visual format. Obviously, at the very end of the pipeline, the system relies on human help – the greater the number of users, the better data quality and throughput. In real-world application, one must motivate users to help. E.g. apart from users that might be intrinsically motivated to help this technology get wider accepted and used, a web portal could offer a free subscription or free access to otherwise paywalled content to users that participate in the validation process. Note that this system could also be used "internally" where the crowd would be the portal authors.

For instance, a journalist could write an article, and when done, give simple true/false feedback for his or her article and thus help create semantic content without any knowledge in the semantic web area. The novelty of our works consists of the semi-automated, crowd validated approach to the construction of the high-quality semantic repository and of the proposed NLP model with large number of relations. In the following sections, data processing pipeline is described in detail: Section 2 comments related work, Section 3 describes data acquisition and NLP pre-processing, Section 4 describes evaluated machine learning algorithms, model construction and evaluation of that model; Section 5 describes resolving of semantic entities, Section 6 gives a short overview of system architecture, Chrome extension and corresponding web service and other user related components; and Section 7 provides final remarks and discusses potential future work. A running example is provided throughout the paper.

2 Related Work

A recent approach to machine reading is described in [6]. Authors use precomputed bag-of-keywords to extract named entities by doing exact word matching and try to detect relations by comparing words from sentence to ontology predicates. Although this approach could in theory recognize many relations, it is limited by the need for relation to be expressed with ontology predicate name in text. Additionally, it is expensive because it requires creation of bag-of-keywords and is not applicable to unknown entities. Although system described in our paper extracts fixed predefined number of relations, it represents a generalised and more robust approach since extraction is not built around keywords.

In [12], authors apply distant supervision by using Freebase repository (now part of Wikidata) and free text from Wikipedia to infer relation features. Pair of named entities in some relation is taken from Freebase, sentences which contain these entities are found on Wikipedia and features that represent relation are extracted from that sentences. These features are later used to train a multiclass logistic regression classifier. The benefit of this approach is that there is no need for manual training data annotation and they extract a large number of relations (around 100). Authors report precision of 67.6% and a large number of false negatives, which means a low recall. Their F1-score is not reported but could be calculated to a value of around 30%. Their results show that their model is good at classifying recognized relations but not so good at recognizing them. All in all, results are good considering number of relations, but in our paper, we decided to focus on a smaller set of relations with goal of having higher recall while maintaining good precision.

A notable approach is described in [9]. FRED is an online tool that orchestrates various state of the art tools and performs additional processing to convert text to RDF and internally connected OWL ontologies. FRED uses a tool called Boxer to parse free text into Discourse Representation Structures from which a RDF graph is built. FRED creates ontology from text and each part of text is mapped to some kind of semantic relation, but only some of these new predicates directly represent meaning from a real-world relation of entities and additional work is necessary to increase usability of this new knowledge. [21] reports FRED 's F1-score of 83% on relation extraction task. This is impressive, but our concern is with the usability of this data, since relations which human reviewers have marked as true are all defined within ontology of submitted text. We have compared our results on the running example with FREDs by using their demonstration website [16]. FRED generated around 800 relations (!) with many of them being relations for sentence structure and words with their meanings, not only named entities. Such number of relations makes it hard for humans to submit larger amounts of text and evaluate the results. All named entities are extracted regardless of their involvement in a relation with another named entity. It extracted two out of three of our relations within its own ontology, while "colleague" relation was missing. That possibly happened because relation is not indicated with the sentence structure, although it can be extracted from semantics of the sentence. Also, minor errors occurred (e.g. *"...President Obama went outside the chain of command"* got linked to *The Chain Buffy the Vampire Slayer* comic DBPedia entity). Never the less, FRED's results are impressive, but in our opinion a bit overwhelming and additional work is required to filter or resolve interesting relations. Our approach is more focused and less general purpose than FRED's. We chose to extract fixed number of relations with predefined meaning which is bound to known predicates from the start. It is a more manageable and robust approach though omitting knowledge outside of its scope.

In a recent distant supervision approach described in [22], authors work on improving three main issues with this approach: data sparsity, noise and lexical ambiguity. To that goal they perform relation extraction across sentence boundaries using unsupervised coreference resolution and use statistical methods to strategically select training data. They build their own corpus by taking seven Freebase classes and their $\sim$7 top properties (43 relations in total), extracting selected number of entities per class which have all chosen properties, and then retrieving texts from web which contain entity and some relation by using Google Search. Following that, they prepare their own set of features for each relation instance and train multiclass logistic regression classifier, similar to work in [12]. Authors do not report F1 score, but they report per-class values for precision, recall and top line for recall (the number of all relation tuples appearing in the corpus divided by the number of relation tuples in the knowledge base). From that, with adjustments for top line, average F1 score could be calculated to around 38%. They improved overall score when compared to [12], but again recall is relatively low.

Additionally, we reviewed models which extract relations between any two nominals with greatest performance. Only publicly available model is Stanford Relation Extraction available within Stanford CoreNLP package. Its development is described in [18]. Although this model is used as a tool and offers relatively simple training of new relations, the absence of possibility to modify feature set and to configure learning process and usage of logistic regression as a learning algorithm has made it unreliable for this paper's problem. This model uses logistic regression algorithm to extract specific relations in domain of sport texts and in best case achieves F1-score of 50.2%. Logistic regression tries to separate examples with plane that separates two classes by minimizing total error. Since model was trained on one domain, learning was easier because there was smaller range of values which can be in features and represent one class. That is why examples are more distant in feature space which implies a smaller level of non-linearity is necessary to separate them – logistic regression is able to do it without mapping into high dimension space. Despite that, classification of eight relations could not achieve higher F1-score. That is why logistic regression was removed from consideration in this paper.

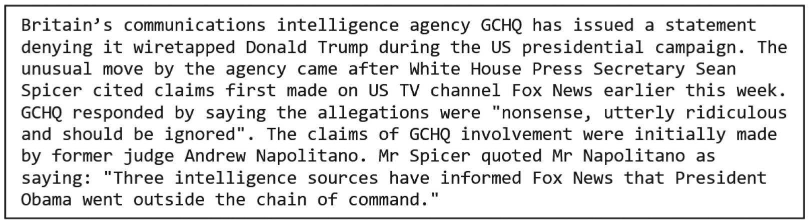
Second model is not available publicly and is described in [10]. This paper uses algorithm SVM (Support Vector Machines) which uses different mathematical procedure during training and hence can give noticeably different results than logistic regression. Classifier can be better or worse, which depends on domain, but generally SVM gives better results. SVM additionally offers usage of kernel functions i.e. comparison functions for examples. By using RBF function (Radial Basis Function) classifier can theoretically learn very large non-linearities, with certain limits. Paper describes classifier for generic news domain (ACE 2005 dataset) which is similar to this paper and uses RBF function for example comparison because of expected non-linearity of examples from texts which cover multiple domains and contain same relations. Authors achieve F1-score of 58.8% on news domain with extraction of five relations: role, at, social, near and part. This result is good for relation extraction area, which implies that features and algorithm hyperparameters were well selected, but it is still far from F1-scores of some other areas like NER where it is around 90%. Their exact feature set was not used in this paper because it has achieved listed F1-score on news data and five relations and would probably yield worse results with thirteen relations and smaller size of training data.

Neural networks provide yet another approach to this problem. Authors in [23] and [24] used it for relation extraction of relations between nominals, not strictly between named entities. By using convolutional neural network, [23] avoids feature creation and their evaluation, since network will automatically learn best features from sentences. Authors train their relation extraction system only with raw sentences marked with the positions of the two entities of interest. On relation extraction problem they achieve F1 score of 61% on ACE 2005 dataset, while on relation classification task (categorize the entity mention pairs that are known to represent some relations) they achieve 82% F1 score on SemEval 2010 task 8 set. Although they achieve excellent results on relation classification, their results imply that relation extraction, which could be divided into relation detection and classification, is still an area without definitive solution, unlike e.g. named entity recognition.

[24] leveraged recurrent neural network and its memory features for relations extraction and classification. They achieve F1 score of 48% on ACE 2004 dataset and 55% on ACE 2005 dataset for relation extraction problem. On relation classification problem they achieve macro F1 score of 84% on SemEval 2010 task 8 set. From these results similar conclusion can be drawn as for the previous model.

3 Data Acquisition and NLP Preprocessing

Data is periodically extracted from five English language web portals chosen by the type of news they publish and popularity: CNN, BBC, Huffington Post, New York Times and The Guardian. RSS (Rich Site Summary or Really Simple Syndication) standard is used for fetching new texts. Using URLs from the feed, the whole articles are fetched and parsed. All expressions that are not sentences of a news text like e.g. titles or subtitles of page, are removed and text is forwarded to the next phase in data pipeline – NLP processing. NLP processing is performed with NLTK [2] and Stanford NER tools [7] in the following order: named entity resolution, sentence selection, part-of-speech tagging and lemmatization. Figure 2 shows a text paragraph before NLP preprocessing.



```
Britain’s communications intelligence agency GCHQ has issued a statement
denying it wiretapped Donald Trump during the US presidential campaign. The
unusual move by the agency came after White House Press Secretary Sean
Spicer cited claims first made on US TV channel Fox News earlier this week.
GCHQ responded by saying the allegations were "nonsense, utterly ridiculous
and should be ignored". The claims of GCHQ involvement were initially made
by former judge Andrew Napolitano. Mr Spicer quoted Mr Napolitano as
saying: "Three intelligence sources have informed Fox News that President
Obama went outside the chain of command."
```

Figure 2 Example text paragraph before NLP processing.

Table 1 Processed text from Figure 2

Word	POS Tag	Lemma	NER tag
Britain	NNP	Britain	LOCATION
’s	POS	’s	O
communications	NNS	communication	O
intelligence	NN	intelligence	O
agency	NN	agency	O
GCHQ	NNP	GCHQ	ORGANIZATION
has	VBZ	have	O
issued	VBN	issue	O
a	DT	a	O
statement	NN	statement	O
denying	VBG	deny	O
it	PRP	it	O
wiretapped	VBD	wiretapped	O
Donald	NNP	Donald	PERSON
Trump	NNP	Trump	PERSON
during	IN	during	O
the	DT	the	O
US	NNP	US	LOCATION
presidential	JJ	presidential	O
campaign	NN	campaign	O
–	–	–	O

Table 1 shows the result of NLP processing of the first sentence in the Figure 2. Three types of named entities were used: Person, Location and Organization. NLTK uses Penn Treebank tagset [15], e.g. NNP stands for “Proper noun, singular”, POS is “Possessive ending”, etc.

Processed sentences are now fed to the main part of the pipeline – machine learning model for relation extraction.

4 Relation Extraction

Relation extraction is a challenging problem and one considered not to be solved yet. Currently there is no procedure for building models for extracting relations with great precision from free or domain-specific text. In this paper, we employ a different approach to relation extraction from the ones previously attempted (see Related Work).

Annotation and relation recognition is done at a sentence level which is a common approach. This means that relations are classified from set of all possible relations between all named entities in a sentence. One learning example represents one relation between two entities. One sentence can, and usually does, form multiple examples. Negative examples are all relations between entities that are not explicitly classified as one of chosen target classes. In the remainder of this section we explain how we tested and constructed the model for the relation extraction.

4.1 Model Training Dataset Features

The proposed initial relation set contains eighteen relations (Table 2). This set is hand-picked from a list of schema.org relations between used NER types (Organization, Person, Location) by looking how interesting a relation is and how common it is in text. Named entities comprising relation and opposite relation are listed. Direction of relation is the same as order of appearance of entities in sentence. Opposite relation is necessary when main relation consists of two entities of the same type and it is not commutative e.g. it is not the same if first person is child of the second, or the second is the child of the first.

We decided to build our own training dataset, which allows us to freely annotate our target relations that we selected for our application and not to depend on datasets annotated through distant supervision. Additionally, all datasets which can be used for relation extraction tasks are designed and annotated for relations extraction between two nominals. Dataset is freely available on project's Github page. To train the model two text corpora from NLTK [2] collection were chosen – Reuters and Brown corpus. Reuters dataset consists of 10788 news documents grouped in 90 potentially overlapping categories, with the

Table 2 Relations considered for extraction process

Relation	Entity Pair	Opposite Relation
colleague	person – person	
homeLocation	person – location	
knows	person – person	
nationality	person – location	
parent	person – person	children
relatedTo	person – person	
spouse	person – person	
brand	organization – organization	isBrandOf
employee	organization – person	
memberOf	organization – organization	member
location	organization – location	
parentOrganization	organization – organization	subOrganization
containsPlace	location – location	containedInPlace

total of 1.3 million words. Brown dataset consists of 15 disjunctive categories with the total of 1.15 million words. As initial learning set 5000 sentences was chosen from Reuters dataset, but with the condition that sentence needs to contain two or more named entities. Sentences were acquired with random sampling of documents from collection of all documents and NLP preprocessed as previously described.

Since Reuters dataset has very small amount of interpersonal relations, preprocessing procedure was conducted with larger part of Brown dataset, but with more strict condition – sentence needs to contain at least two persons. Total of 2373 sentences from both datasets were manually annotated, with around 70% of sentences from Reuters dataset and rest from Brown dataset. Figure 3 shows the number of named entities and Figure 4 shows the number of relations in the train dataset.

Certain relations have a small number of examples or are very similar to some other relation which prevents them from being learned with good precision and additionally disrupt learning of larger relations. With the goal of increasing future model accuracy some relations were merged with their most similar relation. All "nationality" relations are changed to "homeLocation", "memberOf" and "member" to "parentOrganization" and "subOrganization" respectively, "knows" to "colleague" and "isBrandOf" to "brand". Relations "nationality",

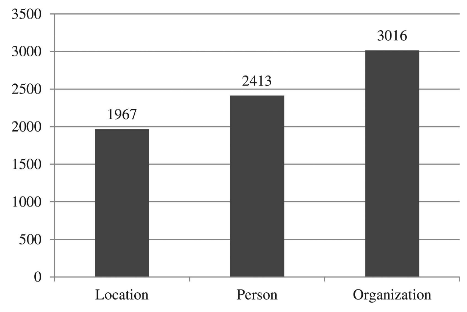


Figure 3 Number of named entities.

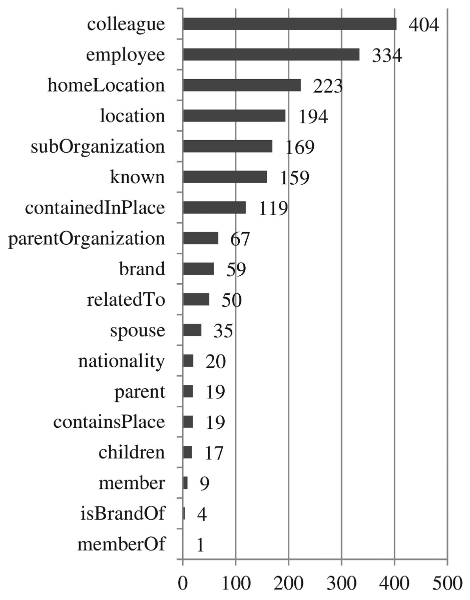


Figure 4 Number of relations in the train dataset.

“memberOf”, “member” and “isBrandOf” are too rare to be learned by the model, while “knows” is too similar to the “colleague” relation.

These replacements do not add semantic error to prediction except “isBrandOf” – “brand” replacement with opposite relation. This replacement is acceptable because “isBrandOf” relation is rare in the texts, so the error of mere replacement will not be noticed during model usage. Contrarily, certain error would still occur had relations not been merged since “isBrandOf” relation would not be learned.

4.2 Model Training

To train a model it is necessary to choose features which will represent a relation. Features selection was done using the “kitchen-sink” approach as follows: relation is initially represented with all features that make sense in given problem and that are considered good for whole or just part of example. After that model is trained and evaluated iteratively and with each iteration subset of features is removed or added with the goal of measuring their influence on prediction. Table 3 lists the final set of features used for relation extraction.

Several classification algorithms were evaluated: Random Forests [11], AdaBoost (Adaptive Boosting) [8], k-NN (k nearest neighbours) [4] and SVM (Support Vector Machines) [5]. All algorithms were evaluated with nested k-fold cross validation [14] with five outer folds and three inner folds. Algorithm implementations from Scikit-learn [13] library were used. To search the space of possible hyperparameter values, we have used random sampling with ten samples instead of exhausting search yielding great performance improvements in training and evaluation, both time and space wise – both processes were an order of magnitude faster.

Sampling is done uniformly and without repetitions, and the whole procedure is repeated five times (number of outer folds). Before training, all features and relation classes are converted to numeric values by using one-hot encoding. Additional preprocessing step is performed, which includes numeric value normalization. Values are scaled into some range with a chosen function, because otherwise numeric feature with greater value would have greater weight in model training. Normal distribution scaling was used, which analyses all feature values and performs scaling by using resolved normal distribution.

First algorithm to be tested was random forests algorithm. This algorithm trains an assigned number of decision trees, each on its own subset of input data, which may overlap. Algorithm allows for a set of parameters like number of trees, number of features which cause node splitting, number of node leaves etc. This algorithm has shown to be a good choice for this problem because it trains multiple internal classifiers which allow modelling relations with overlapping

Table 3 Features used for relation extraction model

Feature	Description
NE1 type	NER type of first entity
NE1 length	Number of words that are part of first entity
NE2 type	NER type of second entity
NE2 length	Number of words that are part of second entity
distance	Number of words between entities
word1 type	POS type of first word in first entity
word2 type	POS type of first word in second entity
word1 before type	POS type of word before first word in first entity
word1 after type	POS type of word after first entity
word2 before type	POS type of word before first word in second entity
word2 after type	POS type of word after second entity
common NE words	Do the entities have common words. Comparison is done with case-insensitive lemmas
location between	Is there an LOCATION entity between two entities of current relation
person between	Is there an PERSON entity between two entities of current relation
organization between	Is there an ORGANIZATION entity between two entities of current relation
family words len	Number of words between entities that are in “family” gazetteer
family word	First of words between entities that are in “family” gazetteer
spouse words len	Number of words between entities that are in “spouse” gazetteer
spouse word	First of words between entities that are in “spouse” gazetteer
organization words len	Number of words between entities that are in “organization” gazetteer
contains separators	Number of chosen punctuation marks between entities

features. One subset of trees learns to recognize relations with one range of features, while other subset learns relations with another range of features. In our evaluations, random forests algorithm achieved average F1-score of 52,5% which is good for the relation extraction domain.

Second algorithm to be tested is k-NN algorithm. k-NN is relatively simple algorithm which classifies example to the class of majority of k-nearest examples in feature space. k-NN achieved an average F1-score of 38.4%.

Third algorithm tested was SVM. Multiple kernel functions were used, and each was evaluated separately. F1-score for linear kernel is 45,7%. RBF kernel achieves F1-score of 48,6%. SVM achieves very good results with regards to the domain, but next algorithm has proven to be better.

Finally, we've considered the AdaBoost algorithm. AdaBoost is not a learning algorithm by itself, but an algorithm for performance improvement of existing algorithms. It trains given algorithm on a subset of training data, evaluates it and adds incorrectly classified test examples to the training set of next iteration with a choice probability greater than plain uniform sampling. Base algorithm used with AdaBoost was decision tree. Besides that, k-NN and SVM were tested. In case of k-NN there was no improvement regarding standalone usage while in case of SVM computing complexity was too great for practical usage or even evaluation. Table 4 contains list of tested hyperparameters with their descriptions and value ranges.

AdaBoost with decision tree achieves an average F1-score of 56,2%. Table 5 shows per-class F1-score values. Weaker results in "containsPlace" relation could be explained with the small number of examples, and "subOrganization", "parentOrganization" and "relatedTo" with nonexistence of appropriate features that could describe them or with necessity for a more complex model that could separate them from other relations.

Table 4 List of tested hyperparameters

Hyperparametar	Description	Value Range
n estimators	Number of decision trees	From 1 to 300 with step of 10
base estimatormax depth	Max depth of each tree	1, 5, 10, 20, 40, 70, 100, unlimited
base estimator-max features	Number of important features for node splitting	"sqrt", "log2", all features

Table 5 Per-class F1-score values of AdaBoost with decision tree algorithm

Relation (Class)	F1-Score (%)
containedInPlace	64,95
children	76,95
containsPlace	18
location	60,75
subOrganization	31,61
homeLocation	71,83
parentOrganization	19,06
spouse	59,55
colleague	65,07
relatedTo	27,12
parent	56,47
brand	75,31
employee	69,34

Table 6 Extracted relations from Figure 2

Subject	Predicate	Object
GCHQ	location	Britain
White House Press	employee	Sean Spicer
Sean Spicer	colleague	Andrew Napolitano

Hyperparameters chosen for training of the final model are following:

- n estimators = 180
- base estimator – max depth = unlimited
- base estimator – max features = all features

Since it has the largest score of all evaluated algorithms, AdaBoost algorithm was chosen and trained with listed hyperparameters on the whole dataset. Evaluation performed with nested k-fold cross validation shows greater F1 score than distant supervision in relation extraction for semantic web area which means that this model solves our problem described at the beginning of the paper. Additionally, to the best of our knowledge such approach was not used in any of the related works, and it represents a different learning approach to the relation extraction.

For example, Table 6 contains relations recognized with AdaBoost from the text in Figure 2.

5 Semantic Processing

After model finishes extracting relations from text, negative relations are ignored, while positive go through semantic processing. Goal of semantic processing is to match named entities from relations to the existing entities in semantic web repositories. Repositories chosen for search are DBPedia, Wikidata and YAGO. Search procedure for matching an entity is conducted by a series of queries against the semantic repositories as follows:

1. Fetch all entities for which English label matches regular expression which is created by concatenating all named entity words with ".+" in between them (any single character repeated one or more times).
2. Fetch abstracts from repositories for all entities from previous step.
3. Fetch number of connected objects for each entity from step 1. Goal of this metric is to represent popularity and importance of entity because more important entities will have more connected knowledge.
4. Choose one entity per repository based on fetched data metrics and similarity of text entity with names of semantic entities.

Semantic repositories are queried using SPARQL queries. Code snippet in Listing 1 attempts to retrieve entity and respective abstract for the NER type "Person" and entity (name) "Andrew Napolitano". This is a query template for the step 1 where bold parts are, of course, replaced with whatever entity and type is begin resolved. Lookup tables have been manually created for mapping our NER types to the respective types from online semantic repositories (e.g. the lookup table contains an entry matching our "Person" to dbo: "Person").

Listing 1. SPARQL query matching NER type "Person" and name "Andrew Napolitano"

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-
                      ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT DISTINCT ?entity ?label ?abstract
```

```
WHERE {
      ?entity rdf:type dbo: "Person" .
      ?entity rdfs:label ?label .
      FILTER (LANG(?label) = "en") .
      FILTER (REGEX(STR(?label), "Andrew.+Napolitano",
                      'i')) .
      OPTIONAL {
               ?entity dbo:abstract ?abstract .
               FILTER(LANG(?abstract) = "en")
     }
}
```

Decision about correct entity is done by calculating three accuracy metrics, combining them and choosing the largest score. First metric is similarity of sentence where the entity was found with abstracts of the fetched semantic entities. This metric is calculated as TF-IDF (Term Frequency-Inverse Document Frequency). The idea of TF-IDF metric is to give larger weight to a document which contains some query keywords but with condition that the same words are not present in most of the other documents in a set. This way important documents are retrieved based on uncommon words in query.

Abstracts are considered as documents and sentence as a query. Such practice is good because, in free text, entities are often described with few words when mentioned for the first time which could indicate correct semantic entity since there are many with the same name.

Same entity could be referenced throughout the text after being introduced for the first time. When an entity is introduced in a text, it will probably be listed in its full form e.g. name and surname of a person, and later will probably be listed in a short form. The goal is to connect the later entities with the correct longest one located in a sentence which describes it more closely. For this task case-insensitive 3-gram comparison is used. Comparison adds two spaces of padding to each end of text expression and splits it to 3-grams with one sign offset between subsequent 3-grams. Similarity between two expressions is calculated as a ratio of mutual 3-grams. Similarity threshold used in this paper is 25%. If none of the already reviewed entities matches currently observed, it is considered as new entity and is added to a set of reviewed entities. If currently observed entity matches one from a set of reviewed entities and if the reviewed entity is longer, reviewed entity is used, and if it is shorter, it is replaced with observed entity which is then used. Each NER type has separate set of reviewed entities.

Second metric is importance metric. It serves as a correction factor in case when TF-IDF weights are similar (e.g. two cities with same name), with the goal of selecting the more important entity as being more likely to be mentioned in the text. Code snippet in Listing 2 shows SPARQL importance query and grammar for generating filter. "##ENTITY##" placeholder is replaced with entity URI from the previous step.

Listing 2. SPARQL query for calculating the importance metric

```
SELECT ?entity (COUNT(?relatedValues) AS ?count)
  WHERE { ?entity ?p ?relatedValues .
                 FILTER (##EXPANSION##)
        }
  GROUP BY ?entity

##EXPANSION## := ##EXPANSION## || ?entity =
                 <##ENTITY##>
##EXPANSION## := ?entity = <##ENTITY##>
```

Since the initial query uses wide matching regular expression, goal of the third and last metric is to use the similarity between entity names to narrow the set of possible semantic entities. Names of fetched semantic entities are compared with text entity by using case insensitive 3-gram comparison. Final score is calculated as a product of mentioned three metrics and entity with the largest score is considered as a match for text entity. If no semantic entity is found, new one is created in “http://www.fer.hr” namespace. Table 7 shows the aforementioned metrics for the top three candidates from DBPedia repository for the named entity “Britain” from the Figure 1. The first row (entity) is chosen.

Search of the Wikidata repository differs in the first step of described procedure because Wikidata contains a large number of entities and

Table 7 Metrics for the “Britain” entity from Figure 1

Entity	Metric 1	Metric 2	Metric 3	Total
Great Britain	0.0170	0.9266	0.5	0.0079
Roman Britain	0.0135	1.0	0.5	0.0067
Kingdom_of_Great_ Britain	0.0125	0.6204	0.2963	0.0023

Table 8 Semantic resolver evaluation results

No Results	0%	33%	50%	66%	75%	100%
16	19	1	4	2	16	42

regular expression query would cause timeout errors. Additionally, resolving the semantic entity type is expensive operation since Wikidata marks every entity with the most specific type and without all its parent types. Since this paper uses broad NER types – Person, Organization and Location, this is a problem. To solve that, Wikidata is searched with label equality match and additional Wikidata entities are resolved from DBPedia by using objects of OWL “sameAs” predicate. This can possibly result in two Wikidata entities.

Evaluation of this process for resolving semantic entities was performed on 100 random entities from news articles and each resolved semantic entity was manually checked. Table 8 shows the evaluation results. Not each queried repository will have data for each entity, so some entities will have less than maximum of four semantic entities. On average, for each entity there is 3.07 semantic entities when considering only entities which have some semantic entities and 2.58 when considering all of them. Table column explanations are given below:

- No results: Semantic repositories have zero results for text entity. This is ignored when calculating accuracy.
- Percentage: Percent of semantic entities which is correct for its respective text entity.

Evaluations showed accuracy of 71%. An entity is considered correctly resolved if majority of resolved URIs reference the correct entity from text. All entities are saved with their respective relation for truthfulness assessment.

6 System Architecture

The proposed system is structured in a service-oriented fashion (Figure 5). Functionality of the NLP model and semantic resolver are exposed through web API (Application Programming Interface). Service tasks are:

- processing texts from clients,
- periodical processing of texts from chosen web portals to cache results,
- managing relation storage, and
- relation visualization.

Service's client is a Chrome extension which extracts text from the predefined set of web portals (if the user is currently reading them) and displays the extracted relations from the current article for user validation. This way a more accurate knowledge graph can be built,

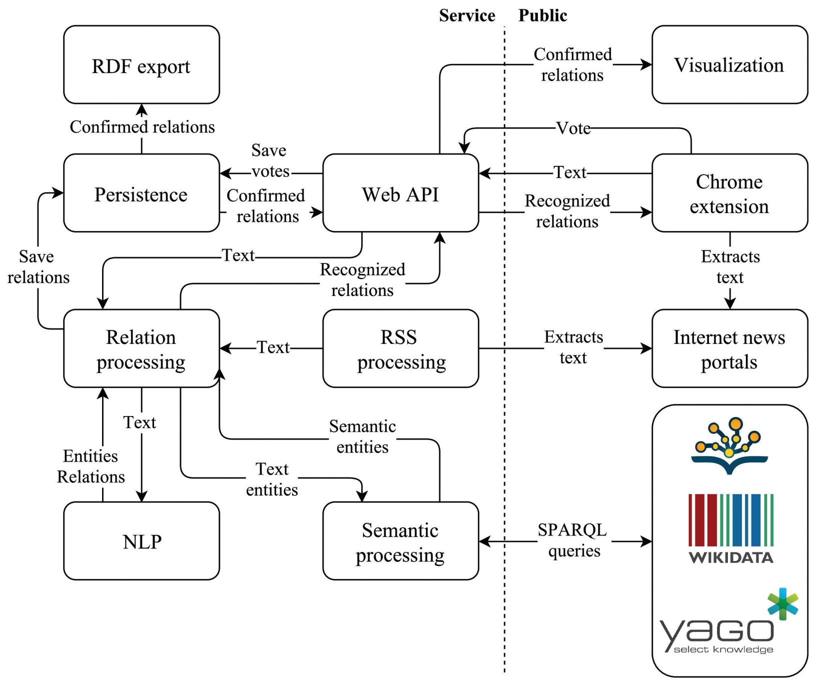
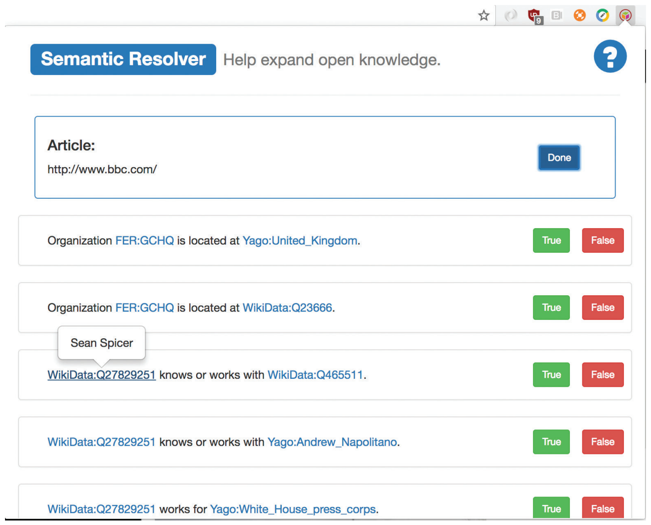


Figure 5 System architecture.



Semantic Resolver Help expand open knowledge.

Article:
http://www.bbc.com/
Done

Organization FER:GCHQ is located at Yago:United_Kingdom. True False

Organization FER:GCHQ is located at WikiData:Q23666. True False

Sean Spicer

WikiData:Q27829251 knows or works with WikiData:Q465511. True False

WikiData:Q27829251 knows or works with Yago:Andrew_Napolitano. True False

WikiData:Q27829251 works for Yago:White_House_press_corps. True False

Figure 6 Chrome extension interface for RDF triplets validation.

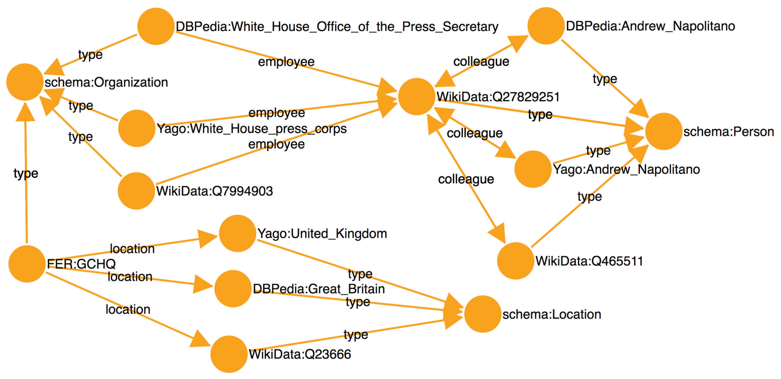


Figure 7 Semantic graph for relations in Figure 1.

and continuous model improvement is possible. Figure 6 shows the Chrome extension interface. A user can simply click on one or more True/False buttons to (in)validate a statement. Additional data for entities in relations is available via tooltip.

The overall data processing pipeline is a lasting process and is not fit for the synchronous paradigm (where the extension asks for relations for some arbitrary text). It is necessary to mitigate long response times from semantic repositories, and processing time in general. That is why we periodically (every few hours) process the articles from chosen web-portals on the server-side to acquire relations in advance: relations are cached and served when and if requested by users (extension) later on. This way, when user opens article from a chosen web portal, relations will be quickly delivered to extension.

Apart from processing the relations in advance, we also apply some other improvements to help speed up the overall process (e.g. caching of semantic entities per text, parallelization of query execution per repository, etc.). Of course, URLs, relations and sentences with entities, validation feedback and other data are stored in a database.

Relations that are confirmed by enough people are consider valid and exported in two formats. First format is a RDF/XML serialization of the RDF graph with relations. Subjects and objects of graph are semantic entities which were resolved during semantic processing while relations are mapped to respective predicates from "schema.org" ontology. Symmetric relations like "spouse" are stored in both directions. For each entity its type is also added to the graph by using "rdf:type" predicate and type from "schema.org" ontology. Resulting XML file is publicly available through web API. Service contains a component which periodically every few hours exports new confirmed relations. Once exported relations are not evaluated for their accuracy again. The same state of relations is also exported through service as a visualization which consists of an interactive directed graph. Figure 7 shows graph visualization for relations in Figure 1.

7 Conclusions and Future Work

In this paper a novel, semi-automated system for extraction and validation of semantic relations from unstructured natural language text was presented. The system uses NLP and machine learning algorithms to extract relations and, by doing so, removes additional work from information publishers. System extracts a good number of relations when looked from practical application perspective, while having better F1 score than related work in semantic relation extraction. Human crowd verification was added at the end of the process to mitigate the potential errors in relation extraction and to distribute the workload over large number of people who do not have to be familiar with concepts of semantic web. Evaluation metrics of system are excellent, and usability of the system is very encouraging.

Projects source code is available at: https://github.com/goran-rumin/Semantic-Resolver. Future work includes improving features for certain, more complex relations, adding continuous model training based on user feedback and extending the set of relations and named entity types.

References

[1] DBPedia, accessed November, 2018.

[2] Steven Bird and Edward Loper. Nltk: the natural language toolkit. In Proceedings of the ACL 2004 on Interactive poster and demonstration sessions, page 31. Association for Computational Linguistics, 2004.

[3] Christian Bizer, Robert Meusel, and Anna Primpeli. Web data commons – rdfa, microdata, embedded json-ld, and microformats data sets, accessed November, 2018.

[4] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. IEEE transactions on information theory, 13(1): 21–27, 1967.

[5] Nello Cristianini and John Shawe-Taylor. An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, 2000.

[6] Arooj Fatima, Arsalan Ghazi, and Cristina Luca. Semantic graph from free-text. In Optimization of Electrical and Electronic Equipment (OPTIM) and 2017 Intl Aegean Conference on Electrical Machines and Power Electronics (ACEMP), 2017 International Conference on, pages 1132–1137. IEEE, 2017.

[7] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In Proceedings of the 43rd annual meeting on association for computational linguistics, pages 363–370. Association for Computational Linguistics, 2005.

[8] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55(1):119–139, 1997.

[9] Aldo Gangemi, Valentina Presutti, Diego Reforgiato Recupero, Andrea Giovanni Nuzzolese, Francesco Draicchio, and Misael Mongioví. Semantic web machine reading with fred. Semantic Web, 8(6):873–893, 2017.

[10] Gumwon Hong. Relation extraction using support vector machine. In International Conference on Natural Language Processing, pages 366–377. Springer, 2005.

[11] Breiman L. and Cutler A. Random forests – classification description, accessed November, 2018.

[12] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, pages 1003–1011. Association for Computational Linguistics, 2009.

[13] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. Journal of machine learning research, 12(Oct):2825–2830, 2011.

[14] Christian Petersohn. Temporal video segmentation. Jorg Vogt Verlag, 2010.

[15] Beatrice Santorini. Part-of-speech tagging guidelines for the penn treebank project (3rd revision). Technical Reports (CIS), page 570, 1990.

[16] STLab. Fred home, accessed November, 2018.

[17] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In Proceedings of the 16th international conference on World Wide Web, pages 697–706. ACM, 2007.

[18] Mihai Surdeanu, David McClosky, Mason R Smith, Andrey Gusev, and Christopher D Manning. Customizing an information extraction system to a new domain. In Proceedings of the ACL

2011 Workshop on Relational Models of Semantics, pages 2–10. Association for Computational Linguistics, 2011.

[19] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. Communications of the ACM, 57(10): 78–85, 2014.

[20] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Soren Auer. Quality assessment for linked data: A survey. Semantic Web, 7(1):63–93, 2016.

[21] Aldo Gangemi. A comparison of knowledge extraction tools for the Semantic Web. The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Proceedings, volume 7882 of Lecture Notes in Computer Science, pages 351–366, Springer, 2013.

[22] Isabelle Augenstein, Diana Maynard and Fabio Ciravegna. Distantly supervised Web relation extraction for knowledge base population. Semantic Web, 7:335–349, 2016.

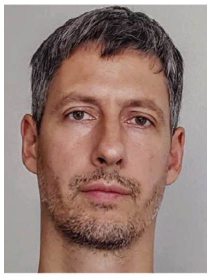
[23] Thien Huu Nguyen and Ralph Grishman. Relation Extraction: Perspective from Convolutional Neural Networks. Proceedings of NAACL-HLT 2015, pages 39–48. 2015.

[24] Makoto Miwa and Mohit Bansal. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 1105–1116. 2016.

Biographies

Goran Rumin is a software engineer in Infobip company currently working on infrastructure application development. He received his

B.Sc. and M.Sc. in computing from the University of Zagreb (Croatia), Faculty of Electrical Engineering and Computing in 2015 and 2017 respectively. He worked on various projects related to machine learning, high availability, service monitoring and security. His interests include machine learning and semantic web.



Igor Mekterović is currently an associate professor at the University of Zagreb (Croatia), Faculty of Electrical Engineering and Computing. He received his PhD. degree in 2008 from the same university. His research interests are in the areas of databases, data warehouses, web development and bioinformatics.