KIEV: a Tool for Extracting Semantic Relations from the World Wide Web*

Naimdjon Takhirov
NTNU
NO-7491 Trondheim, Norway
takhirov@idi.ntnu.no

Fabien Duchateau
Université Lyon 1
LIRIS, UMR5205
F-69622, Lyon, France
fduchate@liris.cnrs.fr

Trond Aalberg
NTNU
NO-7491 Trondheim, Norway
trondaal@idi.ntnu.no

Ingeborg Sølvberg
NTNU
NO-7491 Trondheim, Norway
ingeborg@idi.ntnu.no

ABSTRACT

Deriving knowledge from information stored in unstructured documents is a major challenge. The proliferation of knowledge sharing communities such as Wikipedia urge for automatic methods to construct a knowledge base consisting of entities and their relationships for advanced querying. More specifically, binary relationships representing a fact between two entities can be extracted to populate semantic triple stores or large knowledge bases. In this paper, we present our novel tool KIEV to fulfil this task. It combines a discovery process and a verification process for the entities and the type of relationship. We finally demonstrate three use cases for which KIEV is useful.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

General Terms

Design, Experimentation, Performance

Keywords

Relation extraction, Knowledge bases, Web mining

1. INTRODUCTION

There is a great amount of hidden structured information in text documents. Information available on the Web is particularly growing, and one of the next research challenges deals with the conversion of unstructured documents into exploitable facts such as relationships between entities, so that machines can interpret these facts. A major challenge with the data from the Web is that, on one hand, it contains valuable cultural data that might be missing from semantic knowledge bases, and on the other hand, it requires a great deal of intelligent processing of this data because high quality data is mixed up with low quality noisy text. The Linked Open Data (LOD) aims at making this vision a reality. In this domain, the building of knowledge bases such as DBpedia, MusicBrainz or Geonames is crucial. As a consequence, knowledge harvesting [3] and more generally large-scale acquisition of open-domain information extraction from the Web [1, 2, 4, 6, 8] are emerging fields with the goal of acquiring knowledge from textual content.

Challenges. The task of extracting relationships from large textual collection, such as the Web, has been explored in different ways. In this context, several challenges need to be tackled. The first one deals with the detection and the disambiguation of entities, which can have various labels (e.g., *Samuel Clemens* and *Mark Twain* represent the same person). In a pattern-based system, another issue is the generalization of the patterns, to remove the noise from the sentences. The identification of the type of relationship between two entities is another problem inherent to the power of expression of the languages. Finally, the Web is a huge collection of documents and our prototype needs to be a scalable application.

Contributions. In this paper, we propose our knowledge extraction tool **KIEV**, which stands for **K**nowledge and **I**nformation **E**xtraction with **V**erification. By exploiting a large collection of Web documents, the basic idea is to identify entities which are frequently detected together and to derive their type(s) of relationship based on the surrounding sentences. The verification step includes machine learning techniques to compute the type of relationship between two entities. In addition, the two entities are verified by linking them to common knowledge bases such as DBpedia or Freebase. Contrary to similar works, (i) KIEV defines a sophisticated verification process to avoid the detection of irrelevant examples; (ii) it supports flexible patterns (use of frequent terms) and (iii) extension of the entities; (iv) our tool is not limited to the discovery of the type of relationship, it can also perform other use cases such as entity list search and example discovery.

2. OVERVIEW OF KIEV

The technical and novelty aspects have been described in two papers [5, 6]. In the former, the approach is named SPIDER and it focuses on the extraction of relationships in a large scale context.

*This work has been partially funded by a PHC Aurora project and supported by the French "Ministère des Affaires Étrangères" (MAE), the French "Ministère de l'Enseignement Supérieur et de la Recherche" (MESR), and the "Norges Forskningsraad".

The latter approach, KIEV, is an extension of SPIDER, and it fosters a verification step based on machine learning techniques and interlinking. This demo paper aims at presenting the benefits of KIEV in various scenarios (see Section 3), and therefore we only briefly describe the approach in the rest of this section.

Figure 1 depicts the global overview of KIEV. Given a type of relationship, KIEV requires a collection of documents and a few training examples (verifying the types of relationship) to bootstrap a continuous run. Our collection is the ClueWeb09[1] category B collection (50 millions of English webpages). It is indexed with Hadoop, thus enabling efficient indexing and searching. The first step in KIEV consists of **discovering examples** from the textual collection. It is based on semantic tagging which combines Named Entity Recognition, Part of Speech tagging and Pattern Recognition. This discovery process generates many examples for the concepts contained in a sentence. Thus, a verification of the relevance for these examples is performed with two other processes. The former checks if the extracted entities are effectively related with the type of relationship using a **machine learning classifier**. The latter process **links both extracted entities** of an example to their corresponding entities on the LOD cloud. Once an example is verified, it can be used as a training example to improve the classifier, but also to reinforce the confidence score of a pattern during the discovery process [5]. Let us describe the three main processes.

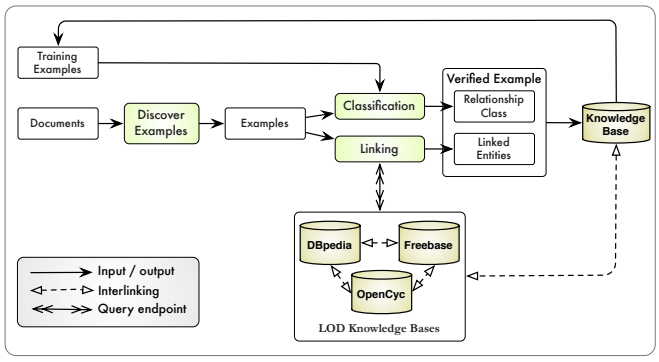

Figure 1: Overview for KIEV

Discovering examples. This process is illustrated with Figure 2. To discover examples, KIEV uses Named Entity Recognition (NER) techniques to extract entities (mainly nouns) in a document [1]. All pairs of entities which are close are considered as candidate examples. The idea is to find more documents containing a pair of entities. However, an entity may be identified by several labels. Thus, KIEV extends the label of the entities by discovering their alternative labels using LOD knowledge bases (e.g., *common.topic.alias* for Freebase, *wikiPageRedirects* for DBpedia), which requires the correct identification of the entity on the knowledge base [6]. When the two entities have been extended, we query the collection to obtain a set of documents which contain a mention (i.e., one of the labels) of both entities. The intuition deals with the detection of frequent terms surrounding the entities. All frequent terms are also extended using the Wordnet dictionary. Frequent terms enable the creation of generic patterns, which can be used later to discover more examples for the same type of relationship. When different patterns confirm that two entities are frequently found closely in many documents, these two entities form an example which is sent to the verification processes.

Verifying the type of relationship. Two entities extracted from textual documents may be linked or not by a type of relationship. This verification step is performed by a machine learning classifier which takes as input the two entities [5]. Given a set of features for the two entities (e.g., the frequency and the presence of frequent terms, the length and structure of the best-ranked patterns which generated the example, the average spamscore of the documents containing the entities), the classifier computes their type(s) of relationship if any.

Verifying the relevance of both entities. Entity Linking aims at discovering local entity's correspondence in another data source. In our context, we need to verify the relevance of the two entities of an example. The idea is to query the descriptive text attributes of LOD knowledge bases (e.g., *common.topic.article* for Freebase, *dbpedia-owl:abstract* for DBpedia) and to compare these text attributes to the context of an entity (sentences, frequent terms, etc.). This method ensures acceptable results for detecting the correct LOD entity [5]. Finally, a strategy is adopted to validate an example (i.e., an entity may not exist in a LOD knowledge base). In our prototype, an example is confirmed if at least one of the entities is matched to a knowledge base and if the type of relationship is verified by the classifier.

3. DEMONSTRATION SCENARIOS

This last section describes the three use cases to be presented at the conference. Each use case mainly depends on the input(s) that the user provides (among the two labels and a type of relationship).

3.1 Discovering the Type of Relation

This use case shows how to discover the possible type(s) of relationship between two entities. This use case is crucial in digital libraries for instance, when legacy data is converted to a semantic format: the semantics of the legacy format may not detail the relationship between a cultural work and a person (e.g., the cover). Let us imagine that a user searches for the type of relationship between *Bored of the Rings* and *Lord of the Rings*. By utilizing KIEV, the first result is *parody*. This scenario requires from the user to provide the two labels which represent both entities. Clicking the `Run` button launches the extension process to discover alternatives labels for the inputs. If the disambiguation process is not sufficient to select the correct LOD entity, KIEV asks the user for validating the correct LOD entity. Then queries composed of all possible pairs of alternative labels are sent to the ClueWeb09 collection. The set of returned documents is analyzed to extract the frequent terms surrounding the labels of the entities. Generic patterns are built from the sentences and the set of frequent terms. Finally, the verification step for the type(s) of relationship enables to verify the type of relationship based on features about the frequent terms, the patterns, the documents. The type(s) of relationships are ranked according to their support score and presented to the user.

3.2 Entity List Search

In this second scenario, users are interested in a list of entities which satisfy the rest of the relation, e.g., *the laureates of a Nobel Prize*. This task is very similar to the *entity search* research field [7], but we do not have any information about the types. This use case is useful to many many applications such as Web search and question answering. A practical benefit for semantic services deals with complex queries, where a user searches for *a list of museums which have abstract art paintings*. On KIEV's interface, the user can type in the label of an entity and select a type of relation-

[1]ClueWeb09, http://lemurproject.org/clueweb09.php

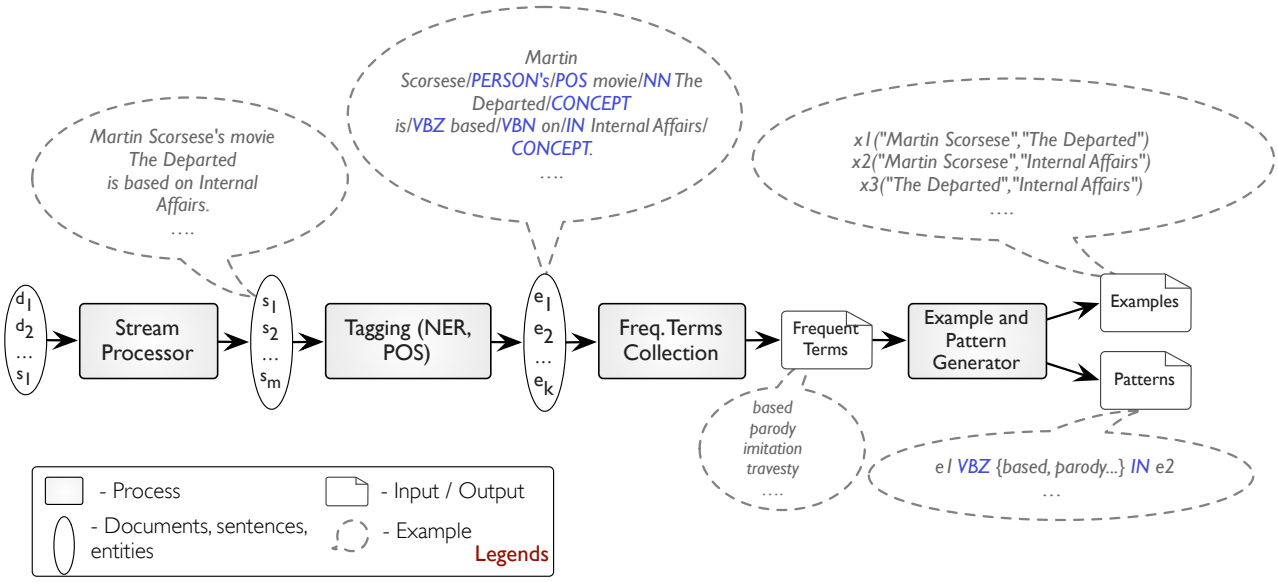

Figure 2: Workflow for Discovering Examples

ship[2]. KIEV automatically extends the entity's label with the extension component and it generates and reuses patterns that could have been previously generated for that type of relationship. Then, the process runs similarly as in the first use case, except that the tool queries the collection for only one entity. The text surrounding the entity is compared to the patterns (presence of frequent terms). If the pattern is a model for the sentence, then the NER component extracts all possible nouns as candidates for the second entity. The verification step runs for each candidate, and the validated ones are displayed to the user.

3.3 Discovering Examples

The last scenario aims at discovering pairs of entities (i.e., examples) for a given type of relationship. This task is crucial in a continuous system, which needs more examples to go on running. From the user point of view, let us imagine a database course instructor who needs to create Relational tables with their data: by *listing the name of presidents of countries*, KIEV might help her/him populating the tables. To perform this task, a user simply selects the type of relationship in the list (from validated training examples) and clicks on the `Run` button. KIEV retrieves all stored patterns associated to the given relationship, and it searches in the collection of documents for all frequent terms from these patterns. Sentences containing these frequent terms are analyzed using Part-of-Speech (POS) tagging and a similarity value is computed to assess the similarity between the pattern and the sentence. In other words, we evaluate whether the generic pattern can serve as a model for the sentence. If the similarity value is above a predefined threshold, the NER component extracts the pair of entities. KIEV finally displays all pairs of entities to the user. Note that performing reasoning directly on knowledge bases may provide better results, but KIEV can discover examples for complex types of relationships which do not exist in ontologies.

4. CONCLUSION

We have presented KIEV, which aims at discovering binary relationships from full-text documents. Its main advantages over similar tools are the multiple use cases which can be executed, the verification step which improves accuracy and the linking of entities to LOD knowledge bases. In the future, we intend to explore the extraction of ternary relationships, for instance those which are time-dependent.

[2]The list of types of relationship depends on the training examples stored in KIEV.

5. REFERENCES

[1] T. Hasegawa, S. Sekine, and R. Grishman. Discovering relations among named entities from large corpora. In *Proceedings of ACL*, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[2] Mausam, M. Schmitz, S. Soderland, R. Bart, and O. Etzioni. Open Language Learning for Information Extraction. In *Proceedings of EMNLP-CoNLL*, pages 523–534. ACL, 2012.

[3] N. Nakashole, M. Theobald, and G. Weikum. Scalable knowledge harvesting with high precision and high recall. In *Proceedings of WSDM*, pages 227–236, 2011.

[4] N. Nakashole, G. Weikum, and F. M. Suchanek. Discovering and exploring relations on the web. *PVLDB*, 5(12):1982–1985, 2012.

[5] N. Takhirov, F. Duchateau, and T. Aalberg. An evidence-based verification approach to extract entities for knowledge base population. In *Proceedings of ISWC*, pages 575–590. Springer, 2012.

[6] N. Takhirov, F. Duchateau, T. Aalberg, and I. Sølvberg. An Integrated Approach for Large-Scale Relation Extraction from the Web. In *Proceedings of APWeb*, pages 163–175. Springer, 2013.

[7] T. Tran, P. Mika, H. Wang, and M. Grobelnik. Semsearch'11: the 4th semantic search workshop. In *Proceedings of WWW (Companion Volume)*, pages 315–316. ACM, 2011.

[8] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: a probabilistic taxonomy for text understanding. In *Proceedings of SIGMOD*, pages 481–492. ACM, 2012.

APPENDIX: Screenshots of KIEV

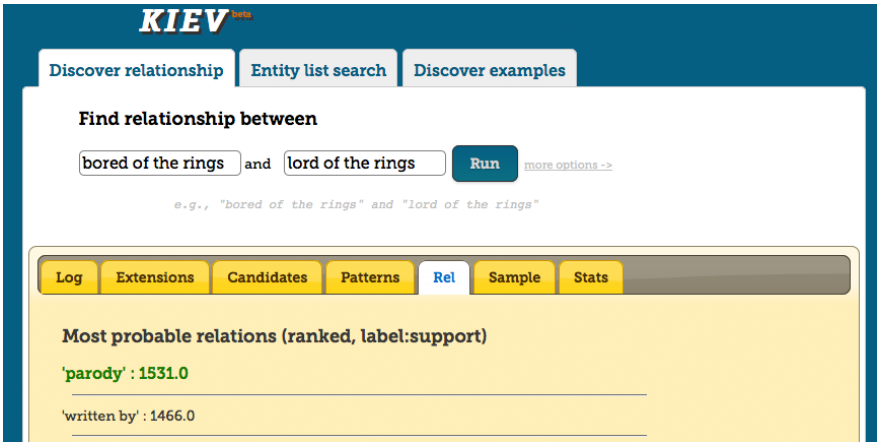

Figure 3: An Example of Use Case 1 (Discovering the Type of Relation): the labels "bored of the rings" and "lord of the rings" are provided by the user. The type(s) of relationship are then displayed according to a support score.

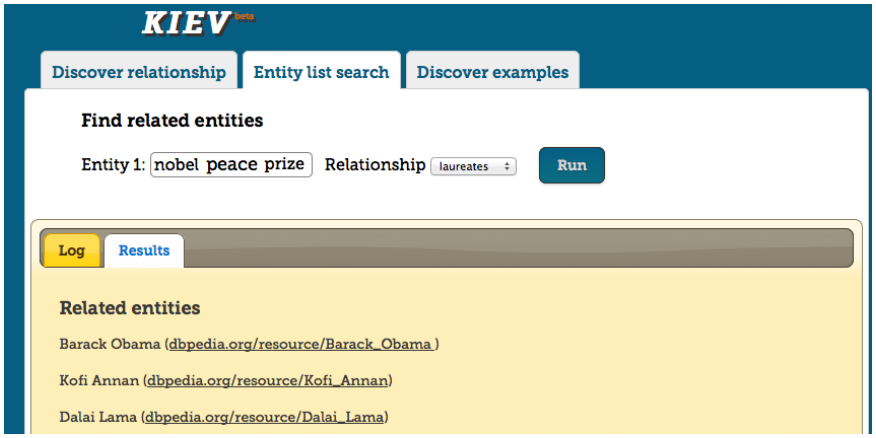

Figure 4: An Example of Use Case 2 (Entity List Search): the user provides an entity and selects a type of relationship. KIEV outputs the possible values for the second entity, which have been verified by linking to a LOD knowledge base. Note that the LOD entity for Dmitry Medvedev has not been discovered, but the adopted strategy only requires one of the entities to be validated.

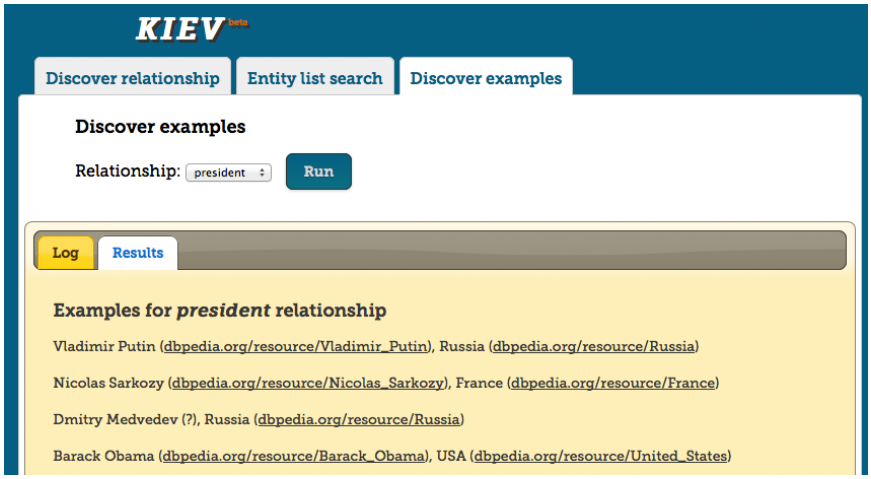

Figure 5: An Example of Use Case 3 (Discovering Examples): the user only selects a type of relationship in the list, and KIEV returns all examples (pair of entities) which satisfy the type of relationship. Note that the oldness of the ClueWeb dataset (2009) does not guarantee up-to-date results (e.g., Nicolas Sarkozy is not anymore the French president).