An Information Extraction Plugin for *RapidMiner 5*

Felix Jungermann
Technical University of Dortmund
Artificial Intelligence Group
felix.jungermann@cs.tu-dortmund.de

Abstract

In this paper we are presenting a *RapidMiner 5* - Information Extraction Plugin[1] which allows the use of information extraction (IE) techniques within the open source datamining software *RapidMiner* [1]. The plugin can be seen as an interface between natural language and IE- or datamining-methods, because it converts documents containing natural language texts into machine-readable form – preserving the original structures of the language – in order to extract interesting information like special entities and relations between these. The plugin is very modular and easy to use, which makes it applicable and extendable for different domains and tasks.

1 Introduction

Nowadays more and more information is available spread all over the internet or other huge document collections. The information is present on websites (containing pure text on the one hand and html-code on the other hand), in documents – pdf-documents for instance –, or in log-files and so on. To process this (daily growing) huge amount of information manually is impossible. Therefore IE-techniques are used for the automatic identification of selected types of entities, relations, or events in free text, as [2] says. While some IE-systems process IE-tasks like for instance Named Entity Recognition (NER) in a somehow black-boxed way, we present a very modular system, which can easily be adjusted and extended for already known or new tasks. The plugin presented in this paper is an update of the (not publicly published) version of the Information Extraction plugin for *RapidMiner 4* [3].

In contrast to only using the IE-functionalities, the already available datamining operators of *RapidMiner* allow the use of datamining-techniques in addition

[1] publicly available at http://www-ai.cs.tu-dortmund.de/SOFTWARE/IEPLUGIN
The plugin is under permanent development to improve, and extend its features. Contact the author if any questions arise.

to IE-techniques, or the analysis of information extracted out documents finally can be done using the existing datamining methods.

The paper further is structured as follows: Section 2 states three paradigms of data analysis in order to motivate the need of an Information Extraction plugin. Section 3 defines the schematic process of an information extraction task. Section 4 finishes the paper with a conclusion.

2 Paradigms of Data Analysis

This Section defines three types of analysis of datasets – namely *traditional datamining*, textmining and information extraction. Two of them (*traditional datamining* and textmining) already are supported by *RapidMiner* or its existing plugins. Information extraction methods and techniques will be supported by the plugin presented in this paper.

Traditional Datamining:
We use the term *traditional datamining* to describe tasks which process data originally containing information units (examples) which are defined by its attributes and the corresponding values. In most of these tasks the particular examples are independent of each other and the used techniques do not take in account other examples while processing a specific one. The independent handling of examples results in a *relaxed* processing of datasets – examples can be put in arbitrary batches during a cross-validation, for instance.

Textmining:
Textmining [4] originally converts a bunch of documents into a dataset which again can be processed by traditional datamining-techniques. One document is represented by one example. This means that a user just can compare different documents or predict the class/label of one particular document. It is not possible to extract multiple information units out of the document. The conversion/processing of the particular documents into information units is the crucial point.

It is remarkable that the original structure of the document and the text of the document is destroyed in most of the textmining tasks – for instance if the bag of word (BOW) representation of a document is created. But, if one is interested in interesting facts nested in the documents this structure is important.

Information Extraction:
During IE a document is split into several tokens which are represented as examples. These examples could be processed using datamining methods, but – as we will mention – preseving the structure and using structured models delivers better results. In contrast to textmining, processing one example for one document, for every document more than one example is created.

For the task of NER – which is part or preprocess of IE – a document is split into its sole words which are classified to be a known entity or not. In this

case, a token represents a word, but other tokens are conceivable.

Formally, NER is the task to predict the best label-sequence Y for a given token-sequence X, having seen a sufficient amount of training-pairs $< x^{(i)}; y^{(i)} >$ Traditional entity-types (used in the first NER-tasks at the *MUC 6* [5]) are *PERSON*, *LOCATION* and *ORGANIZATION*.

The first machine-learning based approaches for NER ignored the order of tokens.They just split the document into tokens and processed each of these tokens as an example being independent of other tokens/examples.

But the neighborhood of a token is important to make correct decisions for a token. [6] describes external and internal evidence as being necessary to extract the correct semantics for a given word.

Therefore, a next step was to encode the neighborhood of a specific token into the attributes for the token itself. But the examples and the decisions being made for them still were handled independently.

Further research showed that structured models deliver better results than independent models. These models like for instance conditional random fields (CRF) by [7] are working on multiple tokens at a time. This tokens can be structured in various ways – the most simple case is a linear chain – and the prediction/class/label for a token is conditioned by the predictions of the tokens in the neighborhood of the particular token. To realize this behaviour, it is necessary to respect the order of the tokens. CRFs for instance are predicting the best label-sequence Y for a given token-sequence X directly, whereas non-structured models predict the labels step-by-step. In the case of NER, CRFs are working on sentence-level which means that the sentence's structure in all cases has to be conserved.

This results in many restrictions, and multiple available operators cannot be used. If a sentence consists of some examples, and the structure must be conserved, a normal cross-validation cannot be used, because it might destroy the structure by selecting just some examples of the sentence and not all of them.

For nearly every information extraction task the structure of the document and the text of the document is needed and has to be preserved by the Information Extraction plugin.

3 Information Extraction Plugin

Figure 1 shows the schematic architecture of the Information Extraction plugin.Unfortunately, we are presenting just an exemplary application flow of an NER-task here because of the space limitations in this paper. The plugin offers much more operators like for instance techniques to process whole sentences by parsers in order to find related entities like in [8].

Input and Tokenization:
If a preprocessed dataset for information extraction tasks is not available, one

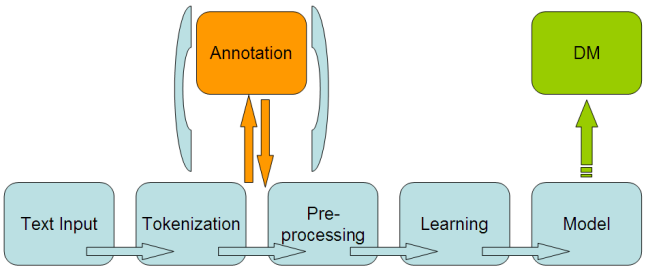

Figure 1: Schema of the Information Extraction plugin process

has the possibility to use the Textprocessing plugin (*RapidDoc*) to load documents into *RapidMiner*. Figure 2 presents the collaboration of Textprocessing and Information Extraction plugin. A document is read into the process, after that the operator *Documents to Data* converts every document into one example which contains an attribute storing the complete text of the document. To tokenize the document (preserving the order of the tokens) the Information Extraction plugin offers tokenizers like the sentence- and the word-tokenizer. These tokenizers work in a cascaded way. Tokenizing a document directly into sole words will give a user no information about the sentences and if a word is in another sentence than another word. Tokenizing a document into its sentences first and tokenizing the sentences into words afterwards delivers more and useful information.

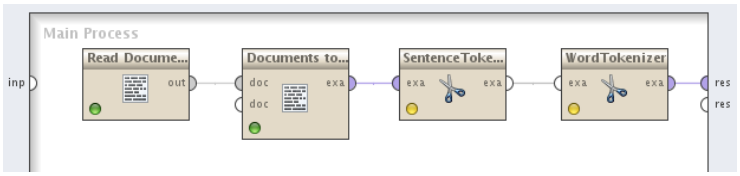

Figure 2: Loading and tokenizing a document

Visualization:
Using the available visualizers allows users to have a look on or to annotate documents. Figure 3 shows an annotated text. It is possible to visualize different attributes in texts. On the left side of Figure 3 named entities are colorized, and on the right side, the id-number of the sentences are colorized.

Preprocessing:
Preprocessing is an important requirement for information extraction. There are different techniques needed for different tasks in information extraction. If we have a look on NER we already know that tokens should be enriched by contextual information. Additionally, examples have to contain generalized information concerning the token in order to deliver predictive models. Famous generalizations are suffixes, prefixes or n-grams of the token.

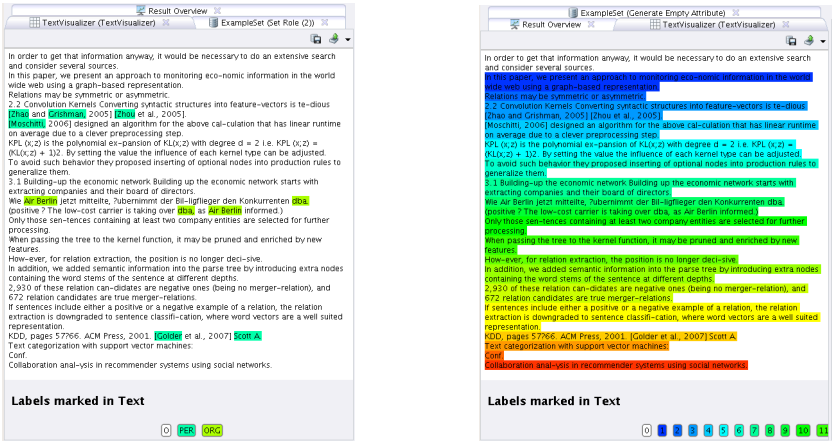

Figure 3: A document colorized by different attribute-values

Learning:
A CRF-operator can be used to train a structured model on such an enhanced exampleset, and the already available datamining operators can be used to compare the structured to an "unstructured" model. For the evaluation of IE tasks the same validation-measures like for other datamining tasks can be used (mostly, *precision*, *recall* and *f-measure*). But sometimes a classification is just correct, if the whole sequence of words is classified correctly (e.g. classifying 'Washington' as a Location and forgetting to classify the following 'D.C.' as a Location, too, causes a mistake and is not partially correct).

Figure 4 shows a complete information extraction task. In this case, an already tokenized dataset is read, after that the dataset is enriched by contextual information. Finally, a CRF-model is trained and applied on the same data – which of course is just done for exemplary reasons.

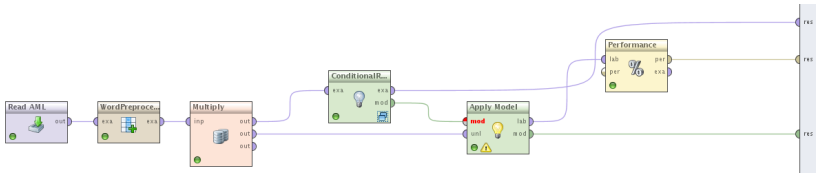

Figure 4: Learning and applying a NER-model

4 Conclusion and Future Work

We presented an Information Extraction plugin for *RapidMiner 5*. The plugin offers information extraction methods by tokenizing documents and by provid-

ing an example for each token. The tokenization can be done in a cascading manner allowing different levels of tokens which are needed for various information extraction tasks.

The current version of the plugin mostly is based on the exampleset architecture of RapidMiner. But it is planned to base most of the Information Extraction plugin functionalities into the Document-Object offered by the Textprocessing plugin (*RapidDoc*).

References

[1] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler, "YALE: Rapid Prototyping for Complex Data Mining Tasks," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)* (T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, eds.), (New York, USA), pp. 935–940, ACM Press, 2006.

[2] R. Grishman, "Information extraction," in *Handbook of Computational Linguistics Information Extraction*, ch. 30, Oxford University Press, USA, 2003.

[3] F. Jungermann, "Information extraction with rapidminer," in *Proceedings of the GSCL Symposium 'Sprachtechnologie und eHumanities'* (W. Hoeppner, ed.), pp. 50–61, Universität Duisburg-Essen, Abteilung für Informatik und Angewandte Kognitionswissenschaft Fakultät für Ingenieurwissenschaften, 2009.

[4] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proceedings of the European Conference on Machine Learning* (C. Nédellec and C. Rouveirol, eds.), (Berlin), pp. 137 – 142, Springer, 1998.

[5] R. Grishman, "Muc-6 website." http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html, 08 2010.

[6] D. McDonald, "Internal and external evidence in the identification and semantic categorization of proper names," in *Corpus Processing for Lexical Acquisition* (B. Boguraev and J. Pustejovsky, eds.), pp. 21–39, Cambridge, MA: MIT Press, 1996.

[7] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, Morgan Kaufmann, San Francisco, CA, 2001.

[8] M. Had, F. Jungermann, and K. Morik, "Relation extraction for monitoring economic networks," in *Proceedings of the 14th International Conference on Applications of Natural Language to Information Systems (NLDB)* (E. M. R. W. M. Horacek, H.; Metais, ed.), vol. 5723 of *Lecture Notes in Computer Science*, pp. 103–114, Springer Berlin / Heidelberg, 2009.