NLATool: An Application for Enhanced Deep Text Understanding

Markus Gärtner[1], Sven Mayer[2], Valentin Schwind[2], Eric Hämmerle[2], Emine Turcan[2], Florin Rheinwald[2], Gustav Murawski[2], Lars Lischke[2], Jonas Kuhn[1]

[1]University of Stuttgart, Institute for Natural Language Processing (IMS)
Pfaffenwaldring 5B, 70569 Stuttgart, Germany

[2]University of Stuttgart, Institute for Visualisation and Interactive Systems (VIS)
Pfaffenwaldring 5A, 70569 Stuttgart, Germany

[1]{firstname.lastname}@ims.uni-stuttgart.de

[2]{firstname.lastname}@vis.uni-stuttgart.de

Abstract

Today, we see an ever growing number of tools supporting text annotation. Each of these tools is optimized for specific use-cases such as named entity recognition. However, we see large growing knowledge bases such as Wikipedia or the Google Knowledge Graph. In this paper, we introduce NLATool, a web application developed using a human-centered design process. The application combines supporting text annotation and enriching the text with additional information from a number of sources directly within the application. The tool assists users to efficiently recognize named entities, annotate text, and automatically provide users additional information while solving deep text understanding tasks.

1 Introduction

A wide range of subfields in natural language processing (NLP) nowadays see systems emerging that solve their respective tasks with sufficiently high-quality levels. Especially tools for automatic entity recognition, entity linking or coreference resolution have advanced rapidly in recent years. Those tasks are also common sub-problems of general (human) text understanding. Usability and their actual applicability to real-world use cases are however often neglected aspects in the development of NLP tools.

Even analysts can benefit from the hints such an automatic preprocessing might provide. Unfortunately, the majority of NLP applications are tailored to a rather technically skilled user audience of experts and also typically focus on specific singular problems. The absence of widely applied standards for representing linguistic data and annotations further hampers interoperability of said systems. As a result, it is often difficult for technically less skilled users or non-experts in the field or other disciplines such as digital humanities (DH) to employ those tools successfully for assistance in their own work.

In this paper, we present a web application that aims to fill this gap. It is designed to assist users in analysis tasks that require deep text understanding without demanding expert or technical knowledge. It combines automatic predictions for several tasks centered around entity recognition and coreference resolution with information derived from a knowledge base. As a result, the tool offers a rich visualization of texts and the entities mentioned in them through an easy to use web interface.

2 Related Work

The main task performed by our system is comparable to entity linking, that is linking mentions of entities in unstructured data such as raw text to their corresponding entries in a knowledge base. Due to natural ambiguities and variations in the way entities can be referenced or mentioned, entity linking remains a challenging task for automatic systems and even for annotators or analysts. A variety of approaches and implementations thereof exist, and we refer to Shen et al. (2015) for a comprehensive survey of those.

There are several types of systems related to entity linking that are relevant in the context of this contribution: The simplest are entity disambiguation or linking implementations that also subsequently provide a visualization of their output. Most systems belong to this category such as DBPedia Spotlight[1] (Daiber

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/. See Section 4 for licensing information regarding the actual software.

[1]http://demo.dbpedia-spotlight.org

et al., 2013) or the YODIE (Gorrell et al., 2015) module for GATE[2]. They rarely provide more than rudimentary highlighting of the found entities and generally are not aimed at more assistive functions or visualizations. Similar to this, tools for wikification (i.e., annotating mentions with the Wikipedia link of the respective entity) include for instance the Illinois Cross-Lingual Wikifier (Tsai and Roth, 2016).

More advanced in terms of visualization, TASTY (Arnold et al., 2016) implements an as-you-type approach to interactive entity linking. It allows users to write a text through the application's own interface and be provided with a live outline of complementary information, such as a picture or article link. Its design is however limited in the scale of how much of this complementary information is visible at once.

3 Design

To determine requirements and desired features of text analysis and information extraction, we performed a qualitative user study with six computational linguistics (CL) experts (23-38 y.) from our faculty. Semi-structured interviews were conducted to determine desired functionality and features. After providing informed consent, participants received in the first phase of the survey four texts in random order (passages from *The Bible*, *Critique of Pure Reason* by Kant, *The Earthquake in Chile* by von Kleist, and excerpts from the *Nuremberg Trials*) to answer questions about comprehension of the texts. No input by the experimenter was provided. Afterward, we asked participants about features that would help to extract the information. In the second phase, we introduced our project and the first prototype mockups. We asked which tools they normally would use and which features they desire for information extraction. All interviews were transcribed and annotated using Atlas.ti[3].

Our results revealed five feature requests: (1) named entity recognition and filtering for information extraction, (2) tools for named entity density analysis, (3) a tool for additional notes and comments, (4) tools for segmenting and structuring text, and (5) fast inquiries based on background web search performed by the application. Data analysis of the interviews revealed four desired named entity main classes: persons, organizations, locations, and misc. For understanding text, participants asked for summaries of content related information outside the text column. Furthermore, participants desired an overview of named entity occurrences (corpus statistics) and a web-based entity search providing top matches and suggestions of named entities from the Google Knowledge Graph or Wikipedia. For named entity analysis, the participants in our study requested additional information with maps, photographs, and additional text. For misc data, the system should determine the closest match and dynamically present extracted information via text, image, or map. The participants also desired a correction feature of named entities. Additionally, we found that with the rise of high resolution screens and multiscreen setups in today's office environments a new tool should support making use of the increased screen space.

The requirement analysis of features for text annotation, and fast information extraction was used to design paper prototypes and mockups of the tool. In a series of design sessions with interaction designers and NLP professionals we found that a web application is best to implement our Natural Language Analysis Tool (NLATool). Furthermore, by using well established interaction patters we can further support the usability.

4 System

The CL community today already offers a quite wide range of very mature tools for numerous specific NLP tasks. This web-based solution is designed in a modular way to make use of existing analysis infrastructure and enable easy integration of other tools. As the main module, we used Stanford dependency parser (CoreNLP) (Manning et al., 2014) and its' features for text analysis. Additionally we used the Google Knowledge Graph to obtain information about the named entities beyond the text itself.

We implemented the NLATool as a web application . The main view is the text view where the user can view, edit, and delete named entities, the *text component*. However, it also presents an overview of all additional information beyond the text, the *research component*. We followed the metaphor of a split screen to enable the user to see both side by side. The *text component* takes up the left side of the screen

[2] `https://gate.ac.uk`
[3] `https://atlasti.com`

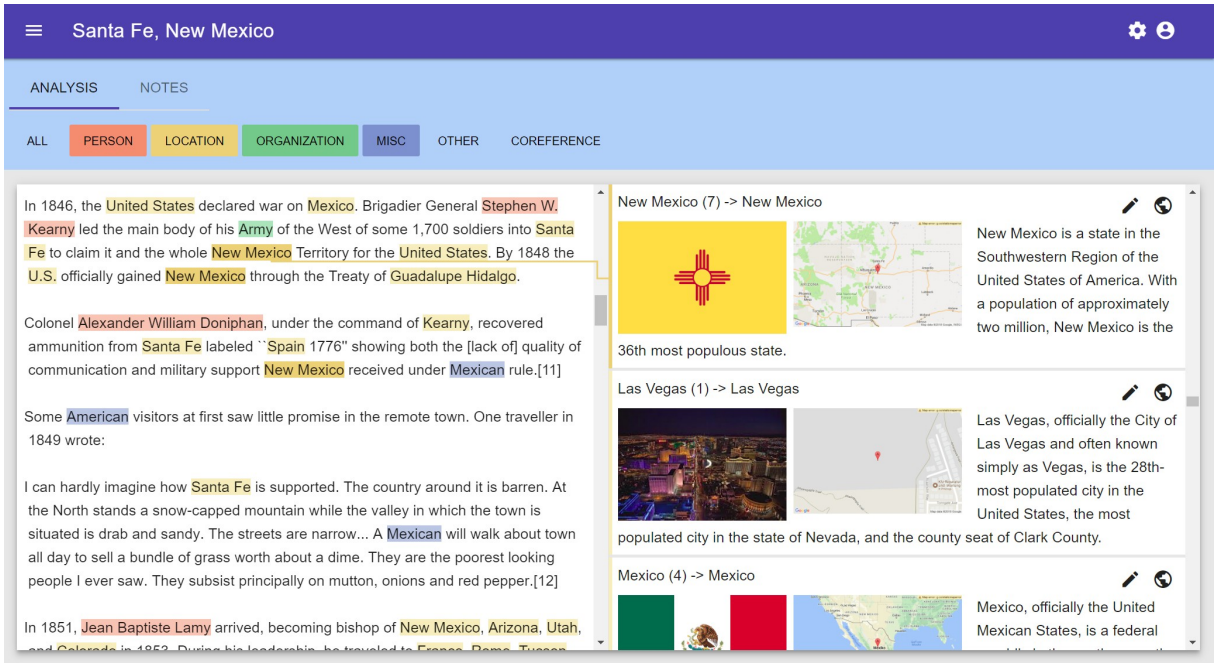

Figure 1: Screenshot of the system showing the menu on top, the text component on the left and the research component on the right. The systems presents the wikipedia article of "Santa Fe, New Mexico", which here is our example text. By hovering over "New Mexico" all related words are highlighted in a more saturated yellow to allow for a fast visual search. The yellow line between the word "New Mexico" and research result fosters spacial relation.

and the *research component* the right side. Additionally, above both views we implemented a menu strip similar to Microsoft Word and Excel. In the menu, the user can turn off and on specific highlighting functions for various named entities classes and enable coreference highlighting. Furthermore, the menu enables the user to switch into *edit mode* and *comment mode*. When switching into one of them the right side, the *research component* gets replaced. In *edit mode*, the user is guided through step by step instruction to add, change, or delete a certain named entity class which corresponds to one or more words. The system internally works with the 7-classes model; however, to not clutter the text we group Date, Time, Duration, Number, Set, and Ordinal into one group named "Other". Additionally, the user can link a different entity to the Google Knowledge Graph in case the previously linked entity is considered incorrect. In *comment mode*, the user can view all comments in a document, and add, change, or delete a comment with is either a global comment or conncted to a span of words.

Beyond the *text view*, the tool offers a view to input plain text which will then be analyzed and presented in the text view. Additionally, as all text and the corresponding comments and edits are stored on the server, the tool also offers to load previous texts. User management is not fully implemented yet as the current intent of the tool is used in a "host your own server" approach as quite often privacy regulations or licenses might not allow to upload text to third-party servers.

The front end of the tool is designed using the material.io framework which delivers the look and feel of a web app. The tool itself uses node.js as the main component and a MySQL database for fast and easy deployment. For the underlying annotation text analysis we use CoreNLP running in a server instance, this allows the node.js server to parse text which was uploaded by the user immediately. For the communication between node.js and the CoreNLP server we build on a free wrapper implementation named node-corenlp[4]. The source code of the system is publicly available under a GNU v3.0 license on GitHub[5]. This allows users to host their own server and enables the community to use the tool more effectively.

[4] https://github.com/gerardobort/node-corenlp
[5] https://github.com/interactionlab/NLATool

4.1 Research Feature

We use CoreNLP to extract all named entities from a given text during initial text setup. Then for each named entity we request the corresponding entry in the Google Knowledge Graph using rest API calls. Each unique result is presented to the user in the *research component* by the comprehensive text given by the Knowledge Graph. If a photo is available, it gets presented alongside the text to further support the user. For Locations and Organizations, an address is also available with the help of the Google Static Map API. For those cases we additionally show a small map directly within the user interface. By clicking on the map a larger interactive version opens in a new tab. For additional information beyond the initial text, the user can click on the text, and third-party content such as the corresponding Wikipedia article opens. As we request further data about every named entity, the Google Knowledge Graph returns the same results for similar requests, such as "Albert Einstein" and "Einstein" both return the same result. We use this property to group all named entities with the same result to reduce the initial clutter of the tool. The user can ungroup them if the initial grouping was incorrect and add new words to a group which was not identified by CoreNLP.

4.2 Highlighting Feature

To foster spacial relation between words and the corresponding research result we first highlight all related words whenever hovering over a word, and second, we add a connection line between the hovered word and the research result (see Figure 1). In case the research result is not in the viewport the elements gets scrolled into the view automatic. When hovering a research element, we use the same approach to foster spacial relationship, the line is drawn, and the words are highlighted.

4.3 Multiscreen Feature

To better support users during text analysis, we implemented a feature to make use of the screen space. Instead, of just having one pair of *text component* and *research component* and scrolling both, we enabled the system to support multiple pairs next to each other in one row. This allows having one pair of the *text* and *research component* per screen. The number of splits can be defined by the user. When using a split view the view changes from a scrolling method to a page flipping implementation. Each *text component* is filled with text but only up to an amount that visually fits into the component.

5 Conclusion

In this paper, we present our new NLATool to support analysis in deep text understanding. In detail, we present a web app based on node.js which combines the established text processing pipeline CoreNLP and the Google Knowledge Graph. We developed the tool using the human-centered design process to better support analysts in their work. Beyond the text, we support the analysts with more insights by presenting additional information gained from the Google Knowledge Graph right within the user interface.

Acknowledgments

This work was in parts funded by the German Research Foundation (DFG) within Cluster of Excellence in Simulation Technology (EXC 310/2) at the University of Stuttgart, the project C04 of SFB/Transregio 161 and project INF of SFB 732.

References

Sebastian Arnold, Robert Dziuba, and Alexander Löser. 2016. Tasty: Interactive entity linking as-you-type. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*.

Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. 2013. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems*.

Genevieve Gorrell, Johann Petrak, and Kalina Bontcheva. 2015. Using @twitter conventions to improve #lod-based named entity disambiguation. In *Proceedings of the 12th European Semantic Web Conference on The Semantic Web. Latest Advances and New Domains*, New York, NY, USA. Springer-Verlag.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*.

W. Shen, J. Wang, and J. Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *Knowledge and Data Engineering, IEEE Transactions on*.

Chen-Tse Tsai and Dan Roth. 2016. Illinois cross-lingual wikifier: Grounding entities in many languages to the english wikipedia. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*.