An English Reading Tool as a NLP Showcase

Mahmoud Azab   Ahmed Salama   Kemal Oflazer
Carnegie Mellon University-Qatar
Doha, Qatar
{mazab,ahmedsaa,ko}@qatar.cmu.edu

Hideki Shima   Jun Araki   Teruko Mitamura
Carnegie Mellon University,
Pittsburgh, PA, USA
{hideki,junaraki,teruko}@cs.cmu.edu

Abstract

We introduce -*SmartReader*- an English reading tool for non-native English readers to overcome language related hindrances while reading a text. It makes extensive use of widely-available NLP tools and resources. *SmartReader* is a web-based application that can be accessed from standard browsers running on PCs or tablets. A user can choose a text document from the system's library they want to read or can upload a new document of their own and the system will display an interactive version of such text, that provides the reader with an intelligent e-book functionality.

1 Introduction

Reading texts in a second language presents the language readers with a number of comprehension problems, especially when the reader does not have access to aids that would enable her to get over them including the problem of unknown words interpretation, unrecognized and forgotten names, difficult and hard-to-understand sentences, and lack of or forgetting the prior context in a former session of reading. There are many NLP-based tools, that offer various kinds of aids, to non-native English readers to help them in understanding a document. Many tools focus on assisting the reader in understanding of a specific word which may lead to better comprehension and vocabulary acquisition such as (Nerbonne et al., 1997) and (Eom et al., 2012). Some other tools focus on assisting the reader and second language learner with highlighting different patterns in the documents and providing the learner a visually enhanced version of the document (Meurers et al.,).

SmartReader is an implementation of a NLP-powered tool to aid in reading texts in English by non-native readers of the language which aims to make reading an active and interactive experience. In this paper, we present the underlying client-server architecture of *SmartReader*; for a detailed presentation of the user functionality provided by *SmartReader*, we refer the reader to Azab et al. (2013).

The main contribution of *SmartReader* is the integration of NLP tools and resources under the UIMA framework within a client-server architecture. The resulting web-accessible reading application can run on various browser platforms to help secondary language learners of English overcome language hindrances. Although currently *SmartReader* has currently been developed for English, it is *language independent*; it can easily be adapted to another language provided the relevant annotation tools and resources are available.

2 System Overview

Our system is based on client-server architecture as shown in Figure 1. The server is responsible for annotating plain text with NLP-related annotations and retrieving them based on the user's interactions. The client is a standard web browser running on PCs or touch tablets and interacts with a server running under a Tomcat web server. It passes user queries to the server and presents menu options and responses to the user. All annotations that are needed to respond to user requests (except for summarization), are produced, by preprocessing the text documents through a series of document annotators and storing their outputs in a UIMA Document Library file accessible to the server

SmartReader is based on significant preprocessing and annotation of texts using many publicly and fairly mature available NLP components for English, integrated in a UIMA (Unstructured Information Management Architecture)

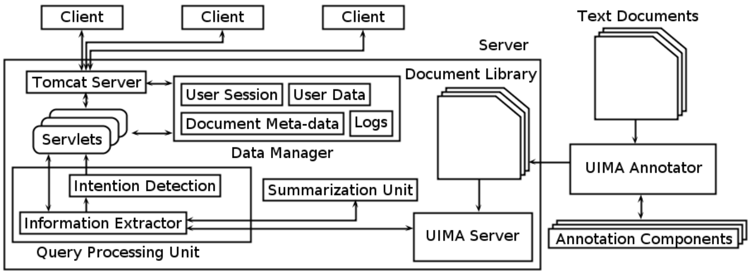

Figure 1: Client-Server Architecture

based server. UIMA is an architecture for the management and analysis of unstructured information such as text and voice, which is based on referential annotation (Ferrucci and Lally, 2004). It facilitates developing and integrating different text analysis engines and annotators. The input can be monotonically enriched while passing from one NLP analysis engine to the next, using a common data repository to all components (Götz and Suhre, 2004). UIMA also supports a flexible combination of individual NLP components into larger processing pipelines. Thus, we can re-use the same annotations (e.g., segmentation, tokenization, POS tagging) for all the next NLP components. It also provides a very powerful querying and search mechanisms for retrieving the annotations from the annotated documents.

The *SmartReader* server has two major functionalities: (1) annotating documents through preprocessing with UIMA annotators, (2) query processing in response to user requests.

Documents Annotation: During annotation, the input plain text is passed to a preprocessor to validate and normalize its orthography. Using Stanford CoreNLP tools, we segment the text into sentences, and then tokenize and perform POS tagging.[1] We then use the following NLP components to annotate the text:

- The **Stanford Dependency Parser** (De Marneffe et al., 2006), provides grammatical relation annotations for each word within the sentence.
- The **Stanford Named Entity Recognizer** (Finkel et al., 2005) and then the **Stanford Co-reference Resolution** (Lee et al., 2013; Lee et al., 2011; Raghunathan et al., 2010) are used to determine the entities in the text and the relationships between them.
- A simple **Word Sense Annotator** based on the Princeton WordNet (Fellbaum, 1998) is used as a broad-coverage machine-readable dictionary of English. As many words in WordNet have more than one sense, we narrow down the available senses by incorporate morphological analysis and part-of-speech filtering then annotate words with the most frequent WordNet sense under the selected part-of-speech. We are currently working on integrating a word sense disambiguation annotator.
- A **Compound Annotator** identifies and looks up the meaning of the phrasal verbs and the compound nouns in the text from WordNet.
- An **In-text Question Answering Annotator** assigns the questions to the related named entities, and ranks them. This is done in two phases. In the first phase, questions are generated using Heilman's question generator tool (Heilman and Smith, 2010). This tool generates a list of questions on every sentence by performing a set of syntactic and semantic transformations. Then, it ranks the generated questions for each sentences according to certain features. In the second phase, we go through the previously annotated named entities and the coreference chains they belong to and assign to every single mention a set of related questions that are generated in the first phase.

Once annotated, a document is loaded into the library. Figure 2, shows the annotation components that each document goes through. Figure 3 shows a logical view of a subset of the annotations for one sentence.

Query Processing: All queries from the user client application are translated into a character offset in the text. Thus when this character offset is passed to UIMA, it returns efficiently all the

[1] `http://nlp.stanford.edu/software/corenlp.shtml`

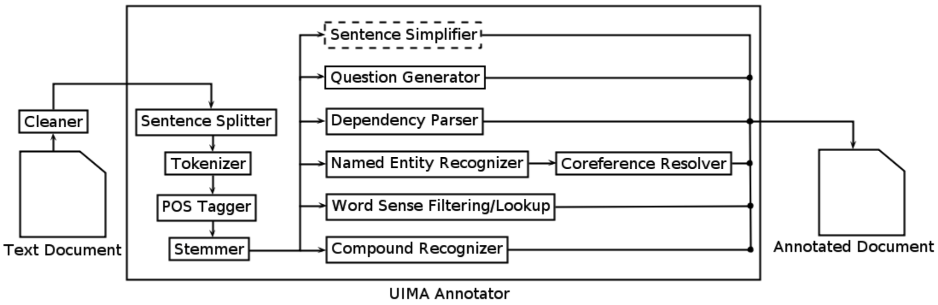

Figure 2: Document Annotation (modules with dotted lines are under development)

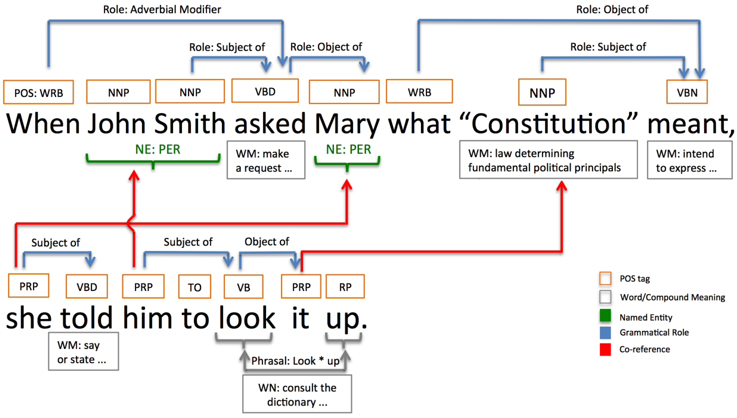

Figure 3: UIMA annotations for one sentence

annotations associated with the word overlapping with that position. These are which are then interpreted by the query processing unit as described earlier.

This step is responsible for processing the user queries and returning the required information. It consists of the following modules:

- **Information Extractor module:** This module efficiently searches a loaded document, then extracts and prepares the relevant information in response to the user's query.
- **Intention Detection module:** This module receives the annotations from the Information Extractor module then orders the set of response options to be presented to the user. It assumes that the annotations available for word/selection and its context indicates the intent of the user making the query.

3 Summary of User Functionality

From a reader's perspective, *SmartReader* is a web-based browser application, that runs on many browsers running on PCs or touch tablets; so on the reader side no additional software is needed. It has a simple and intuitive web interface to sign up/in, browse available texts to the system's library and to upload user's texts. Users also have the option to either upload their input text or try available preprocessed documents in the system library. After uploading/opening a text, *SmartReader* then loads an interactive version of the text into a distraction-free tab, and then the reader can start interacting with the text either by clicking on a word or selecting any segment of text.

The system in turn queries the server, which takes into account all the annotations around the clicked/selected word's/segment's and based on these annotations, highlights a segment of the text depending on the selection context, and presents a response, which most likely fits the reader's intent at the click position.[2] For instance:

- If the reader clicks on a content word, *SmartReader* will present the word meaning, along with word type, sentence examples including the inquired word, as the default response. In case the clicked word is a part of a (possibly discontinuous) compound verb/noun, the tool highlights the whole compound structure and provides its meaning.

[2] For a much more detailed overview of user functionality, please refer to Azab et al. (2013).

- If the reader clicks on a pronoun, the system will inform the reader to whom this pronoun refers by highlighting both the pronoun and the antecedent in context. It will also provides the reader with the ability to navigate through all previous and future mentions in the text.

- The reader can also inquire about the grammatical role of a word within the sentence. *SmartReader* provides the reader with the grammatical role in a user-friendly fashion by mapping dependency labels to more descriptive and meaningful labels.

- The reader can explore beyond the default response by using the additional menu items provided: for instance she may select from a set of questions that *SmartReader* can generate involving a selected named-entity and get the response.

- Beyond these sentences/words interactions, *SmartReader* provides the reader with different levels of text summarization such as multi-section and whole document summarization. For this purpose, we use the Mead toolkit for English to provide the summarization functionality. [3]

4 Conclusion

We presented the implementation and architecture of a tool for helping non-native readers of English text to overcome language related hindrances while reading text. Our tool dubbed *SmartReader* can also be seen as a showcase of English NLP tools and resources that have been built by the NLP community, integrated into an e-book reader application. Our system architecture is general so that *SmartReader* can be adapted to more languages provided annotations resources are available for use in the UIMA framework. We are currently completing our implementation and are in the process of planning a test deployment for students for experimentation.

Acknowledgments

This publication was made possible by grant NPRP-09-873-1-129 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

[3] Available at `http://www.summarization.com/mead/`

References

Mahmoud Azab, Ahmed Salama, Kemal Oflazer, Hideki Shima, Jun Araki, and Teruko Mitamura. 2013. An NLP-based reading tool for aiding non-native English readers. In *Proceedings of RANLP*, Hissar, Bulgaria.

Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Soojeong Eom, Markus Dickinson, and Rebecca Sachs. 2012. Sense-specific lexical information for reading assistance. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*.

Christiane Fellbaum. 1998. WordNet: An electronic lexical database. *The MIT Press*.

David Ferrucci and Adam Lally. 2004. UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of ACL'05*.

T. Götz and O. Suhre. 2004. Design and implementation of the UIMA common analysis system. *IBM Syst. J.*, 43(3):476–489.

Michael Heilman and Noah Smith. 2010. Extracting simplified statements for factual question generation. In *Proceedings of the 3rd Workshop on Question Generation*.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CONLL shared task. In *Proceedings CONLL'11*, pages 28–34.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, pages 1–54.

Detmar Meurers, Ramon Ziai, Luiz Amaral, Adriane Boyd, Ar Dimitrov, Vanessa Metcalf, Niels Ott, and UniversitŁt Tbingen. Enhancing authentic web pages for language learners.

John Nerbonne, Lauri Karttunen, Elena Paskaleva, Gabor Proszeky, and Tiit Roosmaa. 1997. Reading more into foreign languages. In *Proceedings of ANLP'97*.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of EMNLP'10*, pages 492–501.