Medstract – The Next Generation

Marc Verhagen
Computer Science Department
Brandeis University, Waltham, USA
marc@cs.brandeis.edu

James Pustejovsky
Computer Science Department
Brandeis University, Waltham, USA
jamesp@cs.brandeis.edu

Abstract

We present MedstractPlus, a resource for mining relations from the Medline bibliographic database. It was built on the remains of Medstract, a previously created resource that included a bio-relation server and an acronym database. MedstractPlus uses simple and scalable natural language processing modules to structure text and is designed with reusability and extendibility in mind.

1 Introduction

In the late 1990s, the Medstract project (Pustejovsky et al., 2002) set out to use common Natural Language Processing techniques and employ them to access relational information in Medline abstracts. Medstract used a set of pipelined Python scripts where all scripts operated on in-memory objects. The output of this pipeline was a set of relations, indexed by the PubMed identifier of the abstract in which they appeared. A Perl script proposed potential acronyms using a set of regular expressions on named entities in Medline abstracts. Both relations and acronyms were fed into an Oracle database, where access to these datasources was enabled by a set of Perl CGI scripts. The code, however, was not made public and was not maintained in any serious fashion after 2004. Developers of the system dispersed over the world and the Medstract server fatally crashed in 2007.

Here, we describe the resurrection of Medstract. One goal was that code should be open source and that installation should not depend on idiosyncracies of the developer's machine, which was a problem with the inherited code base. Reusability and extendability are ensured by following the principles embodied in the Linguistic Annotation Format (LAF) (Ide and Romary, 2006). In LAF, source data are untouched, annotations are grouped in layers that can refer to each other and to the source, and each layer is required to be mappable to a graph-like pivot format. For MedstractPlus, each component is set up to be independent from other layers, although of course each layer may need access to certain types of information in order to create non-trivial output. This allows us to swap in alternative modules, making it easier to experiment with different versions of the tagger and chunker for example. We now proceed to describe the system in section 2 and finish with the current status and future work in section 3.

2 System Design and Implementation

The general design of MedstractPlus is presented in Figure 1. The Lemmatizer creates what LAF calls the base-segmentation, a first layer of tokenized text that is the input to processing modules associated with other layers. The Lemmatizer incorporates a Python version of the Brill Tagger, extended with entries from the UMLS Thesaurus.

The Semantic Tagger is a group of components using (i) regular expressions for finding simple types like URLs, (ii) dictionary lookup in the UMLS type and concept lists as well as other typed word lists, (iii) off-the-shelf components like the Abner gene tagger (http://pages.cs.wisc.edu/ bsettles/abner/) and (iv) a statistical disambiguation model for genes trained on the GENIA corpus.

Proceedings of the 2011 Workshop on Biomedical Natural Language Processing, ACL-HLT 2011, pages 184–185,
Portland, Oregon, USA, June 23-24, 2011.

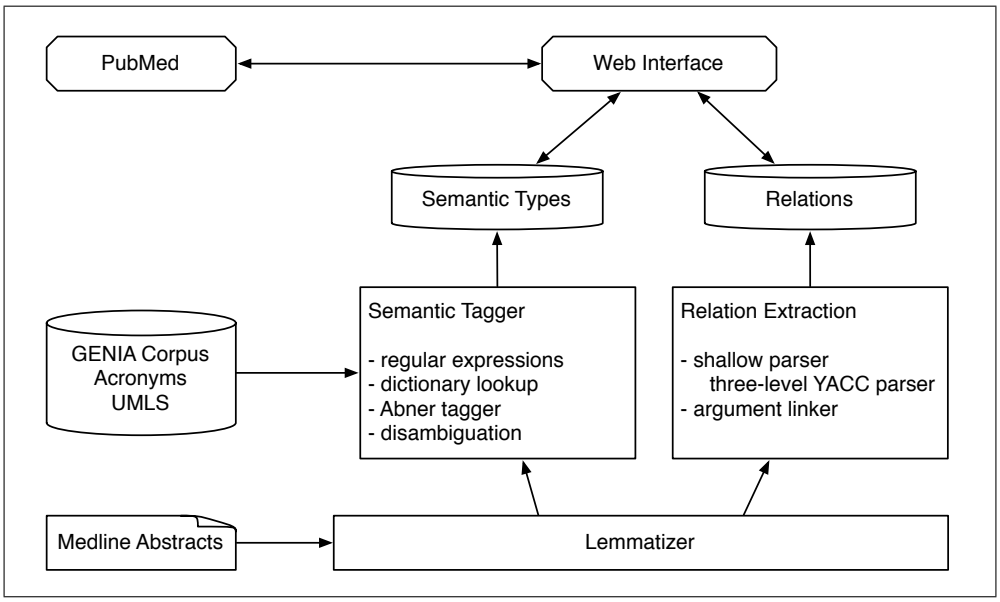

Figure 1: Overview of the MedstractPlus Architecture

The Relation Extraction component now contains a three-level 59-rule YACC parser that, starting with simple low-level chunking of noun and verb groups, proceeds to add more complex noun phrases and subordinated phrases. The argument linker produces binary relations, using a finite-state machine that runs on the data created by the shallow parser.

An advantage of this data-driven approach is that processing can be split up. A complete run of MedstractPlus on all Medline abstracts would take approximately 30 days on a entry-level desktop. But some relatively stable components like the Lemmatizer and the shallow parser (the latter being the most time-consuming component) can be run just once and subsequent runs can be restricted to those components that were changed.

The Web Interface gives access to the types and relations in a fairly standard way. In its current prototype form, it allows a user to type in a gene and then view all relations that the gene participates in. Alternatively, a pair of genes can be given.

3 Current Status and Future Work

The basic architecture depicted in Figure 1 is in place, but some components like the type disambiguator are in embryonic form. The web interface and the source code are or will be available at http://medstractplus.org.

Extensive additions to the basic typing and relation extraction component groups are in progress and the Relation Extraction component can be extended with specialized rule sets for specific relations like *inhibit* or *phosphorylate*. The interaction with the PubMed server is now limited to providing links. But the plan is that the MedstractPlus server will also query PubMed for relation pairs in case its own database provides little information. This approach can be extended to other relation servers like Chilibot (http://www.chilibot.net/), thereby moving towards a system than presents merged relations from the MedstractPlus database as well as relations from other servers.

Acknowledgments

This work was supported by the National Institutes of Health, under grant number 5R01NS057484-04.

References

Nancy Ide and Laurent Romary. 2006. Representing linguistic corpora and their annotations. In *Proceedings of the Fifth Language Resources and Evaluation Conference (LREC)*, Genoa, Italy.

James Pustejovsky, José Castaño, Roser Saurí, Anna Rumshisky, Jason Zhang, and Wei Luo. 2002. Medstract: Creating large-scale information servers for biomedical libraries. In *Proceedings of ACL'02*.