Knowtator: A Protégé plug-in for annotated corpus construction

Philip V. Ogren
Division of Biomedical Informatics
Mayo Clinic
Rochester, MN, USA
Ogren.Philip@mayo.edu

Abstract

A general-purpose text annotation tool called Knowtator is introduced. Knowtator facilitates the manual creation of annotated corpora that can be used for evaluating or training a variety of natural language processing systems. Building on the strengths of the widely used Protégé knowledge representation system, Knowtator has been developed as a Protégé plug-in that leverages Protégé's knowledge representation capabilities to specify annotation schemas. Knowtator's unique advantage over other annotation tools is the ease with which complex annotation schemas (e.g. schemas which have constrained relationships between annotation types) can be defined and incorporated into use. Knowtator is available under the Mozilla Public License 1.1 at http://bionlp.sourceforge.net/Knowtator.

1 Introduction

Knowtator is a general-purpose text annotation tool for creating annotated corpora suitable for evaluating Natural Language Processing (NLP) systems. Such corpora consist of texts (e.g. documents, abstracts, or sentences) and *annotations* that associate structured information (e.g. POS tags, named entities, shallow parses) with extents of the texts. An *annotation schema* is a specification of the kinds of annotations that can be created. Knowtator provides a very flexible mechanism for defining annotation schemas. This allows it to be employed for a large variety of corpus annotation tasks.

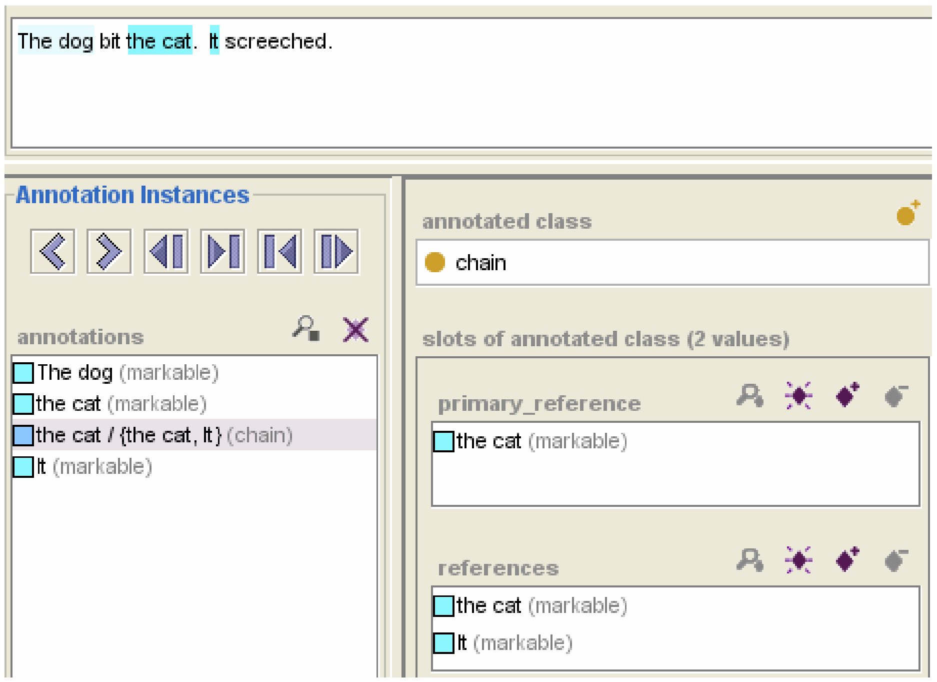
Protégé is a widely used knowledge representation system that facilitates construction and visualization of knowledge-bases (Noy, 2003)[1]. A Protégé knowledge-base typically consists of class, instance, slot, and facet frames. Class definitions represent the concepts of a domain and are organized in a subsumption hierarchy. Instances correspond to individuals of a class. Slots define properties of a class or instance and relationships between classes or instances. Facets constrain the values that slots can have.

Protégé has garnered widespread usage by providing an architecture that facilitates the creation of third-party plug-ins such as visualization tools and inference engines. Knowtator has been implemented as a Protégé plug-in and runs in the Protégé environment. In Knowtator, an annotation schema is defined with Protégé class, instance, slot, and facet definitions using the Protégé knowledge-base editing functionality. The defined annotation schema can then be applied to a text annotation task without having to write any task specific software or edit specialized configuration files. Annotation schemas in Knowtator can model both syntactic (e.g. shallow parses) and semantic phenomena (e.g. protein-protein interactions).

2 Related work

There exists a plethora of manual text annotation tools for creating annotated corpora. While it has been common for individual research groups to build customized annotation tools for their specific

[1] http://protege.stanford.edu



The dog bit the cat. It screeched.

The dog (markable)
the cat (markable)
the cat / {the cat, It} (chain)
It (markable)

chain

the cat (markable)

the cat (markable)
It (markable)

Figure 1 Simple co-reference annotations in Knowtator

annotation tasks, several text annotation tools have emerged in the last few years that can be employed to accomplish a wide variety of annotation tasks. Some of the better general-purpose annotation tools include Callisto[2], WordFreak[3] (Morton and LaCivita, 2003), GATE[4], and MMAX2[5]. Each of these tools is distributed with a limited number of annotation tasks that can be used 'out of the box.' Many of the tasks that are provided can be customized to a limited extent to suit the requirements of a user's annotation task via configuration files. In Callisto, for example, a simple annotation schema can be defined with an XML DTD that allows the creation of an annotation schema that is essentially a tag set augmented with simple (e.g. string) attributes for each tag. In addition to configuration files, WordFreak provides a plug-in architecture for creating task specific code modules that can be integrated into the user interface.

A complex annotation schema might include hierarchical relationships between annotation types and constrained relationships between the types. Creating such an annotation schema can be a formidable challenge for the available tools either because configuration options are too limiting or because implementing a new plug-in is too expensive or time consuming.

3 Implementation

3.1 Annotation schema

Knowtator approaches the definition of an annotation schema as a knowledge engineering task by leveraging Protégé's strengths as a knowledge-base editor. Protégé has user interface components for defining class, instance, slot, and facet frames. A Knowtator annotation schema is created by defining frames using these user interface components as a knowledge engineer would when creating a conceptual model of some domain. For Knowtator the frame definitions model the phenomena that the annotation task seeks to capture.

As a simple example, the co-reference annotation task that comes with Callisto can be modeled in Protégé with two class definitions called *markable* and *chain*. The *chain* class has two slots *references* and *primary_reference* which are constrained by facets to have values of type *markable*. This simple annotation schema can now be used to annotate co-reference phenomena occur-

[2] http://callisto.mitre.org
[3] http://wordfreak.sourceforge.net
[4] http://gate.ac.uk/. GATE is a software architecture for NLP that has, as one of its many components, text annotation functionality.
[5] http://mmax.eml-research.de/.

ring in text using Knowtator. Annotations in Knowtator created using this simple annotation schema are shown in Figure 1.

A key strength of Knowtator is its ability to relate annotations to each other via the slot definitions of the corresponding annotated classes. In the co-reference example, the slot *references* of the class *chain* relates the *markable* annotations for the text extents ‘the cat’ and ‘It’ to the *chain* annotation. The constraints on the slots ensure that the relationships between annotations are consistent.

Protégé is capable of representing much more sophisticated and complex conceptual models which can be used, in turn, by Knowtator for text annotation. Also, because Protégé is often used to create conceptual models of domains relating to biomedical disciplines, Knowtator is especially well suited for capturing named entities and relations between named entities for those domains.

3.2 Features

In addition to its flexible annotation schema definition capabilities, Knowtator has many other features that are useful for executing text annotation projects. A consensus set creation mode allows one to create a gold standard using annotations from multiple annotators. First, annotations from multiple annotators are aggregated into a single Knowtator annotation project. Annotations that represent agreement between the annotators are consolidated such that the focus of further human review is on disagreements between annotators.

Inter-annotator agreement (IAA) metrics provide descriptive reports of consistency between two or more annotators. Several different match criteria (i.e. what counts as agreement between multiple annotations) have been implemented. Each gives a different perspective on how well annotators agree with each other and can be useful for uncovering systematic differences. IAA can also be calculated for selected annotation types giving very fine grained analysis data.

Knowtator provides a pluggable infrastructure for handling different kinds of text source types. By implementing a simple interface, one can annotate any kind of text (e.g. from xml or a relational database) with a modest amount of coding.

Knowtator provides stand-off annotation such that the original text that is being annotated is not modified. Annotation data can be exported to a simple XML format.

Annotation filters can be used to view a subset of available annotations. This may be important if, for example, viewing only named entity annotations is desired in an annotation project that also contains many part-of-speech annotations. Filters are also used to focus IAA analysis and the export of annotations to XML.

Knowtator can be run as a stand-alone system (e.g. on a laptop) without a network connection. For increased scalability, Knowtator can be used with a relational database backend (via JDBC).

Knowtator and Protégé are provided under the Mozilla Public License 1.1 and are freely available with source code at http://bionlp.sourceforge.net/Knowtator and http://protege.stanford.edu, respectively. Both applications are implemented in the Java programming language and have been successfully deployed and used in the Windows, MacOS, and Linux environments.

4 Conclusion

Knowtator has been developed to leverage the knowledge representation and editing capabilities of the Protégé system. By modeling syntactic and/or semantic phenomena using Protégé frames, a wide variety of annotation schemas can be defined and used for annotating text. New annotation tasks can be created without writing new software or creating specialized configuration files. Knowtator also provides additional features that make it useful for real-world multi-person annotation tasks.

References

Thomas Morton and Jeremy LaCivita. 2003. *Word-Freak: An Open Tool for Linguistic Annotation*, Proceedings of NLT-NAACL, pp. 17-18.

Noy, N. F., M. Crubezy, et al. 2003. *Protege-2000: an open-source ontology-development and knowledge-acquisition environment.* AMIA Annual Symposium Proceedings: 953.