MAE and MAI: Lightweight Annotation and Adjudication Tools

Amber Stubbs
Department of Computer Science
Brandeis University MS 018
Waltham, Massachusetts, 02454 USA
astubbs@cs.brandeis.edu

Abstract

MAE and MAI are lightweight annotation and adjudication tools for corpus creation. DTDs are used to define the annotation tags and attributes, including extent tags, link tags, and non-consuming tags. Both programs are written in Java and use a stand-alone SQLite database for storage and retrieval of annotation data. Output is in stand-off XML.

1 Introduction

The use of machine learning for natural language processing tasks has been steadily increasing over the years: text processing challenges such as those associated with the SemEval workshops (Erk and Strapparava, 2010) and the I2B2 medical informatics shared tasks (i2b2 team, 2011) are well known, and tools for training and testing algorithms on corpora, such as the Natural Language Tool Kit (Bird et al., 2009) and the WEKA tools (Hall et al., 2009) are widely used.

However, a key component for training a machine for a task is having sufficient data for the computer to learn from. In order to create these corpora, human researchers must define the tasks that they wish to accomplish and find ways to encode the necessary information, usually in some form of XML, then have relevant data annotated with XML tags.

The necessity of corpus annotation has led to a number of useful tools, as well as assessments for tool usability and standards for linguistic annotation. A recent survey (Dipper et al., 2004) examined what attributes an annotation tool should have for it to be most useful, and the Linguistic Annotation Framework (LAF) describes the desired properties of an annotation framework to ensure interoperability and utility (Ide and Romary, 2006).

The Multi-purpose Annotation Environment (MAE), and the Multi-document Adjudication Interface (MAI) were designed to be easy to begin using, but have enough flexibility to provide a starting point for most annotation tasks. Both programs are written in Java and use a stand-alone SQLite database[1] for storage and retrieval of annotation data, and output standoff XML that is compliant with the abstract LAF model. Both of these tools are available from `http://pages.cs.brandeis.edu/~astubbs/`

2 Related Work

As previously mentioned, there are already a number of annotation tools in use—Dipper et al. examined five different programs; additionally Knowtator (Ogren, 2006), GATE (Cunningham et al., 2010), Callisto (MITRE, 2002), and BAT (Verhagen, 2010) have been used for various annotation tasks; and the list goes on[2]. However, as Kaplan et al. noted in a paper about their own annotation software, SLAT 2.0 (2010), much annotation software is not generic, either because it was designed for a specific annotation task, or designed to be used in a particular way. BAT, for example, utilizes a layered annotation framework, which allows for adjudication at each step of the annotation process, but this makes

[1] http://www.zentus.com/sqlitejdbc/

[2] See http://annotation.exmaralda.org/index.php/Tools for a reasonably up-to-date list of annotation software

Proceedings of the Fifth Law Workshop (LAW V), pages 129–133,
Portland, Oregon, 23-24 June 2011.

tasks difficult to modify and is best suited for use when the schema is not likely to change. GATE was built primarily as a tool for automated annotation, and Callisto, while excellent for annotating contiguous portions of texts, cannot easily create links—it requires the user to create an entire task-specific plug-in. Knowtator provides links and extent tagging, but comes as a plug-in for Protégé[3], a level of overhead that users may find daunting. Similarly, the Apache UIMA system (Apache, 2006) is well developed and supported but presents a very steep learning curve for task creation.

As for adjudication, while some software has built-in judgment capabilities (GATE, BAT, Knowtator, and SLAT, for example), that functionality does not stand alone, but rather relies on the annotations being done in the same environment.

All of the tools mentioned are well-suited for their purposes, but it seems that there is room for an annotation tool that allows for reasonably complex annotation tasks without requiring a lot of time for setup.

3 Simple Task Creation

One of the defining factors that Dipper et al. (2004) identified in evaluating annotation tools is simplicity of use–how long does it take to start annotating? Upon examining various existing annotation tools, they found that there was often a trade-off between simplicity and data quality assurance: tools that have an open interface and loose restrictions for tag sets tended to have lower quality data output, while tools that require a specification could output better data, but took a little longer to get running.

MAE and MAI attempt to find a middle ground between the two extremes: they require task definitions in the form of slightly customized Document Type Definition (DTD) files, which are used to define the tags and their attributes but are not difficult to create or modify[4].

There are two types of tags that are primarily used in annotation: extent tags (sometimes called 'segments' (Noguchi et al., 2008)) which are used to mark a contiguous portion of text as having a specific characteristic, and link tags, which are used to create a relationship between two extent tags. MAE and MAI support both of these tag types, and additionally support non-consuming extent tags, which can be useful for having an annotator mark explicitly whether or not a particular phenomena appears in the document being annotated.

DTD creation is quite simple. If, for example, an annotator wanted to look at nouns and mark their types, they could define the following:

```
<!ELEMENT NOUN (#PCDATA)>
<!ATTLIST NOUN type
  (person|place|thing|other)>
```

The "#PCDATA" in the first line informs the software that NOUN is an extent tag, and the second line gives NOUN an attribute called "type", with the possible values defined in the list in parenthesis.

Creating a link is equally simple:

```
<!ELEMENT ACTION EMPTY >
<!ATTLIST ACTION relationship
   (performs|performed_by)>
```

The "EMPTY" marker indicates that the tag is a link, and the attributes and attribute values work the same way as for extent tags.

4 MAE

Once the DTD is created and files are preprocessed, the user loads the DTD and a file into MAE. The text to be annotated appears in the window, and a window at the bottom of the screen holds a table for each tag (see Figure 1). When a user selects an extent and creates a tag, some information about the tag is automatically added to the table: the start and end locations of the tag, and the text of the extent. Additionally, MAE will automatically generate a document-unique ID number for that tag so that it can easily be referenced in links.

The user can then add in any information about the attributes by filling in the table at the bottom of the screen. In the text window, the color of the extent that has been tagged is changed to the color associated with that tag. If there are multiple tags in a location, the text is underlined as well. Highlighting tagged text in the window will also highlight any table rows associated with that tag, including link tags. This makes it easy for the annotator to see what information has already been added about that text.

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

[4] In the future, a GUI will be added to MAE that will make the DTD creation process easier.

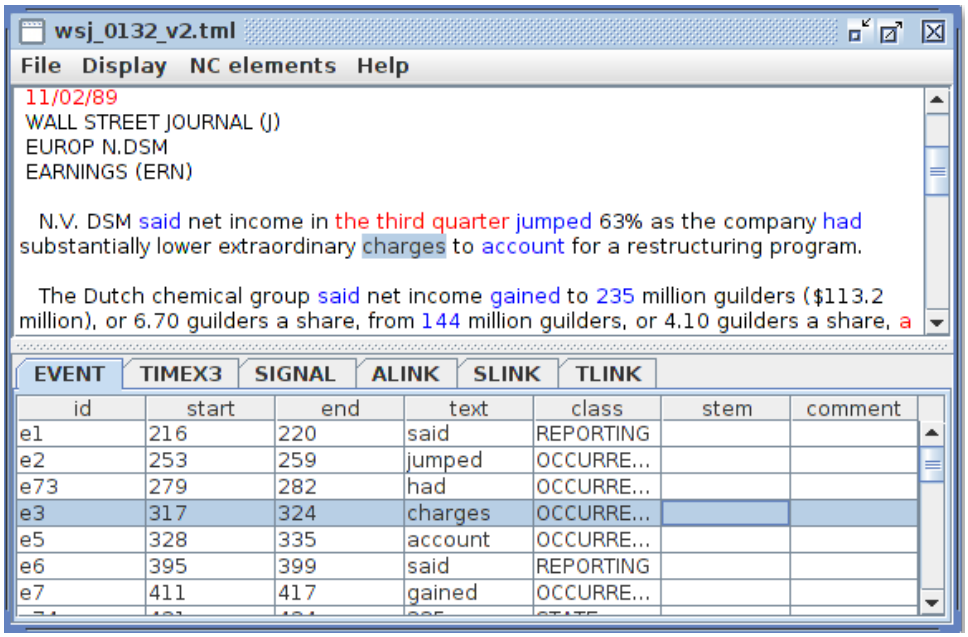

Figure 1: TimeML annotation in MAE.

Non-consuming tags are created from the menu at the top of the screen. Links are created by holding down the control key (or the command key on Macs) and clicking on the two tags that will be linked. A window pops up that allows the user to link either to the tags at the specified locations, or to any non-consuming tags that have been created in the document.

5 Output

Once the user is done annotating, they can save their work in an XML file. MAE outputs (and takes as input) UTF-8 encoded files, so it can be used to annotate any character set that is representable in UTF-8, including Chinese. The output is compliant with the LAF guidelines (Ide and Romary, 2006).

5.1 System testing

MAE is currently being used for a variety of annotation tasks: medical record annotation, eligibility criteria assessment, and for a university course on corpus creation. Annotation tasks in that course range from opinion annotation to tense and aspect in Chinese verbs. It is currently being used on Windows, Mac, and Linux.

6 MAI

MAI is built on the same back-end code as MAE, making them easily compatible. Like MAE, using MAI begins with loading a DTD. Then the adjudicator can load each annotation of a text that they would like to create a gold standard for. As each new document is added, MAI loads the tag information for each annotation into the database for quick reference.

Once all the files are loaded, the adjudicator selects the tag they want to review from the left part of the screen. The text is then color-coded to reflect the agreement of the annotators: blue if all the annotators agree that a tag of the selected type should be at a location, red if only a subset would place a tag there, and black for locations where no tag is present (see Figure 2).

When text is highlighted by the adjudicator, the information about each annotator's tag and attributes for that location is filled in on a table to the right of the screen. From there, the annotator can either fill in the values for the gold standard by hand, or copy the values from one annotator directly into the gold standard column and modifying them as needed. Once the adjudicator is satisfied with the gold standard they can add the annotation to the database by

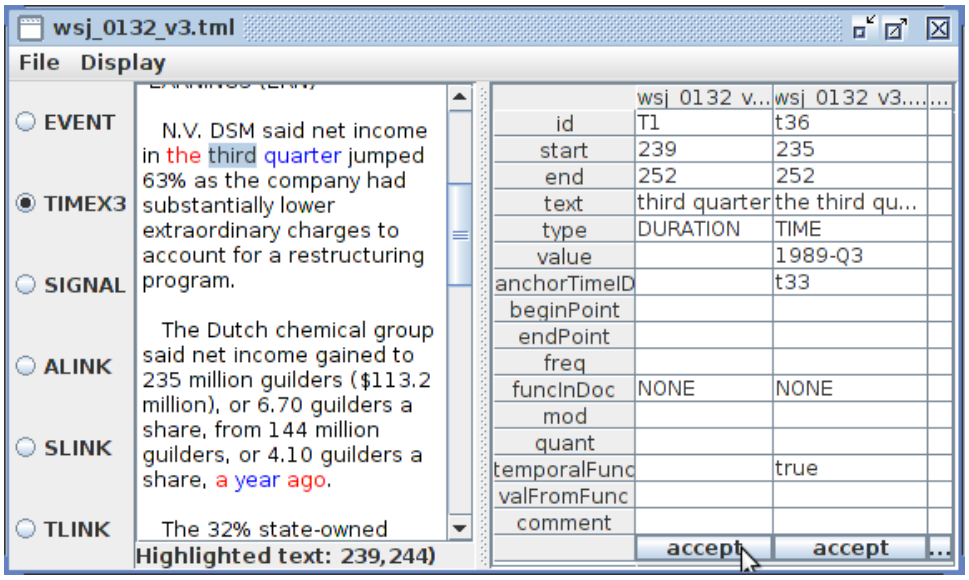

Figure 2: The extent adjudication table in MAI

clicking the "accept/modify" button at the bottom of the gold standard column. At this point, MAI will generate a new ID for that tag, and the color of the adjudicated font will become green.

At the time of this writing, the algorithms for link and non-consuming tag adjudication have not been fully worked out for use inside of MAI. However, once the extent tags have been adjudicated, the annotator can choose to export the non-consuming tags and link tags that involve "approved" extent tags into an XML file, along with the adjudicated extents. This partially-judged file can then be loaded into MAE, where it is easier to display and modify all the relevant information.

6.1 System testing

As with MAE, MAI has been used for the various annotation projects for a course on corpus creation, as well as a medical record annotation task. This program is still under development, but so far adjudications tasks with MAI have proved successful.

7 Conclusions and Future Work

While MAE and MAI do not represent a new frontier in annotation software, I believe that their ease of use, portability, and clean visualization will make them useful tools for annotation projects that do not want to invest in the time required to use other existing software, and for adjudicators that want an easy way to fix discrepancies between annotators. Admittedly, tasks involving heirarchical annotations would require one of the more sophisticated tools that are currently available, but there are still many tasks that do not require that level of complexity that MAE and MAI can be used for.

There is room for improvement in both of these programs: fully implementing link adjudication in MAI, allowing for more customization in the visualizations would make them more enjoyable to use, and expanding the functionality to make them more useful for more tasks (for example, allowing links with multiple anchors instead of just two). Both MAE and MAI are under development, and improvements to both will be made over the coming months.

Acknowledgments

Funding for this project development was provided by NIH grant NIHR21LM009633-02, PI: James Pustejovsky

Many thanks to the annotators who helped me identify bugs in the software, particularly Cornelia Parkes, Cheryl Keenan, BJ Harshfield, and all the students in the Brandeis University Spring 2011 Computer Science 216 class.

References

Apache. 2006. Unstructured information management architecture. http://uima.apache.org/.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc, Sebastopol, CA, first edition edition.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, and Ian Roberts, 2010. *Developing Language Processing Components with GATE*, 5 edition, July.

Stefanie Dipper, Michael Götze, and Manfred Stede. 2004. Simple annotation tools for complex annotation tasks: an evaluation. In *Proceedings of the LREC Workshop on XML-based Richly Annotated Corpora*, pages 54–62, Lisbon, Portugal.

Katrin Erk and Carlo Strapparava. 2010. Semeval-2. http://semeval2.fbk.eu/semeval2.php.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explorations*, 11(1).

i2b2 team. 2011. i2b2 shared task. https://www.i2b2.org/NLP/Coreference/Main.php. accessed Feb. 2011.

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

Dain Kaplan, Ryu Iida, and Takenobu Tokunaga. 2010. Slat 2.0: Corpus construction and annotation process management. In *Proceedings of the 16th Annual Meeting of The Association for Natural Language Processing*, pages pp.510 – 513.

MITRE. 2002. Callisto website. http://callisto.mitre.org/index.html. accessed Dec. 17, 2010.

Masaki Noguchi, Kenta Miyoshi, Takenobu Tokunaga, Ryu Iida, Mamoru Komachi, and Kentaro Inui. 2008. Multiple purpose annotation using slat - segment and link-based annotation tool -. In *Proceedings of 2nd Linguistic Annotation Workshop*, pages pp.61 – 64.

Philip V. Ogren. 2006. Knowtator: a protégé plug-in for annotated corpus construction. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 273–275, Morristown, NJ, USA. Association for Computational Linguistics.

Marc Verhagen. 2010. The brandeis annotation tool. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).