Abstract

We introduce the brat rapid annotation tool (BRAT), an intuitive web-based tool for text annotation supported by Natural Language Processing (NLP) technology. BRAT has been developed for rich structured annotation for a variety of NLP tasks and aims to support manual curation efforts and increase annotator productivity using NLP techniques. We discuss several case studies of real-world annotation projects using pre-release versions of BRAT and present an evaluation of annotation assisted by semantic class disambiguation on a multi-category entity mention annotation task, showing a 15% decrease in total annotation time. BRAT is available under an open-source license from: http://brat.nlplab.org

1 Introduction

Manually-curated gold standard annotations are a prerequisite for the evaluation and training of state-of-the-art tools for most Natural Language Processing (NLP) tasks. However, annotation is also one of the most time-consuming and financially costly components of many NLP research efforts, and can place heavy demands on human annotators for maintaining annotation quality and consistency. Yet, modern annotation tools are generally technically oriented and many offer little support to users beyond the minimum required functionality. We believe that intuitive and user-friendly interfaces as well as the judicious application of NLP technology to support, not supplant, human judgements can help maintain the quality of annotations, make annotation more accessible to non-technical users such as subject domain experts, and improve annotation productivity, thus reducing both the human and financial cost of annotation. The tool presented in this work, BRAT, represents our attempt to realise these possibilities.

2 Features

2.1 High-quality Annotation Visualisation

BRAT is based on our previously released open-source STAV text annotation visualiser (Stenetorp et al., 2011b), which was designed to help users gain an understanding of complex annotations involving a large number of different semantic types, dense, partially overlapping text annotations, and non-projective sets of connections between annotations. Both tools share a vector graphics-based visualisation component, which provide scalable detail and rendering. BRAT integrates PDF and EPS image format export functionality to support use in e.g. figures in publications (Figure 1).
2.2 Intuitive Annotation Interface
We extended the capabilities of STAV by implementing support for annotation editing. This was done by adding functionality for recognising standard user interface gestures familiar from text editors, presentation software, and many other tools.

In BRAT, a span of text is marked for annotation simply by selecting it with the mouse by “dragging” or by double-clicking on a word. Similarly, annotations are linked by clicking with the mouse on one annotation and dragging a connection to the other (Figure 2).

BRAT is browser-based and built entirely using standard web technologies. It thus offers a familiar environment to annotators, and it is possible to start using BRAT simply by pointing a standards-compliant modern browser to an installation. There is thus no need to install or distribute any additional annotation software or to use browser plug-ins. The use of web standards also makes it possible for BRAT to uniquely identify any annotation using Uniform Resource Identifiers (URIs), which enables linking to individual annotations for discussions in e-mail, documents and on web pages, facilitating easy communication regarding annotations.

2.3 Versatile Annotation Support
BRAT is fully configurable and can be set up to support most text annotation tasks. The most basic annotation primitive identifies a text span and assigns it a type (or tag or label), marking for e.g. POS-tagged tokens, chunks or entity mentions (Figure 1 top). These base annotations can be connected by binary relations – either directed or undirected – which can be configured for e.g. simple relation extraction, or verb frame annotation (Figure 1 middle and bottom). n-ary associations of annotations are also supported, allowing the annotation of event structures such as those targeted in the MUC (Sundheim, 1996), ACE (Doddington et al., 2004), and BioNLP (Kim et al., 2011) Information Extraction (IE) tasks (Figure 2). Additional aspects of annotations can be marked using attributes, binary or multi-valued flags that can be added to other annotations. Finally, annotators can attach free-form text notes to any annotation.

In addition to information extraction tasks, these annotation primitives allow BRAT to be configured for use in various other tasks, such as chunking (Abney, 1991), Semantic Role Labeling (Gildea and Jurafsky, 2002; Carreras and Marquez, 2005), and dependency annotation (Nivre, 2003) (See Figure 1 for examples). Further, both the BRAT client and server implement full support for the Unicode standard, which allows the tool to support the annotation of text using e.g. Chinese or Devanagari characters. BRAT is distributed with examples from over 20 corpora for a variety of tasks, involving texts in seven different languages and including examples from corpora such as those introduced for the CoNLL shared tasks on language-independent named entity recognition (Tjong Kim Sang and De Meulder, 2003) and multilingual dependency parsing (Buchholz and Marsi, 2006).

BRAT also implements a fully configurable system for checking detailed constraints on annotation semantics, for example specifying that a TRANSFER event must take exactly one of each of GIVER, RECIPIENT and BENEFICIARY arguments, each of which must have one of the types PERSON, ORGANIZATION or GEO-POLITICAL ENTITY, as well as a MONEY argument of type...
Figure 3: Incomplete Transfer event indicated to the annotator

Money, and may optionally take a place argument of type location (LDC, 2005). Constraint checking is fully integrated into the annotation interface and feedback is immediate, with clear visual effects marking incomplete or erroneous annotations (Figure 3).

2.4 NLP Technology Integration

BRAT supports two standard approaches for integrating the results of fully automatic annotation tools into an annotation workflow: bulk annotation imports can be performed by format conversion tools distributed with BRAT for many standard formats (such as in-line and column-formatted BIO), and tools that provide standard web service interfaces can be configured to be invoked from the user interface.

However, human judgements cannot be replaced or based on a completely automatic analysis without some risk of introducing bias and reducing annotation quality. To address this issue, we have been studying ways to augment the annotation process with input from statistical and machine learning methods to support the annotation process while still involving human annotator judgement for each annotation.

As a specific realisation based on this approach, we have integrated a recently introduced machine learning-based semantic class disambiguation system capable of offering multiple outputs with probability estimates that was shown to be able to reduce ambiguity on average by over 75% while retaining the correct class in on average 99% of cases over six corpora (Stenetorp et al., 2011a). Section 4 presents an evaluation of the contribution of this component to annotator productivity.

2.5 Corpus Search Functionality

BRAT implements a comprehensive set of search functions, allowing users to search document collections for text span annotations, relations, event structures, or simply text, with a rich set of search options definable using a simple point-and-click interface (Figure 4). Additionally, search results can optionally be displayed using keyword-in-context concordancing and sorted for browsing using any aspect of the matched annotation (e.g. type, text, or context).

3 Implementation

BRAT is implemented using a client-server architecture with communication over HTTP using JavaScript Object Notation (JSON). The server is a RESTful web service (Fielding, 2000) and the tool can easily be extended or adapted to switch out the server or client. The client user interface is implemented using XHTML and Scalable Vector Graphics (SVG), with interactivity implemented using JavaScript with the jQuery library. The client communicates with the server using Asynchronous JavaScript and XML (AJAX), which permits asynchronous messaging.

BRAT uses a stateless server back-end implemented in Python and supports both the Common Gateway Interface (CGI) and FastCGI protocols, the latter allowing response times far below the 100 ms boundary for a “smooth” user experience without noticeable delay (Card et al., 1983). For server side annotation storage BRAT uses an easy-to-process file-based stand-off format that can be converted from or into other formats; there is no need to perform database import or export to interface with the data storage. The BRAT server in-
installation requires only a CGI-capable web server
and the set-up supports any number of annotators
who access the server using their browsers, on any
operating system, without separate installation.

Client-server communication is managed so
that all user edit operations are immediately sent
to the server, which consolidates them with the
stored data. There is no separate “save” operation
and thus a minimal risk of data loss, and as the
authoritative version of all annotations is always
maintained by the server, there is no chance of
conflicting annotations being made which would
need to be merged to produce an authoritative ver-
sion. The BRAT client-server architecture also
makes real-time collaboration possible: multiple
annotators can work on a single document simulta-
neously, seeing each others edits as they appear
in a document.

4 Case Studies

4.1 Annotation Projects

BRAT has been used throughout its development
during 2011 in the annotation of six different cor-
pora by four research groups in efforts that have
in total involved the creation of well-over 50,000
annotations in thousands of documents compris-
ing hundreds of thousands of words.

These projects include structured event an-
notation for the domain of cancer biology,
Japanese verb frame annotation, and gene-
mutation-phenotype relation annotation. One
prominent effort making use of BRAT is the
BioNLP Shared Task 2011,1 in which the tool was
used in the annotation of the EPI and ID main
task corpora (Pyysalo et al., 2012). These two
information extraction tasks involved the annota-
tion of entities, relations and events in the epige-
netics and infectious diseases subdomains of biol-
ogy. Figure 5 shows an illustration of shared task
annotations.

Many other annotation efforts using BRAT are
still ongoing. We refer the reader to the BRAT
website2 for further details on current and past an-
notation projects using BRAT.

4.2 Automatic Annotation Support

To estimate the contribution of the semantic class
disambiguation component to annotation produc-
tivity, we performed a small-scale experiment in-
volving an entity and process mention tagging
task. The annotation targets were of 54 dis-

1http://2011.bionlp-st.org

2http://brat.nlplab.org

3Document word count and expected annotation count,
were estimated from the output of NERsuite, a freely avail-
able CRF-based NER tagger: http://nersuite.nlplab.org

Table 1: Total annotation time, portion spent se-
selecting annotation type, and absolute improve-
ment for rapid mode.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Total</th>
<th>Type Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>45:28</td>
<td>13:49</td>
</tr>
<tr>
<td>Rapid</td>
<td>39:24</td>
<td>-6:04 09:35</td>
</tr>
</tbody>
</table>

Figure 5: Example annotation from the BioNLP Shared Task 2011 Epigenetics and Post-translational Modifications event extraction task.
Figure 6: Allocation of annotation time. Green signifies time spent on selecting annotation type and blue the remaining annotation time.

5 Related Work and Conclusions

We have introduced BRAT, an intuitive and user-friendly web-based annotation tool that aims to enhance annotator productivity by closely integrating NLP technology into the annotation process. BRAT has been and is being used for several ongoing annotation efforts at a number of academic institutions and has so far been used for the creation of well-over 50,000 annotations. We presented an experiment demonstrating that integrated machine learning technology can reduce the time for type selection by over 30% and overall annotation time by 15% for a multi-type entity mention annotation task.

The design and implementation of BRAT was informed by experience from several annotation tasks and research efforts spanning more than a decade. A variety of previously introduced annotation tools and approaches also served to guide our design decisions, including the fast annotation mode of Knowtator (Ogren, 2006), the search capabilities of the XConc tool (Kim et al., 2008), and the design of web-based systems such as MyMiner (Salgado et al., 2010), and GATE Teamware (Cunningham et al., 2011). Using machine learning to accelerate annotation by supporting human judgements is well documented in the literature for tasks such as entity annotation (Tsuruoka et al., 2008) and translation (Martínez-Gómez et al., 2011), efforts which served as inspiration for our own approach.

BRAT, along with conversion tools and extensive documentation, is freely available under the open-source MIT license from its homepage at http://brat.nlplab.org

Acknowledgements

The authors would like to thank early adopters of BRAT who have provided us with extensive feedback and feature suggestions. This work was supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan), the UK Biotechnology and Biological Sciences Research Council (BB-SRC) under project Automated Biological Event Extraction from the Literature for Drug Discovery (reference number: BB/G013160/1), and the Royal Swedish Academy of Sciences.
References


