Dain Kaplan, Ryu Iida, Kikuko Nishina, Takenobu Tokunaga

Slate – A Tool for Creating and Maintaining Annotated Corpora

Recent research trends of the last five years show that richly annotated corpora inspire novel research. These richly annotated corpora are indispensable for progressing research, but also more difficult to manage and maintain due to increasing complexity – what is needed is a way to manage the annotation project in its entirety. However, annotation project management has received little attention, with tools predominately focusing on single document annotation. Therefore, we define a list of corpus creation and management needs for annotation systems, and then introduce our multi-purpose annotation and management system `Slate` to address these needs through use of a case study, showing how project management is essential to creating good corpora.

1 Introduction

A look at recent research of the last five years related to language resources reveals an increasingly growing wealth of literature utilizing corpora in many languages on an ever wider variety of topics, including spoken corpora, fluency, gender and age differences, semantic analysis, parallel corpora, work targeting poor-resource language development, and so on.These works have been made possible by the existence and development of corpora.We can see from research trends that richly annotated corpora inspire novel research. Yet, they are also more difficult to manage and maintain due to their increasing complexity. Our goal should then be to encourage the creation and extension of such richly annotated corpora by making it as easy as possible, regardless of discipline.It is important for the annotation tool to allow corpus builders to experiment with a variety of ways for expressing their tasks, i.e., to be flexible in how the annotation schema is expressed, since it dictates the style of work.

We aim to facilitate this kind of corpus creation by developing an annotation tool called `Slate`[1] Kaplan et al. (2010)which is not only capable of adapting to many kinds of annotation tasks, but differs from many other existing tools in that it also covers management of the annotation process, which is increasingly important as the sizes of corpora continue to grow Davies (2009). In the following sections we outline the needs for corpus creation and management (Section 2), briefly introduce our annotation and management tool `Slate` (Section 3), and then demonstrate its utility with a real world case study in Section 4. We briefly follow this up with more examples of use (Section 5), explain other annotation tools and types (Section 6), and then conclude (Section 7).

[1] Segment and Link-based Annotation Tool, *Enhanced*: `http://www.cl.cs.titech.ac.jp/slate`

2 Corpus Creation and Management Needs

As those that utilize language resources continue to diversify and grow in number, so naturally will the uses of the corpora. A more richly annotated corpus provides more value than a collection of raw text; not to underplay the importance of such corpora, but a corpus with various annotations opens new windows for new research and new ways of analyzing this new data. Therefore the barrier to entry for creating new corpora, or extending existing corpora should be as small as possible. Further, as these resources grow and expand, their evolving complexity also necessitates a means for *managing this complexity*, or they themselves will become unmanageable and the resources quickly unusable.

Thus we propose a set of needs that will be indispensable for this task of maintaining and managing, and also creating new annotated corpora. Traditional single document-style annotation tools may be increasingly difficult to use if you wish to coordinate annotation by multiple annotators, insure the proper versions are used and verify/check on progress. An annotation system that fulfills these needs will allow management of creation of the annotation resource as a whole, rather than piecemeal on a per-document-only basis. With simple corpora, managing the project may not be a concern, but in larger projects – and as corpus-based techniques continue to grow and advance so do the sizes of the corpora Davies (2009) – it becomes a serious issue. In order to truly facilitate corpus creation we must therefore also address the annotation project as a whole.

Let us then specify what is needed from an annotation system. Dipper et al. (2004) proposed seven categories for what is needed from an annotation tool (diversity of data, multi-level annotation, diversity of annotation, simplicity, customizability, quality assurance and convertibility).They can, however, be thought of as targeting the *document-level*. The following list serves as a compliment to this list, operating at a more macro, annotation *project-level*. As our list is complimentary, we do not wish to reiterate what was already well said, and therefore skip needs related specifically to the act of annotation (such as appearance).

(1) **User and role management**. For some annotation projects, a simple, single document-oriented annotation tool may be sufficient (such as if the data is small and there is only one annotator), but with larger, more complex corpora, it may be advantageous to have a system to manage the overall project, including roles for an *annotator* (who performs the annotation work), and for an *administrator* (who configures the annotation schema and oversees the project).

(2) **Delegation and monitoring of work**. The system should allow for an administrator to assign/reassign work to annotators, and to monitor their progress. In addition, each annotator should be isolated – or "sandboxed" – from others, so that they are not biased by others' work.

(3) **Adaptability to new annotation tasks**. The system must be flexible enough to accommodate a new annotation task. If administrators cannot easily create a

new project and define the annotation requirements (i.e. the annotation schema) then the system will not be useful to them.

(4) **Adaptability within the current annotation task**. During the lifespan of a project, it often meets with many changes, especially during its initial phases. It is crucial that the system allow for the adjustment of the annotation schema.

(5) **Diffing and merging**. Creating a corpus often entails the comparison of data from multiple annotators on a single resource and reconciling any differences (i.e. *diffing and merging*), to finalize the corpus (create a gold standard). However, if multiple annotators' data is desired, this should also be possible.

(6) **Versioning of corpora**. A corpus is a product, and like any other product, may go through lifecycles.There must be a way to package the corpus, and in the case of fixes or amendments, a way to add them and repackage it. Without management of versioning, the "current state" of the annotation project is all that is known; for large projects it is important to identify milestones or to mark[2] a given state.This way changes will not unknowingly be included into a release.

(7) **Extensibility in terms of layering**. As corpora are continuing to grow in size, it is no wonder that they are also becoming more complex. We have seen recently the layering of corpora upon one another. Examples of this include the Penn Discourse Treebank Miltsakaki et al. (2004) or VP Ellipsis Bos and Spenader (2011) on top of the Penn Treebank Marcus et al. (1993), or the NAIST Text Corpus atop the Kyoto Text Corpus Iida et al. (2007). The system should allow for adding new layers upon previous ones

(8) **Extensibility in terms of tools**. With larger, more complex corpora, it may be difficult for annotators to do all annotation by hand. Having a mechanism for calling plugins to process data allows for various kinds of semi-automatic tagging (e.g. *annotation projection*) while data is within the system. In addition, plugins could allow for changing the way tags are selected from tag-sets, or automatically tagging areas found by the plugin to be similar to areas already manually annotated.

(9) **Extensibility in terms of importing/exporting**. The system should be able to call plugins to convert the data during import and export. This allows the system to be agnostic to the data format and support a variety of standardized formats.Including generated comments into the format may facilitate in human verification of the exported resource.

(10) **Support for multiple languages**. Research today is carried out in a variety of languages, and the system should support them.

[2] Known in source-code versioning control as "tagging".

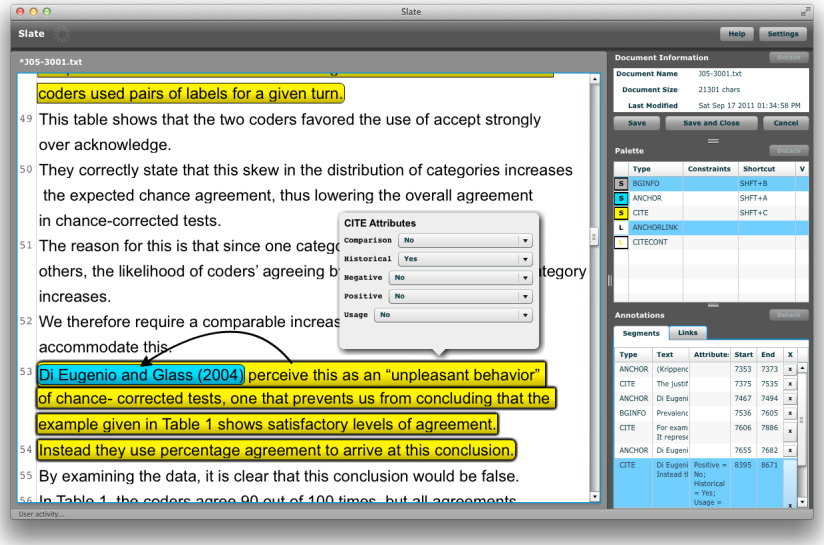

Figure 1: An example of the English Citation Corpus (see Section 5), showing an annotation of a citation's text pointing to an annotation of its citation anchor, and attribute panel with attributes related to the CITE tag.

3 `Slate` – A Web-based Annotation and Project Management Tool

We have developed `Slate` to try to meet the needs defined in Section 2.[3] One goal of the system is to be general enough to allow for a variety of textual annotation tasks. A conceptual framework is needed for representing annotations internally. ATLAS (Flexible and Extensible Architecture for Linguistic Annotation) Bird et al. (2000) is one possibility, very generalized and thus suitable for many tasks. Since our goal is textual annotation, we opted for using Segments and Links Takahashi and Inui (2006); Noguchi et al. (2008), a framework optimized specifically for this case. In this framework, annotations are stored in a stand-off[4] format.[5] Two concepts are used in annotation: *Segments* and *Links*. A *Segment* is defined as a span of text with a typed[6] label. Any textual annotation will be defined as a kind of Segment; using a stand-off

[3] At the time of writing, (5), (7), and (8) have not yet been fully realized.

[4] Meaning that the data is *not* directly embedded in the original text such as in XML, but separate to it; this supports partially overlapping annotations not possible in XML-like formats.

[5] `Slate` in fact uses a relational database for storing data while it is within the system.

[6] "Typed" as in "types" of Segments are defined, and those definitions are used to place those types as labels on spans of text.

format allows Segments to partially overlap. Similarly, a *Link* is a typed-relationship between two Segments. Any Segment or Link definition may further define attributes. For instance, a "Named Entity" Segment definition could have an attribute "Category" with possible values "Person", "Place", or "Thing".

Tag definitions are configured through a graphical user interface (GUI) within a Web-browser by the project administrator.Since `Slate` is both an annotation tool, and a tool for annotation project management, it allows project administrators to split up groups of documents to be annotated into smaller subsets, and to assign those subsets to annotators. `Slate` is Web-based, so annotators need not install any specialized software, only needing access to the internet. All of the annotators' changes are saved to the server, so their work can be monitored by the project administrator.

The current implementation is geared towards textual markup; but we have designed the system to be flexible, and it would be possible to replace the annotation canvas suited for textual markup with another suited for a different task, such as editing phrase-structure. (This is not impossible with the current annotation canvas, but definitely impractical.) `Slate` also supports an I/O plugin framework, so supporting an existing standard is only a matter of writing the interfacing code to read/write the format for importing/exporting.For more technical details, please refer to Kaplan et al. (2010).

The annotation screen is shown in Figures 1 and 2. For simplicity, we will use Figure 1 for explaining the annotator UI (as the example is in English). The left-side of the screen shows the main annotation panel; here is where the text to be annotated is shown. The annotator creates new annotations by selecting the type of Segment (tag) they wish to create from the middle panel on the right (where it says "Palette"), or by pressing that Segment's keyboard shortcut to select it, and then clicking and dragging the mouse over the text-span they wish to annotate, much like selecting text with a mouse in a text processor. To create a Link between two Segments, the annotator clicks and drags the mouse from the source to the destination Segment. The end result is an arrow (or undirected line if the Link definition is specified as such) between them.

The Palette in Figure 1 shows five definitions, three Segment types, and two Link types (the single letter in the leftmost column of the Palette panel shows the definition as a "S" for Segment, or "L" for Link). The Segment and Link definitions that appear here are determined by the schema that the administrator has defined for the project, as mentioned above. The main annotation panel shows a CITE Segment (yellow) with a Link (black arrow) to an ANCHOR Segment (teal). In the case that Segments overlap, Figure 1 shows that they become nested within one another.

If a Segment definition has attributes, the annotator may click on a selected Segment to open the attributes panel, where he/she may edit any values as specified by the Segment definition (compare Figure 1 with Figure 2). Links may also have attributes, and are editable in the same manner as for Segments.

The bottom right of the screen shows all created Segments and Links, letting the annotator easily navigate through the document. The top right of the screen has

buttons for saving or canceling changes, and basic information about the document, like when it was last saved.

4 A Case Study: Non-native Japanese Learner Composition Errors Corpus

The best way to see how `Slate` can facilitate creation of language resources without the need for developing a custom tool is to look at real-world examples. In this section we present a case study for an annotation task from Japanese Education in the Humanities, showing the problems encountered with their previous method of annotation, and how `Slate` helps to resolve many of these issues, increase the precision of the resulting data, and diversify the corpus as a whole. See Section 5 for a quick look at other examples.

4.1 Japanese Learner Composition Errors

During the process of learning a foreign language, one often makes mistakes along the way. A corpus of such common mistakes is indispensable to language instructors for snipping the bud of bad habits before they form; understanding learner tendencies for errors is crucial in designing lesson plans, general instruction, creating special learner dictionaries, devising tests, etc. As a compliment to modern day compositional support systems, such as `Natsume`[7], it is also possible to create compositional tools that automatically warn learners of potential mistakes as they are making them.

The underlying corpus texts are essays written by Japanese language learners on a series of topics. Currently there are over 261 essays (with 3,500 sentences) written by more than 164 learners.

4.2 Before `Slate`

At the time the corpus project began, there was no Japanese learner composition errors corpus in digital form, and therefore no specialized tool for its creation. The corpus creation started by using Microsoft Excel, which at the time, was an easily accessible tool at hand. However, using Excel proved to be a trying experience; some of the more major limitations encountered are outlined below.

Limitations on Data Integrity

I-1 Excel does not enforce formatting or style within cells; as a result an annotator's method for describing errors would very day-by-day based on his/her current disposition (e.g. using an arrow instead of parentheses, etc.).

I-2 Copy and paste made it difficult to keep consistent format, including the possibility for inserting accidental changes into the data.

I-3 The overall consistency of the data was poor due to I-1 and I-2, which made systematic analysis of the data difficult.

[7] `http://hinoki.ryu.titech.ac.jp/natsume/`

I-4 As a result of I-3, measuring inter-annotator agreement was also difficult.
I-5 The flow of text was difficult to grasp by looking at an Excel cell, often meaning the entire sentence could not be seen at once, increasingly likelihood for errors in annotation.
I-6 Many columns made the possibility for entering data in the wrong cell high.

Limitations on data diversity

D-1 Non-contiguous learner errors (such as a grammatical construction begun at the start of a sentence, but not finished until the end) where difficult to describe.
D-2 Annotating a sentence containing multiple leaner errors was not possible.
D-3 Relations between sentences, where a mistake is started or continued, were not possible.

4.3 With `Slate`

A screenshot of the annotation screen using `Slate` for this project is shown in Figure 2. This project has only one Segment definition, shown in purple, which marks composition errors. The attributes panel has a number of values related to the type of error, its correction, etc.

By converting the existing data, the project was able to migrate existing annotations into `Slate`. Once data is imported, the underlying text that is annotated is not changed, as a result this remedies miscopied data issues (I-2). The underlying text is never changed because all annotations are stored separately from the text; this allows multiple annotators to annotate the same document without being interfered or biased by one another. This makes inter-annotator agreement I-4 possible in the sense that now multiple annotators can work on the same document.

As explained briefly in Section 3, `Slate` does not predefine annotation tag-sets, but instead lets an administrator define them. Once the tag-set(s) are defined, all annotation is constrained to them. This means the user's input is limited, preventing issues where no system was constraining them from freely formatting text (I-1).

Because `Slate` is a visual tool, it eliminates many of the limitations prevalent in the previous annotation method. Non-contiguous errors (D-1) and errors that span multiple sentences (D-3) are possible to annotate with the ability to use *Links* to connect the *Segments*. Multiple annotations within the same sentence (D-2) is also now possible (see Figure 1). As a visual tool, it also means that the annotator is able to see the flow of text (I-5). Since users are directly interacting with what they are annotating, there is higher accuracy of annotation (I-6).

Further, now as the team of annotators grows to include members spread across the globe, there is no issue in sending and receiving work, as it is all managed online by `Slate`.

Most importantly, because `Slate` outputs a specified format with all the annotations, analysis (I-3) is now more straightforward.

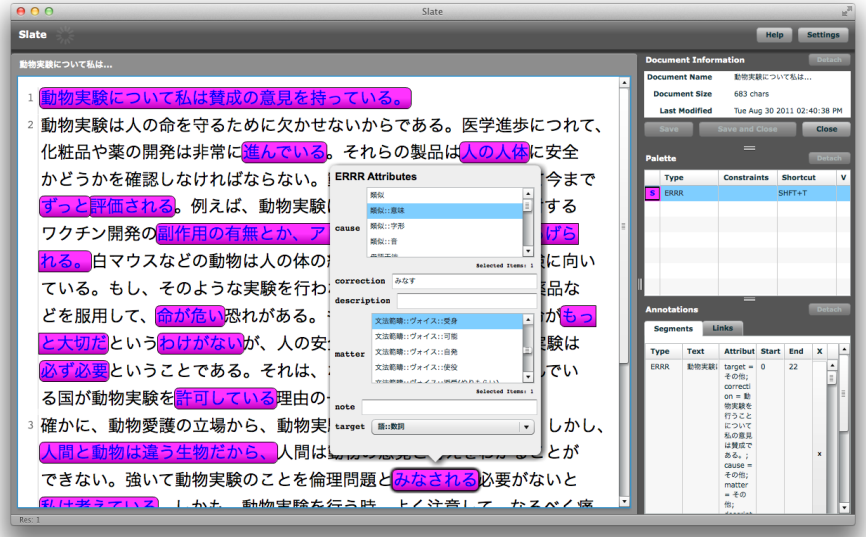

Figure 2: An example of composition errors annotated using Slate, with the panel showing editing attributes for a given Segment.

5 Other Corpus Creation Projects using `Slate`

Though space prevents a more detailed look, a few other projects utilizing `Slate` are briefly described below.

- **Japanese Blog Anaphora Corpus** – A corpus composed of blog texts, with annotations for the expressions that hint at the anaphoric relations (such as "this" or "that"), and the actual known referent of each, with explicit references between them. There are attributes on the annotations to classify them into various categories, such as between explicit and implicit anaphors. The anaphors and their referents are realized using Segments, and coreference relations using Links. `Slate` allows for free definition of attributes on annotations, so various kinds of simultaneous classifications are possible simply by defining multiple attributes for each annotation type.

- **Japanese Q&A Website Corpus** – A corpus composed of question threads[8] posted on Q&A Web sites, containing annotations marking the parts in the text

[8] I.e. the hierarchical flow of posts made by users asking a question and receiving answers, with possible additional posts by the initial asker with clarification about the problem along the way.

that indicate background about the problem setting, parts of the text that describe the problem itself, and parts that are directly answered by other users (the answers that directly correspond to parts of the question(s) asked). Two potential uses for such a corpus include improving the precision of Q&A site searching by knowing more semantically about the questions and their answers, and which parts of the text reflect each (e.g. limiting your search to within problem setting descriptions only). `Slate` allows for annotating these various parts of text, and for specifying relations between them quite freely.

- **English Citation Corpus** – A corpus of research papers with the citations in each annotated, including directly related background information that pertains to the citation, and various classifications for each citation, such as marking it as a comparison/contrast, refuting a claim, etc. In the case that the citations are non-contiguous Kaplan et al. (2009), they are connected to each other using *Links*. `Slate` allows for the various classifications using attributes on the annotations, and lets you link annotations together to make creation of this corpus possible. A sample of the annotation screen in `Slate` is shown in Figure 1.

6 Comparison with similar / related tools

In this paper we have shown how `Slate` can be a viable solution to many annotation tasks, especially for enabling those otherwise unable, to create richer, more precise language resources. It has been designed with corpus management functionality, an area overlooked by most other tools of this kind. There are, however, many other tools available that satisfy a wide range of different tasks to varying degrees. Major differences are outlined below.

In essence three kinds of annotation tools exist, ranging from specialized to general: project-specific tools, task-oriented tools, and generalized or multi-purpose tools. In addition to these three kinds of tools, there are tools that run as desktop applications, and those that are Web-based; further, there are tools that are single document-centric, or that are concerned only with the current document being opened by the user, and those that attempt to help the user manage the annotation task as a whole (by grouping files, providing comprehensive search, etc.). In general, non-Web-based tools have several drawbacks related to data management and data consistency, such as maintaining consistent versions of the files on multiple computers, transferring the files to the annotators' computers (including potential licensing issues involved as well as file sizes), redistributing the files to different annotators in the case of reassignment, and remembering which files have been distributed to which annotators.

Project-specific tools are very specialized. Usually they support a single corpus format, so they can only work with a single project. Such tools are good in one sense because they meet the specific needs of an annotation project, but require resources for their development/maintenance, and may also not be compatible with other corpora, and thus not support layering (preventing the development of richer,

more complex resources). Some more recent examples of project-specific tools include a Korean treebank Park et al. (2006), lexical chains Stührenberg et al. (2007), image annotation Russell et al. (2005), and sentiment tagging Francisco et al. (2011). Of these, some are Web-based Stührenberg et al. (2007); Russell et al. (2005); Francisco et al. (2011). **Task-specific tools** are more generalized than project-specific tools, but still often exist because of a certain project and its descendants, such as a treebank annotator reading and writing files in Penn Treebank format. Some recent examples include: text alignment to speech Draxler (2005), word-alignment Madnani and Hwa (2004), syntactic annotation Noguchi et al. (2006), and frame-set annotation Choi et al. (2010b,a). Some are desktop applications Madnani and Hwa (2004); Choi et al. (2010b,a), and some Web-based Draxler (2005); Noguchi et al. (2006).

Multi-purpose tools are capable of adapting to a variety of tasks. The most famous of these may be MMAX2 Müller and Strube (2006), and also the NITE XML Toolkit Carletta et al. (2005). Glozz Widlöcher and Mathet (2009) and Word Freak Morton and LaCivita (2003) have plugin architectures and nice interfaces, comparable in many ways to `Slate`, but lack project management and as easy to use schema definition UI. For multimedia annotation ELAN is a popular choice Auer et al. (2010). There are other tools in various states of activity/dormancy Asan and Orăsan (2003); Cunningham et al. (2002); Dennis et al. (2003); Mueller and Strube (2001); Callisto (2002).

The tool most similar to `Slate` is probably Djangology Apostolova et al. (2010), which is a Web-based tool supporting project management. Its UI however, does not allow for visual creation of links/relations between segments like `Slate` does. The other multi-purpose tools are not Web-based, nor do they support management of the annotation project.

7 Conclusion

There is of course no silver bullet to annotating corpora; their creation comes at the cost of substantial time and labor. However, because of this there is all the more reason to make sure that the time and effort spent are only spent where they are necessary, and are kept to a minimum everywhere else. To this end, we have tried to make sure `Slate`'s ability to manage corpus creation and to create a flexible annotation schema allows for different styles of working. We have also designed the system with extensibility in mind, so that future developers may more easily adapt it to new tasks and standards.As the various project examples have shown, by allowing freedom in how the schema is defined `Slate` can adapt to a number of textual annotation tasks. Further, it allows annotators to get started quickly since no setup is required, and provides project administrators instant access to the annotators' work.

`Slate` is still under active development as we work towards realizing all the needs outlined in Section 2. The project website is: `http://www.cl.cs.titech.ac.jp/slate/`.

Acknowledgements

This work has been supported in part by the Grant-in-Aid for Scientific Research Priority Area Program "Japanese Corpus" (2006 – 2010), sponsored by MEXT (Ministry of Education, Culture, Sports, Science and Technology – Japan).

References

Apostolova, E., Neilan, S., An, G., Tomuro, N., and Lytinen, S. (2010). Djangology: A light-weight Web-based tool for distributed collaborative text annotation. In *Proc. of the 7th conference on International Language Resources and Evaluation (LREC'10)*, pages 3499–3505.

Asan, C. O. and Orăsan, C. (2003). Palinka: A highly customisable tool for discourse annotation. In *Proc. of the 4 th SIGdial Workshop on Discourse and Dialogue, ACL '03*, pages 39–43.

Auer, E., Russel, A., Sloetjes, H., Wittenburg, P., Schreer, O., Masnieri, S., Schneider, D., and Tschöpel, S. (2010). ELAN as flexible annotation framework for sound and image processing detectors. In *Proc. of the 7th conference on International Language Resources and Evaluation (LREC'10)*, pages 890–893.

Bird, S., Day, D., Garofolo, J. S., Henderson, J., Laprun, C., and Liberman, M. (2000). ATLAS: A flexible and extensible architecture for linguistic annotation. *CoRR*, cs.CL/0007022.

Bos, J. and Spenader, J. (2011). An annotated corpus for the analysis of VP ellipsis. *Language Resources and Evaluation*, 45(2):1–32.

Callisto (2002). `http://callisto.mitre.org`.

Carletta, J., Evert, S., Heid, U., and Kilgour, J. (2005). The NITE XML Toolkit: Data model and query language. *Language Resources and Evaluation*, 39(4):313–334.

Choi, J. D., Bonial, C., and Palmer, M. (2010a). PropBank FrameSet annotation guidelines using a dedicated editor, Cornerstone. In *Proc. of the 7th conference on International Language Resources and Evaluation (LREC'10)*, pages 3650–3653.

Choi, J. D., Bonial, C., and Palmer, M. (2010b). PropBank instance annotation guidelines using a dedicated editor, Jubilee. In *Proc. of the 7th conference on International Language Resources and Evaluation (LREC'10)*, pages 1871–1875.

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proc. of the 40th Annual Meeting of the ACL*, pages 168–175.

Davies, M. (2009). Contemporary American English (1990-2008+). *International Journal of Corpus Linguistics*, 14(2):159–190.

Dennis, G., Sherman, B., Hosack, D., Yang, J., Gao, W., Lane, H. C., and Lempicki, R. (2003). DAVID: Database for annotation, visualization, and integrated discovery. *Genome Biology*, 4(5).

Dipper, S., Gotze, M., and Stede, M. (2004). Simple annotation tools for complex annotation tasks: An evaluation. In *Proc. of the LREC Workshop on XML-based Richly Annotated Corpora*, pages 54–62.

Draxler, C. (2005). Webtranscribe – an extensible Web-based speech annotation framework. In *Proc. of the 8th Text, Speech and Dialog conference (TSD'05)*, pages 61–68.

Francisco, V., Hervás, R., Peinado, F., and Gervás, P. (2011). EmoTales: Creating a corpus of folk tales with emotional annotations. *Language Resources and Evaluation*, 45.

Iida, R., Komachi, M., Inui, K., and Matsumoto, Y. (2007). Annotating a Japanese text corpus with predicate-argument and coreference relations. In *Proc. of the Linguistic Annotation Workshop*, pages 132–139.

Kaplan, D., Iida, R., and Tokunaga, T. (2009). Automatic extraction of citation contexts for research paper summarization: A coreference-chain based approach. In *Proc. of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries*, NLPIR4DL '09, pages 88–95.

Kaplan, D., Iida, R., and Tokunaga, T. (2010). Annotation process management revisited. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 3654–l3661.

Madnani, N. and Hwa, R. (2004). The UMIACS word alignment interface. `http://www.umiacs.umd.edu/~nmadnani/alignment`.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Miltsakaki, E., Prasad, R., Joshi, A., and Webber, B. (2004). The Penn Discourse Treebank. In *Proc. of the 4th conference on International Language Resources and Evaluation (LREC'04)*.

Morton, T. and LaCivita, J. (2003). WordFreak: an open tool for linguistic annotation. In *Proc. of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Demonstrations - Volume 4*, NAACL-Demonstrations '03, pages 17–18.

Mueller, C. and Strube, M. (2001). MMAX: A tool for the annotation of multi-modal corpora. In *Proc. of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 45–50.

Müller, C. and Strube, M. (2006). Multi-level annotation of linguistic data with MMAX2. In *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pages 197–214.

Noguchi, M., Ichikawa, H., Hashimoto, T., and Tokunaga, T. (2006). A new approach to syntactic annotation. In *Proc. of the 5th conference on International Language Resources and Evaluation (LREC'06)*, pages 1412–1417.

Noguchi, M., Miyoshi, K., Tokunaga, T., Iida, R., Komachi, M., and Inui, K. (2008). Multiple purpose annotation using SLAT – segment and link-based annotation tool. *Proc. of 2nd Linguistic Annotation Workshop*, pages 61–64.

Park, S.-Y., Song, Y.-I., and Rim, H.-C. (2006). A segment-based annotation tool for Korean treebanks with minimal human intervention. *Language Resources and Evaluation*, 40(3-4):281–289.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2005). Labelme: A database and web-based tool for image annotation. Technical report, Massachusetts Institute of Technology.

Stührenberg, M., Goecke, D., Diewald, N., Mehler, A., and Cramer, I. (2007). Web-based annotation of anaphoric relations and lexical chains. In *Proc. of the Linguistic Annotation Workshop*, pages 140–147.

Takahashi, T. and Inui, K. (2006). A multi-purpose corpus annotation tool: Tagrin. *Proc. of the 12th Annual Conference on Natural Language Processing*, pages 228–231. (in Japanese).

Widlöcher, A. and Mathet, Y. (2009). La plate-forme Glozz: environnement d'annotation et d'exploration de corpus. In *Actes de TALN 2009*, Senlis, France. (in French).