Technical Infrastructure at Linguistic Data Consortium: Software and Hardware Resources for Linguistic Data Creation

Kazuaki Maeda, Haejoong Lee, Stephen Grimes, Jonathan Wright
Robert Parker, David Lee, Andrea Mazzucchi

3600 Market St., Suite 810, Philadelphia, PA, 19104, USA
Linguistic Data Consortium, University of Pennsylvania
{maeda, haejoong, sgrimes, jdwright, parkerrl, david4, amazzu}@ldc.upenn.edu

Abstract

Linguistic Data Consortium (LDC) at the University of Pennsylvania has participated as a data provider in a variety of government-sponsored programs that support development of Human Language Technologies. As the number of projects increases, the quantity and variety of the data LDC produces have increased dramatically in recent years. In this paper, we describe the technical infrastructure, both hardware and software, that LDC has built to support these complex, large-scale linguistic data creation efforts at LDC. As it would not be possible to cover all aspects of LDC's technical infrastructure in one paper, this paper focuses on recent development. We also report on our plans for making our custom-built software resources available to the community as open source software, and introduce an initiative to collaborate with software developers outside LDC. We hope that our approaches and software resources will be useful to the community members who take on similar challenges.

1. Introduction

Linguistic Data Consortium (LDC) at the University of Pennsylvania has participated as a data provider in a variety of government-sponsored programs that support development of Human Language Technologies. These programs include DARPA GALE (Global Autonomous Language Exploitation), DARPA MADCAT (Multilingual Automatic Document Classification Analysis and Translation), DARPA Machine Reading, NIST TAC (Text Analysis Conference), ACE (Automatic Content Extraction), LCTL (Less Commonly Taught Languages), NIST Open MT (Machine Translation) Evaluation, NIST RT (Rich Transcription) Evaluation, NIST SRE (Speaker Recognition Evaluation) and NIST LRE (Language Recognition Evaluation). As the number of projects increases, the quantity and variety of the data LDC produces have increased dramatically in recent years. For the DARPA GALE program, for example, LDC has collected and distributed a total of more than 15,000 hours of high quality broadcast audio data in Arabic, Chinese and English, and distributed it to the participating sites.[1] The total storage needs for various data within LDC has exceeded 100 Terabytes as of this writing. In addition to the quantity, the variety of data has also increased significantly. The types of linguistic data LDC collects and creates include newswire articles, weblog and newsgroup postings, wiki entries, video and audio recordings of broadcast speech and spontaneous speech, images of handwritten and printed text, transcripts, parallel and comparable text, word-alignment parallel text, treebank annotations, and various text and speech annotations.

This paper describes the technical infrastructure, both hardware and software, that we built in order to support these complex, large-scale linguistic data creation effort at LDC. As it would not be possible to cover all aspects of LDC's technical infrastructure in one paper, this paper focuses on recent development.

[1]Some of the data is currently in the pipeline for general publications for LDC members.

Later in this paper, we will also report on our plan for making our custom-built software resources available to the communities outside LDC. While our software is typically developed to meet the needs of projects with a tight timeline and evolving requirements, we are committed to contributing to the human language technology community, by releasing our custom-built software as open source software whenever possible. We will also describe an initiative for collaborating with software developers outside LDC.

2. Hardware Infrastructure

2.1. Data Storage

LDC's data storage needs have increasingly required extreme flexibility and expansion capability. However, some elements of existing infrastructure such as legacy hardware and systems, vendor-dependence of the main storage system and a slow tape-based backup system are not conducive to meeting these goals. Our initial data storage system, based on a family of NAS (Network Attached Storage) nodes, has reached a performance limit typical of this storage method. Moreover, the effort to scale up storage according to the increasing needs of the project has become a more time-consuming process with discontinuous steps.

The technical solution LDC adopted recently, which will gradually replace the current storage infrastructure, is based on a deep level of virtualization of the storage, providing an in-house SaaS (Storage as a Service) framework. This new framework promises thin provisioning – that is, gradually increasing the storage size on an as-needed basis – and tiered storage – that is, using various types of storage media, including our preexisting hardware, in a transparent way to the users.

2.2. Data Backups

In order to improve the efficiency of data backups, LDC divides data storage into two types: static data storage and dynamic data storage. The static data storage holds data that typically will not be modified, such as video and audio source data. The dynamic data storage holds all other kinds of data. The static storage is backed up using a tape backup system only when updates are made. The dynamic storage is backed up by both data replication and daily incremental tape backups.

3. Software Infrastructure

3.1. Overview

When there is a need for software tool for a project, LDC's project managers and technical staff evaluate existing solutions and the cost of building new software tools. While third-party solutions are adequate in some cases, there are many cases where custom-built software tools are necessary or more desirable in order to meet the project specifications and timeline. This section focuses of the software infrastructure designed and implemented by the LDC's technical staff.

3.2. Web Text Data Collection

Web text is a valuable resource for human language technology development. Initially at LDC, data collection efforts of web text, such as weblog and news group posts, were done individually for each project, resulting in sometimes overlapping effort, incompatible formats, duplicated contents, and so on. There was a clear need for a centralized approach for the collection of web text. To address these issues, LDC has developed an extensive software framework. The *WebCol* framework, which is implemented in Python, provides a variety of services for the collection of web text. This framework consists of the data storage module, the data-tracking database module, and the web module. The data storage module interfaces with a physical storage system, and implements functionalities for storing and retrieving data from the physical storage system. The database module interfaces with the data tracking database in MySQL and provides an API for accessing the tracking database. The web module provides functions for accessing websites and downloading data.
For each new weblog or a newsgroup source, the user of the framework supplies a harvester and a formatter, which are written for the particular source. The rest of the collection is automatic. The harvesting occurs on a dedicated server at the scheduled intervals, and each harvested post is given a document ID, formatted, and stored in the data storage.
As of this writing, LDC has harvested and processed more than 10 million documents of Arabic, Chinese and English web text data.

3.3. Parallel Text Harvesting and Alignment

One of the resources in highest demand at LDC in recent years is parallel text. While manual translations provide excellent resources for the development of human language technologies, high resource costs limit the amount LDC can produce. An alternative approach is to identify, harvest and use existing parallel or comparable text. Previously, we reported our approach below for creating such harvested parallel text (Maeda et al., 2008b). This process includes the following steps:

- Automatic harvesting of data
- Metadata extraction and formatting
- Document mapping using the BITS system
- Sentence-level alignment using the Champollion toolkit

This approach is now incorporated into a regular data production pipeline on certain data sets, such as multilingual newswire sources. In the past few years, LDC has created over 82,000 document pairs of automatically aligned parallel text in Arabic and English and 67,000 document pairs of potential parallel text in Mandarin Chinese and English.

3.4. Transcription Harvesting and Alignment

Transcripts provided by broadcast networks are another attractive source of existing harvestable resources. Many broadcast sources, such as CNN and MSNBC, provide transcripts for their programs. In most cases, such transcripts are simple text transcripts and are not time-aligned to the media files.
After appropriate intellectual property rights negotiations, LDC harvests online transcripts. For the alignment of English transcripts to the corresponding audio files, LDC uses a forced aligner tool it has developed using the Hidden Markov Toolkit (HTK) (Young et al., 2000). This tool was designed to handle regions of speech that are not transcribed and noises in the audio files such as commercial breaks.
This approach worked well for time-aligning transcripts with English broadcast news and broadcast conversation audio data. For the latest GALE distillation task, LDC created over 200 hours of English transcripts automatically aligned to audio data using this approach.

3.5. Handwriting Data Collection

The goal of the DARPA MADCAT program is to create human language technologies to convert foreign language text images into English transcripts. LDC creates the training and evaluation reference data for the MADCAT program, consisting of scanned image files of handwritten text, zone annotation of tokens and lines, Arabic transcripts and their English translations.
One of our approaches for data creation involves creation of handwritten text from existing Arabic newswire and web text and their translations. We assign "scribes" to copy the text onto paper, under a variety of defined writing conditions, such as paper type (lined or unlined), writing instrument, and writing speed. LDC has developed a web-based workflow management system that supports this data collection. This system has the following functionalities:

- Register participating scribes and maintain personal information about the scribes
- Make assignments for scribes and generate printable assignments in pdf format and generate specifications for writing conditions.
- Maintain a tracking database of scribes, data, and data assignments

This system shares many implementation details and design philosophies with the LDC MT Post Editing Workflow system (Maeda et al., 2008a). As with the MT Post Editing Workflow system, the implementation of the system is done in PHP and CodeIgnitor with a back-end database in MySQL.

3.6. Data Exploration

The exploration of text data for documents containing certain types of information is a common prerequisite for annotation tasks. LDC maintains a large collection of text data, such as newswire articles, weblog posts, newsgroup posts, manual transcripts, closed captions, ASR output, manual translations, harvested translations, and MT output. A data exploration tool should allow the annotator to perform searches on text corpora and record information about the found documents.

The ACE Data Exploration toolkit (Strassel et al., 2008) is one such tool LDC has created. More recently, LDC created a similar tool, using the same technologies, for the NIST TAC KBP (Knowledge Base Population) track (McNamee et al., 2010). LDC selected candidate evaluation entities from a snapshot of the English Wikipedia and identified documents containing those entities from LDC's newswire, web text and transcription collection (Simpson et al., 2010).

3.7. Data Triage

Data triage is a task in which a human makes a small number of judgments, often a yes/no judgment, on text, audio, video or image files, in order to select a set of data well-suited for downstream tasks.

LDC has a simple tool named ScanFiles that allows annotators to make judgments on a small number of files. LDC also has ACK, the annotation collection kit, which is a user-definable, web-based toolkit (Maeda et al., 2008a).

These tools are suitable for cases where the number of annotators and data sets are relatively small. For larger tasks, we needed a tool that automatically manages a large number of assignments, annotators, and that allows access from outside the LDC network.

For the DARPA MADCAT program, LDC has developed and incorporated a data triage functionality in the workflow management system described in Section 3.5.. LDC is currently undertaking the creation of a more flexible, user-configurable, tool incorporating the features of these existing tools.

3.8. Word Alignment

Parallel text manually aligned at the word-level is an expensive yet invaluable resource for the development of MT technologies. LDC has created word-aligned Arabic-English and Chinese-English parallel text resources for the DARPA GALE program. During the first few phases of GALE, LDC used a third-party word alignment tool, which was adequate for the initial purposes. As the LDC annotation teams gained more experience with the word alignment annotation tasks, and as the task specifications evolved to address the requirements of the program, the need for a new tool became apparent. In order to address the feature requests and feedback from the LDC annotation teams, LDC's technical staff has developed a new word alignment tool. The new tool is designed to maximize annotation efficiency and accuracy as well as enabling new annotation tasks. New features include the following: 1) a table-display of links, tokens, and link types, 2) a full display of both source and translation text for better understanding of the context, 3) support of new annotation tasks, such as the use of linguistic tags (Li et al., 2010), 4) visual indicators on the token alignment panel and the full-text panels to avoid missed links, and 5) built-in support for QC processes.

The implementation of this tool is based on the same set of technologies as other recent stand-alone annotation tools developed at LDC: Python, Qt and PyQt. Figure 1 shows a screen shot of this tool used in a Chinese-English word alignment task.

3.9. Machine Reading Annotation

The recently begun DARPA Machine Reading (MR) Program (Strassel et al., 2010) requires LDC to carry out a novel program of text annotation, and new software was developed to support the task. The demands of the MR Program are such that the developed software would likely be useful to other annotation projects. The MR Program is structured around a series of "Use Cases", which are composed of small domain-specific sets of annotation categories, supported by an ontology for formal representation, source text and annotations on that source text which demonstrate the mapping of ontological elements. Each Use Case is designed to promote and test a particular functionality in the NLP software that research sites develop. Because each new Use Case may be based in a new domain in terms of ontology and corpus, the MR Program actually involves a series of independent annotation projects, each developed and executed relatively quickly. In a superficial sense, this requires a series of custom annotation tools. The approach taken to developing the supporting software was to distinctly separate the general annotation functionality from the task specific design elements. The primary code base is written in Python, using Qt and PyQt, as is common with LDC projects. Much of the design for each annotation task, e.g. annotation categories and GUI layout, is defined in a separate file in a domain specific language, rather than in Python. The creation of this domain specific language for the configuration files not only facilitates rapid prototyping by simplifying the mutable portions of "code" but also allows non-programmers to manipulate the design without specific knowledge of Python.

Given this highly customizable design, the annotation software allows the user to define annotation categories, lay them out in the GUI as desired, and connect text extents to instances of those categories. The annotations are realized as underlines in the source text, and the underlines can be stacked to allow for overlapping annotations. The color of underlines and annotation categories can be customized in the configuration file to supplement the visual representation. A coreferencing feature is also available. Code in the configuration file will automatically categorize annotations as desired, and then within each category, the annotator can drag and drop the annotations to indicate that they refer to the same entity. The configuration file also allows for user-defined constraints that when violated produce run-time warnings with the GUI. The annotator can then navigate to the problematic annotations and modify them before saving.

MR annotation is somewhat peculiar in the presence of the ontology, the potential for divergence between the ontology and the annotation categories, and the implementation of the ontology in RDF (Resource Description Format). Because the ontology is expressed in RDF, and more importantly because the ontology has its own design requirements, the ontological categories are not always convenient annotation categories. This is not by design, but simply due to the practicalities of human annotation. The MR annotation software was designed to manipulate two sets of categories, each with their own native format. The annotator works with user-defined categories that may or may not resemble the ontological categories required by the program, and the annotation software handles the mapping between the two representations. Furthermore, the software performs certain data manipulation functions aside from annotation that can be performed from the command line without opening the GUI. For example, error reports or histograms can be generated based on annotated data.

Figure 2 shows a screen shot of this tool.

4. Roadmap for Software Resource Sharing

4.1. Distribution of Existing Software Tools

In addition to creating data, annotation task specifications and data formats, we believe that our custom-built software resources would benefit the community greatly. Many of software resources described or mentioned in this paper are already available to the community as open source software – some as part of a larger software infrastructure (e.g., AGTK) and some as independent releases. Other software projects are in the process of preparing for a public release, and some have been given to users on a request-only basis.[2]

However, the process has not always been done consistently. One of the primary reasons for this was the lack of a standard practice at LDC to convert software designed for projects into open source software.

To address this issue, LDC is currently reviewing and standardizing the process of releasing software resources created for projects to the public. The review points include a best practice for naming software packages, selecting an open source license, selecting a distribution method, documentation, and user support. The standardization of software distribution will allow us to release our software as open source software in a more consistent manner.

4.2. Speech Annotation Toolkit (SATK)

Another aspect of software resource creation and dissemination we are planning to work on is collaboration with software developers outside LDC. The Speech Annotation Tool (SATK) is one such initiative. SATK is planned to be a repository of software components that are useful in creating various software tools for speech annotation and analysis tasks.

Examples of software components to be included in SATK include waveform display, spectrogram display, annotation table-display transcription display, metadata annotation display, pitch tracking display, formant tracking display, file format conversion, data import/export for interoperability with other software, such as Transcriber, Praat, R, MATLAB/Octave, HTK and NLTK. Some of such components have already been designed, created and used in AGTK TableTrans, LDC XTrans and LDC MDE tool (Maeda et al., 2008a; Maeda et al., 2006).

An initial goal of SATK is to create a collection of software components that allows software developers to create various applications to be used not only in language resource creation, but also in other fields, such as linguistic research and education and second language education.

[2]For more information, visit `http://www.ldc.upenn.edu/tools`.

5. Summary

LDC's technical staff has created a software and hardware technical infrastructure for supporting extensive and expanding linguistic data resource creation effort at LDC. Most of the software resources introduced in this paper are currently or will be available as open source software to the community. We hope that our approaches and tools will be useful to the community members who take on similar challenges.

Acknowledgement

This work was supported in part by the Defense Advanced Research Projects Agency, Program Grants No. HR0011-06-1-0003 and HR0011-08-0004. The content of this paper does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

6. References

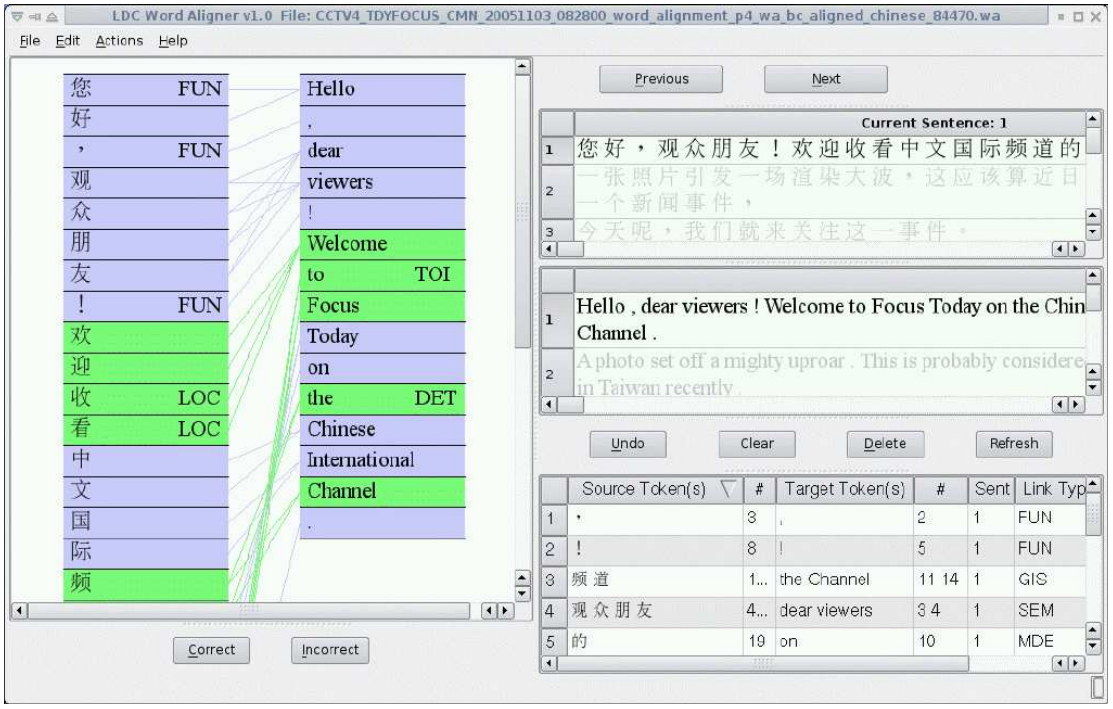
Xuansong Li, Niyu Ge, Stephen Grimes, Stephanie M. Strassel, and Kazuaki Maeda. 2010. Enriching word alignment with linguistic tags. In *Proceedings of LREC 2010*.

Kazuaki Maeda, Haejoong Lee, Julie Medero, and Stephanie Strassel. 2006. A new phase in annotation tool development at the Linguistic Data Consortium: The evolution of the Annotation Graph Toolkit. In *Proceedings of LREC 2006*.

Kazuaki Maeda, Haejoong Lee, Shawn Medero, Julie Medero, Robert Parker, and Stephanie Strassel. 2008a. Annotation tool development for large-scale corpus creation projects at the Linguistic Data Consortium. In *Proceedings of LREC 2008*.

Kazuaki Maeda, Xiaoyi Ma, and Stephanie Strassel. 2008b. Creating sentence-aligned parallel text corpora from a large archive of potential parallel text using BITS and Champollion. In *Proceedings of LREC 2008*.

Paul McNamee, Hoa Trang Dang, Heather Simpson, Patrick Schone, and Stephanie M. Strassel. 2010. An evaluation of technologies for knowledge base population. In *Proceedings of LREC 2010*.


LDC Word Aligner v1.0 File: CCTV4_TDYFOCUS_CMN_20051103_082800_word_alignment_p4_wa_bc_aligned_chinese_84470.wa
File Edit Actions Help
Previous
Next
Undo
Clear
Delete
Refresh
Correct
Incorrect

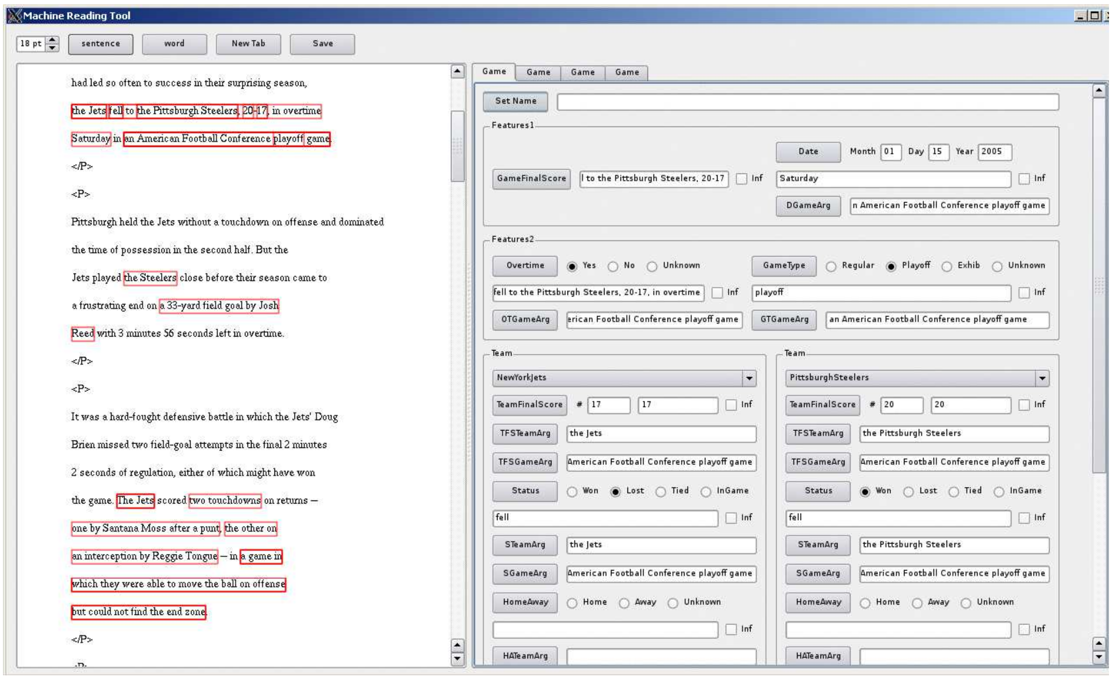
您 FUN	Hello
好	,
， FUN	dear
观	viewers
众	!
朋	Welcome
友	to TOI
！ FUN	Focus
欢	Today
迎	on
收 LOC	the DET
看 LOC	Chinese
中	International
文	Channel
国	.
际	
频	

	Current Sentence: 1
1	您好，观众朋友！欢迎收看中文国际频道的
2	一张照片引发一场渲染大波，这应该算近日一个新闻事件，
3	今天呢，我们就来关注这一事件。

1	Hello , dear viewers ! Welcome to Focus Today on the Chin Channel .
2	A photo set off a mighty uproar . This is probably considere in Taiwan recently .

	Source Token(s)	#	Target Token(s)	#	Sent	Link Typ
1	，	3	,	2	1	FUN
2	！	8	!	5	1	FUN
3	频 道	1...	the Channel	11 14	1	GIS
4	观 众 朋 友	4...	dear viewers	3 4	1	SEM
5	的	19	on	10	1	MDE

Figure 1: LDC Word Alignment Tool



had led so often to success in their surprising season,

the Jets fell to the Pittsburgh Steelers, 20-17, in overtime

Saturday in an American Football Conference playoff game.

</P>

<P>

Pittsburgh held the Jets without a touchdown on offense and dominated

the time of possession in the second half. But the

Jets played the Steelers close before their season came to

a frustrating end on a 33-yard field goal by Josh

Reed with 3 minutes 56 seconds left in overtime.

</P>

<P>

It was a hard-fought defensive battle in which the Jets' Doug

Brien missed two field-goal attempts in the final 2 minutes

2 seconds of regulation, either of which might have won

the game. The Jets scored two touchdowns on returns —

one by Santana Moss after a punt, the other on

an interception by Reggie Tongue — in a game in

which they were able to move the ball on offense

but could not find the end zone.

</P>

Date: Month 01 Day 15 Year 2005

GameFinalScore: l to the Pittsburgh Steelers, 20-17 ☐ Inf | Saturday ☐ Inf

DGameArg: n American Football Conference playoff game

Overtime: [✓ Yes] No Unknown | GameType: Regular [✓ Playoff] Exhib Unknown

fell to the Pittsburgh Steelers, 20-17, in overtime ☐ Inf | playoff ☐ Inf

OTGameArg: erican Football Conference playoff game | GTGameArg: an American Football Conference playoff game

NewYorkJets	PittsburghSteelers
TeamFinalScore # 17 17 ☐ Inf	TeamFinalScore # 20 20 ☐ Inf
TFSTeamArg: the Jets	TFSTeamArg: the Pittsburgh Steelers
TFSGameArg: American Football Conference playoff game	TFSGameArg: American Football Conference playoff game
Status: Won [✓ Lost] Tied InGame	Status: [✓ Won] Lost Tied InGame
fell ☐ Inf	fell ☐ Inf
STeamArg: the Jets	STeamArg: the Pittsburgh Steelers
SGameArg: American Football Conference playoff game	SGameArg: American Football Conference playoff game
HomeAway: Home Away Unknown	HomeAway: Home Away Unknown
☐ Inf	☐ Inf
HATeamArg:	HATeamArg:

Figure 2: LDC Machine Reading Annotation Tool

Heather Simpson, Stephanie Strassel, Robert Parker, Hoa Trang Dang, and Paul McNamee. 2010. Wikipedia and the web of confusable entities: Experience from entity linking query creation for TAC 2009 Knowledge Base Population. In *Proceedings of LREC 2010*.

Stephanie Strassel, Mark Przybocki, Kay Peterson, Zhiyi Song, and Kazuaki Maeda. 2008. Linguistic resources and evaluation techniques for evaluation of cross-document automatic content extraction. In *Proceedings of LREC 2008*.

Stephanie Strassel, Dan Adams, Henry Goldberg, Jonathan Herr, Ron Keesing, Daniel Oblinger, Heather Simpson, Robert Schrag, and Jonathan Wright. 2010. The DARPA Machine Reading program - encouraging linguistic and reasoning research with a series of reading tasks. In *Proceedings of LREC 2010*.

Steve Young, Dan Kershaw, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. 2000. *The HTK Book (for HTK Version 3.0)*. Cambridge University, Cambridge, UK.