# Data Acquisition and Linguistic Resources

Editor: Stephanie Strassel, Caitlin Christianson, John McCary, William Staderman

[1. Data Acquisition and Linguistic Resources 1](#_Toc251583166)

[1.1. Introduction 2](#_Toc251583167)

[1.2. Data Collection, Distribution, and Management 4](#_Toc251583168)

[1.2.1. Linguistic Resource Management for Large-scale Evaluation Programs 4](#_Toc251583169)

[1.2.2. TextCollection for Volume 7](#_Toc251583170)

[1.2.3. Distribution of GALE Resources 10](#_Toc251583171)

[1.2.4. Technical Infrastructure Supporting Large-scale Linguistic Resource Creation 13](#_Toc251583172)

[1.3. Human Annotation 16](#_Toc251583173)

[1.3.1. High-volume, High-efficiency Multilingual Transcription 16](#_Toc251583174)

[1.3.2. Creating Translation Resources to Support MT System Development 28](#_Toc251583175)

[1.3.3. Word Alignment for Improved Machine Translation 36](#_Toc251583176)

[1.3.4. Distillation Training Data Annotation 46](#_Toc251583177)

[1.3.5. Arabic Treebanking 61](#_Toc251583179)

[1.3.6. OntoNotes: A Large Training Corpus for Enhanced Processing 74](#_Toc251583181)

[1.4. Automatic Annotation 85](#_Toc251583182)

[1.4.1. Speech Segmentation and Its Impact on Spoken Document Processing 85](#_Toc251583183)

[1.4.2. CADIM Arabic Tools 97](#_Toc251583184)

[1.4.3. Chinese Statistical Parsing 97](#_Toc251583185)

[1.4.4. Evaluating the Impact of Word Segmentation on Chinese Parsing 111](#_Toc251583186)

[1.4.5. Combining Discriminative Re-ranking and Co-training for Parsing Mandarin Speech Transcripts 119](#_Toc251583187)

# Introduction

One of the potentially profound impacts of the GALE program is the legacy of data created during GALE. This legacy includes multiple languages (Arabic, Chinese, and English), multiple genres (newswire, broadcast news, newsgroups/blogs, and broadcast conversations), semantic annotation (propositions, coreference, names, and word sense), all in large volumes compared to prior work. This legacy has the promise of powering research for many years to come.

Data played a fundamental role in enabling the development of GALE’s next generation of translation and distillation technologies, since a common element throughout GALE technologies has been the use of statistical learning algorithms.

Traditionally data collection for machine translation has consisted primarily of large volumes of parallel corpora as well as some work in more detailed treebanking of collected data.  The GALE program required far more rigorous annotation and data specification procedures in order to optimize the engines’ performance and enable exploration of new approaches, delving much further into the individual and, at times, contradictory and ambiguous facets of foreign languages.

Section 1.2 covers Data Resource Management including large-scale broadcast and text data collection. In order to meet the need of performers for more varied, targeted, open source data against which to test and train their machines, LDC and its partners devised an unparalleled foreign language media collection program which quite literally spanned the globe, with collection sites in the U.S., the Middle East, Africa, Asia and elsewhere.  LDC and its partners were tasked not only with collecting unclassified foreign language training data, but also with the fair and equitable management, distribution and control of test data to the various competing performers as well as evaluation entities such as NIST, without which measuring progress in a scientific manner would have been impossible.

Section 1.3 covers the diverse types of manual annotation that enrich the linguistic data in GALE. Multiple types of annotation fostered new research directions, such as

1. Parallel corpora manually aligned at the word level (Section 1.3.3)
2. Distillation queries and answers (Section 1.3.4)
3. Dependency trees to support translation (Section 1.3.6)
4. Integrated syntactic and semantic annotations, including parse trees, propositions, coreference, names, and word senses (Section 1.3.7).

Section 1.4 reports on the use of algorithms for automatic annotation of data. This is explored in the context of rich transcription of speech, of segmenting and parsing Arabic, of Chinese word segmentation in the context of parsing, and of parsing Chinese spoken language.

# Data Collection, Distribution, and Management

# Large Scale Multilingual Broadcast Data Collection to Support Machine Translation and Distillation Technology Development

Authors: Kevin Walker, Christopher Caruso, Denise DiPersio

# Choosing Broadcast Sources

**6.**

**6.1.**

**6.2.**

**6.2.1.**

**1.**

**1.1.**

**1.2.**

**1.2.1.**

**LDC’s local collection**

In the beginning of the GALE program, LDC’s priority was ramping up collection to meet program targets, i.e., 1000 hours each of broadcast news (BN) and broadcast conversation (BC) in each of Arabic, Chinese and English. Later, LDC added additional receivers to the collection system to increase the range of its Arabic collection, responding to sponsor requests for greater representation of programming across the Arabic-speaking region, particularly, the Gulf region and Iraq. A number of Arabic sources were available from free-to-air (FTA) satellites transmitting over the Philadelphia area. LDC designed program surveys of the various sources that ran roughly twice per hour for several days, since these types of channels do not maintain scheduling information. LDC also coordinated with Hong Kong University of Science and Technology (HKUST), Medianet in Tunisia and MTC/ELRA in Morocco to obtain additional sources.

LDC’s GALE broadcast collection includes programming from the 26 Arabic sources, 13 Chinese sources and 3 English sources. shows the combined total number of hours of broadcast programming collected by LDC and remote sites in the first four years of the GALE program through December 2009.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Collection Type | P1 Hours | P2 Hours | P3 Hours | P4 thru |
| Arabic BN | 751 | 3896 | 3944 | 4940 |
| Arabic BC | 700 | 2813 | 3125 | 3153 |
| Chinese BN | 918 | 1231 | 1055 | 1452 |
| Chinese BC | 1242 | 1484 | 1427 | 1729 |
| English BN | 558 | 655 | 632 | 511 |
| English BC | 1041 | 1234 | 1035 | 1351 |

Table 1.: LDC broadcast collection for GALE

**LDC’s outsourced collection**

GALE’s goal to create end-to-end language systems meant that the broadcast collection should incorporate programming from as wide a variety of sources as possible. Accordingly, LDC obtained data collection assistance from several groups located within geographical areas of interest with access to targeted satellite and local programming. LDC’s broadcast collection programmer designed a portable collection platform that was deployed outside of the continental US.

# Broadcast Collection System Design and Operation

Part of the design intent driving the development of LDC’s broadcast collection system was that it be modular and regularized. That meant that all of the recording nodes should be interchangeable, that filenames and database fields should follow consistent, formal rules and that signal interconnects should be consistent. The receivers feed into an audio/video (A/V) matrix switch so that any source can be routed to any receiver simply by changing an entry in the schedule.

Initial recordings consist of video, stereo audio, and, in the case of English source, closed captions. LDC collects both audio and video data for each recording so that this material can be reusable for a variety of research purposes and because having access to the video portion of a given broadcast aids troubleshooting system functions and makes auditing more reliable, more efficient, and less error-prone. Recordings are typically transcoded to MPEG-4/AVC at 1Mbps shortly after capture.

The collections system is illustrated in the block diagram below:

Figure 1.: LDC Broadcast Collection Functional Block Diagram

Collection database

The database is the heart of the collection and contains all essential information: it is a history of all of the recordings that have been made; it has configuration and status information for all recorders; it has information about all receivers and associates specific programs of interest with the appropriate receiver; it contains a schedule of all recording jobs that need to be executed, along with their status; and finally, it stores all audit judgments associated with a given recording.

The following diagram depicts the collection database:

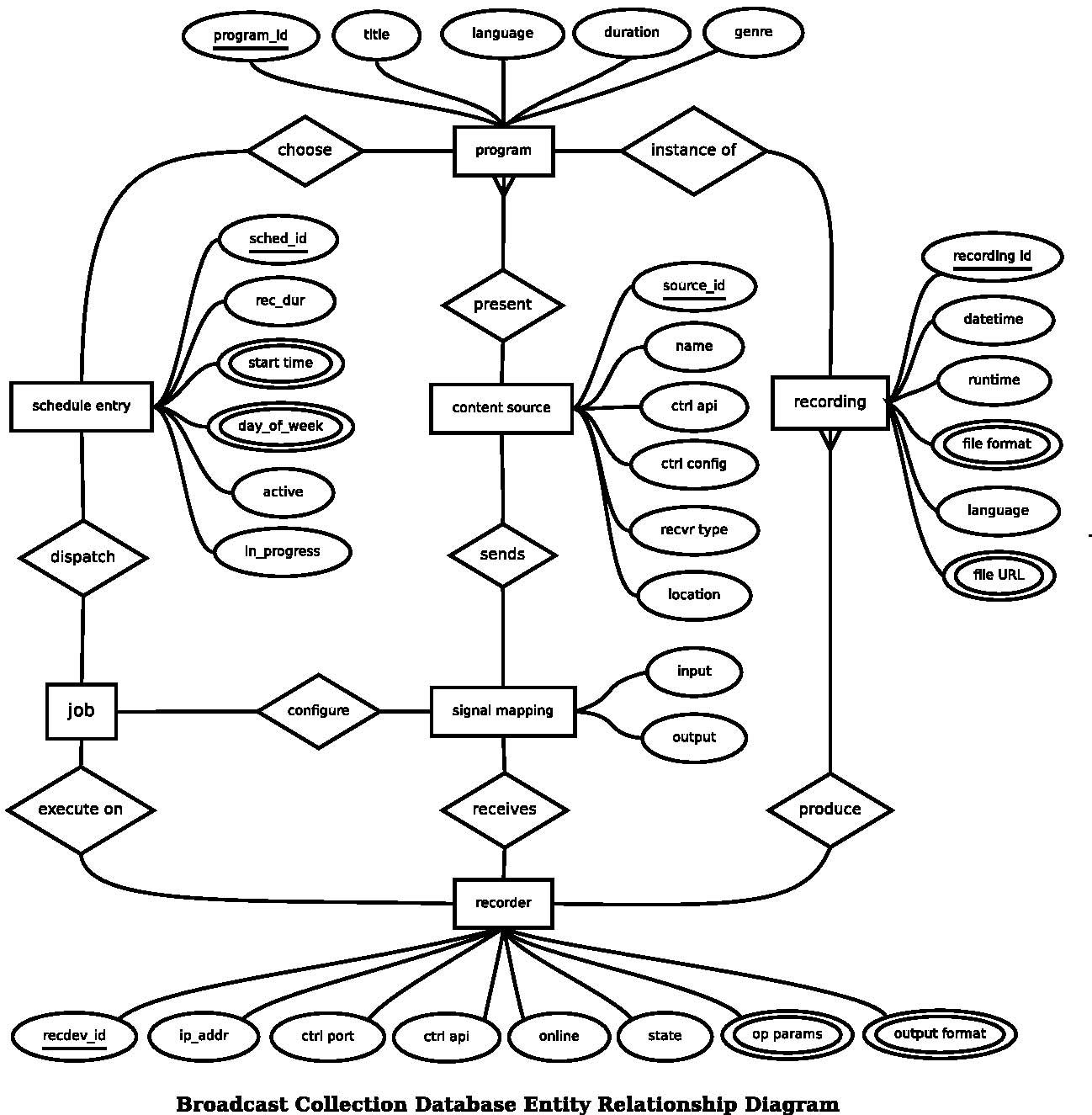


Figure 1.: Broadcast Collection Database Entity Relationship Diagram

#### 1.2.1.3. Automatic Speech Recognition (ASR) Technology

LDC has integrated three client ASR systems (from GALE research sites BBN, IBM and SRI) into its daily collection processes for the duration of the GALE program. Transcripts are automatically generated on the BBN and IBM systems for all locally-collected Arabic and Chinese audio data. The text output is used for downstream data selection. Audio data from LDC’s remote broadcast collections is not processed daily due to time constraints, but is instead run as needed as part of the GALE evaluation data selection process. Data is processed on the SRI system by request.

Locally-collected broadcasts are automatically pooled onto a centralized server as they are recorded and processed. This server then supplies the extracted audio portions to the different ASR systems which generate transcripts for the previous day's broadcasts.

#### Portable Broadcast Collection Platform

LDC’s portable broadcast collection platform is a digital video recording (DVR) system capable of recording two streams of A/V material simultaneously. It supports analog CATV (NTSC and PAL) and FTA DVB-S satellite programming and is capable of operating outside of the United States. It has a very small footprint and is suitable for transportation as a piece of carry-on luggage. The portable platform and the main LDC collection system share the same code base and rely on a modular, unified hardware specification. Improvements in the main collection platform therefore translate into benefits for both platforms.

#### 1.2.1.5. Conclusion

Using a highly-automated design and open source utilities, the GALE broadcast collection system represents a significant achievement in delivering volumes of high-quality broadcast data from multiple programming sources and geographic locations. The system has performed impressively. Through GALE Phase 3, LDC had delivered over 11,000 hours of broadcast audio to GALE research sites. A large portion of that data has been transcribed and translated as well, thus providing the GALE community with a considerable quantity of training, devtest and evaluation data.

# Text Collection for Volume

Authors: Haejoong Lee, Denise DiPersio, Robert Parker

#### Web Data Collection

GALE has required huge amounts of web data. With the launch of the program, the goal was to collect two million words for each language. To date, the number has increased to twenty million words for each language. LDC developed a web data collection framework to support GALE. The web collection project for GALE currently collects data from 47,857 weblogs or newsgroups which are hosted by 94 web sites using the framework.

In this section, we describe the details of the framework, and discuss benefits and limitations of the framework, and a future development plan.

The framework also provides a common ground on which various management applications can be built. For example, a collection management GUI makes it possible for a non-technical user to manage the entire collection process. Another program is capable of producing a real time collection status report out of the tracking database. These tools help streamline the collection process and reduce the management cost.

**Data Model**

Most weblog and newsgroup sites are structured in a similar way. This structure can be generalized by a three-level hierarchy. At the top, there is a site that hosts individual weblogs or discussion groups. The weblogs and discussion groups form the middle level. Each of the weblogs or discussion groups in turn contains threads of messages, which form the bottom level. In this hierarchy, we call the topmost entities *site*s, middle ones *group*s, and bottom ones *thread*s.

The basic data unit of the web data collection framework is a thread. Thread can be viewed as a tree whose root node is either a blog post or an initial newsgroup post. Comments by viewers for weblogs, and replies to a previous post for discussion groups form the inner nodes and leaves of the tree.

As an example, the weblog site blogspot.com is a *site*. This site hosts numerous weblogs, which are *group*s, contributed by bloggers. Each weblog has blog posts, e.g. daily coverage of presidential election. Such blog posts become *thread*s.

**Collection Process**

Under the framework, a collection task is defined by a group of sub-processes running in parallel. This includes harvesting, formatting, token counting and language identification processes.

The harvester process identifies and downloads web documents in raw HTML format. Downloaded HTML files are converted into SGML-based format by the separate formatting process.

The token counting process queries the tracking database to get a list of threads that haven’t been counted. For each thread in the list, a token counting algorithm is applied to get the token count. The result is the number of white-space-separated tokens for languages like Arabic and English, and number of characters for Chinese. The result is stored in the tracking database.

Language identification is a process of verifying the language of a thread. When a thread is downloaded, a new record for the thread is added to the thread table of the tracking database. By default, the language field of this record is set to the language of the thread’s group. It is not uncommon for a group to have threads of different languages. For example, some groups have both Arabic and English threads. Also, it is possible that the language initially assigned to the group is wrong. Thus, it’s necessary to have a process to verify the language of downloaded threads. For language identification, a language model trained with character *n*-grams of the target language is used. For a given document, the language model returns a floating point number indicating the likelihood of the document being in the language in question. This number is stored in the tracking database.

Note that the same token counting and language identification programs can be used across different sites. On the other hand, each site has its own harvester and formatter. Thus, a collection process usually consists of one token counting process, one language identification process, and several harvester processes with the same number of formatter processes, all of which running in parallel.

#### Newswire Collection

As of the commencement of GALE in 2005, LDC was collecting newswire in the target languages from six Arabic sources, two Chinese Sources and five English. In late 2006, new web sources were added to support GALE (three additional Arabic and four additional Chinese sources).

These new resources represented a 50% increase in LDC’s Arabic collection and a 56% increase in the Chinese collection. The average monthly volumes represented by the GALE newswire collection from Phase 2 forward are 13.4 million words/month for Arabic, 64.4 million characters/month for Chinese and 35.3 million words/month for English.

Most of LDC’s GALE newswire sources are harvested from provider websites, but some are received via satellite modem, FTP uploads and downloads, NNTP (Network News Transfer Protocol), proprietary clients and email.

#### Future Work

LDC developed the web data collection framework during the early months of the GALE program. Since then, it has been driving the web data collection project for GALE. At the same time, it has been debugged, modified and improved. However, there is still large room for improvement.

First of all, extracting information from the input HTML data is done by harvester and formatter code crafted by human. This is time-consuming and sometimes becomes a bottleneck during an early stage of a collection project. As an attempt to solve this problem, automatic pattern learning algorithms such as (Pennman, 2009) and (Holovaty, 2007) are being studied.

Another area requiring improvement is to automate the collection process, which currently requires human intervention on a regular-basis. Human intervention is needed because the harvester processes are often confronted by various types of errors such as a bug in the script, temporary problem of a web server, a system problem or a formatting problem of the HTML data it was trying to download, etc.

Although the time required to resolve such issues is minimal, automation that performs analysis of errors followed by an appropriate action is desired for smoother data collection process. Moreover, instead of harvesters being launched manually by human, a scheduler is being researched that initiates harvester and formatter processes just as frequently as indicated by data volume that sites or groups produce. LDC is planning to add these capabilities for fuller automation of the web data collection process.

LDC continues to work to refine its newswire procedures. This is particularly evident in the evolution of the collection and processing of newswire data from the web. Initially, the web collections were very simple bulk fetches from provider websites, resulting in the collection of duplicate or unwanted content. Over time, LDC has developed more intelligent web-crawling processes to reduce these issues, as well as to exploit newer technologies such as RSS (Rich Site Summary) and ATOM syndication. In addition, LDC has instituted automated reporting to monitor the collection and conversion processes and to provide alerts for any discrepancies.

#### Conclusion

LDC has collected a very large volume of text data for the GALE program. The amount of text data LDC collects and processes each month is 242.8 million words: 151.2 million words of weblog and newsgroup data, and 91.6 million words of newswire data[[1]](#footnote-1). Robust and efficient text collection infrastructures have been crucial for such a huge achievement. The web data collection framework has provided a scalable and cost-effective method for collecting web data for GALE. LDC continues to use the newswire collection infrastructure for GALE with great success. It was developed in early 1990’s and is evolving ever since. Further research and development for improvement on both the web data collection framework and newswire collection infrastructure are in progress.

# Distribution of GALE Resources

Author: Denise DiPersio

# Introduction

The particular challenge posed by the GALE distribution objective -- to distribute significant volumes of data timely and efficiently while conforming with external licensing and contractual arrangements – required an organization with expertise in distributing a broad range of resources to many users. LDC was particularly well-suited for this task; prior to GALE, it had distributed tens of thousands of corpora on media and via web download under various licensing schemes to thousands of users worldwide.

The GALE distribution objective required the performance of five principle tasks: creating a licensing structure; developing a data distribution plan for each GALE phase; preparing the data for release; distributing the data sets; and establishing a Secure Copy Server (SCP) server for rapid sharing of copyrighted data within GALE teams. Each of these tasks is described below.

#### Licensing

LDC developed evaluation license agreements stipulating that GALE participants would have to agree to use the data provided by LDC solely for GALE-related purposes.

#### Data Distribution Plan

GALE established ambitious goals for data collection, creation and distribution: 2000 hours each of broadcast news and broadcast conversation plus five million words of newsgroups and blogs –a large portion of which was transcribed, translated and treebanked – in each of Arabic, Chinese and English each year. In order to assure that GALE sites received a steady flow of data for training and testing purposes, LDC developed a plan to provide quarterly data deliveries. In response to community feedback, LDC changed that strategy to provide two large training data releases timed to be of maximum value to the sites as they prepared for scheduled evaluations.

At the commencement of the program, LDC provided a kickoff release to GALE research sites that included existing publications as well as previously unreleased data relevant to the program. In the first year, new data was distributed in four quarterly releases. Starting in the second year, training data releases were frontloaded at the request of GALE sites in order to allow as much time as possible to prepare for the evaluation task. In practice, this meant making two large distributions per phase, typically in the fall and spring.

Midway through year four, LDC had distributed nearly 7000 copies of 509 distinct corpora to forty-five sites in seven countries. That data includes 159 corpora from the LDC catalog designated as being relevant to GALE and approximately 200 new corpora. The latter consists of various training, test and development data (“ecorpora” or corpora desginated for an evaluation community such as GALE), FOUO corpora (“G-corpora,” only for use in the GALE program), restricted corpora (“R-corpora,” data whose distribution is confined to a limited group) and donated data.

#### 1.2.3.4. Data Preparation

GALE task managers and programmers prepared the data, and conducted task-specific data quality checks for content and format before it was mastered for release. The Publications Group performed additional quality assurance that included: checks for file integrity, permissions and file access; checking character ranges and parsing XML in text files; and checking headers in audio files. For media releases, each duplicate was automatically compared with the master using md5 checksums, and random manual checks are conducted on duplicated media. For web download releases, the Publications Group verifies content and completeness before preparing the download.

#### 1.2.3.5. Data Distribution

There were two principal methods for distribution depending on the size of the dataset: HTTP distributions were downloadable from LDC’s Members web site by authenticated GALE participants.

#### 1.2.3.6. SCP Server

The SCP server was established in year one in response to requests from sites to allow for rapid sharing of copyrighted data and annotations thereof within GALE teams at different geographic locations. Such activity was otherwise prohibited by the GALE license agreement and by LDC’s agreements with data providers, that is, data could only be shared within one organization at a single geographic site. For security reasons, access to the server was by means of public/private keypairs only.

#### 1.2.3.7. Conclusion

GALE has conferred a measurable benefit to the data needs of the linguistic research community. Most of the data collected and/or created for use in GALE will be released in the LDC catalog. As of this writing, some year one materials were already available, and other data was in the pipeline. This is consistent with LDC’s mission to share linguistic resources as broadly as possible.

Furthermore, GALE’s volume and distribution requirements provided the acid test for existing processes and have resulted in the establishment of an efficient infrastructure for data distribution. This includes the creation of stable master copies, media shipment tracking, perseverance of downloadable releases and rapid data sharing via the SCP server. That infrastructure can now be propagated to other common task projects with relative ease.

# Technical Infrastructure Supporting Large-scale Linguistic Resource Creation

Authors: Kazuaki Maeda, Andrea Mazzucchi and Christopher Cieri

#### Introduction

The DARPA GALE program set a new level of technical requirements for the Linguistic Data Consortium (LDC due to the quantity of training and evaluation data, the diversity, the manual and automatic transcripts, manual and automatic translations, found parallel text, word aligned parallel text, distillation annotations and treebank annotations. This paper provides an overview of the technical infrastructure supporting the GALE program at LDC.

#### Data Storage Infrastructure

The data storage requirements for GALE included large storage capability, flexible expansion capability and solid data security. The size of the dynamic data, such as annotations, software and documentation, for GALE exceeded more than 1 TB for each phase, which is more than 10 times larger than previous resource creation projects such as TDT (Cieri et al., 2002). The size of the static data, such as audio, video, newswire and web text data, is even larger. The total hours of broadcast news collection per phase, which includes both video and audio data, grew from approximately 5,000 hours in Phase 1 to more than 11,000 hours in Phase 3. The volume of web text collection grew from almost none before the GALE collection to approximately 7 million threads in Phase 4. The total size of the data storage for GALE has reached 90 TB as of this writing.

#### Software Infrastructure

The challenges presented by large-scale GALE data creation necessitate software tools that maximize efficiency. Most of GALE data collection, annotation, and distribution tasks use custom-built software developed by LDC's technical staff. Software tools are written to support annotation according to guidelines and other specifications, and to output consistent data formats.

**Annotation Tools:** In order to maximize the efficiency of annotation software development, we have adopted a common set of technologies for stand-alone annotation tools, such as Python, Qt, PyQt and MySQL, which are compatible with those previously developed at LDC, such as AGTK (Maeda et al., 2006). This allows the developers to reuse software components, and allows flexibility for allocation of software developers to available tasks as all developers are familiar with these technologies. Other tools use web-based technologies, such as PHP and Java Script. Annotation tools developed for GALE include XTrans, the Translation QC Tool, the SU annotation tool (Glenn et al., 2010), the Distillation Annotation Toolkit (Song et al., 2010), the broadcast audio auditing tool (Walker et al., 2010), and the GALE evaluation snippet selection tool.

**Annotation Workflow Management Tools:** LDC has developed a custom annotation workflow management system called AWS (Annotation Workflow System), used for most in-house manual annotation tasks at LDC. LDC has also developed a workflow management system, specifically for the Machine Translation Post-Editing task. The MT Post-editing system is web-based, allowing MT Post Editors to log in, receive their current assignments, annotate them, and submit them via the system (Maeda et al., 2008a).

**Parallel Text Alignment Tool:** LDC has developed two toolkits that are used for automatically aligning parallel text for GALE. One is the BITS (Bilingual Internet Text Search) toolkit (Ma and Liberman, 1999). The document alignment module of this toolkit was used for the document aligning task. The other toolkit is called the Champollion toolkit (Ma, 2006). This toolkit was designed for sentence alignment of noisy data, and proved to be very useful for alignment of parallel text harvested from web sites (Maeda et al., 2008b).

**Content Duplication Identification Tool:** It is inevitable to have some degree of content duplication in the raw collected data. For example, newswire agencies often distribute multiple versions of the same article with updates, or fixes; the same clips may be used in multiple broadcast programs. LDC runs a content duplication identification tool on the evaluation snippets, and on the source files selected for translation training data.

**Language ID Tool:** A language identification toolis run over LDC's web text collection to confirm that the majority of the collected text is in the target language. This is important for multilingual sources and noisy sources, such as newsgroup text, in which postings in non-target languages are often observed.

**Arabic Morphological Analyzer:** The LDCStandard Arabic Morphological Analyzer (SAMA), which is based on the Buckwalter Arabic Morphological Analyzer (BAMA) (Buckwalter, 2004), is run on the input data to the Arabic Treebank annotation task. The BAMA tool was developed by former LDC staff member Tim Buckwalter, and has been extended as the SAMA tool. The SAMA tool is extensively used in LDC's Arabic language projects.

#### Training Data Creation

LDC is responsible for creating the following types of training data for GALE:

* Broadcast
* Web Text
* Transcription
* Parallel Text
* Distillation
* Word Alignment
* Arabic Treebank
* English Translation Treebank

#### Evaluation Data Creation

LDC also creates development (devtest) and evaluation (eval) data for GALE MT evaluation programs. While the basic data creation process is the same for the training data, the evaluation data creation requires more detailed quality control and secure handling of the data. The evaluation data process is done in close collaboration with the National Institute of Standards and Technology (NIST), which administers the GALE MT evaluation.

**MT Evaluation Snippets:** Unlike the trainingdata, the GALE MT evaluation uses snippets of documents or broadcast recordings. The evaluation snippets are created in the following steps.

1. Data collection
2. Program-level broadcast audit
3. Decision of evaluation epoch by GALE data committee
4. Manual selection of evaluation snippets
5. Transcription (for BC and BN) and SU annotation (for NW and WB)
6. Duplicate content identification
7. Down-sampling of evaluation pool by NIST
8. Initial Translation
9. Complete QC of source and translation
10. Final selection of evaluation snippets by NIST

In each stage above, snippets that are found problematic or unsuitable are dropped from the data set, so over-selection of data is crucial. For the Phase 4 evaluation, LDC selected four times as much data for the initial pool than remained in the final evaluation data set.

**Data Security:** All GALE evaluation data is protected with strict data security control. Evaluation data on the file servers is protected with Unix group access control, and only LDC personnel who are working on the GALE program are permitted to be in this restricted group.

#### Conclusion

The technical infrastructure described in this paper represents the work done for the GALE program, which is the largest scale resource creation effort in the history of the LDC. However, there is no doubt that the work for GALE was built on the earlier linguistic resources creation efforts at LDC, and will be the basis for future work at LDC.

# Human Annotation

# Linguistic Resources for Transcription and Translation

Authors: Meghan Lammie Glenn, Lauren Friedman, Stephanie M. Strassel, Zhiyi Song, Gary Krug, Kazuaki Maeda, Haejoong Lee, and Christopher Caruso

#### Introduction

The GALE program required significant volumes of new training data -- hundreds of hours of transcribed and annotated speech, plus hundreds of thousands of words of parallel text -- for each language and genre being evaluated in the program. As such, LDC's initial focus was on developing efficient, scalable processes for data selection, transcription and translation. The primary emphasis was on quantity and coverage, especially with respect to training data for Machine Translation (MT) and Automatic Speech Recognition (ASR) systems. Large volumes of Arabic and Chinese text and audio were collected to provide background language models and data for unsupervised training, and a sizeable portion of the collected data was then selected for manual transcription and translation. Over time, as the archive of existing training data expanded and GALE system performance improved, requirements for new linguistic resources evolved accordingly, shifting from rapid, high volume resource production to a more refined and targeted selection of material to address specific gaps in coverage. Each stage of the GALE program has motivated advances in LDC's resource creation pipeline. New annotation tasks and guidelines, quality control procedures and enabling infrastructure like annotation software and data formats have been developed at regular intervals to keep apace with evolving program requirements. As GALE system performance improves, LDC has increasingly relied on ASR and MT system output to augment and improve the data production pipeline.

This paper describes LDC’s methodology for creating linguistic resources for transcription and translation, covering both training and evaluation data. We describe data selection; annotation including transcription, translation, and pilot efforts; quality control practices; and the novel technical infrastructure developed to support the unique demands of GALE.

#### Training Data Production

**Data Selection**

Although some transcribed audio and parallel text may be harvested online or adapted from existing resources, the bulk of the training data developed for GALE is created manually. Training data production starts with selection of material for manual transcription and translation. This material is chosen from Arabic and Chinese source newswire, web text, broadcast news and broadcast conversation collected by LDC for GALE, following procedural guidelines developed by LDC in consultation with NIST, DARPA and GALE research teams. Detailed volume and genre requirements for each round of selection are established by LDC and GALE sites, and refined at regular intervals to meet evolving program goals.

For each batch of data to be transcribed or translated, a selection pool is prepared that adheres to requirements for (1) source and program variety, (2) genre balance, (3) broadcast/publication dates, and (4) content of existing transcript and translation resources. The selection pool is heavily weighted to prefer sources and programs that are underrepresented in previous training data releases. Priority is also given to recently collected data and/or material from epochs close to the current evaluation epoch. Once the selection pool is established, human annotators quickly review each candidate, excluding items that do not meet requirements. Because GALE is primarily focused on transcription and translation of Modern Standard Arabic and Mandarin Chinese, candidates that contain large volumes of non-MSA or non-Mandarin Chinese dialects are typically excluded from selection. Similarly, material whose content focus is deemed inappropriate -- coverage of sporting events, spam or other offensive material, soap operas and the like -- is also excluded.

**New Selection Methodology**

After substantial volumes of high-quality, broad coverage training data for translation and transcription became available during the initial phases of the program, there was growing interest from GALE research sites and LDC alike in developing refined data selection techniques that would especially target high-payoff data; i.e., novel material that does not duplicate the salient features of existing training data, and material that is especially difficult for ASR and MT systems. In response to this need, LDC began working closely with research teams to develop a series of "smart data selection" techniques to guide selection of new transcription and translation training data. These new approaches rely both on understanding the properties of existing training data (for instance, what n-grams already occur in sufficient numbers in existing parallel text?), and on detecting data on which system performance is unexpectedly poor.

**Smart Data Selection for Transcription**

As a starting point, LDC and GALE research teams collaborated to install production MT and ASR systems at LDC, and to integrate these systems into LDC's regular collection pipeline (Walker et al., this volume). Collected audio recordings are run daily through the ASR systems, yielding two dozen or more hours of ASR output per language per day. GALE researchers worked with LDC to develop a process that could leverage this resource by tapping into a simple intuition: if multiple ASR output streams have high rates of agreement with one another for a single recording, ASR error rates for that recording are probably low, whereas low agreement rates likely signal high ASR error rates. Audio recordings showing moderately high rates of disagreement among ASR systems could then be targeted for manual transcription, to produce high-value training resources when compared to resources selected using traditional methods.

To implement the new approach, segmented ASR output for Arabic and Chinese is first stripped of all segmentation markers and converted into plain text for more accurate comparison. The resulting files are then processed by sclite (Fiscus et al., 2006), a scoring script developed by NIST, which generates a report of the word error and agreement rates among the different systems for each show’s recording. The result of the process is a list of programs ranked by scoring agreement for all programs in a given selection pool.

A pilot study conducted by LDC during GALE Phase 3, which compared the output of two ASR systems, demonstrated a “sweet spot” between 30% and 80% agreement. Annotators audited the ranked list of audio files and recorded information about each file, noting lengthy music segments, foreign language speech, non-native speakers, commercials, and the like. Recordings with 30% or lower agreement tended to contain extended commercials, music programs, heavy regional speech variation, or recording errors. Those with 80% or higher agreement tended to be “talking-head” style read speech, usually by a single speaker. LDC leverages this information to exclude files from the two extremes during selection. The programs between the 30% and 80% error range may contain speech phenomena for which there is not yet enough training data for ASR systems.

Once the outlier programs have been removed from the selection pool, annotators review the programs to further eliminate those which may not be suitable for GALE transcription and translation due to their content or level of speaker interaction. The files approved during this process are then manually transcribed.

An added benefit to leveraging ASR technology and scoring recordings according to the level of agreement among shows is that programs with exceptionally low or high levels of matching segments may be further examined to determine the causes of variation, enabling LDC to identify potentially problematic audio elements of broadcasts. Another potential use for these reports would be to more quickly identify “challenge” sets of broadcast genres for system development.

This ASR-assisted selection approach has been implemented to select half of all new training data transcripts.

**Smart Data Selection for Translation**

Starting in Phase 3, data selection for translation underwent a similar transformation. Whereas previous selection methods produced full-document translations (a whole newswire article, blog post or broadcast story), the new method targets selection of individual high-yield sentences -- where high-yield is defined having features that are novel compared to existing stores of training data. LDC collaborated with GALE research teams, in particular IBM and SRI, to developed scripts to implement the novel selection method. The process begins by establishing a table of n-grams for sentences in existing training data sets. Next, LDC prepares a pool of candidate documents for a given language and genre. The script is run over this data pool, and generates a list of candidate sentences, scoring each according to its uniqueness against the table of existing n-grams. Humans quickly scan the resulting ranked list of candidate sentences, discarding inappropriate material prior to manual translation. Initial feedback from GALE teams is positive, and this sentence-based method is now exclusively used to select new translation training data.

#### Training Data for Transcription

**Manual Transcripts**

While some manual transcription and related speech annotation activities are performed by local LDC staff, we also maintain relationships with a number of international partner sites and commercial agencies who possess the infrastructure and know-how to rapidly produce high-volume, low-cost training transcripts and translations of sufficient quality. All transcription partners undergo regular hands-on training conducted by LDC annotation staff, and all partners utilize LDC infrastructure including transcription guidelines and user interfaces, to ensure consistent quality. Moreover, small volumes of data are regularly subject to dual, independent transcription to provide material for analysis of human transcription variation and to ensure that inter-transcriber agreement both within and across sites falls within an acceptable range.

**Transcription Task Specifications**

All transcribers, whether they are working at LDC or at one of our partner sites, are required to follow formally-defined transcription specifications to ensure efficiency and consistency among all transcripts, regardless of source language. Guidelines differ depending on the purpose of the data, though the goal for all transcription approaches is essentially the same – to produce a verbatim transcript with speaker and story identification in the most efficient manner. Training data requirements place a higher value on procuring large volumes of transcripts with less emphasis on gold standard perfection, whereas evaluation data requirements reverse these priorities. This section describes the training data transcription approaches employed within GALE.

**Quick Transcription**

Quick transcription (QTR) for broadcast news and broadcast conversation aims to quickly produce a (near-) verbatim transcript, time-aligned at the phrase or pause-group level, with markup limited to story boundaries and speaker turns/IDs. The emphasis is on efficiency, with a goal of producing fairly accurate transcripts in 5 to 10 times real-time (in other words, 5 to 10 hours of transcription effort per hour of speech). QTR is used only for training data production.

**Quick Rich Transcription**

Quick-Rich transcription (QRTR) was defined at the outset of the GALE program and is the default manual transcription method for GALE. The goal of QRTR is to balance efficiency and completeness, producing transcripts that are maximally useful for downstream manual and automatic annotation tasks including translation, distillation and Treebanking. Like QTR, QRTR results in verbatim, time-aligned transcript with story boundaries and speaker turn identification, but QRTR also incorporates features like sentence-unit (SU) identification and sentence-based time alignment. (LDC, 2008).

The methodology for identifying sentence units in spontaneous speech grew out of work performed during the metadata extraction (MDE) evaluation of the DARPA EARS program, which aimed to annotate verbatim transcripts of spontaneous speech in order to make them maximally readable (LDC, 2004). Annotations included removing disfluent non-content words such as filled pauses or backchannels, and labeling each SU according to its type, such as statement, question, or incomplete utterance.

QRTR incorporates SU annotation into the audio segmentation process, which is the first stage of transcription. A sentence unit (SU) is a natural grouping of words produced by a single speaker. SUs have semantic and syntactic cohesion. When creating SU boundaries for spoken language, the transcriber’s goal is to identify a semantically and syntactically cohesive group of words that constitute a reasonable sentence-like unit. The segmentation remains consistent for all downstream tasks, permitting a mapping from the source audio signal to the source transcript to the English translation and annotations for each sentence unit. Transcription rates for QRTR are targeted at 15 to 20 times real-time.

#### Overview of QRTR Rules

Transcription rules are consistent for all three source languages. The elements of a quick rich transcript include: verbatim transcription, time-aligned story boundaries; speaker turn, section and sentence type identification; speaker identification, and standard treatment of common spoken phenomena (e.g., hesitation sounds, spoken letters or numbers, and disfluencies).

**Segmentation**

Transcription begins with audio segmentation, which involves marking structural boundaries including story boundaries, speaker turns and sentences. Speakers are identified by name where possible, or by a unique identifier, such as “speaker1”. Transcribers also indicate the speaker’s type – male, female, child, or other – and native or non-native speaker status. Story boundaries are inserted when there is a change in topic, and stories are labeled according to their style: talking-head broadcast news sections are labeled “reports”; discussions or informal chats are labeled “conversations”. Musical interludes, commercials, public service announcements, and periods of silence are left un-segmented, or in some instances may be segmented and labeled “non-trans” and left untranscribed.

**Verbatim Transcription and Orthography**

Once audio has been segmented into smaller units, annotators transcribe the content of each segment. Transcribers use standard Mandarin Chinese, Modern Standard Arabic, and North American English orthography. Special conventions are used to flag certain speech phenomena like disfluencies, mispronounced words, and non-target language speech. Quality control checks verify the format of the resulting file as well as overall transcript quality.

**Foreign or Colloquial Speech**

When transcribers encounter speech not in the target language, they identify the language but do not attempt to transcribe the region.

Colloquial speech regions are transcribed but are marked as being “non-standard.” Due to the prevalence of dialectal speech in Arabic broadcast conversation files, transcribers mark speech that is not Modern Standard Arabic as such, for example:

<non-MSA> </non-MSA>

When the transcriber understands the dialect, he/she transcribes the utterance, for instance:

<non-MSA> كمان </non-MSA>

While transcribers do not label specific Arabic dialects during the transcription process, identifying non-MSA regions at this stage may help downstream processing, error analysis, or setting the stage for Arabic linguists to revisit those regions as a separate dialect annotation task.

**Web Transcripts**

To supplement manual transcription efforts, LDC routinely harvests available web transcripts[[2]](#footnote-2) in Arabic, Chinese and English for sources that are part of our ongoing audio collection, and conducts periodic searches to identify transcripts from new broadcast sources. Plain text transcripts are extracted from HTML files, converted to UTF-8, and divided into sentences based on punctuation characters. While web transcripts are typically verbatim or near-verbatim, the format differs considerably from LDC's native transcription format in several ways. First, web transcripts are not time-aligned with the audio; second, they do not follow LDC-style transcription conventions; finally, story boundaries and sentence units (SUs) have not been explicitly labeled.

**Transcription Volume**

LDC has released over 10,000 hours of training transcripts in Arabic, Chinese and English from Phase 1 through Phase 4 of GALE. The total number of transcript hours released, and the distribution of those transcripts by language and genre, varies from phase to phase depending on the specific requirements expressed by the GALE research teams, and also in part due to the availability of collected and audited broadcast data.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Language** | **Genre** | **Phase 1** | **Phase 2** | **Phase 3** | **Phase 4** |
| Arabic | BC | 454 | 798 | 667 | 459 |
| BN | 640 | 751 | 557 | 171 |
| Chinese | BC | 975 | 478 | 280 | 262 |
| BN | 586 | 492 | 341 | 310 |
| English | BC | 763 | 128 | 154 | 282 |
| BN | 507 | 122 | 151 | 65 |

Table 1.: Volumes, in hours, of manual and web training transcripts distributed through Phase 4 of GALE.

#### Training Data: Translation

All statistical machine translation systems require significant linguistic resources for system training; GALE required LDC to produce these materials on a larger scale and with greater variety and agility than ever previously undertaken. In response, it was necessary for LDC to develop novel methods, technical infrastructure, and best practices to create a pipeline that is flexible, scalable and efficient. While the quality of training data is important, it must be balanced against the unwavering demand for high volume, low cost, and rapid distribution -- in well-established translation genres like newswire, but also in diverse and challenging new genres like talk shows and weblogs. As the GALE program progresses, the translation pipeline must also become more refined, and the efficiency of the processes and the quality of the data improve in tandem. This section fully describes the translation pipeline, which includes not only the actual translation process, but also numerous preparation and quality control measures that shape the final product.

**Before Translation**

LDC conducts multiple data preparation stages before sending source documents to translation agencies. First, all files put through the translation pipeline are either manually selected or – if automatically-selected – manually reviewed to reject non-target content, as previously discussed in the Data Selection section.

**Corpus-wide Scans**

Once the translation pool has been identified, the corpus is automatically scanned to remove duplicate documents and to ensure that there is no overlap between the training and evaluation sets. This crucial step further focuses the use of resources on novel data and is one of many safeguards to preserve the integrity of evaluation sets.

**SU Annotation**

Selected data is segmented into sentence units (SUs). SU segmentation is performed manually or automatically depending on the genre and timeline. In general, newswire data can be automatically segmented; QRTR transcripts are manually segmented during transcription; QTR transcripts and web documents must be manually segmented prior to translation. Automatically-segmented data is manually checked and corrected to confirm that logical semantic segments have been created. SU segmentation is preserved throughout the training data pipeline.

**Pre-processing**

Before the data pool is outsourced for translation, it is sub-sampled in order to achieve the best diversity and balance among sources and genres, according to requests or requirements for each dataset. The final set of segmented source data is then sent out in a special UTF-8 text format where each numbered segment of source data is paired with a corresponding blank numbered line. For example, an outgoing Arabic source file would look like this:

<ar=1> Arabic text

<en=1> [blank line]

<ar=2> Arabic text

<en=2> [blank line]

Translators perform the translation one segment at a time, inputting the English on the blank line for each source segment. This process was developed to create perfect alignment at the sentence level between the source and translation. Using this translation format, manual or automatic alignment at the sentence-level is not needed downstream; the translation is aligned as it is generated.

These translator-ready files are then collected into “kits” for outsourcing to vetted commercial translation agencies. Kits are customized for each agency based on target volume, agency expertise, file length, and level of difficulty. Different sources are distributed evenly across agencies so that any observations about a particular source are not invalidated by different agencies’ variability in style or quality.

**Manual Translation**

In addition to the technical infrastructure it has developed, LDC has also produced extensive documentation to systematize and standardize large-scale human translation for the GALE program. While variability cannot be completely eliminated in a human task such as translation, where there will always be multiple correct translations, it is controlled as much as possible through the provision and enforcement of detailed and rigid translation guidelines developed for GALE, which instruct translators to emphasize accuracy and fidelity to the source text, and provide clear guidance on how to approach special cases (LDC, 2009).

These guidelines include instructions and examples to aid translators in approaching documents collected for GALE – which differ significantly from the data translators usually encounter – as well as requirements for how and by whom documents must be translated. LDC uses multiple translation teams for each language, and each team must have at least one translator native in the source language and one native in the target language.

Translators are required to follow the guidelines’ specifications for translating speech disfluencies, factual errors, filled pauses, proper names, and more. Instructions for handling potential challenges – like typos, neologisms, emoticons, and other features of web data – are also included. Addressing these special cases in the translation guidelines assures consistency where there would otherwise be variability if individual translators relied on their own best judgment. Furthermore, clear markup that indicates typos, translator uncertainty, and made-up words allows sites and evaluators to treat these instances differently when necessary.

The translation guidelines are a living document, constantly updated to include more examples and address new translation issues. Releases include READMEs to indicate which version of the guidelines has been applied to each particular data set. Distinct guidelines have been developed for Arabic and Chinese source languages so that language-specific issues can be adequately addressed. Translation agencies are always required to use the most up-to-date version of the guidelines to ensure that all data is translated according the same standards.

#### After Translation

**Post-processing and Sanity Checks**

Once a set of data has been translated, incoming translations are processed using a suite of scripts that extract the English lines from the merged source/translation documents and perform a bevy of sanity checks. These checks ensure that all files have been returned, all files are correctly encoded, all source segments are present, all segments have corresponding translations, and the data format is intact. Thus, most problems with incoming translations are discovered automatically and can often be rectified automatically as well. Identifying problems at this early stage allows LDC enough time to return files to agencies for retranslation when errors cannot be easily fixed by LDC.

**Manual Quality Control**

Manual quality control (QC) occurs after this series of automatic sanity checks. Even when translations are created by skilled translators and then proofread closely, erroneous translations and areas of ambiguity frequently exist in raw translations. Certain genres that are included in GALE, especially broadcast conversation and web data, provide particular challenges to human translators, as noted previously. In addition, since translation is not an exact science, two independent translators will typically produce two different translations for an identical source document. However, human translations produced for MT training and evaluation must be exact and fully expressive: the translation must convey no more and no less information than the source.

The quality control processes undertaken for training and evaluation data are vastly different. Evaluation translation quality control is discussed in detail in Section 1.3.1.3. For training data, a subset of each translation delivery (between 10 and 20 percent of the total word count) is checked by bilingual LDC annotators trained in the appropriate procedure. Annotators apply specific scoring mechanisms according to a rubric included in the translation guidelines provided to agencies. Translation errors are categorized as: syntactic, lexical, poor English usage, or typographic (significant spelling or punctuation mistakes). Each category is given a certain number of points. Deliveries that receive a failing score (too many points) are rejected and returned to the agency; payment is withheld until corrections are completed on the entire translation set (not just the files that were reviewed) and the revised translation delivery meets QC standards.

#### Training Data: Parallel Text Harvesting

LDC has also developed a set of software tools including BITS (Ma and Liberman, 1999) to identify potential parallel text resources among online archives of multilingual documents (Maeda, et al., 2008). LDC uses this software to scan and harvest likely parallel documents from the web on a regular basis. These sources include newswire articles from multilingual news agencies, such as Agence France Presse and Xinhua News Agency.

Parallel text documents on the web come in a variety of formats. After harvesting, files are converted to a text format with a predefined set of SGML or XML markups, and the document mapping module of the BITS system is then run to identify pairs of possible parallel documents. Once pairs are identified, each document is automatically segmented into sentences and then processed by the Champollion sentence aligner (Ma, 2006) to create sentence mapping tables. For the GALE program, LDC has created and distributed over 82,000 automatically-harvested document pairs containing likely Arabic-English parallel text, and over 67,000 document pairs for Chinese-English.

#### Evaluation and Development Data Production

One of the greatest challenges encountered in producing linguistic resources for GALE is creating the gold standard source and reference sets for evaluation. The GALE evaluation paradigm relies on a carefully constructed test set, which includes careful manual selection and transcription (for audio genres) of the evaluation data pool. On the surface, creation of test data for machine translation is straightforward: take the set of evaluation documents and manually translate them. But like any task involving human judgment, “translation” is not a monolithic task and there are multiple decision points along the way. As the provider of evaluation data, LDC must consider not only the fully articulated requirements for test data – the type stated in an evaluation plan – but also hidden assumptions and implicit requirements that are equally important in constructing appropriate data for evaluation.

**Data Selection**

Evaluation data is selected carefully, according to very specific guidelines. An evaluation epoch is first decided collaboratively by LDC, DARPA and NIST. LDC compiles candidate files from that epoch to construct a pool of newswire, broadcast audio and web data. Each candidate file (audio or text) is examined carefully to identify snippets that meet the criteria specified in the selection task specification. These text and audio snippets typically correspond to a topically-cohesive story selected from each document or recording. Information about topic category, presence of dialect, and acoustic and dialect features of the audio file are noted by annotators.

**Segmentation and Transcription**

Segmentation for evaluation data is especially important, since MT output is scored at the segment level. Poorly constructed segments can lead to ambiguity and then inaccurate MT scores. Segmentation errors that affect the meaning or interpretation of the text are always corrected, even if they are discovered at the final stage of the pipeline.

Sentence Units (SUs) are manually identified for the newswire and web snippets. Selected audio snippets are carefully transcribed and segment boundaries identified by professional transcription agencies and LDC annotators, following an enhanced version of the Quick Rich Transcription (QRTR) specification. Transcription of evaluation data differs from the training data approach in that it requires multiple manual reviews of a file to produce a gold standard reference transcript. Transcribers first create a basic quick-rich transcript, with speaker and sentence unit annotation and a verbatim transcript. Senior transcribers then review each utterance carefully to verify the transcripts' accuracy, making modifications where necessary. The lead transcriber conducts a final review over all transcripts before marking them complete, and the reference transcripts are then passed to the gold-standard translation process.

**Translation and Manual Quality Control**

Transcribed and segmented files are reformatted into a human-readable translation format, and assigned to the best professional translators for careful translation, following LDC’s standard GALE translation guidelines (LDC, 2009).

After translation, reference files undergo several additional stages of annotation and quality control to correct errors, finalize and correct segmentation, improve translation adequacy, add translation variants, standardize proper nouns, verify technical terms and so on, with the ultimate goal of having gold standard translations that are absolutely faithful to the source data in terms of meaning, fluency, structure and style.

In order to standardize the translations and produce an appropriate reference for evaluation, LDC has developed a six-step translation and QC process:

1) A source-language dominant bilingual translator produces a preliminary translation emphasizing accuracy;

2) A target-language dominant bilingual translator revises the translation to improve fluency;

3) A source-language dominant bilingual annotator checks translation for errors and omissions;

4) A source-language dominant bilingual senior annotator checks for remaining errors, improves fluency, corrects and standardizes named entities;

5) A target-language dominant bilingual annotator improves fluency and adds translation variants where required;

6) A target-language monolingual annotator reviews for fluency and consistency, and flags questionable regions.

Steps 1 and 2 are largely the same as those used in the creation of training data, but additional quality control loops are added to meet the evaluation data standards. The translations delivered after Step 2 are considered final and complete by the agencies; the subsequent steps are an above-and-beyond layer added by LDC in order to ensure the highest possible confidence in the released gold standard.

Steps 3 through 6 are performed in-house. In Step 3, the annotator focuses only on correcting egregious errors. The main objective of Step 4 is to resolve any nuanced issues with the translation, while verifying total fidelity to the source – a requirement for the GALE evaluation. In Step 5, fluency problems are corrected and translation variants are introduced to clarify regions of ambiguity.

Relative to Step 5, Steps 3 and 4 are extremely time-intensive. For the most recent GALE evaluation, Step 3 averaged 25 minutes per Arabic document and 21 minutes per Chinese document. (Each document is between 150 and 350 tokens, where a token is an Arabic word or Chinese character.) Step 4 averaged 13 minutes per Arabic document and 15 minutes per Chinese document. Step 5, however, averaged just 6 minutes per document for both languages, since the translation is generally in excellent condition by the time it reaches Step 5.

The final check, Step 6, is a quick but thorough read-through of all of the translations, with an eye to any errors that may have been introduced inadvertently in previous steps. The Step 6 reviewer must also ensure that the translations read as correct, fluent English, independent of the source text. Each stage requires a different level of expertise, and while six different translators might produce six different translations, the six-person translation and QC team – who work independently but consecutively on one working document – is designed to achieve a higher level of consistency and predictability. Step 5, where translation variants are introduced when required, is especially important for ensuring that any alternate – but equally accurate – interpretations of the source text are included in the final translation reference.

**Development Data**

In addition to training and evaluation data, LDC produces development datasets, which are typically small-volume, carefully-created datasets that resemble evaluation data in terms of preparation and total volume, and are carefully selected from the epoch immediately before the evaluation collection epoch. In terms of production procedures, devtest sets fall somewhere between the carefully controlled process for evaluation data, and the high-volume rapid-turnaround approach taken for training data production. Requirements for development data vary across phases, and are typically specified by GALE research teams. For instance, in GALE Phase 4 LDC selected, transcribed, annotated, and translated a large devtest corpus of over 600 snippets per language per genre. A smaller subset of each phase's devtest set is typically enhanced with additional annotation including word alignment and multilingual Treebanking.

#### Annotation Tools and Infrastructure

The constantly evolving data requirements of the GALE program have required LDC to develop an innovative suite of supporting software and related technical infrastructure, in order to construct an adaptable and extensible pipeline for transcription and translation that appropriately balances concerns of quality, efficiency, complexity and flexibility.

**XTrans**

Among the challenges faced in creating manual transcripts for training or evaluation purposes is the efficiency of the transcription software itself, since an efficient tool can boost productivity and provide opportunities for richer annotation or research. Broadcast recordings present the challenge of containing overlapping speech from multiple speakers on a single audio channel. Creating segments and transcripts for such overlapping utterances is a function that many currently available transcription tools lacked. To meet demands for increased volumes of spontaneous speech, which often include conversations among multiple, simultaneous speakers, LDC required an efficient and straightforward tool for transcribing overlapping speech. Furthermore, the need to produce large volumes of transcripts with a consistent file format across multiple languages promoted the development of a new transcription tool.

XTrans, the resulting speech annotation toolkit (Glenn et al., 2009), supports a full range of speech annotation tasks including quick and rich transcription and annotation of broadcast audio, telephone speech and meetings. Powered by Qt’s international language support, XTrans can be used for transcription tasks in many different languages. It is easily ported to most UNIX derivatives, Microsoft Windows and Mac OS X. XTrans consists of several re-usable components such as text and waveform display. Most of the components are written in Python with some components written in C++ (LDC, 2007).

The Qt text widget also supports bi-directional text input for right-to-left (RTL) languages like Arabic. This functionality allows transcribers to insert English-language metadata tags in the Arabic language transcript. Clicking on a segment in the transcript highlights the corresponding segment in the waveform, and vice versa.

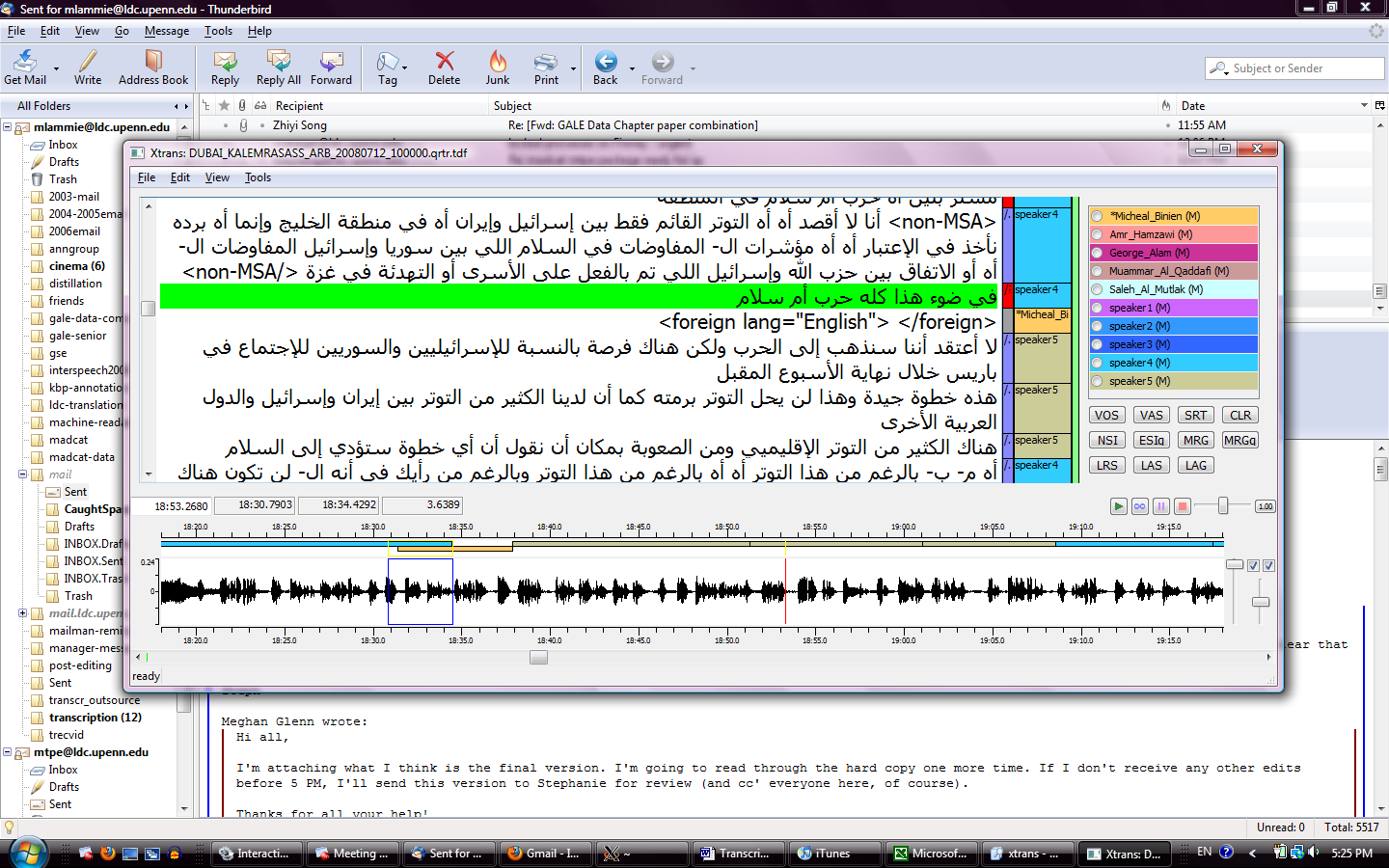


Figure 1.: Arabic broadcast conversation transcription session in XTrans.

XTrans alleviates the challenge of efficiently annotating multiple overlapping speakers. The tool incorporates the concept of a *virtual speaker channel* (VSC). Each VSC corresponds to one speaker, rather than to any particular physical channel in a sound file. A VSC may also be used to represent background noise or other non-speaker sound sources. The addition of this feature allows transcribers to transcribe a potentially unlimited number of simultaneous speakers without having to resort to any cumbersome markup.

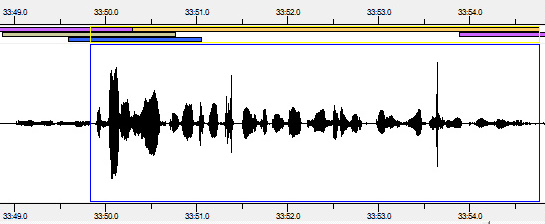


Figure 1.: Close-up of overlapping utterances in XTrans.

Each speaker in the recording is randomly assigned a color by the tool. The segments for that speaker are coded with the same color in the transcript window. The waveform display also shows the span of each segment in the audio channel as a color-coded box. Overlapping utterances are depicted in the waveform display by stacked boxes, as shown in Figure 1.4. The overlapping segment information is then extrapolated from the times in the transcript file, as shown by the start and end times of the two segments in the following example:

|  |  |  |  |
| --- | --- | --- | --- |
| **Start** | **End** | **SpeakerID** | **Transcript** |
| 394 | 400 | speaker1 | 那一会儿我们跟您详细聊一聊,给我们讲讲您的故事,好吗,好谢谢何伯伯 |
| 394.2 | 399 | speaker2 | 呃呃, 呃呃呃, 呃呃呃 |

Table 1.: Overlapping segments in plain text transcript.

XTrans incorporates a number of quality control functions that improve the accuracy of speaker identification by the transcriber, such as the ability to listen to random segments or all segments from a selected speaker. In addition, transcribers can listen to all unsegmented audio “gaps” in the transcript, which assists in identifying regions of missed speech. It is easily configured to make transcription efforts more efficient -- for instance, all segmentation functions can be performed using keyboard shortcuts instead of a mouse. Users may also add their own keybindings or modify existing keybindings to suit their comfort and preference.

**Translation QCTool**

The multi-stage translation quality control process for evaluation datasets is facilitated by QCTool, an annotation tool written in Python and based on XTrans. QCTool allows annotators to view source documents and translations side-by-side and edit the working copy of a translation. It also includes functionality for viewing and reverting to previous translation versions, flagging sections for further review, playing back source audio data, and displaying edits as they are made.

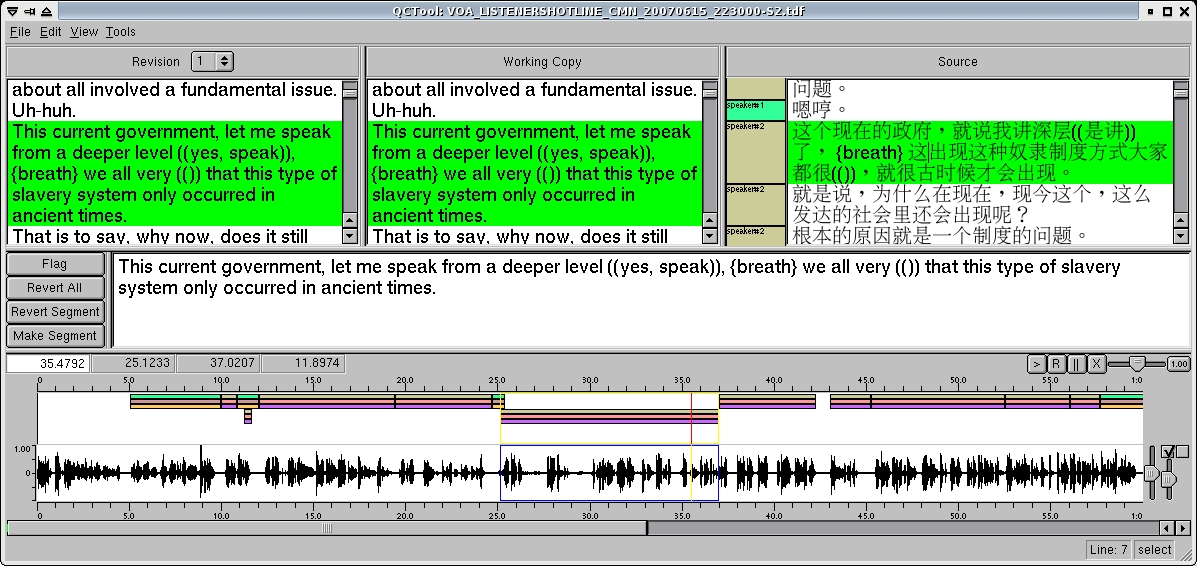


Figure 1.: Screenshot of QCTool.

QCTool was developed by reusing and specializing components of XTrans. A simple analysis shows that QCTool reused 99% of the XTrans code. About 8% of the QCTool code was newly added, due to the object-oriented design of XTrans. For example, the text display required enhancement to ensure that corresponding segments are aligned when displayed on the tree-panel text display. It is this approach that made the rapid development of the tool possible.

As with LDC's other annotation tools, QCTool is fully integrated with LDC's Annotation Workflow System (AWS), which manages documents, directories, permissions, and assignments, and tracks annotator efficiency and progress. Together, QCTool and AWS ensure that each intermediary version of the translation is stored for later training and analysis. In LDC’s production pipeline, QCTool is integrated with AWS for steps 3, 4 and 5 of the six-step process.

Before entering the AWS workflow, the file contains two layers of text: source text and a preliminary translation. At each step of the workflow, AWS adds an exact copy of the previous translation layer. The annotator at each step works on the most recently added layer, which is displayed in the “Working Copy” panel, to correct and improve that layer. One of the previous translations is displayed on the “Revision” panel, and the user can select which revision to display. The source text is displayed on the “Source” panel.

QCTool uses a slightly modified version of the original data model used by XTrans. The original data model is essentially a table where each row represents a segment or sentence. The translation QC process involves several layers of such tables. For example, source text, preliminary translation and corrections from additional QC steps each form a layer. In QCTool, these layers are combined into one by means of a table ID; that is, each row is augmented by an ID of the table it belongs to. The new data model is physically stored using the same file format used by XTrans. However, only QCTool recognizes the additional information to display each row in an appropriate text panel.

QCTool increases the speed and accuracy of the quality control process by giving the annotator access to all of the relevant information in one place, and making it much easier to visualize revisions and identify problematic regions.

**Translation Pipeline Database**

LDC’s core translation infrastructure is grounded in a custom MySQL database that tracks every file at every stage of the translation pipeline and allows the steps outlined above to be followed smoothly, efficiently, and consistently.

This database stores information on data features (e.g. token count, language, genre) and partitions (devtest, training, evaluation); tracks assignments, deadlines, and payments for outsourced translation; and captures information on data location and storage that is exploited by downstream processing scripts. The database is also the backend for the Translation Extranet, which allows files to be stored securely but easily accessed and managed by external agencies and LDC administrators via a simple web interface.

This database is the key to data management of this scale and complexity. Fields are generally populated via batch imports, and files are added at the planning stages so that each step in the pipeline can be tracked and recorded.

Querying allows programmers and managers to drill down and retrieve all of the relevant information associated with a particular file (e.g., which kit it belongs to, when it was delivered, what the QC score was, which agency translated it, whether it has been processed, etc.) as well as all the files associated with a particular stage (e.g., processed but not released), feature set (e.g., Arabic Broadcast News from Phase 2 Training), source (e.g., Xinhua), translation agency, and so on. Processing scripts hook into the database to improve efficiency and allow certain fields to be automatically updated in real time. Additionally, reports synthesizing various points of interest can be generated on the fly for better planning, management, and information-sharing.

**Transcription and Translation File Format**

The output of XTrans and QCTool is a simple Tab Delimited Format (TDF), which is the primary file format for transcripts and translations produced at LDC. This file format represents data as a set of "records", which are in turn a set of "fields" separated by tab characters. Each record refers to a particular segment in the transcript.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Label** | **Description** | **Format** |
| 1 | file | file name or id | Unicode |
| 2 | channel | audio channel | Int |
| 3 | start | start time | Float |
| 4 | end | end time | Float |
| 5 | speaker | speaker name or id | Unicode |
| 6 | speakerType | speaker type | Unicode |
| 7 | speakerDialect | speaker dialect | Unicode |
| 8 | transcript | transcript | Unicode |
| 9 | section | section id | Int |
| 10 | turn | turn id | Int |
| 11 | segment | segment id | Int |
| 12 | sectionType | section type | Unicode |
| 13 | suType | SU type | Unicode |

Table 1.: The 13 fields of a record, or segment, in XTrans .tdf format.

In addition to the body of segments, there are several lines of meta-information in the .tdf file. The first line declares the above field specification for segments in the following form:

file;unicode channel;int start;float ...

The second and third lines specify where the location and types of the section boundaries. For example,

;;MM sectionTypes [u'report', u'nontrans', None]

;;MM sectionBoundaries [0.0, 425.3, 9999999.0]

These lines mean that the first section starts at 0.0 second and its type is "report", and that the second section starts at 425.3 seconds and its type is "nontrans", and this is the last section (9999999.0 is always the last field in this line).

#### Pilot Annotation

LDC transcription and translation efforts aim to maximize the utility of data for GALE teams; this vision includes producing pilot annotations and experimenting with new approaches to provide researchers with additional resources to support system development and testing.

**Supralexical Annotation**

In GALE Phase 3, LDC performed pilot annotation to enrich devtest and unsequestered evaluation transcripts from previous GALE phases. The supralexical annotation pilot aimed to maximize usefulness of existing transcripts for research teams, and involved in-line transcript annotation and named-entity annotation. The supralexical annotation task was performed in two stages: the first stage required annotators to listen to the audio file while annotating the transcript and included the following tasks:

* marking filled pauses
* confirming accuracy of speaker info (gender & native/non-native status)
* adding in-line notation about bandwidth, which included surrounding transcribed words that sounded like telephone speech with <telephone> </telephone> tags, and digitally-transferred field reports with <field report> </field report> tags
* inserting speaker-created and background noises
* adding Arabic dialect descriptions for non-MSA utterances

Since Arabic dialect annotation is a specialist task, supralexical annotators only changed non-MSA tags when they could confidently identify the dialect. Annotators chose one of the following dialect descriptions: (1) Maghrebi, (2) Egyptian/Sudanese, (3) Levantine, or (4) Gulf/Iraqi. When annotators could not confidently identify dialect being spoken, they retained the non-MSA tags from the original transcript. Any region not tagged with a dialect ID or as non-MSA was assumed to be MSA (LDC, 2009).

The second stage of supralexical annotation involved marking named entities as Person, Title, Organization, or Location (LDC, 2006).

|  |  |  |  |
| --- | --- | --- | --- |
| **Language** | **# files BC** | **# files BN** | **Approximate total hours** |
| Arabic | 102 | 152 | 8.5 |
| Chinese | 157 | 144 | 10 |

Table 1.: Data volumes annotated for the Phase 3 supralexical annotation pilot.

**Translation Pilot Experiments**

LDC’s streamlined translation processes have led to a previously impossible level of adaptability to new techniques or datasets and allow quick responses to innovative data ideas. For example, LDC performed a pilot experiment called “MT Plus” wherein source data was run through in-house MT systems and then sent out to translation agencies for correction. GALE sites were interested in comparing MT-based translation to standard from-scratch translation.

Pilot experiments have also been undertaken in translation from English into Arabic/Chinese, in first-pass translation without quality control, and in sentence-level translation. LDC's efficient infrastructure means that resources are maximized and start-up costs are minimized.

* + - 1. Conclusion

This paper described LDC’s efforts to produce training and evaluation data, including transcripts and translations, to support the DARPA GALE program. GALE's challenging performance targets demand linguistic resources on a scale and complexity never before encountered. LDC has developed a robust and flexible pipeline, combining enabling technical infrastructure, detailed task specifications and fully documented best practices. Over the life of the program resource creation at LDC has become more efficient and adaptive, with increased emphasis on automation and utilization of emergent GALE technology to improve and augment the data pipeline. These combined efforts have enabled LDC to meet and often exceed requirements for large volumes of translation and transcription training data, high quality evaluation and development sets, and informative annotation experiments that are responsive to the ever-evolving needs of system developers, evaluators and program sponsors.

# Word Alignment for Improved Machine Translation

Authors: Xuansong Li, Xiaoyi Ma, Stephen Grimes, Stephanie Strassel, Gary Krug, and Dalal Zakhary

#### Introduction

Manual word alignment annotation at LDC is a part of the GALE project and is an ongoing five-year effort. This project has produced Arabic-English and Chinese-English word alignment corpora with the amounts of data shown in the following chart (“P1”, “P2”, “P3”, “P4” for Phase 1, Phase 2, Phase 3, Phase 4; “k” for 1,000):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Genre | P1(k) | P3(k) | P4(k) | Total(k) |
| Arabic | BN | 25 | 79 | 89 | 193 |
| BC | 31 | 105 | n/a | 136 |
| NW | 105 | n/a | 266 | 371 |
| WEB | 94 | 26 | 85 | 205 |
| Chinese | BN | 35 | 95 | n/a | 130 |
| BC | 33 | 75 | 57 | 165 |
| NW | 102 | n/a | 149 | 251 |
| WEB | 102 | 33 | 126 | 261 |

Table 1.: Data volume

#### Data and Tool

**Data type and source**

Two types of source languages involved in the current task are Arabic and Chinese. In the first year of the GALE project, the data types used included newswire, broadcast news, broadcast conversation (talk shows, call-in shows), newsgroups and weblogs. Newswire text came from tree-bank data while other types were harvested at LDC and translated by different translation agencies. In Phase Three of GALE, data types were reduced to broadcast news, broadcast conversation, and web (including newsgroup and weblogs). These data are now or will be tree-banked. The data source is shown via the following chart:

|  |  |
| --- | --- |
| Genre | Chinese Data Source |
| BN/BC | 2005-2006 CCTV, 2005 Phoenix TV, 2006 CCTVNEWS, CCTV4, HUBEI 2007 Phoenix TV, CCTV, CCTVNEWS, China Central TV 2005-2006, CCTV, 2005 Phoenix TV |
| NW | 1994-1998 Xinhua |
| WEB | Online, VOA |

Table 1.: Chinese data source.

|  |  |
| --- | --- |
| Genre | Arabic Data Source |
| BN/BC | 2005 Aljazeera, 2001 Nile TV, 2005-2006 Al Arabiyah, 2007 Saudi Nightly News, Dubai TV, Al Iraqiyah, Kuwait TV News, Lebanese Broadcast Corp, Saudi Nightly News 2005 Aljazeera,2001 Nile TV |
| NW | 2002 An Nahar |
| WEB | Online, VOA |

Table 1.: Arabic data source.

The data was selected from the released source of LDC GALE Phases two and three.

**Tokenization**

For a more precise text and punctuation alignment, both the source and translation were tokenized automatically, without human interventions. The tokenization of English follows the same guidelines used in Penn English Treebank: split words by white spaces, split clitics/contractions (such as shouldn’t, I’m, etc.), separate punctuations from the preceding/following words. Apostrophe S (‘s) is treated as a separate token. Penn English Treebank treats most hyphens as separate tokens; however, some are treated as part of words. Arabic tokenization follows the ATB tokenization scheme by directly extracting tree tokens from ATB for word alignment tasks.

Because of a lack of word boundary, Chinese tokenization is challenging due to segmentation issues. We finally chose character-based files for tokenization with the following considerations. First, word segmented files are normally automatically done by a word segmentation tool and a predefined monolingual dictionary. Yielded errors need to be manually corrected, which is another added process to word alignment itself. Second, character-based system can be easily adapted to any existing segmentation system. Third, character-based files provide the flexibility of ignoring a word boundary when necessary and breaking down a two-character word to minimum semantic units, which greatly benefits cross-cultural findings and more subtle linguistic descriptions. Fourth, the convenience of handling new words is obvious with character-based files, such as proper nouns and newly coined words. These types of words usually require a great effort in manual correction after the texts have been word-segmented. Finally, Chinese input texts at character level can simplify data pre-processing. Therefore, in tokenization we treat each Chinese character as a separate token. All hyphens are separate tokens. In addition, we separate punctuations from the preceding/following characters.

**Data structure and format**

The input data, including both source and translation data, are in tab delimited format (.tdf). The annotation results are stored in an xml-like format, as illustrated in the following:

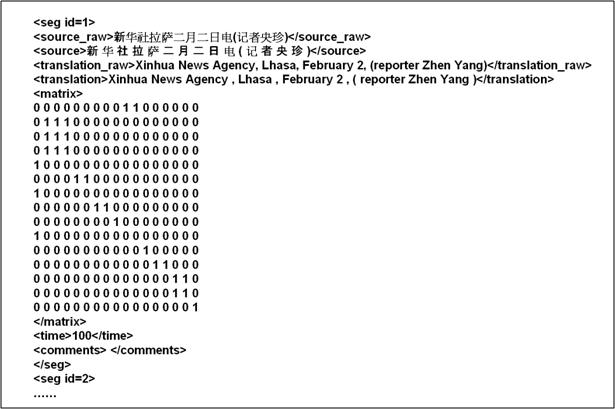


Figure 1.: Data File Format.

The key tags are:

**Source**: the source sentence (in Arabic or Chinese).

**Translation**: the English translation.

**Matrix**: a two dimensional array **M** indicating alignment among words. The rows contain the indices to the English tokens and the columns, the into the source (Arabic/Chinese) sentence tokens. The value of each cell can be 0, 1 or 2.

represents the "Not Translated" English tokens, represents the "Not Translated" source tokens. A value of 1 indicates "Not Translated" and "Correct," meaning a word alignment cannot be found, but the meaning is conveyed in the translation. A value of 2 indicates "Not Translated" and "Incorrect," meaning a word alignment cannot be found, and the translator has made an error (missing or inserted word), which is rare. For example, if M[3,0] is 1, it indicates that the third English token is not translated but the translation is correct.

If is 1 or 2, it indicates there is a link between English token i and source token j. A value of 1 indicates the translation is correct, and value of 2 indicates incorrect, which is rare. There are cases where more than one English token are aligned to more than one source token, in which case, links will be drawn between every English token and every source token. For example, if English tokens i and align to source token j and and will be set to 1 or 2, depending on whether the translation is correct.

**“Unaligned\_sentence” between matrix tags indicates the sentences** rejected by annotators during annotation in cases where sentences are half translated, foreign text, stylistic text of peculiar data types, or locally ill-formatted. For instance, in broadcast conversations, there are fragments of music which are represented as blank lines, and in newsgroups and weblogs, there is foreign text, email address or web links.  All data are encoded in UTF8.

**Tool**

The alignment tool is a simplified version of a JAVA-based word alignment tool developed by Carol Nichols (cln23+@pitt.edu) and Rebecca Hwa (hwa@cs.pitt.edu) at the University of Pittsburgh (Nichols, 2005). The original tool has a client-server model and works for Chinese-English only, and the server side is very CPU intensive. To better serve our task, the server side is discarded and only client side (GUI interface) is kept. Instead of JAVA applet, the tool has been changed to work as a stand-alone GUI application, and Arabic support has been added. The GUI interface has been improved for both visualization and functionality of annotation. The tool presents each pair of tokenized sentences vertically, which are clickable for one-to-one, one-to-many, many-to-one, and many-to-many links. To handle noisy data or style features of different data types which are not appropriate for alignment, a “reject-segment” button was created to conveniently exclude the inappropriate data from alignment while maintaining the integrity of source data.

#### Annotation Guidelines

**Guideline compilation process**

Guidelines compilation began with the annotation of a pilot set of files by native Arabic and Chinese lead annotators. The first guidelines draft was written based on initial annotation results, with reference to the Annotation Style Guide for the Blinker Project (Melamed, 1998). After several rounds of discussions, the draft was revised to form the first version. Two Chinese annotators and two Arabic annotators then joined in and annotated the trial sets according to these guidelines. Their annotation results were examined carefully for differences, which were discussed and tackled again in revising the guidelines. These guidelines were consistently updated to tackle new findings. During the annotation process in phase1 and phase 3, guidelines were updated several times due to newly found un-tackled language phenomena or direct requirements from the GALE research sites. The basic approaches of guidelines, however, remain the same.

**Guideline approaches**

Two versions of guidelines have been developed: Chinese-English and Arabic-English. The same framework is adopted for the two types of guidelines, with the general principles and strategies discussed first, then the specific rules and language issues. Ample examples are provided to support the annotation rules. The most recently updated word alignment guidelines GALE\_Arabic\_alignment\_guidelines\_v3.0.pdf and GALE\_Chinese\_alignment\_guidelines\_v3.0.pdf can be accessed and downloaded from [http://projects.ldc.upenn.edu/gale](http://projects.ldc.upenn.edu/gale/task_specifications/).

Word alignment is a task performed between parallel texts, involving two languages and addressing universal linguistic phenomena as well as cross-cultural features. To respond to universal linguistic issues, general principles are discussed, while specific rules are stipulated to tackle cross-culturally different topics.

**Language universals**

* 1. Alignment types:

Two types of links (translated-correct and translated-incorrect) and two types of markups (“not-translated correct” and “not-translated incorrect”) are established to capture linguistic information and cross-cultural elements.

* 1. Translated:

Pure linguistic translation equivalence refers to semantic “deep structure” equivalence either with or without lexical/functional surface-structure equivalence. A word can be translated in various ways. As long as the meaning is adequately expressed, the word is semantically “translated” in spite of variations in surface-structure. Passive voice in one language can be translated into active voice in another, or there may be divergences in diction, as in:

高兴(happy)死(die)了.

Extremely happy.

All such variant versions are semantic translations of the source, and thus “correct” links. This is true for all languages. Aligning correct links is important to word alignment task because valid translation pairs can be conveniently extracted to construct translation lexicons. Most of the links in our task are of this kind. We designed “Translated incorrect” link type to cover instances where the text is improperly translated in meaning or grammatically wrong.

* 1. Not-translated:

If a word is not translated or conveyed, thus causing information loss, then it is regarded as “not translated”, and naturally is “incorrect.” Therefore, “not-translated incorrect” refers to cases with a loss of deep structure meaning and an absence of surface structure representation, as in the following example, “butter,” is missing in deep structure as well as surface structure in translation.

他(he)买(buy)了牛奶(milk)黄油(butter)和(and)鸡蛋(egg)。

إشترى الحليب والزبدة والبيض.

He bought milk and eggs.

Translating cross-cultural thought inevitably involves translation variations, which are realized by means of expansion, deletion, summarization, explanation, supplement, combination, and reformation in the light of the special needs of readers. Such adaptations and translation variations lead to superficial or surface structure changes, such as overt additions or omissions. These words are extraneous. Deleting them, however, would corrupt the correctness of the sentence, as with determiners which are absent in Chinese but present in English. The additions or omissions are, from a cross-cultural perspective, obligatory and systematic. Special grammatical rules in different languages can be revealed by studying omissions or additions in translation. To this end, the “not-translated correct” markup is designed to label these extras in source/translation.

Our primary concern is how to adequately categorize these superficially extraneous words or phrases. They can be divided into two types: functional (grammatical) and semantic. Furthermore, they can be either at word-level, sentence-level, or discourse level. In English translations, the functional (synsemantic) words, such as prepositions, determiners, subordinating conjunctions, certain particles, auxiliary, and modal verbs are mostly word-level or local level tokens and cannot be neglected. They are so rudimental in revealing the hidden information regarding constituent dependencies that leaving them unaligned would lose valuable dependency information. Moreover, word-level synsemantic words contribute to revealing the complete semantic equivalence between languages, without which the language would be unacceptable. Therefore, synsemantic words at the word level are attached to their autosemantic head words and they jointly constitute “translated correct” links. For instance, “the” can be attached to “reporter,” and “the reporter🡪记者 (reporter)” is an aligned “translated-correct” link. Describing word relations at a local level by the attachment approach is import for word alignment tasks because word relations can help to capture local level syntactic information.

Extraneous words at sentence level usually carry strong cross-cultural or discourse significance. They are not important for semantic equivalence. Without them, the translations are still correct or acceptable. Therefore, these added words with no direct lexical equivalence can be marked as “not translated correct.” They are not-translated in the sense that they have no written equivalents. However, they are correct in the sense that they are needed for being more like English or Chinese, or they are grammatically needed. Language idiosyncrasies play an important role here. Some words appear or disappear depending on the peculiarities of a specific language. For instance, “都 (all)” in the following example does not carry meaning, but it makes the Chinese sentence more fluent. Marking these words is important to machine translation in the sense that a grammatical/rule-based translation model can choose to ignore such words.

国家(nation)每(every)年(year)都(all)拨(allocate)专(special)款(fund)用(use)于(for)开展(develop)残疾人(handicapped)体育(sports)活动(activities).

Every year the nation allocates special funds to be used in developing handicapped sports activities.

Currently, pragmatic and contextual features are also covered under “not-translated correct” category. Discourse-level extraneous words rely on discourse clues and no direct lexical correspondence could be found locally within a phrase or sentence. For these cases, “not-translated correct” is the best solution. For instance, in the following example, the word 中国 (China) is omitted from translation and can be labeled as “not-translated correct.” Here, 中国 (Chinese) 排球 (volleyball) 队 (team) is semantically equivalent to “the volleyball team” if we carry on the same topic. However, if they are examined at the word-level or locally, we can never say that these two terms are semantically equivalent. They are equivalent only by adding discourse clues.

中国(Chinese)排球(volleyball)队(team)昨日(yesterday)抵达(arrive)美国(the U.S.)。中国(Chinese)排球(volleyball)队(team)今天(today)下午(afternoon)与(and)当地(local)纽约(New York)队(team)进行(have)了比赛(match)。

Chinese volleyball team arrived in the U.S. yesterday. The volleyball team had a game with the local New York team this afternoon.

Likewise, insertions or omissions of personal or genre stylistic features may not be translated, and no pair equivalency can be detected. They are not very important semantically, thus are treated as “not-translated correct” if no match is found.

* 1. Minimum Match:

Another alignment issue concerns the recognition of link boundaries. The “minimum match” approach is proposed for finding complete and minimal semantic translation units. In word alignment, the principle of a word-for-word link is strictly followed, that is, the smallest number of words will be preferred, such as the case “鲜(fresh)” in “鲜(fresh)花(flower)” is paired to “fresh” in “fresh flowers” instead of treating “鲜(fresh)花(flower)” as a whole correspondent to “fresh flowers.” This minimum unit approach is important to word alignment because without these minimum unit alignments, minimum syntactic structures units built on these units then cannot be adequately established.

However, some links are many-to-many links since they cannot be separated from each other. Such cases include abbreviations like “世界(world)卫生(health)组织(organization)🡪WTO,” idiomatic expressions/transliteration phrases like “齐(in line with)头(head)并(and)进(march)🡪keep abreast with,” “heart is racing🡪 热(hot)血(blood)沸腾(boil),” “in your dreams 🡪 في المشمش ,” “the youngest child🡪 آخر العنقود ,” and other inseparable cases. Hyphenated words in English are treated as one unit if the subparts of hyphenated words are inseparable. However, if salient one-to-one links exist on both sides, they are paired separately, especially with nonce words.

1. Language peculiarities:

While a hidden matching semantic structure of two languages with cross-cultural features can be revealed by systematically reoccurring omissions/insertions, the study of peculiarities of one language can indicate idiosyncrasies with no systematic matching structure in other languages. A large part of the guidelines is reserved for these language idiosyncrasies with specific rules, which have been augmented with new findings of language peculiarities during annotation process. Arabic guidelines deal with Arabic specific features, including equational sentences, empty subject, cliticization of determiners, prepositions, pronouns, and conjunctions, as well as idioms and particular Arabic interrogative words with no equivalent in English. In Chinese-English alignments, ­­featured idiosyncratic topics include non-inflection, topicalization, measure words, duplication, tense and aspects, various types of helping words, etc. Chinese 的, sometimes described as “evil” or “notorious” due to its uncertain features, requires several annotation rules for a comprehensive description. A glimpse of the following examples can well tell its multi-function and extreme uncertainty idiosyncrasies: adjective modifier as in a); adverb modifier as in b); possession modifier as in c); noun modifier as in d); location (general) modifier as in e); clause modifier as in f); tense modifier as in g). During annotation, 的 is either aligned to its counterparts as in c), d), and f) or attached to related words as in a), b), e), and g).

a) 红红(red) 的🡪red (的attached to红红)

b) 高兴(happy)地🡪 happily (地attached to高兴)

c) 中国(China) 的产品(product)🡪 China’s prouct(的aligned to “‘s”)

d) 人生(life) 的历程(cause)( 🡪 the cause of life(的aligned to “of”)

e) 桌子(table)上(on)的书(book)🡪 book on the table (的attached to上)

f) 经历(experience)过(past)战争(war)的(DE)人(person)🡪 those who have experienced wars(的aligned to “who”)

g) 呈交(submit)的报告(report)🡪 report submitted (的attached to呈交)

#### Evaluation

To test the inter annotation rate, dual annotation is done using AWS workflow system, which blindly assigns a number of files to two annotators at the same time. The results are then computed to show the rate of difference. Because of translation quality and genre of files, the agreement rate for newswire files is obviously higher, as shown in , whereas the rate for web and broadcast news is relatively lower, with a range of 80%-90%. shows the inter-annotator agreement of two annotators on 4 newswire files (F1, F2, F3, and F4). The first row of the table shows the total alignment links from each annotator. The agreement is computed based on their common links. When re-assigning the same file to the same annotator after an interval of about one month, the intra annotation rate shows a good result of 95%-99% for the newswire data.



Table 1.: Inter-annotator Agreement on 4 Newswire Files.

#### Future Work

Word alignment is a complex task requiring an exhaustive description of translation unit equivalence between source and target languages.  The work reported in this paper established a framework for alignment.  With IBM we then jointly designed and deployed an enhanced alignment and tagging annotation scheme as part of the Phase 4 Chinese-English alignment task.  As a next step, this tagging scheme will be implemented for other language pairs including Arabic-English.

Another avenue for further research is the inclusion of additional levels of alignment.  Initial investigations during Phase 4 on automatic post-processing to induce higher level alignment proved fruitful; this has yielded a promising automatic annotation structure for creating multi-level alignments during Phase 5.  The focus of Phase 5 word alignment will be on infrastructure and technologies for streamlining the creation of multi-level parallel aligned Treebank corpora.

# Distillation Training Data

Authors: Zhiyi Song, Heather Simpson, Stephanie M. Strassel, Robert Parker, Kazuaki Maeda

#### Introduction

Distillation is the final stage of language processing in the GALE program, in which relevant information is extracted from foreign language and English input and concisely presented to users in English. It is not a key-word search, and does not involve summarization, but rather it utilizes language analysis techniques to identify information relevant to a user’s query, with the aim of extracting all available relevant information and eliminating redundancy in that information. The GALE Distillation evaluation is designed to quantify the amount of relevant and non-redundant information a distillation engine is able to produce in response to a given template query.

To support the distillation task, LDC creates training corpora, annotation tools and related resources.

#### Training Data Resources

The training resources included:

* Collections of raw source text in English, Chinese, and Arabic, that serve as input to training and test corpora for Distillation
* Queries in English, Chinese, and Arabic that conform to designated templates
* manual annotation in English, Chinese and Arabic for relevant documents
* relevance judgments of system-extracted snippets

#### Distillation Training Corpora

The Phase 1 Distillation training materials were derived from LDC’s TDT4 and TDT5 text corpora. The two corpora contain texts in all three languages (Arabic, Chinese and English) and all four genres (newswire, web data, broadcast news transcripts and broadcast conversation transcripts). The annotation includes:

* searching for relevant documents and providing yes/no judgments
* extracting snippets (noun phrases, clauses or sentences that contain relevant information to the query)
* resolution of pronouns and certain types of temporal and locative expressions
* creating nuggets (atomic pieces of information from a snippet that an annotator considers a valid answer to the query)
* creating nugs (coreference of semantically equivalent nuggets within a language)
* creating supernugs (coreference of equivalent nuggets across languages)

Phase 1 testing involved 10 templates. For this phase, a total of 248 English, Chinese and/or Arabic queries were annotated.

Queries conform to one of the ten template types. Query responses include document and snippet relevance judgments, nuggets, nugs and supernugs. Not all queries have been exhaustively annotated for a given feature, given resource constraints during corpus development.

Phase 2 queries conform to one of seventeen template types. In Phase 2, LDC imposed a three-hour limit on the time that annotators can spend on the annotation of each query, so not all queries have been exhaustively annotated.

In addition, to improve the quality of the training data, relevance judgment, and nugget coreference within and across the source languages were added.

Relevance judgments of system-extracted English snippets produces additional much-needed training data for the teams to improve the systems’ relevance performance. Detailed discussion of this change is in section 1.3.4.4.

To address the coreference problem, the Phase 1 nug and supernug creation were dropped from the annotation pipeline. These tasks were replaced by the entailment judgment task, judging entailment relationships between pairs of English nuggets. This change brings the following benefits:

* Reduces ‘missed coreference’ errors and ‘wrong coreference’ errors.
* Produces a more general-purpose corpus which can be used not just for distillation. There is currently no entailment corpus available outside of the GALE Distillation program, although there are at least 2 projects/challenges where systems compete in this task.
* Produces ‘fully-annotated’ queries. This was not feasible for very large queries with a lot of nuggets in Phase 1 in part due to the complexity of the nug/supernug workflow.
* Improves evaluation of inter-annotator agreement and significantly improved inter-annotator consistency itself.

Take the following two pairs of nuggets as examples:

*Nugget 1: 14 suicide bombers killed 28 weekend revelers*

*Nugget 2: 28 civilians were killed*

*Nugget 1: Tariq Aziz was a minister in Saddam's government*

*Nugget 2: Tariq Aziz was an Iraqi official in the Saddam regime*

In Phase 1, annotators struggled to agree whether the pairs of nuggets needed to be coreferenced, as the nuggets contain related information, but not completely identical information. In Phase 2, instead of coreferencing the nugget pairs, annotators judge the nugget pairs to be either “Nugget 1 is redundant to Nugget 2, Nugget 1 is equivalent to Nugget 2, Nugget 1 conflicts with Nugget 2, or Nugget 1 is not related to Nugget 2”.

In Phase 2, three of the new templates require names, locations, and dates as relevant answers, instead of propositions. These templates are different from propositional templates and therefore, should be handled differently.

* *Template 2 WHAT people |organizations|countries ARE INVOLVED IN [event] AND WHAT ARE THEIR ROLES?*
* *Template 11 FIND ACQUAINTANCES OF [person]*
* *Template 17 WHERE HAS [person] BEEN AND WHEN?*

In Phase 1, there was only one template which falls into this category: *Template 11 FIND MUTUAL ACQUAINTANCES OF [person] AND [person]* which is simplified to *FIND ACQUAINTANCES of [person]* in Phase 2. To illustrate the challenge the non-propositional templates brought up, here are some examples of the nuggets from Phase 1:

*Query: FIND ACQUAINTANCES OF [Howard Dean]*

Snippet: *Democrat Howard Dean has asked retired Army Gen. Wesley Clark to support his presidential campaign if Clark decides not to enter the race*

Nugget: *Howard Dean has asked Wesley Clark to support his presidential campaign*

In the nugget above, *Wesley Clark* is the relevant answer to the query. However, creation of the lengthier propositional nugget is required by both the guidelines and tool kit. The problem with requiring a propositional nugget as an answer is that the smallest proposition encompassing the query answer also includes background information outside of the direct answer to the query, as in the example above. . In Phase 1, this problem was worked around by requiring that evaluation participants extract the list of person entities from LDC’s nuggets.

To solve the problem fundamentally, LDC introduces the annotation of focus core nuggets in Phase 2, with the relevant piece of information that directly answers the query being focused, while the rest of the nugget provides background information. For example:

*FIND ACQUAINTANCES OF [Katie Couric]*

Snippet: *She [Katie Couric] did say that it was a "cool idea," and that her 10-year-old-daughter, Carrie*

Nugget: *She [Katie Couric] has [[a 10-year-old-daughter named Carrie]]*

In this example, *[[a 10-year-old-daughter named Carrie]]* is annotated as the focus core nugget, providing a direct answer to the query.

In phase 2, LDC released the training data containing:

* Snippet and nugget annotation of 255 English queries, 211 Chinese queries and 192 Arabic queries for all 17 templates
* Relevance judgment of 89 queries. For each query, LDC judged up to 50 snippets extracted by systems.
* Entailment judgment of 60 queries.

At present, a corpus was assembled that contains both text and transcribed speech documents in all three languages. Also three more templates were added bringing the total templates to 20. This new corpus contains texts or pointers to previously-distributed texts in all three languages (Arabic, Chinese, and English), and all four genres (newswire, web data, broadcast news transcripts and broadcast conversation transcripts). For English audio data, LDC additionally provided transcripts automatically time-aligned with the source audio via our forced alignment tool. The corpus contain the following annotations:

* Snippet and nugget annotation of 172 English queries, 161 Chinese queries and 167 Arabic queries for all 20 templates
* Relevance judgment of system-extracted snippets for 66 queries.

#### Manual annotation in English, Chinese and Arabic

During the first phase, Arabic, Chinese, or English native speaker junior annotators first issued a query to LDC's search engine and retrieve a set of candidate documents in their language. They read each document to determine whether it contained information relevant to the query, and then identified snippets of relevant text within each document. Pronouns and certain types of temporal and locative expressions were resolved at this stage. They then created a nugget for each fact expressed in the snippet. Semantically equivalent nuggets for a single query and a single language were grouped into nugs by senior annotators. After all the nugs for a given query had been established independently for each language, bilingual lead annotators mapped English and Arabic nugs, and Chinese and English nugs. Arabic and Chinese nugs coreferenced with the same English nug were automatically merged to a trilingual supernug. Arabic and Chinese nugs which are not mapped to English nugs at this stage were reviewed by a team of bilingual lead annotators to map them to each other through their English translation. All Arabic and Chinese nugs were then translated into English. The coreference of nuggets into nugs, and nugs into supernugs, is intended to identify redundant information. The resulting list of supernugs for each query represents a complete list of "facts" in English, drawn from all of the multilingual, multi-genre documents that were considered to contain responses to this query.

Queries for the ten Phase 1 templates were created. Queries conformed to the format of each template filling in the variable information with specific entities or events. The queries were created in English and then translated into Arabic and Chinese so that the same queries are annotated in all three languages.

Since the training data production and evaluation are run by two different agencies, GALE Distillation participants have requested specifically that training data production need to follow the same guidelines used in evaluation data production. LDC, the training data creation team adopted the evaluation team’s relevance judgment and nugget creation guidelines and developed training data annotation guidelines with internally relevant information.

In Phase 2, as stated above, 7 additional templates were added and the annotation for the training data was change to include:

* entailment judgments between pairs of nuggets
* relevance judgment of system-extracted snippets
* entailment judgment of nuggets

**Relevance Judgment Annotation of System-extracted Snippets**

The Relevance Judgment annotation task involves judging system-extracted snippets for their relevance to a given query. Systems extracted up to 50 snippets per query. Annotators judged these snippets as "Relevant" or "Irrelevant", based on the relevance guidelines provided by the evaluation team. Snippets for which the relevancy was impossible to determine, either because of poor translation quality or lack of context, were judged as "Unknown". Selected portions of data are dually annotated for the purpose of assessing annotator consistency.

**Entailment Judgment Annotation**

In the Entailment Judgment annotation task, annotators tagged ordered pairs of nuggets for each query with the following 5 tags: (1) Redundant, (2) Equivalent, (3) Contradicting, (4) NotRelated and (5) NotSure, according to LDC's annotation guidelines. Randomly selected portions of data were dually annotated for the purpose of assessing annotator consistency.

After Phase 2 several major changes were implemented:

Change 1: In Phase 2, snippets consisted of noun phrases, clauses, or sentences. At present, snippets are restricted to sentences only. LDC and GALE Distillation participants defined a shared set of segment boundaries for the entire source data collection to ensure consistency.

Change 2: In Phase 2, annotators selected snippets and then extracted nuggets from those snippets, requiring editing of the snippet text. Now, the nuggetization task is replaced by selection of the relevant portion of the snippet which will be automatically nuggetized by the evaluation team.

Change 3: Since potential snippets are restricted to pre-defined sentence boundaries, snippets could be automatically extracted from the selection of the relevant portion of text. Therefore, to maximize annotation efficiency, LDC replaced the snippet and relevant portion selection tasks with a single task in which annotators extracted relevant portions of text equal to or smaller than a sentence.

Change 4: The Phase 2 entailment judgment task was dropped from the annotation pipeline. LDC focused on providing relevance judgments on system-extracted English snippets, as the GALE participants identified the relevance judgments on system-extracted snippets

Presently, the annotation task has two main components:

* relevant portion of snippet annotation in Arabic, Chinese, and English
* relevance judgment of system-extracted snippets

**Snippet/Relevant Portion Annotation**

Queries are created in English and then translated into Chinese and Arabic. For each language, native speaker annotators first find relevant documents and identify portions of text that contain answers to the query in the source document. They then select the relevant portions of text. The snippets, corresponding to the sentences containing the selected relevant portions, are automatically extracted in the annotation output.

**Relevance Judgment Annotation of System-extracted Snippets**

The Relevance Judgment annotation task involves judging system-extracted snippets for their relevance to a given query. Systems were not given a limit on the number of snippets extracted per query. Annotators judged these snippets as "Relevant" or "Irrelevant", based on the relevance guidelines. Snippets for which the relevancy was impossible to determine, either because of poor translation quality or lack of context, were judged as "Unknown".

#### Annotation Tool Kits

To support the training data annotation, initially, LDC developed a tool kit for distillation which has the following functions:

* Search engine returning a relevance-ranked list of documents
* Document relevance judgment and relevant snippet selection
* Nugget creation
* Nugget coreference
* Nug coreference
* Supernug creation

During Phase 2, LDC developed or modified three annotation tool kits to accommodate the changes in annotation requirements.

**Snippets and Nuggets annotation tool kit**

Due to changes in the distillation task and source data set, a few changes were made to the application to support the Phase 2 tasks- notably disabling of the nug and supernug annotation functionality and enabling the creation of core focus of the non-propositional nuggets.

**Entailment Judgment tool kit**

LDC also developed a simple tool for the entailment judgment task. In this task, annotators made judgments on the entailment relationships between all ordered pairs of nuggets for each query. Annotators were presented with pairs of nuggets, which were created automatically, and chose one of the specified entailment types.

**Relevance Judgment tool kit**

The relevance judgment tool was developed to assist the manual judgment of system output. In this task, annotators judged the relevance of system-generated snippets to a query.

After Phase 2, LDC has developed or modified two additional annotation tool kits.

**Query Creation tool kit**

LDC developed a simple tool for the query creation task, which allowed senior annotators to manually input queries intended for downstream annotation into a form that provided output query XML format.

**Snippet/Relevant Portion annotation tool kit**

Due to changes in the distillation task and source data set, a few changes were made to the application to support the current tasks- notably disabling of the nugget annotation functionality and restricting the selection of relevant portion to the sentence or sub-sentence level, utilizing the segmentation boundaries defined for the source data.

#### Conclusion

Distillation was a new application of natural language processing. Consequently, the task as well as the preparation of training data went through a few changes during the program.

Annotation consistency was a major challenge in Phase 1, as the guidelines provided for relevance judgments and nugget/nug/supernug creation were found to be under-specified when applied to real-world data. The relevancy guidelines used for relevance judgments and nugget creation specified the information which should be considered relevant for each template. The specifications for certain templates, particularly unstructured and open ended templates such as Templates 5 and 9, left significant room for subjective interpretation by the annotator, which lead to inconsistency between annotators and difficulty in making annotation decisions for ambiguous cases.

Coreference of nuggets was also a very difficult task producing errors of missed coreference or wrong coreference, as the nature of the task is too difficult for humans in the case of large queries. Additionally, there was a lot of disagreement in annotator judgment of the threshold for difference between nuggets requiring separation into different nugs.

The complexity of the annotation workflow required for Phase 1 annotation was also challenging, particularly because of the nugget coreference within and across languages. Since it would be unreasonable to rely on having a team of Arabic/Chinese/English trilingual annotators, the approach taken for supernug annotation was to divide the workflow into three language pairs: Arabic and English, Chinese and English, Arabic and Chinese. For annotation of Arabic/English and Chinese/English, bilingual annotators can do the work, while for Arabic/Chinese annotation the annotation is the result of consultation between two bilingual annotators: one English/Arabic bilingual and one English/Chinese bilingual. The complexity of the workflow design has some negative impact on the quality of the nugget coreference.

Addressing the issues in Phase 1 led to a much improved Phase 2, with better task definitions and improved relevancy guidelines, but the accuracy of the system output and the inter-annotator consistency were still concerns, as shown in the two accuracy and consistency studies conducted by LDC.

The accuracy and consistency studies used annotator relevance judgments on system output snippets. The sites provided LDC with system output snippets extracted from the Training data. LDC annotators then judged each snippet as either “relevant”, “irrelevant” or “Unknown”, following the relevance guidelines provide by BAE. A judgment of “relevant” was assigned a value of “1” and all other judgments got “0”. The accuracy of each query then is the value of relevant/total count of snippets.

After the Phase 2 evaluation, LDC also judged relevance for the system responses to each evaluation query. Annotators judged whether a system response to a query was “Relevant”, “Irrelevant”, “Partially Relevant”, “too\_little\_information” or “I\_don’t\_know”, following the relevance guidelines provided by BAE. A judgment of “relevant” was assigned a value of “1” and “partially\_relevant” was assigned “0.5”. All other judgments got “0”. The accuracy of each query is the value of (relevant + partially relevant)/total count of responses.

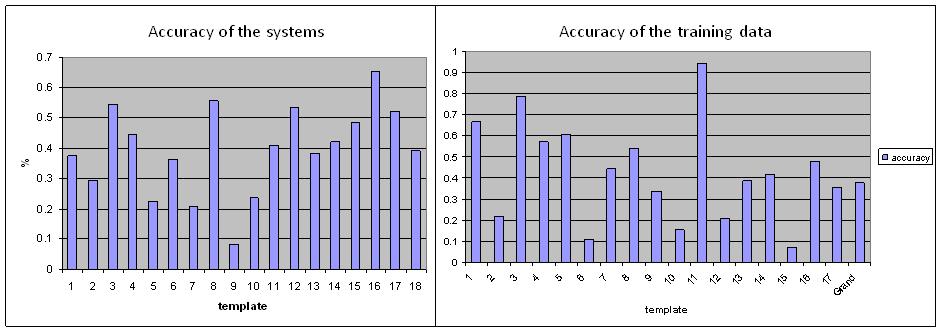


Figure 1.: Accuracy of system output and training snippets.

The average rate of accuracy of the system responses is comparable to that of the training snippets; both are about 39%(see Figure 1.7). Some templates are extremely low; these are also the templates that annotators found very difficult during training data creation.

Inter annotator agreement is also a big concern in regards of relevance judgment. Part of the training system snippets and the entire system response dataset were dually annotated to assess annotation consistency. As shown in Figure 1.8, inter-annotator agreement of both datasets does not reach 80%, with that of the system responses being 59% (left) and that of training snippets 75% (right) which is much lower than the evaluation teams’s inter-annotator agreement which is 89% for the Phase 2 evaluation data annotation (Babko-Malaya, 2008). Since LDC’s two consistency studies were conducted on system output extracted from either English source or Machine Translation materials, the quality of the system output may affect inter-annotator agreement. It would be worthwhile to conduct studies in the future on human generated snippets to see whether the agreement would be higher.

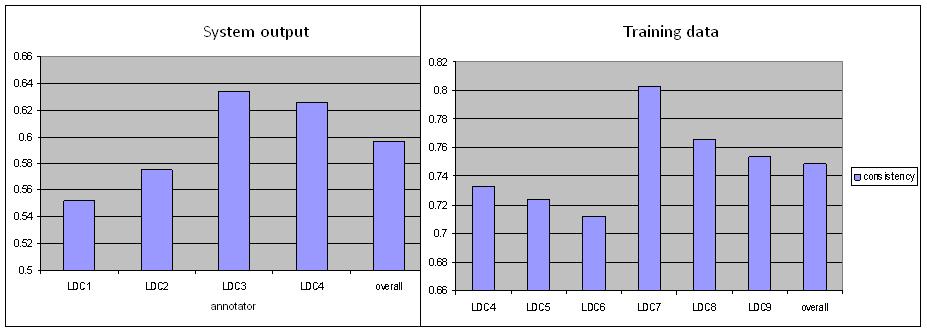


Figure 1.: Inter-annotator agreement

The present distillation task and training system has tried to correct the problems seen in previous phases.

# Arabic Treebanking

Authors: Mohamed Maamouri, Ann Bies, Seth Kulick, Nizar Habash, Reem Faraj and Ryan Roth

#### Introduction

Collections of manually checked syntactic analyses of sentences, or *treebanks*, are an important resource for building statistical parses and evaluating parsers in general. Rich treebank annotations have also been used for a variety of applications such as tokenization, diacritization, POS tagging, morphological disambiguation, base phrase chunking and semantic role labeling.

Two prominent efforts in Arabic Treebanking existed prior to GALE: the Penn Arabic Treebank (PATB) and the Prague Arabic Dependency Treebank (PADT). The main difference between these two resources is the linguistic representation: PATB uses phrase structure and PADT uses dependency representation. Both of these important efforts employ complex and very rich linguistic representations that require extensive annotator training. More concretely, the PATB provides tokenization, complex POS tags and syntactic structure, in addition to diacritizations, lemma choices and some semantic tags (such as TMP and LOC) to distinguish different modifiers. PATB annotation consists of two phases: (a) Morphological/Part-of-Speech (=POS) tagging which divides the text into lexical tokens and includes morphological, morphosyntactic and gloss information, and (b) Syntactic analysis referred to as Arabic Treebanking (=Arabic TB) which characterizes the constituent structures of word sequences, provides function categories for each non-terminal node, and identifies null elements, co-reference, traces, etc. (similar to the Penn English Treebank II style) (Marcus, et al., 1994; Marcus, et al., 1993; Bies, et al., 1995). In addition to the usual issues involved with the complex annotation of data, the work on Arabic treebanking, spearheaded by the LDC’s PATB effort since 2001, has caused the field to come to terms with a number of issues that are specific to a highly inflected language with a rich history of traditional grammar (Maamouri and Bies, 2004) in a variety of ways.

Within GALE, the work on Arabic treebanking was manifested in two efforts. First was the continuing annotation and upgrading of the PATB with a focus on unifying the different PATB parts. Second was a new exploratory annotation effort called the Columbia Arabic Tree Bank (CATiB). CATiB contrasts with PATB (and PADT) in putting an emphasis on faster production with some constraints on linguistic richness. Like PATB, CATiB annotation also consists of two levels of annotation: POS and syntactic. However, because of the different choices made in the CATiB annotation style, these two levels contain different amounts of information. Like PADT, CATiB uses a dependency representation, however with much less morphological information and shallower syntactic relations.

The intent behind both GALE Arabic treebanks is producing more data to serve GALE targets such as improved machine translation using syntactic models. As a result of these efforts, the GALE community now has available to it a variety of morphologically and syntactically annotated Arabic data to use.

This section is divided into two parts describing these two GALE efforts.

#### Upgrading and Enhancing the Penn Arabic Treebank

**Authors: Mohamed Maamouri, Ann Bies, Seth Kulick**

#### Introduction

The LDC Arabic Treebank team has significantly revised and enhanced its annotation guidelines and annotation procedures during GALE. The revision process was initiated by the GALE sponsor and research teams based on lower than expected initial parsing scores and a high level of inconsistencies in the annotation of the Arabic Treebank (ATB) data. Those inconsistencies were the initial targets for improvement in both guidelines and annotator training.

In order to reduce the cognitive load on annotators, both morphological and syntactic annotation guidelines were revised in several respects to be more closely aligned with traditional grammar concepts already familiar to our annotators. We also determined that many of the problems derived from an improper partitioning of the work between different levels, both conceptually and in the actual annotation procedure. Conceptually, subordinating syntactic to semantic needs in certain constructions led to inconsistencies in annotation, as different annotators gave higher priority to one or the other. In order to address such concerns, the decision was made to subordinate semantic needs to syntactic needs in constructions such as إضافة/idafa (noun-noun construct state structures) with quantifiers. Overhauling some key aspects of the morphological/Part-of-Speech guidelines, such as making new, more fine-grained tags for comparatives, quantifiers and numerals, provided more information to the Treebank annotators and to the parser (Maamouri, Bies and Kulick, 2008).

The annotators then underwent a period of intensive training focused on the revised annotation guidelines and difficult linguistic structures, as well as on consistency in general. This training combined with the revised guidelines has resulted in a markedly higher rate of inter-annotator agreement and a more consistently annotated corpus.

We have now completed automatic and partial manual revisions to all of ATB1[[3]](#footnote-3), ATB2[[4]](#footnote-4) and ATB3[[5]](#footnote-5), bringing them into line as far as possible[[6]](#footnote-6) with the new annotation guidelines and greatly improving the annotation consistency and parsing results. The success of this revision process and the improved outcome has been the result of a significant collaboration between data providers and end users.

#### Motivation for Enhanced Annotation

The revision process was initiated based on lower than expected initial parsing scores and on an examination of inconsistencies in the annotation. Parser scores for a statistical parser trained on ATB data were roughly 9 points in absolute f-measure below that of the English Penn Treebank (WSJ) (see section 1.3.5.2.7 for more detail on these scores). Inconsistencies within the Treebank annotation regarding the relationship between Morphological/Part-of-Speech (POS) tags and the syntactic annotation as well as inconsistencies in the annotation of certain syntactic constructions were shown to contribute to the parser performance. Those inconsistencies were therefore the initial targets for improvement both in the guidelines and in annotator training.

Many of the inconsistencies derived from an improper partitioning of the work between different levels, both conceptually and in the actual annotation procedure. Conceptually, subordinating syntactic to semantic needs in certain constructions led to inconsistencies in annotation, as different annotators gave higher priority to one or the other. For example, a quantifier-noun sequence such as “every collection” in Arabic is traditionally expressed in terms of an إضافة/idafa construction, in which the noun is considered dependent on the quantifier, which itself is treated as a noun:

(NP every/all/each\_one |-kul~u |- كُلُّ

(NP collection/group|majomuwEapK|مَجْمُوعَةٍ))

However, in earlier ATB work, this structure was treated as flat

(NP every/all/each\_one|-kul~u|- كُلُّ

collection/group |majomuwEapK|مَجْمُوعَةٍ)

in order to make what is often thought of as the “semantic head” (here, “collection”) more easily accessible to users. However, annotators applied both interpretations, and such structures were inconsistently annotated in the Treebank. The resolution of this type of inconsistency among others led to substantial revision of the annotation guidelines.

#### Improvements to Annotation Guidelines and Procedures

More complete and detailed annotation guidelines overall were developed, and a period of intensive annotator training focusing on the new guidelines and on specific inconsistently annotated constructions followed.

Both POS and Treebank guidelines were revised in several respects, balancing the goals of (1) representing more finely-grained distinctions, and (2) aligning more closely with traditional grammar concepts already familiar to annotators.

#### Morphological/Part-of-Speech Level

We overhauled some key aspects of the POS guidelines, such as making new tags for comparatives and quantifiers, since tagging these simply as NOUN was not informative enough for either the Treebank annotators or the parser. The POS tags for nouns and adjectives in particular were revised to be more fine-grained. In addition to NOUN\_PROP (proper name), the core POS tag of NOUN is now further distinguished as

* NOUN (common noun)
* NOUN\_NUM (number)
* NOUN\_QUANT (quantifier)

The core POS tag of ADJ is also further distinguished as

* ADJ (common adjective)
* ADJ\_NUM (ordinal number)
* ADJ\_COMP (comparative adjective)

The above greater distinctions among nouns and adjectives also follow traditional Arabic grammar categories[[7]](#footnote-7).

Additional POS changes were also made to more closely follow traditional Arabic grammar categories – for example, the number of prepositions was drastically reduced (most of those lexical items are now categorized as NOUN), and many particles are now given several POS alternatives, again closely aligned with traditional categories. For example, the word *fa* had one POS value only in previous Treebank annotation: CONJ. The word *fa* now has four different POS tags available, corresponding to potentially different syntactic contexts and following its four traditional categories: CONJ (for *fa* *Al-EaTf*/فاء العطف , the *fa* of coordination), CONNEC\_PART (for *fA’ Al-rabT/فاء الربط* , the *fa* of connection), RC\_PART (for *fa* *Al-jazA’*/*فاء الجزاء* , the *fa* of reward, response conditional), and SUB\_CONJ (for *fa Al-sababiy~ap*/فاء السببية , the *fa* of causality).

A new POS category of pseudo-verbs has been added to account for the verbal behavior of certain Arabic particles. These are “the sisters of إنَّ <inna” (with the exception of أنَّ “>anna,” the complementizer “that”), a category regarded by Arabic grammarians as having verbal properties, such as subcategorizing for a subject and a predicate or clausal complement. Since these words display verbal behavior although they are not technically verbs, they will now be given the POS tag “PSEUDO\_VERB” and head a VP in the tree.

#### Syntactic/Treebank Level

In order to address concerns such as the inconsistent annotation of quantifiers, the decision was made to subordinate semantic needs to syntactic needs in certain constructions (for example, idafa with quantifiers).

As the idafa structure is a particularly frequent noun phrase structure, this decision affects the annotation of a significant portion of the corpus. In idafa structures syntactically headed by common nouns, the semantic and syntactic head of the noun phrase will be the same noun (as in the “grammar book” example below, where “book” is both the semantic and the syntactic head of the noun phrase).

(NP (NOUN+CASE\_DEF\_NOM

كتاب kitaAbu book)

(NP (NOUN+CASE\_INDEF\_GEN

نحو naHowK grammar)))

كتاب نحوٍ

*(a) grammar book*

vs.

(NP (NOUN\_QUANT+CASE\_DEF\_NOM

every -kul~u – كُلُّ )

(NP (NOUN+NSUFF\_FEM\_SG+CASE\_INDEF\_GEN

collection majomuwEapK مَجْمُوعَةٍ)))

كُلّ مَجْمُوعَةٍ ُ

*every collection*

However, in idafa structures that are syntactically headed by quantifiers (as in the “every collection” example above), the semantic head of the noun phrase is not the quantifier at all, but its dependent noun. The interaction of this idafa structure with the new, more fine-grained POS tags allows the difference in semantic and syntactic heads to be captured. The syntactic/Treebank annotation is based on the syntactic head (the quantifier, “every”). However, the semantic head (the dependent noun, “collection”) is still easily accessible to end-users based on the POS tag NOUN\_QUANT on the quantifier. This interaction of the changes in POS and syntactic annotation guidelines also results in an overall conceptual and practical improvement, since (1) a simple algorithm can recover the necessary semantic information, and (2) inter-annotator agreement is higher.

As with the revision of POS guidelines, the revision of the syntactic annotation guidelines also served to more closely align the Treebank annotation with traditional Arabic grammar categories for several constructions. These include the treatment of comparatives, numbers and numerical expressions and the treatment of several particular pronominal constructions such as separating pronouns/Damiyr Al-faSl/ضمير الفصل and anticipatory pronouns/Damiyr Al$a>n/ضمير الشأن .

Further revisions include a more careful and complete classification of verbs and their argument structure and a more comprehensive approach to the contexts in which gerunds or participles have a verbal reading. For example, a masdar/gerund, active participle or passive participle followed by a PP complement to the regular verb form (PP-CLR) is now shown with a verbal reading.

For a more complete description of the new annotation policies, see the *Arabic Treebank Morphological and Syntactic Annotation Guidelines* (2008)[[8]](#footnote-8).

#### Corrections of Previous Annotation Level

The initial POS annotation is still selected from the morphologically analyzed alternatives provided by the Buckwalter morphological analyzer (BAMA 2004)[[9]](#footnote-9). However, crucial to reducing the number of mismatches between POS tags and syntactic structures is the ability of Treebank annotators to correct certain specified POS tags from the earlier annotation level. The annotation tool has been revised so that Treebank annotators now have the ability to correct specific POS tags such as CONJ 🡪 ADV or PREP 🡪 NOUN. In addition, a careful set of quality control searches based on head rules leads to further manual correction of POS tags in conflict with the tree structure.

#### Improvements in Inter-annotator Agreement and Training

Intensive annotator training focused on agreement and consistency and led to an improvement of inter-annotator agreement scores on the tree structures from an initial f-measure of 86.98% to the much improved f-measure of 94.3%.

The ATB production workflow includes both automatic and manual error correction, along with planned on-going annotator training, and it is hoped that these measures will continue to improve the agreement further. In order to maintain a high rate of inter-annotator agreement, approximately 10% of each corpus is dual blind annotated during production and put through the full workflow. The 94.3% score measured 293 sentences of dual blind annotation from the ATB3 (Annahar) corpus, annotated independently by two Treebank annotators. This score was computed using the standard evalb program (not including function tags).

The initial agreement score was considered to be too low for the purpose of training statistical parsers on the ATB data. The goal was to approach the reported score of 93.8% for the Chinese Treebank. This goal has now been not only met but surpassed, and data produced with this level of agreement is expected to support on-going work on improving parsing results.

#### Revision Process and Quality Assurance

As described above, the annotation process for creating the original corpora consisted of a level of POS/morphological annotation on the original tokens from the text file, followed by separation of various clitic morphemes to create the tokens actually used for treebanking.

The guideline revisions specified changes at various different levels, including tokenization and POS tags as well as the trees. As a result, there are cases in which different decisions are now being taken in the morphological guidelines as to when the original tokens should have been split or not. In order to modify the tokenization to match the current guidelines, it was not possible to do so only by examining individual tokens in the Treebank, since such tokens may themselves be part of a larger original token.

For example, while “limA\*A” formerly existed in the ATB3-v2.0 corpus both as a single token and also split into two tokens (“li” and “mA\*A”), in the revised morphological guidelines it is now treated as one token only. However, the annotation as it existed in the ATB3-v2.0 corpus for the two-token analysis had already split up the word, and the individual Treebank tokens “li” and “mA\*A” were both acceptable tokens unto themselves. It was only in the context of being part of a larger original word that it could be recognized that they needed to be merged back together for this revised release.

Therefore, we created a version of the corpus which associated each original token from the source text file with the one or more Treebank tokens that together make up that original token[[10]](#footnote-10). We then modified the tokenization automatically based on that correlation. This also helped to identify possible tags in some cases for when the range of POS tags was restricted when part of a larger token (e.g., “mA” in the context of “bi+mA” has a smaller range of possible tags compared to “mA” occurring by itself).

The software written for this process was not only useful for automatic changes to adjust the tokenization, but also resulted in a characterization of all original tokens for, roughly speaking, the closed class “function words” that were the focus of the POS revisions. That is, all such tokens were automatically identified in terms of potential component morphemes and possible POS tags for each morpheme (e.g., the “bimA” case just mentioned). This helped ensure the consistency and accuracy of annotation at the token level.

In addition to automatic revisions as above, we significantly improved the post-annotation quality control (QC) process for the ATB. The QC process consists of a series of specific searches targeting several types of potential inconsistency and annotation error, and we increased the number of error searches threefold during the ATB revision process. These error searches are run after annotation is complete, and any errors found via these searches are hand corrected.

#### Parsing Improvement and Analysis

An important goal is to evaluate the increase in parser accuracy as a result of the revisions described in this paper, and to compare the current accuracy to that of parsing on a more established source, namely the Wall Street Journal portion of the English Penn Treebank (PTB). Using a previously proposed data split[[11]](#footnote-11), we trained and tested on each of the revised ATB1, ATB2 and ATB3 individually, as well as the combined ATB123 (738, 845 tokens/words in total), for both the newly revised versions and the older pre-revision releases. In addition, we have trained and tested on an amount of PTB data comparable in size to ATB3, since ATB3 is the largest of the three revised ATB corpora, as well as for the combined ATB123.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tags** | **Parser’s own** | | | **Supplied** | | |
|  | **Old** | **New** | **PTB** | **Old** | **New** | **PTB** |
| ATB3 | 77.5 | 81.0 | 87.6 | 78.5 | 83.2 | 87.2 |
| ATB123 | 78.8 | 82.7 | 88.6 | 79.1 | 84.1 | 88.8 |

Table 1.: Parser Results, with parser choosing its own tags and with parser forced to use given tags.

The parser used was the Bikel adaptation of the Collins parser[[12]](#footnote-12). We ran the parser in two modes. In both, the parser input contains the gold Part-of-Speech tags. The dev section results in Table 1.10 columns 2-4 are for the mode in which the parser used the given tags only for words with which it was unfamiliar from training, and otherwise was free to choose its own tags. In the second mode, shown in Table 1.10 columns 5-7, the parser was forced to use the given tag for each word.

As can be seen in Table 1.10, there is improvement in the score for the parser with the revised data, roughly halfway bridging the gap to the PTB score[[13]](#footnote-13). It is perhaps of note as well that there is a greater distinction in the two ways of running the parser for ATB as compared to PTB. This is perhaps indicative of greater tree/tag consistency in the ATB, or perhaps of a greater share of the burden put on the pos tags. This is a matter for further study, but in both parser modes there is noteworthy improvement for the new compared to the old scores.

In order to better understand the source of the parser improvement, we performed a dependency analysis, as was also done in Kulick, Gabbard, Marcus (2006). Each parser output tree and corresponding gold tree is broken down into a collection of relations, which is a one-level slice of the context-free tree. We have selected some of the most frequent relations for ATB123 and categorized them into two groups, shown in Table 1.11. In this table, the columns are (1) the relation, (2) the frequency of that relation in the new ATB123, and (3-5) the scores for ATB123-old, ATB123-new, and PTB.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Relation** | **% of all relations in ATB123-new** | **ATB123-old  f-measure** | **ATB123-new  f-measure** | **PTB  f-measure** |
| NP 🡪 NOUN NP | 16.75 | 90.4 | 97.4 | n/a |
| PP 🡪 PREP NP | 13.40 | 96.5 | 99.2 | 95.2 |
| Base NP | 12.71 | 84.1 | 90.2 | 95.0 |
| VP 🡪 verb NP | 11.59 | 92.1 | 94.1 | 93.3 |
| SBAR🡪compl S | 2.59 | 91.1 | 92.9 | 92.0 |
| S 🡪 NP VP | 2.03 | 87.4 | 91.3 | 96.3 |
| VP 🡪 VERB PP | 6.44 | 82.6 | 83.4 | 83.5 |
| NP🡪 NPB PP | 3.49 | 73.3 | 75.7 | 86.2 |
| NP 🡪 NP PP | 1.77 | 33.5 | 45.0 | n/a |

Table 1.: Parser accuracy on core syntactic structure relations (top) and PP attachment relations (bottom).

The top portion of Table 1.11 shows relations that make up what might be called the core syntactic structures. For example, the relation NP 🡪 NOUN NP is the idafa construction[[14]](#footnote-14). PP 🡪 PREP NP is the relation of the object of a preposition to the PREP, and so on. A “base NP” (NPB) is an NP without another NP inside it. This table demonstrates the improvement in the parser recovery of these core relations, which seems strongly indicative of increased Treebank internal consistency. One place where there is certainly room for improvement is with the S 🡪 NP VP relation, in which it seems likely that the parser is getting confused over the optionality of the subject placement in Arabic.

The bottom portion of Table 1.11 shows the three relations having to do with PP attachment. Here, while the score for attachment of a PP modifier to a VP is nearly identical to that of the PTB, the score is significantly lower for PP attachment to an NPB (the very low scoring relation for PP attachment to an NP that is not an NPB does not even exist in the PTB). The lower score for the NP 🡪 NPB PP relation is no doubt because of the impact of the idafa structure upon the PP attachment problem, as has been discussed in the literature (see e.g., Kulick, Gabbard, and Marcus, 2006 and Gabbard and Kulick, 2008).

#### Conclusions

We have discussed some of the issues that arise when the Arabic Treebank syntactic annotation is manually enhanced as the first step, ahead of the morphological/Part-of Speech annotation. We outlined an automatic procedure that more closely aligns the POS tags and the Treebank annotation, leading to increased parsing results and additionally providing the annotation pipeline with improved error checking and quality control. The importance of the interaction between POS tags and the tree structure is shown by the increase in parsing results obtained by forcing the parser to use the given tags resulting from this procedure, also indicating the important role that a POS tagger would play in a full Arabic NLP pipeline. In future work, we intend to investigate whether certain of the tags may be more crucial for the parser to get right. It seems reasonable that many of the “function word” particles are particularly crucial.

Revising the annotation of the existing Arabic Treebank corpora to reflect the newly updated guidelines has provided a significantly improved resource to the community. Additional annotation of new data in the improved guidelines style will follow.

The improved ATB guidelines, improving inter-annotator agreement scores, and an expected continuing improvement in parsing scores are the result of a fruitful collaboration between data producers, sponsors and end users, along with the support and time to effect the change. It is hoped that such collaboration will continue to benefit both annotation production and NLP applications in the future.

#### The Columbia Arabic Treebank

Authors: Nizar Habash, Reem Faraj and Ryan Roth

#### Introduction

Under time restrictions, the creation of a treebank faces a tradeoff between linguistic depth and treebank size. This is especially the case for morpho-syntactically complex languages such as Arabic or Czech. Linguistic depth provides the advantage of providing many linguistic features that may be useful for a variety of applications. This comes at the cost of slower annotation as a result of longer guidelines and more intense necessary annotator training. As a result, the deeper the annotation, the slower the annotation process and the smaller the size of the treebank. And consequently, the less data there is to train tools that can benefit from more data.

Two basic ideas inspire the Columbia Arabic treebank (CATiB) approach. First, CATiB avoids annotation of redundant linguistic information. For example, nominal case marks and state (definite, indefinite, construct) in Arabic are determined automatically from syntax and morphological analysis of the words and needn’t be specified by human annotators. Of course, some information is not easily recoverable in CATiB, such as phrasal co-indexation and full lemma disambiguation. Second, CATiB uses a linguistic representation and terminology inspired by Arabic’s long tradition of syntactic studies. This makes it easier to train annotators. CATiB uses an intuitive dependency representation and relational labels inspired by Arabic grammar such as *idafa* (a syntactic construction used for indicating a possession relationship between two nouns) in addition to the universally recognizable predicate-argument labels of *subject* and *object*.

#### CATiB Linguistic Annotation Profile

CATiB uses the same tokenization scheme of the PATB and PADT. However, unlike these resources, the CATiB POS tag set is much smaller. Whereas PATB uses over 400 (tokenized) tags specifying every aspect of Arabic word morphology such as definiteness, gender, number, person, mood, voice and case; CATiB uses six POS tags: NOM (nominals such as nouns, pronouns, adjectives and adverbs), PROP (proper noun), VRB (verb), VRB-PASS (passive verb), PRT (particles such as prepositions or conjunctions) and PNX (punctuation). CATiB uses a dependency representation that models predicate-argument structure (subject, object, etc.) and Arabic nominal structure (idafa [possessive construction], tamyiz [specification construction], modification). Here are all the relations: SBJ (subject of verb or topic of simple nominal sentence), OBJ (object of verb, preposition, or deverbal noun), TPC (topic in complex nominal sentences containing an explicit pronominal referent), MOD (general modifier of verbs or nominals), IDF (idafa in nominal constructions) and TMZ (tamyiz in nominal constructions). No empty categories are made explicit and no co-indexation of phrases. A detailed discussion of the CATiB guidelines is presented in (Habash et al., 2009). Although the CATiB tag set is small, it is appropriate for human annotation purposes. We are able to reproduce a parsing-tailored tag set [size 36] (Kulick et al., 2006) automatically at 98.5% accuracy using features from the annotated trees. Details of this result will be presented in a future publication.

#### CATiB Annotation Process

Although CATiB is independent of previous annotation projects, we made sure to build on top of existing resources and lessons learned. In our CATiB pipeline, we tokenize using the MADA+TOKAN toolkit, trained on PATB (Habash and Rambow, 2005). We initially parse using the MALT parser trained also on a converted version of the PATB (Nivre et al., 2007). The generated trees are then manually checked and modified using the Tred toolkit produced by the team that developed the PADT (Pajas, 2002).

Our annotators are all educated native Arabic speakers – although not necessarily in Arabic linguistics or linguistics in general. The training process took place over two months (150 hours on average). This is much faster than the reported 6-months training for PATB. The speed of annotation of POS and syntax in production mode is over 540 tokens/hour. The speed of verifying the tokenization is over 5K words/hour. In PATB annotation, the reported speed of syntactic annotation alone (no POS) is similar to CATiB’s. When combining this speed with the speed of the PATB POS annotation step, we get an effective 250-300 tokens/hour – almost half the CATiB speed.[[15]](#footnote-15)

We were able to create the CATiB manual from scratch; interview, hire and train annotators; and produce over 228K annotated words all in the span of seven months. In addition to annotation of new data, we automatically converted PATB trees to our CATiB representation – this task is possible primarily because we use less linguistic information. Collectively, the CATiB version 1.0 release contains over 1M tokens (841K words, 31,319 trees), including annotated and converted data.

Our basic inter-annotator agreement scores are around 91.6% on unlabeled parent attachment (for sentences <= 40 tokens) and reach up to 94.7%. A serial annotation experiment, where the annotators corrected already-annotated output, gives results reaching up to 97.8% at 30% overall speed reduction. These results are generally comparable to previous efforts that take longer time to train and annotate. More details about CATiB results are presented in (Habash and Roth, 2009).

# OntoNotes: A Large Training Corpus for Enhanced Processing

Authors: Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, Nianwen Xue

#### Introduction

This paper describes a large multilingual richly annotated corpus that is being made available to the community. We emphasize quality and consistency, targeting interannotator agreement rates at 90%. The data covers multiple genres in English, Chinese, and Arabic, including a significant amount of parallel data. The annotation, intended to capture a skeletal representation of literal meaning, includes parse trees, predicate argument structures, word senses localized in an ontology, coreference, and name types. The resource is delivered as an integrated database, supporting combined queries that access multiple annotation layers. Annual incremental releases are distributed via the LDC.



Figure 1.: levels in OntoNotes.

#### Motivation, Goals, and Rationale

As shown in Figure 1.9, to the baseline structure of parse trees and propositions, OntoNotes adds:

* Referring expressions and the textual phrases they refer to
* Terms disambiguated by word sense and localized in an ontology
* Named Entities

Based on our interpretation of the criteria for successfully applying learning algorithms, our guiding principle has been to find a “sweet spot” in the space of

* Inter-tagger agreement, so that human agreement as a ceiling on algorithm performance is as high as possible.
* Productivity, so that the amount of training data is maximized, given a budget,
* Depth of representation, so that the added semantic features are as deep as possible.

The methodology described here was tested prior to entering production mode, where pilot rounds of annotation were conducted to find the sweet spot above. In particular, only those classes of co-reference satisfying the methodology above during the pilot study are annotated. The methodology has been applied for word sense, with the sense inventory for each word being selected according to the criteria above.

Another dimension of the OntoNotes product is the integration of all of the annotations in a database (Pradhan *et al.,* 2007a), which has at least two benefits:

* Consistency checks on entering each annotation element flag many inconsistencies across annotations for manual correction.
* The data may be searched for phenomena of interest.

This paper illustrates annotation primarily of English, though OntoNotes covers Arabic and Chinese as well. In the next sections, we describe each of the component annotations: treebanking, proposition banking, word sense, ontology creation, coreference, and names. The paper concludes with a summary of related work.

#### Treebanking

The Treebank style in OntoNotes for English is a modification of the Treebank II style for the Penn Treebank (Marcus *et al*., 1993; Marcus *et al., 1994*). For Chinese, the style follows the Chinese Treebank (Xue *et al.*, 2005). (Arabic Treebank annotation is being performed at the Linguistic Data Consortium.) These are annotated with information to make predicate-argument structure easy to decode, including function tags and markers of “empty” categories that represent displaced constituents.

To facilitate merging of the syntactically annotated material with the PropBanked material, both the Treebank style and PropBank style were modified to correct for some small mismatches between the annotations (Babko-Malaya *et el.,* 2006). The major changes to the Treebank stylebook involved modifying the list of verbs considered to take so-called “small clauses” to conform to the argument structures assigned by PropBank, and changing the structures of resultatives to match the PropBank analysis.

The internal consistency of the newly syntactically annotated material for English has been tested, and is quite good. The principal annotator for English reannotated sampled material a year after the original annotation. The F-measure of the newly annotated material against the initial annotation by the EVALB measure was 98.5.

A major editing pass of the OntoNotes Treebank materials is now underway to achieve full consistency with all materials treebanked under the GALE project. The first modification retrofits much of the OntoNotes treebanked materials to conform with the current LDC syntactic style for NPs, crucially adding branching structure whenever the default right branching structure of pre-head modifiers is violated. The second modification changes the tokenization so as to split on most token-internal hyphen. This eliminates some anomalies that resulted from the earlier tokenization, which was based entirely on white space. In “the New York-based company,”, for example, the old tokenization, with “York” attached to “-based” made it impossible to annotate “New York”.

#### Propbanking

PropBanking focuses on annotating the argument structure of verbs, and provides a corpus annotated with semantic roles, including participants traditionally viewed as arguments and adjuncts. The style for English is that of the 1M word Penn Treebank II Wall Street Journal corpus (Palmer *et al.*, 2005). In addition to annotating verbs we are also applying Nombank style annotation to just those nouns with predicate-argument structures that can participate in event coreferences, such as nominalizations and eventive nouns. Links from the argument labels in the Frames Files to FrameNet frame elements and VerbNet thematic roles have been added. This style of annotation has also been successfully applied to other genres and languages. For Chinese, the style is that of (Xue & Palmer, 2009) The same style has also been applied to Arabic (Diab *et. al.*, 2007).

#### Word Sense

One of the daunting challenges was attaining 90% annotator agreement for word sense, since, for example, WordNet inter-annotator agreement averages in the low 70s. Building on results in grouping fine-grained WordNet senses into more coarse-grained senses that led to improved inter-annotator agreement (ITA) and system performance (Palmer *et al.,* 2007), we have developed a process for rapid sense inventory creation and annotation that includes critical links between the grouped word senses and the Omega ontology (Philpot *et al.*, 2005; see Section 5).

Figure 1.10 shows the empirical process for proposing meaningful sense distinctions and determining if they could be annotated at 90% accuracy. A 50-sentence sample of instances is annotated and immediately checked for inter-annotator agreement for all verbs and any noun with frequency over 100. ITA scores below 90% lead to a revision and clarification of the groupings by the linguist. It is only after the groupings have passed the ITA hurdle that each individual group is linked to a conceptual node in the ontology. In addition to higher accuracy, we find at least a three-fold increase in annotator productivity.



Figure 1.: Annotation Procedure

The same methodology has been applied to English, Arabic, and Chinese; the only difference is the starting point for suggesting sense inventories. For English, WordNet has been our starting point of choice. For Chinese, diverse sources are reviewed before hypothesizing an inventory, including entries in web-accessible dictionaries, print dictionaries, samples from the corpora to be annotated and general web searches. For Chinese verbs, the starting point has been the course-grained senses in the frame files created for the PropBank annotation, although print and electronic dictionaries are also consulted. Similar research including access to Arabic WordNet is carried out for each Arabic word prior to authoring its sense inventory file.

#### Verbs

The word sense inventories for English verbs come initially from grouping related WordNet senses (Palmer, et. al., 2007). Subcategorization frames and semantic classes of arguments play major roles in determining the groupings, as illustrated by the grouping for the 22 WN 2.1 senses for *drive* in . The groupings are also linked to the ontology (see Section ). In addition to improved annotator productivity and accuracy, we have found a corresponding improvement in word sense disambiguation performance. Training on this new data, Dligach and Palmer (2008) report 83% accuracy for verbs using a Support Vector Machine and rich linguistic features, which is almost 20% higher than state-of-the art performance on ungrouped, fine-grained senses (Chen and Palmer, 2005). The sense inventories for Chinese and Arabic verbs are created by starting with the PropBank frame files and subdividing the verb entries into more fine-grained senses where deemed appropriate.

|  |  |
| --- | --- |
| GI: operating or traveling via a vehicle  *NP (Agent) drive NP, NP drive PP* | WN1: “Can you drive a truck?”,  WN2: “drive to school,”,  WN3: “drive her to school,”,  WN12: “this truck drives well,”  WN13: “he drives a taxi,”,  WN14: “The car drove around the corner,”,  WN:16: “drive the turnpike to work,” |
| G2: force to a position or stance  *NP drive NP/PP/infinitival* | WN4: “He drives me mad.,”  WN5: “She is driven by her passion,”  WN6: “drive back the invaders,”  WN7: “She finally drove him to change jobs,”  WN15: “drive the herd,”  WN22: “drive the game.” |
| G3: to exert energy on behalf of something *NP drive NP/infinitival* | WN11: “What are you driving at?,”  WN10: “He is driving away at his thesis.” |
| G4: cause object to move rapidly by striking it *NP drive NP* | WN9: “drive the ball into the outfield ,”  WN17 “drive a golf ball,” WN18 “drive a ball” |

Table 1.: Four Groups for “drive”, Compared to the WordNet Senses

#### Nouns

Noun annotation follows a procedure similar to that for verbs. The noun senses are created starting with WordNet and other dictionaries. We aim to double-annotate, at the target agreement level, the 1100 most frequent polysemous English nouns in the entire corpus before the end of 2009, while maximizing overlap with the sentences containing annotated verbs. We have lower targets for the other two languages, which were started later.

Certain nouns carry predicate structure. To ensure conformity with verbs, the structure of nominalizations (*destruction*) and eventive nouns (*party*) is created and assigned by the verb specialists at Colorado.

In order to speed up annotation, we investigated a form of active learning, in which nouns with high agreement in a subset of the whole corpus were used as training data by an automated annotation learner. Unfortunately, different sense distributions across corpora meant that we could not always use results from one year to automatically annotate another year’s data. We investigated various strategies to bootstrap the learning, by mixing into the training data small amounts of annotated data from the new corpus. The results show that even 50 instances from the new distribution permit learning that is accurate enough for about 50% of the high-frequency nouns (Zhu and Hovy, 2007; Zhu *et al.*, 2008).

The OntoNotes (release 1.0) verb and noun word sense data was used in the Semeval-1 (Pradhan *et al.*, 2007b) Overall accuracy over 100 lemmas (65 verbs and 35 nouns) from WSJ corpus, for the best performing system was 86% ― with average over verbs being 78% and over nouns being 89%.

#### Coverage Issues

There are far too many polysemous lexical items for any project to provide exhaustive coverage. Therefore the prioritization of items for annotation is of pressing concern.

Clearly high frequency items provide the most leverage, but they often have a predominant sense (as much as 90% of the data) which can overwhelm annotators with hundreds or even thousands of repetitive examples that will provide little if no system performance improvement. For example, 183 of the 186 instances of the word “bank” in the Ontonotes portion of the WSJ corpus are cases of the first of the 10 senses (a financial institution). In all corpora combined, 607 of the 640 instances of “investment” are the third sense (the activity of investing money for profit). With this type of data, double-blind manual annotation and adjudication is not really necessary.

Of course, prior to the manual annotation the entropy of a word’s sense distribution is unknown. When we are partially through annotation of a given word and it is clear that the majority of its instances fall into one sense, we can then dispense with full double-annotation of all senses, and allow one of the “annotators” to be a trained classifier. This method is described in more detail in (Zhu and Hovy, 2007). This allows for somewhat quicker annotation progress to be carried out for the most common senses, so more time can be devoted to human annotation of rarer words and senses.

The desired aim is a balance between sufficient coverage of high frequency items and maximal coverage of low frequency ones. For these rare words and senses the greatest challenge is finding enough instances to provide adequate training material. We are also exploring techniques such as language modeling for preselecting instances of rare senses from a new corpus (Dligach and Palmer, submitted). In addition we have implemented a data selection plan which supplements our “whole document” based annotation approach with lexical samples for specific lexical items which require greater coverage.

#### Ontology

Standard dictionaries simply list the senses for each word. To support synonym access, inheritance of features and other properties such as predicate frames, links to instances, and so on, we group together the senses that share the same meaning, and then arrange them into a shallow taxonomy that we call the Omega Ontology (Philpot *et al.*, 2005) following the process ofFigure 1.11.

A manual procedure forms *sense pools* by selecting and grouping together individual noun and verb senses that convey the same meaning. (Sense pools correspond to WordNet’s synsets, but are generally less fine-grained.) Each sense pool contains one or more definitions, examples, features, and pointers to the individual senses that comprise it, from which one can access their respective annotated sentences. It is thus possible to assemble for each meaning a set of sentences that contain different target words, each expressing that meaning, in order to train more-powerful sense disambiguation engines.

All sense pools are attached into Omega’s Upper Model (Hovy *et al.*, 2009), a network of some 120 nodes that represent very abstract conceptualizations. Reference to VerbNet semantic classes has been helpful in creating nodes for the verb upper level ontology (Palmer *et. al.*, 2009). To date, about 5000 noun-derived and 3500 verb-derived pools (from English sources) have been created and attached, by multiple annotators who compare their decisions to ensure quality, using a specialized interface. Work is underway to create sense structures also for the other two languages, and either merge them into English-derived pools or attach them to the Upper Model separately. In addition, we have started creating sense structures for the 3000-odd monosemous English nouns occurring in the corpus, and merging or inserting them into Omega.



Figure 1.: Process for adding to the Ontology.

#### Coreference

The coreference annotation in OntoNotes connects coreferring instances of specific referring expressions, primarily NPs that introduce or access a discourse entity. For example, “Elco Industries, Inc.”, “the Rockford, Ill. Maker of fasteners”, and “it” could all corefer. (Non-specific references like “officials” in “Later, officials reported…” are not included, since coreference for them is frequently unclear.) In addition, proper noun premodifiers and verb phrases can be marked when coreferent with an NP, such as linking, “when the company withdrew from the bidding” to “the withdrawal of New England Electric”.

Unlike the coreference task as defined in the ACE program, attributives are not generally marked. For example, the “veterinarian” NP would not be marked in “Baxter Black is a large animal veterinarian”.

However, the sense of “be” is marked so that attributive information is annotated. Adjectival modifiers like “American” in “the American embassy” are also not subject to coreference.

Appositives are annotated as a special kind of coreference, so that later processing will be able to supply and interpret the implicit copula link.

All of the coreference annotation is being doubly annotated and adjudicated. Over the first two years, the overall average agreement between individual annotators and the adjudicated result for non-appositive coreference using the MUC coreference scorer was 86%.

Pradhan *et al.* (2007c) report baseline performance on the OntoNotes coreference data using a standard feature set. Coreference decoding contrasts with decoding other layers in that system performance on coreference still lags very much behind the ITA, in spite of the latter being very high. This is most likely due to the fact that richer semantic and word-knowledge components, in addition to annotation granularity and consistency, are important in identifying co-referring entities. Better learning strategies combined with the accompanying layers in OntoNotes would likely help bridge this gap in the future.

#### Names

Names are also annotated using an 18-type superset of the ACE name guidelines. This supplemental annotation is done in a single pass.

#### Database

Since we are delivering multiple levels of annotation (syntax, propositions, coreference, word sense, ontology, and names), several questions arose:

1. How could we ensure that all the components are consistent with each other, avoiding engineering/formatting inconsistencies?
2. Should the annotations be delivered as independent pieces provided in an integrated representation?
3. What representation would best facilitate use of this information as training data for systems that will be incorporated into applications? Can this representation also support leveraging these additional knowledge sources during the training process?



Figure 1.: Simplified diagram of the interconnections between annotation layers.

We have created a corpus with diverse levels of semantic information integrated in one database (Pradhan *et al.*, 2007a). Figure 1.12 illustrates some of the interconnections captured. The database contains these multiple annotation levels on texts in each of the three target languages.

Each document is stored as a sequence of sentence strings. The token table identifies the word substrings by their offsets, and those tokens in turn are the leaves of a parse tree showing the syntactic structure of the sentence. Supplementary link tables implement many-to-many relations whereby PropBank, word sense, and coreference annotations can be associated with parse tree nodes. Link tables are also used to connect the OntoNotes word senses with the associated PropBank frames. For parallel text that is available in multiple languages, the database also supplies alignment information that maps from each sentence to the associated sentence(s) in the other language.

This process of integrating these disparate types of annotation into a single database identified several types of inconsistencies between the layers. Some of these were theoretical differences For example, the original freestanding PropBank annotation used multiple pointers for a single argument whose immediate value was a trace node that pointed to another, more distant tree node. In the integrated annotation, each PropBank argument is a single syntactic constituent, since the Treebank already provides the trace information.

There were also cases where the syntactic analysis implied by the original PropBank annotation clashed with that in the Treebank. For example, for the phrase “keep their markets active”, the original PropBank analysis had “their markets” and “active” both as arguments of “keep”, while the Treebank treated “active” as the predicate of a small clause. The two groups each made some changes to resolve these inconsistencies (Babko-Malaya *et al.,* 2006).

Other clashes that showed up in the database merge were just incidental, for example, tree structure changes that had been made by one set of annotators but not communicated to the other groups. Resolving all of these inconsistencies helped to ensure a clean, consistent final product, where the relationships between all the layers and within the layers themselves can be efficiently captured in the database schema.

We have also provided an object layer on top of the database layer, written in Python. This layer implements a Python object type for almost every database table. These objects directly represent data values like inverse pointers that are not stored explicitly in the database, and they thus allow for more efficient and flexible data manipulation than when working directly at the level of the underlying database tables, particularly when working across multiple layers. It can also output representations of each individual layers by itself, or a human-readable representation that combines the information in all the layers.

This object layer facilitates defining custom views of the data as well as extracting cross-layer features for use in analysis or in predictive models, neither of which was easily possible before. For example, one could use this API to find out the distribution of named entity types that occur as ARG0 arguments of the predicate “say”, combining information from the Treebank, PropBank, and names layers.

#### Related Work

PropBank I (Palmer *et al*., 2005), developed at UPenn, captures predicate argument structure for verbs; NomBank provides predicate argument structure for nominalizations and other noun predicates (Meyers *et al*., 2004). PropBank II annotation (eventuality ID’s, coarse-grained sense tags, nominal coreference and selected discourse connectives) has been applied to a small (100K) parallel Chinese/English corpus (Babko-Malaya *et al*., 2004). The OntoNotes representation extends these annotations, and allows eventual inclusion of additional shallow semantic representations for other phenomena, including temporal and spatial relations, numerical expressions, deixis, etc.

One of the principal aims of OntoNotes is to enable automated semantic analysis. One state-of-the-art algorithm for semantic role labeling for PropBank style annotation (Pradhan *et al.*, 2005) achieves an F-score of 81.0 using an SVM model. OntoNotes will provide a large amount of new training data for similar efforts.

Other related work falls into two classes: the development of resources for specific phenomena or the annotation of corpora. An example of the former is Berkeley’s FrameNet project (Baker *et al.*, 1998), which produces rich semantic frames, annotating a set of examples for each predicator (including verbs, nouns and adjectives), and describing the network of relations among the semantic frames. An example of corpora annotation is the Salsa project (Burchardt *et al.*, 2004), which produced a German lexicon based on the FrameNet semantic frames and annotated a large German newswire corpus. A second example, the Prague Dependency Treebank (Hajic *et al*., 2001), has annotated a large Czech corpus with several levels of (tectogrammatical) representation, including parts of speech, syntax, and topic/focus information structure. The Tsinghua Chinese Treebank TCT (Zhou, 2003) contains some 2 million Chinese characters, of which half has been treebanked, and manually annotated for syntactic and certain semantic relations, such as causality and conditionals. It covers various genres

Finally, the IL-Annotation project (Reeder *et al.*, 2004) focused on the representations required to support a series of increasingly semantic phenomena across seven languages (Arabic, Hindi, English, Spanish, Korean, Japanese and French). In intent and in many details, OntoNotes is compatible with all these efforts, which may one day all participate in a larger multilingual corpus integration effort.

#### Summary

|  |  |  |  |
| --- | --- | --- | --- |
|  | English | Chinese | Arabic |
| NW | 550 K | 250 K | 300 K |
| BN | 200 K | 300 K | 200 K |
| BC | 200 K | 150 K | – |
| Web | 300 K | 150 K | – |

Table 1.: Planned corpus (token counts).

The plan for the full OntoNotes corpus is shown in Table 1.13, covering three languages and four genres (NewsWire, Broadcast News, Broadcast Conversation, and Web text), and including significant amounts of parallel bilingual data. OntoNotes Version 2.0, released by the LDC in early 2008, covered NW and BN in English and Chinese and NW in Arabic. Version 3.0, to be released in June 2009, will add coverage of BC data in English and Chinese, with additional Arabic NW. It is our hope that this annotation will provide an enduring resource for the community.

# Automatic Annotation

# Speech Segmentation and Its Impact on Spoken Document Processing

Authors: Ostendorf, B. Favre, R. Grishman, D. Hakkani-Tur, M. Harper, D. Hillard,

#### Introduction

Dramatic improvements in automatic speech recognition (ASR) technology make it now possible to explore how language processing techniques designed for text can be applied to spoken language. Ever increasing collections of information are available as speech recordings, including news broadcasts, talk shows, meetings, debates, lectures, hearings, oral histories, and webcasts, among other types of human-directed (vs. computer-directed) communications. ASR can automatically transcribe (albeit imperfectly) the speech in such spoken documents into a stream of words. But to derive content of interest, one would like to be able to apply language processing techniques that have traditionally been developed for written input.

A challenge for the processing of most classes of spoken documents—as compared with text documents—is the lack of overt segmentation information. Text input typically contains punctuation that segments words into sentences and subsentential units. Sentences are further organized into higher-level units such as speaker quotes, paragraphs, sections, chapters, articles, and so on, via formatting. In contrast, when spoken language is processed by an automatic speech recognizer, the output is simply an unannotated stream of words, as shown in the example below. Human listeners can easily segment such spoken input, arriving at the formatted version that follows. To do so they draw on a range of cues, not all of which are fully understood.

|  |
| --- |
| **Unformatted Word Transcripts**  with more american firepower being considered for the persian gulf defense secretary cohen to-day issued by far the administration’s toughest criticism of the u. n. security council without mentioning russia or china by name cohen took dead aim at their reluctance to get tough with iraq frankly i find it uh incredibly hard to accept the proposition that in the face of saddam’s uh actions that uh members of the security council cannot bring themselves to declare that this is a fundamental or material breach uh of uh con-duct on his part i think it challenges the credibility of the security council in europe today secretary of state albright trying to gather support for tougher measures was told by the british and french ... |
| **Formatted transcripts**  Reporter: With more American firepower being considered for the Persian Gulf, defense secretary Cohen today issued by far the administration’s toughest criticism of the U.N. Security Council. Without mentioning Russia or China by name, Cohen took dead aim at their reluctance to get tough with Iraq.  Cohen: Frankly I find it incredibly hard to accept the proposition that in the face of Saddam’s actions that members of the Security Council cannot bring themselves to declare that this is a fundamental or material breach of conduct on his part. I think it challenges the credibility of the Security Council.  Reporter: In Europe today, Secretary of State Albright trying to gather support for tougher measures was told by the British and French ... |

Automatic segmentation is still far from human performance, but significant progress has been made by combining lexical information from a word recognizer, with spectral and prosodic cues. Lexical sequence information provides cues related to syntactic and semantic constraints, and is thus helpful in finding sentence and clause boundaries. For ex-ample, a sentence in English is not likely to end with a determiner. Such cues are, however, subject to degradation from word recognition errors. Lexical cues can also be fairly domain-specific, and may thus perform poorly when training and test data come from different speaking contexts. Spectral information provides cues to speaker and show changes, as well as to non-speech events such as laughter. Prosodic features such as fundamental frequency, duration, and energy patterns provide information about multiple types of segment boundaries. For example, pitch tends to drop before the ends of sentences, and to an even lower value at the end of a topic or paragraph-like unit. Boundaries are often accompanied by pauses and by durational lengthening of phones directly preceding the boundary.

Over the past decade, researchers have explored methods for improving computational models for various levels of segmentation, showing that combining both acoustic and lexical cues provides significant gains in accuracy over a naive pause-based segmentation. More importantly, as will be shown here for a variety of language processing tasks, these segmentations also lead to much better task performance than pause-based segmentations. The experiments described represent a survey of the work of different groups over different time periods controlling for different aspects of segmentation, so the results are not directly comparable. However, the findings together tell a consistent story, specifically that many language processing tasks benefit from linguistic structure (beyond pause units) and that optimizing segmentation for the task is a useful strategy, i.e., the best tradeoff of recall and precision varies depending on the task.

In the remainder of the paper, we describe different types of segmentation useful for spoken document processing, outline popular methods for feature extraction and computational modeling, survey recent results showing the impact of segmentation in several language processing applications, and summarize the findings and open questions.

#### Segmentation in Spoken Language

Language processing technology for text leverages sentence boundary and punctuation information, and the spoken language versions of this technology build on these systems. Thus, sentence segmentation is of particular importance for automatic spoken document analysis and understanding. Sentence boundaries are also important for aiding human readability of the output of automatic speech recognition systems (Jones et al., 2005).

Sentence-level information is but one of many useful levels of structure in language, as evidenced by the additional forms of punctuation (for example, commas) often available in text. For some language analysis tasks, such as parsing and entity extraction, sub-sentence punctuation is of additional value. However, many of these applications may benefit more from an alternative to punctuation: prosodic phrase boundaries. Speakers naturally group words into semantically coherent phrases indicated by timing and pitch cues; these prosodic phrase boundaries often coincide with major syntactic constituent boundaries (particularly those marked with commas and semi-colons) but have a much flatter structure than syntax. They provide smaller (and potentially more useful) units for processing.

Applications of segmentation above the sentence level depend on genre. For example, topic segmentation is important when processing news broad-casts that include multiple stories. Similarly, speaker tracking and possibly role or identity recognition can provide useful structure in genres with multiple speakers. Simply knowing who is speaking (even without an associated name) can improve the read-ability of a speech transcript when there is more than one person talking. Speaker tracking is also useful for automatic analysis of conversation or meeting dynamics and for attribution in question answering. Both speaker and topic segmentation can be useful in speech recognition, for acoustic and language model adaptation, respectively.

#### Computational Modeling Techniques

Two very different types of segmentation algorithms are used: audio diarization and structural segmentation. Audio diarization aims to segment an audio recording into acoustically homogeneous regions, given only features extracted from the audio signal. Audio diarization techniques can include a variety of tasks, such as distinguishing speech from mu-sic or advertisements from news. The term structural segmentation is used here to include tasks that represent linguistic structure (commas, sentences, story/discourse), for which algorithms leverage both acoustic and lexical cues. The two classes of algorithms are treated separately below, followed by a discussion of how different types of segmentation may be combined. We provide an overview of the most popular methods in each case, with the disclaimer that technology is still evolving in this field.

#### Speaker Diarization

Much of the foundation for speaker diarization comes from speaker recognition research; some of the earliest systems were developed to support work on speaker identification in broadcast news. A typical speaker diarization system may be broken down into several “standard” components (Tranter and Reynolds, 2006), with the two main components being “segmentation” and “clustering.” During the segmentation step (or “speaker change detection”), boundaries between acoustic events (typically due to a change of speaker) are located to create homogeneous segments of audio. The dominant approach to segmentation involves computing a generalized log likelihood ratio at candidate boundaries, comparing the likelihoods of the data using two distributions for the subsets of data to the left and right of the boundary vs. a single distribution for the combined set. To determine the cut-off point, typically some form of regularization or prior is used, such as the Bayesian Information Criterion, which effectively adds a penalty for increased numbers of parameters. Then, during clustering, all of the segments belonging to the same speaker are grouped together. The most common approach for the initial speaker clustering is hierarchical agglomerative clustering, which begins with a large number of clusters that are merged pair-wise. The number of speakers is not known a priori, so various heuristics are used to determine the stopping point. Determining the number of speakers can be difficult in applications where some speak only briefly (e.g., in news sound bites), since they tend to be clustered with other speakers.

The segmentation and clustering steps may be iterated until some stopping criteria is satisfied. In subsequent passes, different models may be used, such as hidden Markov models (HMMs) for joint segmentation and clustering. Multi-pass methods are useful for the challenge of handling speaker overlap (in talk shows) and handling noisy conditions (reporters calling in from the field). The most common features used in audio diarization are cepstral features and their derivatives, as in speech recognition except without the normalization aimed at factoring out speaker and channel differences since these are exactly the types of differences that are targeted in diarization.

Speaker diarization performance is typically measured by diarization error rate (DER), which measures the percentage of time that a system incorrectly labels the audio recording based on an automatic mapping of hypothesized speaker clusters to reference speakers according to maximal overlap.[[16]](#footnote-16) Diarization error rates vary depending on the number of speakers, speaker overlap, and acoustic conditions. Hence, DER can vary widely, e.g., from roughly 2% to 12% for different broadcast news sources in (Tranter and Reynolds, 2006).

#### Structural Segmentation

There are two basic modeling approaches used for structural segmentation: 1) detection of boundary events and 2) whole constituent modeling. The approaches can also be combined. Both models are applied after speech recognition, and take advantage of the alignment between words (and the phones therein) and the acoustic speech signal.

Boundary event detection is essentially a sequence tagging problem: for each word in the sequence, assign a boundary label indicating the constituent (or none) ending at that point. As such, any computational model for tagging is applicable here. HMM-like models dominated early work in speech segmentation (Stolcke and Shriberg, 1996). Given the word sequence W and the prosodic features *F*, the most likely event sequence *E* is given by:

The transition probabilities (in) are obtained from an *n*-gram language model, also referred to as a hidden-event language model, that characterizes the event labels and words jointly. The observation posteriors are generated from a prosody model, e.g., a decision tree classifier or neural network. HMMs that are discriminatively trained have also been used for sentence boundary detection (Tomalin and Woodland, 2006).

Maximum entropy (Maxent) and conditional random field (CRF) classifiers have also been investigated for boundary event detection (Huang and Zweig, 2002; Liu et al., 2006). Unlike HMMs, Maxent and CRF approaches provide more freedom to incorporate contextual information and to combine word-based and prosodic features. Both use the exponential form for the conditional probabilities. For example, in Maxent:

A CRF models sequence information, whereas Maxent individually classifies each data sample. The weights (λ) for the features are estimated to maximize the conditional probabilities of the training set. In (Liu et al., 2006), HMM, Maxent and CRF approaches are compared for sentence segmentation of broadcast news and conversational speech, finding that the CRF leads to the best results but by a small margin and at a higher computational cost. Another approach that can accommodate a rich variety of features is based on combining Boostexter with a hidden-event language model (Zimmermann et al., 2006). Boostexter is based on the principle of boosting that combines many weak classifiers, each having a basic form of one-level decision trees using confidence-rated prediction. It has the advantage of good performance with a relatively low cost implementation.

Whole constituent modeling considers both the beginning and the end time of a segment in determining boundary location. For many problems, the cues are local to the boundary, such as for prosodic phrase boundaries. For others, the cues extend over the entire phrase, and the whole constituent approach is preferable. Whole constituent modeling is also useful when a maximum or minimum length constraint is needed. The challenge of modeling the full constituent is in decoding. Since it is impractical to consider all possible constituent onset times, the search space is often reduced by restricting the set of candidate boundaries. Whole constituent modeling has been used for sentence segmentation, story segmentation, and in speaker modeling where both acoustic and lexical cues are incorporated. In sentence segmentation for translation (Matusov et al., 2006), an explicit sentence length model is incorporated in a log-linear combination of language model and prosody model scores. Posterior probabilities identified via boundary event detection can be included in the combination for further improvements (Matusov et al., 2007). In story segmentation, whole constituent modeling is needed for characterizing the topical coherence of sentences in the segment and extracting position-based information about lexical cue words (Rosenberg et al., 2007).

The modeling approaches described above rely on various word-based and prosodic features. Lexical features typically consist of word *n*-grams and part-of-speech *n*-grams. These features are useful for identifying short utterances in spontaneous speech such as backchannels (“uhhuh”, “yeah”), for characterizing sequences of words that are unlikely to be split by a sentence boundary (“the problem”), and for representing words that are likely to start a new sentence (such as “I”). Syntactic features have also been used to improve sentence boundary detection (Roark et al., 2006; Favre et al., 2008b). The features have different representations in different modeling approaches, for example, an *n*-gram language model in the HMM framework or word tuple indicators in discriminative classifier approaches.

Prosodic features reflect information about duration, pause, intonational and energy contours. Features can be extracted from automatic alignments of word and phone transcriptions with the speech signal. Duration features (such as word, pause, and phone durations) are obtained directly from alignment time marks. Since different phones have different baseline durations, duration features are typically normalized for phonetic content. In addition they may be normalized by speaker or speaking rate. Pause duration is known to be important for segmentation, but it poses some challenges because of editing in broadcast news and speaker overlap in conversational speech. Useful pitch and energy features tend to capture differences across the word boundary in question, as well as slopes and normalized level of pitch or energy just before a boundary. In the case of both pitch and energy, features must be appropriately normalized (by speaker for pitch; by channel for energy).

Sentence segmentation is reported either using an F-measure or the sentence error rate, which computes error rate as the number of incorrect boundaries over the total number of boundaries using the mdeval tool.[[17]](#footnote-17) Performance depends on speech genre and recognizer error rate. Conversational genres tend to have shorter sentences but higher error rates. With the caveat that performance continues to improve, example F-measures on automatically transcribed speech are roughly 65% for broadcast news (Favre et al., 2008a) and 64% for broadcast conversations in English. The F for Mandarin broad-cast news is about 75% (Matusov et al., 2007).

#### Multi-level Segmentation

Since the various types of segmentation are generally interdependent and since automatically detected boundaries can be errorful, soft predictions (boundary posteriors) at the different levels can be considered jointly to improve performance. Speaker boundaries based purely on acoustic information of-ten do not align perfectly with sentence boundaries that are based on speech recognizer output. Higher accuracy speaker boundaries can be obtained by adjusting boundary times to match those of nearby sentence boundaries. Similarly, improved story boundary detection is achieved by considering candidate boundary points at more locations than the automatically detected sentence boundaries, either by lowering the threshold for sentence detection (e.g. from probability 0.5 to probability 0.1) or simply by considering all boundaries with a 250ms or greater length pause (Rosenberg et al., 2007). Taking into consideration the higher-level information associated with story boundary detection can potentially feedback into improvements in sentence segmentation. The use of soft decisions on segment boundaries also makes it possible to tune the boundary detection threshold or operating point for specific applications. Work described in the next section shows that this is indeed useful, though the best operating point varies with the different tasks.

#### Applications

Spoken document processing can involve a combination of several tasks, typically starting with speech recognition and speaker segmentation, followed by some basic linguistic analysis such as part-of-speech tagging and parsing, and then involving higher level processing such as translation and information extraction. Automatic segmentation touches on all of these problems, but we will focus on stages after speech recognition. In the examples here, the segmentation types used (speaker, sentence, comma, intonational phrase, and story) employ the basic algorithms described in the third section of this paper. The specific methods vary with genre and with the time period of the work, since this is still an area of research and the best case configurations are evolving.

#### Speaker Role and Identity Recognition

In broadcast news, most speech is from anchors and reporters, but there are excerpts from speeches or interviews, sometimes referred to as “soundbites.” Detecting soundbites and associating them with particular speakers is important for information extraction and attribution in question answering. Using the example in the first section, the task is to associate the speech segments produced by Cohen with his name, given speaker diarization results and ASR transcripts.

Data from Mandarin broadcast news has been used for soundbite and speaker name recognition (Liu and Liu, 2007) using a classification frame-work. Each speaker turn is labeled with one of three roles: anchor, reporter, or soundbite. The features used are based on textual information (mainly word n-grams) from the current segment, the preceding and the following segments. Speaker name recognition takes advantage of the coded behavior typical of broadcast news (i.e., reporters often naming the next or previous speaker). Hypothesized names from the current and neighboring segments are classified in terms of whether or not it is the speaker’s name for a target soundbite segment using keywords and position of the name in the sentence.

In experiments with Mandarin broadcast news, word errors and sentence segmentation errors have different impacts on the soundbite detection and name recognition tasks due to the types of cues they use. For soundbite detection, segmentation errors were more harmful than word errors, with degradation in F-measure of 12% vs. 2%, respectively, com-pared to detection on oracle transcripts. Incorrect sentence segmentation leads to missed cue words to soundbites. For name recognition, the opposite is true: 1% degradation of F-measure due to segmentation errors, compared to 15% with word errors. Since many soundbite speaker names are infrequent, they are less reliably recognized than other words.

#### Tagging and Parsing

Part-of-speech (POS) tagging is the process of marking up a sequence of words with their parts of speech (e.g., noun, verb). Parsing produces a structural analysis of a word sequence with respect to a grammar. High quality automatic sentence segmentation is important for utilizing these techniques most effectively, both for accuracy and for addressing length-dependent complexity issues. Experiments on the impact of comma prediction on POS tagging accuracy of Mandarin broadcast news speech showed that performance was significantly better when using the automatic commas compared to a matched train/test condition with-out sentence-internal punctuation (Hillard et al., 2006). Other work on parsing English conversational speech showed a significant effect of sentence segmentation on parsing whether using the reference word transcriptions (Kahn et al., 2004) or ASR transcripts (Harper et al., 2005): automatic sentence segmentation can recover roughly half the loss in parsing performance due to using pause-based segmentation compared to hand-labeled references (roughly 10 points F-measure recovered).

Further experiments (Roark et al., 2006; Harper et al., 2005) showed that optimizing sentence segmentation thresholds specifically for parse accuracy (vs. sentence segmentation accuracy) yielded greater improvements in parsing (from F of 64.8 to 65.7). When optimizing for parse accuracy, the system tended to produce shorter segments than when optimizing for sentence segmentation accuracy, i.e. trading off precision for recall. The shorter sentence-like segments also benefited a parsing language model used in speech recognition, leading to significant improvements in the SParseval score when word sequences and parses are chosen jointly.

#### Information Extraction

Information Extraction (IE) aims at finding semantically defined entities in documents and characterizing relations between them. Like many text processing tasks, IE systems typically benefit from hand-written punctuation. Studies show that when punctuation is removed, there is an associated loss in performance. For example, missing commas have a dramatic impact on IE (Makhoul et al., 2005), with performance loss typically bigger than that observed when moving from reference to ASR output (for a range of word error rates on English news). Similar results were reported for name tagging on Mandarin broadcast news (Hillard et al., 2006), and it was shown that half of the lost performance was re-covered with automatic comma prediction (from F of 84.9 to 85.4).

Another study (Favre et al., 2008a) confirmed these observations for English IE on speech, and found that optimizing sentence and comma prediction thresholds for IE performance is more effective than optimizing these thresholds separately for punctuation prediction accuracy: improvements in the ACE scores are from 15.6 to 18.4 for relations, and 47.0 to 48.2 for entities. Error analysis showed that punctuation errors can result in merged noun phrases or split entities. The best case performance was obtained by jointly optimizing comma and sentence boundary thresholds but allowing the thresh-olds to vary in detecting entities vs. relations.

#### Machine Translation

In machine translation (MT), sentence segmentation helps provide translations with proper punctuation, but it also impacts word choice since sentence boundaries are incorporated in the language model and they constrain the possible phrase translations. Many systems limit sentence lengths for complexity reasons, which motivates a constituent-based approach to sentence segmentation. The translation application also motivates a new type of feature, introduced in (Matusov et al., 2007) to characterize phrase coverage of the words that span the candidate boundaries to ensure that word sequences with good phrasal translations will not be broken by a segment boundary. The phrase coverage feature is a bigram language model probability. Depending on whether the bigram probability is high or low, there is likely to be a good phrasal translation in the system or not, respectively.

Different sentence segmentation algorithms have been evaluated on large vocabulary Arabic-to-English and Chinese-to-English broadcast news translation tasks using the phrase-based MT system of RWTH (Mauser et al., 2006). The explicit length modeling of the whole-constituent model (using a less sophisticated prosody model and without the phrase coverage feature) did not do as well as the boundary detection approach in terms of sentence segmentation accuracy, but it did lead to better MT performance. MT performance improves by combining the two methods, but the best result was achieved by using the phrase coverage feature, increasing Bleu from 18.1 for fixed-length segments to 21.2 for the MT-optimized sentence predictions. The sentence boundary precision is reduced significantly when the phrase coverage feature is used, but this does not affect the translation because the context at the erroneously inserted boundaries was not captured in MT training anyway. As in the parsing work, MT experiments have shown that a lower detection threshold is better for translation of Chinese (0.2 vs. the minimum error threshold of 0.5), favoring recall over precision or shorter segments. A separate study on Arabic-to-English translation found that longer sentences are better (Matsoukas et al., 2007). Shorter sentences in Chinese are likely to help limit reordering errors, while for Arabic (which has less long distance reordering), longer segments likely provide additional context without much increased risk of reordering mistakes.

While punctuation marks predicted in ASR output can be useful for predicting target language punctuation, they can be also used to guide the MT process. In (Matusov et al., 2007), automatically predicted Chinese commas were used as soft boundaries for reordering in MT search. Reordering across a comma is found to be highly unlikely and is penalized by modifying the lexicalized re-ordering model of the phrase-based MT system (Zens and Ney, 2006), with reduced penalties for lower confidence predictions. In experiments on the Chinese-to-English task, the soft comma constraints did not result in a significant improvement in standard MT development scoring methods (Bleu and Ter). However, the word order in several translated sentences was subjectively better when the soft boundary penalty was applied.

#### Conclusions

In summary, the fact that most language technology used in spoken document processing is designed in large part from written text argues that speech must be made to look more like text for achieving good performance. An important challenge in this respect is speech segmentation, including sentence segmentation at a minimum, but ideally also speaker and topic segmentation for formatting and adaptation, as well as sub-sentence punctuation and/or intonational phrase prediction. Both event-based and constituent-level computational models have been developed for these problems, many of which combine lexical and acoustic cues in detecting boundaries. While these algorithms are far from perfect, in most applications they provide a much better solution than simple pause-based segmentation. Oracle results suggest that further improvements to segmentation algorithms would be useful, though improvements to word recognition have higher impact for some tasks.

In the various applications surveyed here, there is a consistent finding that tuning the segmentation thresholds for the application leads to significant performance improvements over using the threshold that minimizes segmentation error. In many cases (but not all), higher recall is more effective (i.e. shorter sentences), but the optimal threshold varies. This raises the question as to how best to meet the needs of multiple language processing modules, particularly when they all operate on the same hypothesized transcript. One solution is to use a low threshold (more hypothesized boundaries) with confidences associated with the boundaries, so that different downstream modules can use their own threshold. Alternatively (or in addition), the need for different thresholds may reflect a need for different types of structures, including sub-sentence units such as intonational phrases or syntactic chunks.

# CADIM Arabic Tools: Morphological Analysis, Disambiguation and Generation, Tokenization, Diacritization, Lemmatization, POS Tagging and Base Phrase Chunking

Authors: Mona T. Diab, Nizar Habash, Owen Rambow, and Ryan M. Roth

#### Introduction

In this survey paper, we describe some of the NLP tools we developed over the past number of years to process Modern Standard Arabic and Arabic Dialect text. We present a series of tools that vary in their dependence on explicit linguistic rules and linguistic depth, and that target a variety of NLP tasks. AL-MORGEANA and MAGEAD are morphological analysis and generation systems. AL-MORGEANA extends on the Buckwalter analyzer and MAGEAD handles Arabic dialects. MADA is a system for morphological disambiguation including POS tagging, lemmatization and diacritization. TOKAN is a general tokenizer for Arabic that works with MADA. AMIRA is a suite of shallow linguistic processing tools with limited dependence on explicit linguistic rules. AMIRA performs tokenization, POS tagging, and base phrase chunking. We compare some of the functionality of our different systems. We also present several of the NLP applications that have successfully used our different systems.

**Morphological Analysis** refers to the processby which a word (defined orthographically for our purposes) has all of its possible morphological analyses determined. Each analysis also includes a single choice of core part-of-speech (such as noun or verb; the exact set is a matter of choice). A morphological analysis can be either form-based, in which case we divide word into all of its constituent morphemes, or functional, in which case we also interpret these morphemes. For example, in broken (i.e., irregular) plurals, a form-based analysis may not identify the fact that the word is a plural, since it lacks the usual plural morpheme, while functional analysis would.

**Morphological disambiguation** refers to thechoice of a morphological analysis in context. This task for English is referred to as part-of-speech (POS) tagging, since the standard POS tagset, though only comprising 46 tags, completely disambiguates English morphologically. In Arabic, the corresponding tagset comprises up to 330,000 theoretically possible tags, so the task is much harder. Reduced tagsets have been proposed for Arabic as well, in which certain morphological differences are conflated, making the morphological disambiguation task easier. The term POS tagging is usually used for Arabic with respect to some of the smaller tagsets.

**Tokenization** (also sometimes called*segmentation*) refers to the division of a word into clusters of consecutive morphemes, one of which typically corresponds to the word stem, usually including inflectional morphemes. There are two parameters. First, we need to choose which types of morphemes to segment (the tokenization scheme). There is no single correct tokenization scheme. Second, we need to decide whether after removing some morphemes, we regularize the orthography of the resulting segments, since the concatenation of morphemes can lead to spelling changes on their boundaries. For example, the Ta-Marbuta ( ة ħ) appears as a regular Ta (ڌ t) when followed by a pronominal clitic. Usually, the term *segmentation* is only used when no orthography regularization takes place. Orthography regularization is desirable in NLP because it reduces data sparseness, as does tokenization itself.

**Lemmatization** is the mapping of a word formto its corresponding lemma (also known as a *citation form*), the canonical representative of its lexeme. A lexeme is a lexicographic abstraction: it is the set of all word forms that differ only in inflection and possible cliticization, processes which are (almost) always productive. The lemma is a conventionalized choice of one of these word forms to stand for the set. Usually, a lexeme is assumed to have one sense, so that homonyms (such as the two meanings of English bank) are considered two lexemes. If the sense distinction is removed from the definition of lexemes, one sometimes uses the term vocable, which is a purely morphological characterization of a set of word forms. In this chapter, we use the term lexeme without the semantic dimension, i.e., we collapse the concepts of lexeme and vocable.

**Diacritization** is the process of adding missing diacritics (short vowels, the marker of the absence of a short vowel, and the gemination marker). Diacritization is closely related to morphological disambiguation and to lemmatization: for an undiacritized word form, different morphological analyses often correspond to different diacritizations (for example, voice in MSA), and different lemmas can be seen to lead to different diacritizations as well. Often, the choice of the diacritic on the last written letter of the word (without the possessive or object clitic which may be attached) is particularly hard, since it requires syntactic information: in verbs, this diacritic often expresses mood, and in nouns and adjectives, it expresses syntactic case. Thus, it is often common to define a simpler diacritization task which does not choose the word final diacritic.

**Base phrase chunking** is a syntactic task,in which non-recursive multiword syntactic phrases are identified without actual parsing. The relevant notion of phrase needs to be clearly defined depending on where and how these syntactic chunks can be used, and in Arabic, this is not as straightforward as in English due to the complexity of some of the syntactic structures of noun phrases.

The Columbia Arabic and its Dialects Modeling (CADIM) group has produced a set of tools to address these tasks. In this chapter we will present the functionality of several of our basic tools that have been widely used in the NLP community and in our own research. This chapter is a survey chapter, it is meant to highlight the various tools that we have and how they compare with one another. We will also refer to some of the applications where we used our tools.

Our tools can be classified coarsely into a set of deep and shallow processing tools as they vary in their dependence on explicit linguistic representations in their machinery. The deep processing tools depend on explicitly expressed linguistic analyses as an essential component in the processing of the text, and require corpora that have been annotated at this level. The shallow processing tools, on the other hand, learn from surface representations where the morphological information and linguistic knowledge is implicitly represented and the system learns pattern generalizations without reference to explicitly represented linguistic units.

#### ALMORGEANA

ALMORGEANA is a system for Arabic Lexeme-based Morphological Generation and Analysis (Habash, 2007a).[[18]](#footnote-18)

**Desiderata**

We chose the following desiderata forthe design of ALMORGEANA (and also MAGEAD, Section 1.4.4.3): (1) coverage of the language of interest in terms of both lexical coverage (large scale) and coverage of morphological and orthographic phenomena (robustness); (2) a complete mapping between surface word forms to/from a functional representation of morphology; (3) full reversibility of the system so it can be used as an analyzer or a generator; (4) usability in a wide range of natural language processing applications such as MT or IR; and finally, (5) availability for the research community.

**Lexicon**

ALMORGEANAis built on top of the publicly available large-scale database of the Buckwalter Arabic Morphological Analysis (BAMA) (Buckwalter, 2002; Buckwalter, 2004). The database consists of six components: three morphological databases for prefixes, stems, and suffixes; and three tables showing the compatibility of entries from the three morphological databases with each other (prefix-stem, stem-suffix and prefix-suffix). Unlike BAMA, which focuses on analysis to a surfacy form-based representation, ALMORGEANA analyzes to, and generates from the functional (lexeme-and-feature) level of representation. To that effect, the ALMORGEANA lexicon extends the BAMA morphological databases with lexeme and feature keys, which are used in analysis and generation. This work on ALMORGEANA is close in spirit to the ex-tensions to BAMA in the functional morphology sys-tem, ELIXIRFM (Smrž, 2007).

**Analysis**

Analysis in ALMORGEANAis similar toBAMA: the word is segmented into prefix-stem-suffix triples, whose individual presence and bi-lateral compatibility is checked against the BAMA database. The difference lies in an extra step that uses lexeme and feature keys associated with stem, prefix and suffix string sequences to construct the lexeme and feature output. The output of ALMORGEANA includes all possible analyses. Each analysis consists of the diacritized form of the word, its lexeme, its morphological features, and an English gloss. For example the word الكتب *llktb*[[19]](#footnote-19) ‘for the books’ returns the following analysis:

lilkutubi=[kitAb\_1 POS:N l+ Al+ +PL +GEN]=books

Here, *lilkutubi* is the diacritized form of the word. Inside the square brackets, we find the nominal lexeme *kitAb\_1* ‘book’, the proclitic preposition l+ ‘to/for’, the definite article *Al*+ ‘the’, the feature +PL ‘plural’ and the feature +GEN ‘genitive case’. Most of the information in the feature set is directly derivable from the morpheme tags in the BAMA output for the same word: li/PREP+Al/DET+kutub/NOUN+i/CASE\_DEF\_GEN. How-ever the feature +PL indicating plurality is not. It is part of the extension done in ALMORGEANA in processing the BAMA databases.

The challenge of out-of-lexicon words during analysis is handled in two ways similar to BAMA. First, simple spelling expansion is used for cases with common spelling variations such Alef-Hamza- Above (أ Â) spelled without the Hamza (ء ’) as Alef (ا A), and Ya (ي y) spelled without the dots as Alef-Maqsura (ى ý). Second, a back-off analysis mode can be turned on in which prefix and suffix matching and compatibility is used but unattested stems can be hypothesized. A full description can be found in (Habash and Rambow, 2005).

**Generation**

In generation, the input is a lexemeand feature set. The generated output is a fully inflected and diacritized word. For example, [kitAb\_1 POS:N l+ Al+ +PL +GEN] generates *lilkutubi*. The process of generating from lexeme and features is similar to analysis except that lexeme and feature keys are used instead of string sequences. First, the feature set is expanded to include all forms of under-specified obligatory features, such as case, gender, number, etc. Next, all lexeme and feature keys in the ALMORGEANA lexicon that fully match any subset of the lexeme and expanded feature set are selected. All combinations of keys that completely cover the lexeme and expanded feature set are matched up in prefix-stem-suffix triples. Then, each key is converted to its corresponding prefix, stem or suffix string. The same compatibility tables used in analysis are used to accept or reject prefix-stem-suffix triples. Finally, all unique accepted triples are concatenated and output. In the case that no surface form is found, a back-off solution that attempts to regenerate after discarding one of the input features is explored.

See (Habash, 2007a) for more details on ALMORGEANA and an evaluation of its performance.

#### MAGEAD

MAGEAD is a morphological analyzer and generator for the Arabic language family, by which we mean both MSA and the spoken dialects. We shall collectively refer to MSA and its dialects as Arabic variants. For a fuller discussion of MAGEAD (including an evaluation), see (Habash et al., 2005; Habash and Rambow, 2006). For an excellent discussion of related work, see (Al-Sughaiyer and Al-Kharashi, 2004).

MAGEAD relates (bidirectionally) a lexeme and a set of linguistic features to a surface word form through a sequence of transformations. In a generation perspective, the features are translated to abstract morphemes which are then ordered, and expressed as concrete morphemes. The concrete templatic morphemes are interdigitated and affixes added, and finally morphological and phonological rewrite rules are applied. In this section, we discuss our organization of linguistic knowledge, and give some examples; a more complete discussion of the organization of linguistic knowledge in MAGEAD can be found in (Habash et al., 2006).

**Lexeme and Features**

Morphological analysesare represented in terms of a lexeme and features. We define the *lexeme* to be a triple consisting of a root, a *morphological behavior class* (MBC), and a meaning index. We do not deal with issues relating to word sense here and therefore do not further discuss the meaning index. It is through this view of the lexeme (which incorporates productive derivational morphology without making claims about semantic predictability) that we can both have a lexeme-based representation, and operate without a lexicon (as we may need to do when dealing with a dialect). In fact, because lexemes have internal structure, we can hypothesize lexemes on the fly without having to make wild guesses (we know the pattern, it is only the root that we are guessing). Our evaluation shows that this approach does not wildly overgenerate.

We use as our example the surface form, ازدهرت *Aizdaharat* (*Azdhrt* without diacritics) ‘she/it flourished’. The MAGEAD lexeme-and-features representation of this word form is as follows:

(1) Root:zhr MBC:verb-VIII POS:V PER:3 GEN:F NUM:SG ASPECT:PERF

**Morphological Behavior Class**

An MBC mapssets of linguistic feature-value pairs to sets of abstract morphemes. For example, MBC verb-VIII maps the feature-value pair ASPECT:PERF to the abstract root morpheme [PAT\_PV:VIII], which in MSA corresponds to the concrete root morpheme *V1tV2V3*, while the MBC verb-II maps ASPECT:PERF to the abstract root morpheme [PAT\_PV:II], which in MSA corresponds to the concrete root morpheme *1V22V3*. We define MBCs using a hierarchical representation with non-monotonic inheritance. The hierarchy allows us to specify only once those feature-to-morpheme map-pings for all MBCs which share them. For example, the root node of our MBC hierarchy is a word, and all Arabic words share certain mappings, such as that from the linguistic feature conj:w to the clitic w+. This means that all Arabic words can take a cliticized conjunction. Similarly, the object pronominal clitics are the same for all transitive verbs, no matter what their templatic pattern is. We have developed a specification language for expressing MBC hierarchies in a concise manner. Our hypothesis is that the MBC hierarchy is variant-independent, i.e. dialect/MSA independent. Although as more variants are added, some modifications may be needed. Our current MBC hierarchy specification for both MSA and Levantine, which covers only the verbs, comprises 66 classes, of which 25 are abstract, i.e., only used for organizing the inheritance hierarchy and never instantiated in a lexeme.

**MAGEAD Morphemes**

To keep the MBC hierarchy variant-independent, we have also chosen a variant-independent representation of the morphemes that the MBC hierarchy maps to. We refer to these morphemes as *abstract morphemes* (AMs). The AMs are then ordered into the surface order of the corresponding concrete morphemes. The ordering of AMs is specified in a variant-independent context-free grammar. At this point, our example (1) looks like this:

[Root:zhr][PAT\_PV:VIII] [VOC\_PV:VIII-act] + [SUBJUF\_PV:3FS]

Note that as the root, pattern, and vocalism are not ordered with respect to each other, they are simply juxtaposed. The ‘+’ sign indicates the ordering of affixival morphemes. Only now are the AMs translated to *concrete morphemes* (CMs), which are concatenated in the specified order. Our example becomes:

(3) <zhr,V1tV2V3,iaa> +at

Simple interdigitation of root, pattern and vocal-ism then yields the form iztahar+at.

**MAGEAD Rules**

We have two types of rules. *Morphophonemic/phonological* rules map from the morphemicrepresentation to the phonological and orthographic representations. For MSA, we have 69 rules of this type. Orthographic rules rewrite only the orthographic representation. These include, for examples, rules for using the *shadda* (consonant doubling diacritic). For Levantine, we have 53 such rules.

For our example, we get /izdaharat/ at the phonological level. Using standard MSA diacritized orthography, our example becomes Aizdaharat (in transliteration). Removing the diacritics turns this into the more familiar ازدهرت Azdhrt. Note that in analysis mode, we hypothesize all possible diacritics (a finite number, even in combination) and perform the analysis on the resulting multi-path automaton.

We follow (Kiraz, 2000) in using a multi-tape representation. We extend the analysis of Kiraz by introducing a fifth tier. The five tiers are used as follows: Tier 1: pattern and affixational morphemes; Tier 2: root; Tier 3: vocalism; Tier 4: phonological representation; Tier 5: orthographic representation. In the generation direction, tiers 1 through 3 are al-ways input tiers. Tier 4 is first an output tier, and subsequently an input tier. Tier 5 is always an out-put tier.

We have implemented multi-tape finite state automata as a layer on top of the AT&T two-tape finite state transducers (Mohri et al., 1998). We have defined a specification language for the higher multi-tape level, the new MORPHTOOLS format. Specification in the MORPHTOOLS format of different types of information such as rules or context-free grammars for morpheme ordering are com-piled to the appropriate LEXTOOLS format (an NLP-oriented extension of the AT&T toolkit for finite-state machines, (Sproat, 1995)). For reasons of space, we omit a further discussion of MOR-PHTOOLS. For details, see (Habash et al., 2005).

**From MSA to Levantine**

We have modifiedMAGEAD so that it accepts Levantine rather than MSA verbs. Our effort concentrated on the orthographic representation; to simplify our task, we used a diacritic-free orthography for Levantine developed at the Linguistic Data Consortium (Maamouri et al., 2006). Changes were done only to the representations of linguistic knowledge, not to the processing engine. We modified the MBC hierarchy, but only minor changes were needed. The AM ordering can be read off from examples in a fairly straightforward manner; the introduction of an indirect object AM would, for example, require an extension of the ordering specification. The mapping from AMs to CMs, which is variant-specific, can be obtained easily from a linguistically trained (near-)native speaker or from a grammar handbook. Finally, the rules, which again can be variant-specific, require either a good morphophonological treatise for the dialect, a linguistically trained (near-)native speaker, or extensive access to an informant. In our case, the entire conversion from MSA to Levantine was performed by a native speaker linguist in about six hours.

#### MADA

MADA (Morphological Analysis and Disambiguation for Arabic) is a utility that, given raw Arabic text, adds as much lexical and morphological information as possible by disambiguating in one operation part-of-speech tags, lexemes, diacritizations and full morphological analyses. This task is not trivial, due to the morphological complexity of the language. The complexity of the morphology together with the underspecification of the orthography creates a high degree of ambiguity. On average, a word form in the Penn Arabic Treebank (Maamouri et al., 2004) has about 12 morphological analyses.

MADA’s approach distinguishes between the problems of morphological analysis (what are all the different readings of a word without regard to con-text) which is handled by a morphological analyzer such as ALMORGEANA or MAGEAD, and morphological disambiguation (what is the correct reading in a specific context). MADA is a morphologucal disambiguation system. Once a morphological analysis is chosen in context, we can determine its full POS tag, lemma and diacritization in a single step. Knowing the morphological analysis also allows us to tokenize and stem deterministically; this is handled by TOKAN (Section 5) once MADA has finished processing the text.

MADA operates in stages. First, it uses ALMORGEANA (Section 2) internally to produce a list of potential analyses for each word encountered in the text; at this point, word context is not considered. MADA then makes use of up to 19 features to rank the list of analyses. For each feature, a classifier is used to create a prediction for the value of that feature for each word in its context. Fourteen of the features use Support Vector Machine (SVM) classifiers; the remaining features capture information such as spelling variations and n-gram statistics.

Each classifier prediction is weighted using a tuning set, and the collection of feature predictions is com-pared to the list of potential morphological analyses. Those analyses that more closely agree with the weighted set of feature predictions receive higher ranking scores than those which do not; the highest scoring analysis is flagged as the correct analysis for that word in that context.

Once MADA has finished, the user has access to the full morphological analysis for each word in the input. The user may then extract any or all of the analysis information. This is why MADA can be used for a multitude of tasks, including part-of-speech and morphological feature tagging, lemmatization, predicting full diacritization, glossing, stemming and others. Since MADA selects a complete analysis from ALMORGEANA, all decisions regarding morphological ambiguity, lexical ambiguity, tokenization, diacritization and POS tagging in any possible POS tagset are made in one fell swoop (Habash and Rambow, 2005; Habash and Rambow, 2007; Roth et al., 2008). MADA has over 96% ac-curacy on basic morphological choice (including tokenization but excluding syntactic case, mood, and nunation) and on lemmatization. MADA has over 86% accuracy in predicting full diacritization (including syntactic case and mood). Detailed compar-ative evaluations are provided in the following publications: (Habash and Rambow, 2005; Habash and Rambow, 2007; Roth et al., 2008) .

The operation of MADA is versatile and highly configurable. Starting with version 2.0, MADA applies weights to each of the 19 features it uses for better accuracy; these weights are determined on a tuning set and are optimized for different purposes, such as tokenization, diacritization, or POS tagging. These weight sets are included with the package and should be chosen by the user depending on how MADA will be used. However, users can also choose to set these weights directly themselves. By default, MADA attempts to rank complete analyses in terms of overall correctness. By choosing an alternative feature and weight set, it is possible to have MADA focus more specifically on getting a particular analysis aspect correct. For example, users can achieve a 0.4% absolute improvement in POS tagging accuracy if they use the weight set that was tuned for POS tagging, as opposed to the default set. How-ever, the accuracy of the other MADA outputs (the lexeme prediction, for example) may suffer. Starting with version 2.1, MADA also includes a morphological backoff procedure, which can be turned on or off by the user (see Section 2).

#### TOKAN

TOKAN is a general tokenizer for Arabic that pro-vides an easy-to-use resource for tokenizing MADA disambiguated Arabic text into a large set of possibilities (Habash and Sadat, 2006; Habash, 2007a). The decision on whether an Arabic word has a con-junction or preposition clitic is made in MADA; but the actual tokenization of the clitics including han-dling various morphotactics and spelling regularization is done in TOKAN. The tokenization scheme can be used as parameter in machine learning for a variety of applications, such as machine translation, or named-entity recognition (see Section 8).

TOKAN takes as input a MADA-disambiguated file and a tokenization scheme description that specifies tokenization target. Consider the following specification:

“w+ f+ b+ k+ l+ s+ Al+ REST + / + POS +P: +O: -DIAC”

This scheme separates conjunctions, prepositions, verbal particles, the definite article and pronominal clitics and it adds the basic POS tag to the form of the word. The scheme also specifies that diacritics are generated. An analysis of the word وسيكاتبها

wasayukAtibuhA ‘and he will correspondwith her’ would be tokenized as “wa+ sa+ yukAt-ibu/V +hA.” A simpler scheme such as “w+ f+ REST” would simply produce “w+ sykAtbhA.” See (Habash and Sadat, 2006) for a detailed description of several schemes that have become commonly followed since that work was published. TOKAN has a large number of other features that allow the user to perform different kinds of orthographic normalizations or control how the output is ordered and presented as it may fit different needs of different systems. Table 1.14 exemplifies a few of the TOKAN supported schemes with the same sentence.

Internally, TOKAN uses morphological generation (through ALMORGEANA) to recreate the word once different clitics are split off. This approach of back generation allows us to modify the morphological content in a word including, for instance, deleting/defaulting specific features of a word easily. We do this to guarantee the form of the generated word is normalized and consistent with other occurrences of that word. For example, simply splitting the pronominal clitic off a word with Ta-Marbuta ( ة ħ) would keep the Ta-Marbuta in its word-internal form (regular letter Ta,ت t). With TOKAN, the Ta-Marbuta is generated as appropriate (e.g., جولته jwlth ‘his-visit’ is tokenized into ه + جولة jwlh+h ‘visit +his’, not ه+ جولت jwlt+h (which is not a valid spelling).

#### AMIRA

AMIRA is a set of tools built as a successor to the ASVMT toolkit developed at Stanford University (Diab et al., 2004) and described in detail in (Diab et al., 2007b). The toolkit includes a tokenizer, a part of speech tagger (POS) and a base phrase chunker (BPC), also known as a shallow syntactic parser. The technology of AMIRA is based on supervised learning with no explicit dependence on knowledge of deep morphology, hence, in contrast to the previous tools, it relies on surface data to learn generalizations. In general the tools use a unified framework which casts each of the component problems as a classification problem. The underlying technology uses Support Vector Machines in a sequence modeling framework using YAMCHA.[[20]](#footnote-20) The AMIRA tools are highly accurate, very fast, and robust. They allow for a limited number of variable user settings depending on the disambiguation granularity. The AMIRA tools have been widely used for different NLP applications due to their speed and high performance, such as MT, IR, Parsing, NER, and IE.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| وسينهى الرئيس جولته بزيارته الى تركيا | | | | | | |
| Input | wsynhý | Alrˆyys | jwlth | bzyAr\_*ħ* | Alý | trkyA |
| Gloss | and will finish | the president | tour his | with visit | to | Turkey |
| English | The president will finish his tour with a visit to Turkey. | | | | | |
| **Scheme** |  |  |  |  |  |  |
| ON | wsynhy | Alryˆys | jwlth | bzyAr\_ *ħ* | Aˇ lý | trkyA |
| D1 | w+ synhy | Alryˆys | jwlth | bzyAr\_ *ħ* | Aˇ lý | trkyA |
| D2 | w+ s+ ynhy | Alryˆys | jwlth | b+ zyAr\_ *ħ* | Aˇ lý | trkyA |
| D3 | w+ s+ ynhy | Al+ ryˆys | jwl\_ *ħ* +h | b+ zyAr\_ *ħ* | Aˇ lý | trkyA |
| WA | w+ synhy | Alryˆys | jwlth | bzyAr\_ *ħ* | Aˇ lý | trkyA |
| TB1 | w+ synhy | Alryˆys | jwl\_ *ħ* +h | b+ zyAr\_ *ħ* | Aˇ lý | trkyA |
| TB2 | w+ s+ ynhy | Alryˆys | jwl\_ *ħ* +h | b+ zyAr\_ *ħ* | Aˇ lý | trkyA |
| MR | w+ s+ y+ nhy | Al+ ryˆys | jwl +\_ *ħ* +h | b+ zyAr +\_ *ħ* | Aˇ lý | trkyA |
| LEM | Ânhý | ryˆys | jwl\_ *ħ* | zyAr\_ *ħ* | Aˇ lý | trkyA |
| POS1 | V | N | N | N | P | PN |
| POS2 | VBP | NN | NN | NN | IN | NNP |
| ENX | w+ s+ ÂnhýV BP +S3MS | Al+ rˆyysNN | jwl\_ *ħ* NN +h | b+ zyAr\_ *ħ* NN | Aˇ lýIN | trkyANNP |
| DIAC | wasayun.hiy | Alr\_ayˆiy.su | jaw.latahu | biziyAra\_ *ħ* ˜ı | Aˇ ilaý | tur.kiyA |

Table 1.: Some of the preprocessing schemes supported by TOKAN: ON (orthographic normalization), D1, D2 and D3 (different degrees of decliticization), WA (wa+ decliticization), TB1 and TB2 (old and new Arabic Treebank tokenization, respectively), MR (morphemes), LEM (lemmatization), POS1 and POS2 (two possible POS tagsets), ENX (a tokenization equivalent to D3+LEM+POS2 with markers for verbal subject) and DIAC (diacritization)

From the user perspective, the different components of the tool suite could be invoked sequentially, taking raw text in any encoding and producing clitic tokenized, POS tagged and/or base phrase chunked data. They could also be applied directly on some given text. For example POS tagging could be applied on raw text without explicitly invoking tokenization. We briefly review the three main components of the AMIRA suite.

#### AMIRA-TOK

A tokenization scheme is a convention chosen by the researcher, and there are many possibly tokenization schemes. For AMIRA-TOK we focus primarily on clitic tokenization. Clitics in Arabic are typically affixes that latch onto base words. They are mor-phemes that have a syntactic role and affect the orthography and phonology[[21]](#footnote-21) of words. AMIRA tools do not rely on morphological analysis or genera-tion tools in any of its processes. Hence, AMIRA-TOK learns clitic tokenization generalizations from the clitic segmentations present in the Penn Arabic Treebank (PATB) directly without relying on rules explicitly.

**AMIRA Clitics** In addition to the standard PATBclitic tokenization, AMIRA-TOK can also segment off the nominal definite article ال Al. Accordingly, we have modified the training data to include segmentation of the ال Al.[[22]](#footnote-22) The motivation behind seg-menting off the Al is that it has a positive impact on higher order NLP applications especially those involving correspondence to other languages where the definite article is a stand alone word. Moreover, this segmentation reduces the sparseness associated with nominal forms.

In total, AMIRA-TOK segments off the following set of clitics: conjunction proclitics (prefixes) و w, ف f, prepositional proclitics ك k, ك l, ب. b, future marker proclitic س s, verbal particle proclitic ل l, definite article proclitic ال Al), and pronominal enclitics (suffixes) indicating possessive/object pro-nouns. In general, nominals can have 0- 3 possible proclitics from the conjunction, prepositions and definite article set. Verbs can have 0- 2 proclitics from the set of conjunctions and the future clitic marker.[[23]](#footnote-23)

Since multiple proclitics are allowed in Arabic, they follow a specific order if present. The definite article ال Al is always closest to the base word, followed outwardly by the prepositions, then by the conjunctions. Likewise for the verbal proclitics, the future marker or verbal particle is the clitic affixed to the verb directly, followed outwardly by the conjunction. Accordingly, in the example بالحسنات و wbAlHsnAt, ‘and by the virtues’, the proclitic tokenization will be rendered as follows:

و w+ب b+ الAl+ حسنات HsnAt where و w is the conjunction and, ب. b is the preposition by, and ال Al is the definite article ‘the’.

The modeling of clitic tokenization in AMIRA-TOK is exactly the same

**Character Chunking** For AMIRA-TOK, we ap-ply a chunking scheme on the character level casting the problem as a chunk boundary identification and chunk classification problem. We use an IOB annotation scheme, every character in our data (including punctuation) is annotated as: inside a chunk (I), outside a chunk (O), or beginning of a chunk (B). For the I and B tags, we have five possible classes: Pre-fix 1, Prefix 2, Prefix 3, Word, Suffix. This leads to a total of 11 classes in the data: O, B-PRE1, I-PRE2, B-PRE2, I-PRE2, B-PRE3, I-PRE3, B-WORD, I-WORD, B-SUFF, I-SUFF.

Given raw data, the words are segmented into a series of chunks corresponding to the different classes above. Our goal is to produce text where all the word tokens (modulo the clitics) are bona fide words in a dictionary and increasing the level of regularity in the text as a means of normalization. Note that AMIRA-TOK does not perform lemmatization. For example, in our running example بالحسنات و wbAlHsnAt, ‘and by the virtues’, the clitic tokenization AMIRA-TOK reduces it maximally to حسنات + ال + + ب + و w+ b+ Al+ HsnAt, where حسنات HsnAt is kept in its plural form and not reduced to the singular lemma form حسنة Hsnħ.

**Morphotactic Restorations** AMIRA-TOKdoesnot produce stemmed words which are not valid Arabic words. Accordingly, we apply some heuristics to reverse the effect of morphotactics such as the loss of ا A in the definite article ال Al when in the context of the proclitic preposition ل l ‘for’. For example, the input word وللبلاد wll-blAd, ‘and for the countries’, is clitic tokenized as w+l+l+blAd in the system output from the classification. A post-hoc fix is applied to ensure the consistency of the output. The final output is rendered بلاد.+ ال + ل+و w+l+**A**l+blAd. Most of these morphotactic restorations can be deterministically applied.

However, some morphotactics are not deterministic such as those involving the nominal feminine marker (Ta-Marbuta), the Alef-Maqsura, and word final Hamzas. In AMIRA-TOK, we currently handle the former two cases: the Ta-Marbuta and the Alef-Maqsura.

**Ta-Marbuta Normalization**

The Ta-Marbuta ( ة ħ) is a word final nominal feminine marker that is realized as a ت t preceding a pronominal enclitic. Accordingly, a stem after token cliticization ending with a ت t could either be a verb (not a Ta-Marbuta) or a noun (possibly a Ta-Marbuta). In هم# حسنت+ب+و w+b+Hsnt#hm, ‘and by their virtue’, the word-final ت t in Hsnt is an ة ħ converted to ت t in the context of the enclitic هم hm: هم# حسنة+ب+و w+b+Hsnh#hm. As part of AMIRA-TOK, the ت is converted to the corresponding ة ħ. This allows for more accurate POS tagging later as the base word è Hsnh will correctly be identified as a noun rather than verb.

**Alef-Maqsura Normalization**

The Alef-Maqsura (ى ý) always occurs word finally. It is letter that graphically looks like a dotless Ya (ي y), yet phonologically sounds like an Alef ا A). In the context of a pronominal enclitic, it changes to an orthographic ا A. For example a noun such as عصاهم SAhm ‘he-went-against them’, is initially segmented into هم # عصا &SA#hm yet the stem final ا A should be an Alef-Maqsura (ى ý): هم# عصى &Sý#hm. In this case the resulting verb stem عصى« &Sý would be confused with the noun عصا &SA, stick. In most cases the variation of ى ý and ا A is a lexical distinction relating to the underlying root of the word or the word pattern. Accordingly, AMIRA-TOK converts these word final ا A into ى ý.

Both Ta-Marbuuta and Alef-Maqsura normalization are lexically determined; hence, we apply another layer of learning to the problem of classifying word final letters. A stem final ت t either remains a regular ت t or is converted to ة ħ, and the stem final ا A either remains an ا A or is converted to a ى Y. As a result the end output from the AMIRA-TOK is valid surface form words with clitic tokens. AMIRA-TOK performs at a very high F-score measure of 99.2% evaluated using standard Conlleval evaluation metrics.

**User Interface**

**]**In the current version of AMIRA-TOK, the user is allowed ample flexibility in deciding the form of output they would like. The user can decide what level of clitic segmentation they desire, and whether tokenization should be indicated with spaces (changing the token count) or with a plus sign only (preserving token count). We have specified several schemes as illustrated in Tabel 1.15.

#### AMIRA-POS

Part-of-speech tagging is the process by which every word in running text is assigned a POS tag from a predefined tag set. AMIRA-POS assumes the text is clitic tokenized.

**AMIRA POS Tag Sets** The POS tag set (ERTS)used by AMIRA-POS has 72 tags; it is a subset of the full morphological set defined over tokenized text. ERTS is a superset of the reduced PATB tag set (RTS) that comprises 25 tags. In addition to the in-formation contained in the RTS tags, ERTS encodes additional morphological features such as number, gender, and definiteness on nominals. For example, in RTS nouns are tagged as either NN or NNS, indicating only number. In ERTS, nouns tags represent definiteness and gender in addition to number; furthermore, we add the dual as a value to the number feature. A full description of ERTS is presented in (Diab, 2007b).

**POS Tagging** We adopt an SVM-based classification approach using character n-grams as features in our sequence models. Interestingly, the accuracy of the ERTS tagger is 96.13% and the accuracy of the RTS tagger is 96.15%. This suggests that our choice of information to include in our ERTS tag set reflects a natural division in the syntactic space. In (Diab, 2007b; Diab, 2007a), we show that using ERTS improves the quality of downstream processing such as base phrase chunking.

**User Interface** The user has the flexibility to in-put raw or tokenized text in a scheme that is consistent with one of the schemes defined by AMIRA-TOK. Consequently the user may request that the POS tags be assigned to the surface forms. Inter-nally, in case of the raw input, AMIRA-POS runs AMIRA-TOK on the raw text and then performs POS tagging. The output can be presented as tokenized and POS tagged, or without tokenization where the POS tag is assigned to the surface words. In this latter case, the ERTS tagset is appended with the clitic POS tags to form more complex POS tags. The user can choose to either tag with ERTS or RTS.

#### AMIRA-BPC

Base phrase chunking is the process by which a sequence of adjacent words are grouped together to form syntactic phrases such as NPs and VPs. An English example of base phrases would be [I]NP [would eat]V P [red luscious apples]NP [on Sundays]PP . BPC is the first step towards shallow syntactic parsing. Many applications such as information extraction and semantic role labeling in English have been proven to benefit tremendously from BPC at a relatively low loss in performance when compared to the use of deep syntactic parsing.

**Chunking**

AMIRA-BPCproduces the longest possible base phrases with not much internal recursion. The internal recursion is done as a deterministic post process. We have modified the BPC rules to be more appropriate for the Arabic language (Diab, 2007a). Nine types of chunked phrases are recognized using a phrase IOB tagging scheme (described earlier in Section 6.1); Inside (I) a phrase, Outside (O) a phrase, and Beginning (B) of a phrase. The nine chunk phrases identified for Arabic are ADJP, ADVP, CONJP, PP, PRT, NP, SBAR, INTJ, VP. Thus the task is a one of 19 classification tasks (since there are I and B tags for each chunk phrase type, and a single O tag). The training data is derived from the PATB using the CHUNKLINK software[[24]](#footnote-24). CHUNKLINK flattens the tree to a sequence of base (non-recursive) phrase chunks with their IOB labels. For example, a token occurring at the beginning of a noun phrase is labeled as B-NP. We have adapted the CHUNKLINK software to the Arabic language.

However we do acknowledge that an appropriate size of a chunk is most dependent on an end application.

The AMIRA-BPC component is very fast and extremely accurate. The system yields an F1 mea-sure of 96.33%. Different research groups have shown the utility of BPC for different applications such as Machine Translation and information ex-traction specifically in Named Entity Recognition (NER) (Benajiba et al., 2008; Benajiba et al., 2009).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | wbAlHsnAt | wllblAd | fbmktbthm | wsnqwlhA |
| Arabic  Gloss  Translation | وبالحسنات | وللبلاد | فبمكتبتهم | وسنقولها |
| and+by+the+ virtues | and+for+the+countries | then+by+libraries+their | and+will+we\_ say+it |
| And by the virtues | And for the countries | Then by their libraries | And we will say it |
| **Scheme** |  |  |  |  |
| PATB | w+b+AlHsnAt | w+ l+ AlblAd | f+ b+ mktbt #hm | w+ s+ nqwl #hA |
| AMIRA-TOK | w+b+Al+HsnAt | w+l+Al+blAd | f+b+mktb\_h#hm | w+ s+nqwl #hA |
| Conjunction-only | w+ bAlHsnAt | w+ llblAd | f+ bmktbthm | w+ snqwlhA |
| Conjunction-only+Suffix | w+ bAlHsnAt | w+ llblAd | f+ bmktb\_h #hm | w+ snqwl #hA |
| Preposition-only | w+b+ AlHsnAt | w+l+ AlblAd | f+b+ mktbthm | wsnqwlhA |
| Preposition-only+Suffix | w+b+ AlHsnAt | w+l+ AlblAd | f+b+ mktb\_h #hm | wsnqwl #hA |
| Al-only | wb+Al+ HsnAt | wl+Al+ blAd | fbmktbthm | wsnqwlhA |
| Al-only+Suffix | wb+Al+ HsnAt | wl+Al+ blAd | fbmktb\_h #hm | wsnqwl #hA |
| future marker-only | wbAlHsnAt | wllblAd | fbmktbthm | w+s+ nqwlhA |
| future marker+Suffix | wbAlHsnAt | wllblAd | fbmktb\_h #hm | w+s+ nqwl #hA |
| All Prefix Only | wbAl+ HsnAt | wll+ blAd | fb+ mktbthm | ws+ nqwlhA |
| Suffix only | wbAlHsnAt | wllblAd | fbmktb\_h #hm | wsnqwl #hA |
| All Prefixes+Suffix | wbAl+ HsnAt | wll+ blAd | fb+ mktbthm | ws+ nqwl #hA |

Table 1.15: Some of the different AMIRA-TOK clitic tokenization schemes in the change token count mode

**User Interface**

AMIRA-BPCcan accept any levelof preprocessing on input text. AMIRA-BPC internally uses ERTS POS tag set. However, the user may request the RTS as the POS tag set, we have an internal mapping process from ERTS to the RTS POS tagset. Also AMIRA-BPC can produce the BPC tags on any form of AMIRA-TOK consistent schemes in addition to raw input text.

#### Comparison of CADIM Tools

In this section, we compare and contrast the different CADIM tools discussed so far in terms of their design, functionality and performance.

**Design**

As for their design, it may help to contextualize the different tools in terms of their basic use in two suites: The MADA suite and the AMIRA suite (see Figure 1.13). Within the MADA suite, we have an explicit morphological analysis step handled by ALMORGEANA or MAGEAD. Currently, ALMORGEANA is fully integrated and we plan to integrate MAGEAD. MAGEAD uses a different approach to morphology than ALMORGEANA. The difference however should not be evident for MSA. MAGEAD’s advantage over ALMORGEANA is the ease of extensibility to new dialects. The second, in fact core, component in the MADA suite, is the MADA system, which disambiguates the analyses produced by the morphological analyzers. Finally, the TOKAN component makes use of the morphological generation power of ALMORGEANA and MAGEAD to tokenize the disambiguated analysis through regeneration.

In the AMIRA suite, the three components focus on tokenization (AMIRA-TOK), POS tagging (AMIRA-POS) and base-phrase chunking (AMIRA-BPC), in that order.

The AMIRA-BPC component is unique in that it has no parallel in the MADA suite. In fact the output of TOKAN can be designed to look like the output of AMIRA-TOK followed by AMIRA-POS, and then be used as input to AMIRA-BPC.

In term of their design, AMIRA-TOK and AMIRA-POS are different from the MADA suite in that they take a two-step approach to POS tagging: tokenize then tag. In comparison, MADA has a different approach that breaks the problem into three steps (an-alyze, disambiguate, generate), which are orthogonal to AMIRA’s split.



Figure 1.13: The MADA suite and the AMIRA suite.

Although there are three steps in MADA, the decision for tokenization and POS tagging is done together in one-fell-swoop. One way of distinguishing these tools is in terms of the depth of linguistic knowledge needed. AMIRA is shallow in that it focuses on form-based morphology (specifically cliticization) learned from annotated data; whereas MADA uses deeper lexically modeled functional morphology. Another difference between the current MADA suite and the AMIRA-TOK and AMIRA-POS suite is that the former may produce no analysis for a given word if it does not exist in the underlying morphological tools while the AMIRA suite always produces a hypothesized tokenization and POS tag for every word in the text.

In terms of their training needs, the MADA suite expects the presence of both a morphological analyzer and training data for supervised learning, whereas the AMIRA suite only needs annotated training data. The training data could be created through any number of ways, including the use of morphological analyzers followed by human annotation as is done at LDC; but this is not a requirement for the AMIRA suite. These different yet similar requirements put similar limits on the kind of extensions that could be done in either approach. For example, going to an Arabic dialect would require the presence of some morphological analyzer/generator for the dialect for MADA, but not AMIRA. However both need some amount of annotated data to train on.

**Functionality**

Lemmatization and base-phrase chunking. Base-phrase chunking is only handled in the AMIRA suite, but it is in fact a separate module that can be used independently with the MADA suite. The other four applications are handled at once in MADA as part of its common morphological disambiguation process. AMIRA does not handle lemmatization or diacritization. As for tokenization and POS tagging, since MADA goes deeper than AMIRA, a wider set of possible tokenization schemes and POS tags can be out-put by MADA. Although AMIRA is more limited by comparison, it does handle the most commonly used tokenizations and POS tags. Researchers interested in exploring a large number of different sets of tokenizations as features in their systems should consider MADA. Researchers only interested in limited comparisons or specific applications, whose to-kenizations and POS tags are supported by AMIRA should consider AMIRA.

**Performance**

It is hard to compare the performance of AMIRA and MADA suites. Previous attempts by (Habash and Rambow, 2005) show that similar performance is possible on tasks that are shared: specific PATB tokenization and POS tags. AMIRA can be significantly faster than MADA; however, MADA needs to be run only once and a much larger number of tokenizations and POS tags (in addition to other outputs not supported by AMIRA) can be produced by running the fast TOKAN step.

**Applications**

In terms of functionality, we consider five applications: tokenization, diacritization, POS tagging, MADA, TOKAN, and the AMIRA suites have been used by numerous academic and commercial research institutes around the world, including University of Washington, University of Massachussets, University of Maryland, University of Sus-sex, Dublin City University, Cambridge University, SRI, BBN, Fair Isaac Inc., SketchEngine LLC, MIT, RWTH Aachen, Polytechnic University of Catalunya (UPC), Copenhagen Business School, Polytechnic University of Valencia (UPV), and the National Research Center of Canada, among others. The tools have been cited in numerous publications and have been shown to improve performance in a variety of NLP applications such MT, IR, and IE.

**MADA+TOKAN for NLP applications**

In the context of machine translation (MT) from Arabic to English, Habash and Sadat (2006) and Sadat and Habash (2006) explored the use of different preprocessing schemes and their combination. Their results have been followed by different groups of researchers working on Arabic-English MT (Costa-jussà et al., 2006; Crego et al., 2006; Vilar et al., 2008). Diab et al. (2007a) explored the use of MADA-generated diacritizations for MT. Elming and Habash (2007) and Elming et al. (2008) improved automatic word alignment for Arabic-English MT using combinations of different tokenization schemes generated by MADA+TOKAN. See (Habash, 2007b) for more details on different representations of Arabic morphology for MT. Badr et al. (2008) used MADA in the context of English-to-Arabic MT. MADA has also been used to produce features for Named Entity Recognition (NER) (Farber et al., 2008; Benajiba et al., 2008).

**AMIRA for NLP applications**

The AMIRA suite has been successfully used by several groups in the context of text MT, specifically for alignment improvement and reordering within the context of statistical MT (Crego and Habash, 2008), and also for identifying difficult source language text (Mohit and Hwa, 2007). Moreover, the AMIRA suite was used in the context of speech MT (Stroppa and Way, 2006). The AMIRA suite was explored for the purposes of cross language information retrieval in work by Larkey et al. (2007). AMIRA has been used to produce POS tag and BPC features for NER which significantly boost the per-formance of the Arabic NER system (Benajiba et al., 2008; Farwell et al., 2007).

**Case Study of a Complex ASR+MT system**

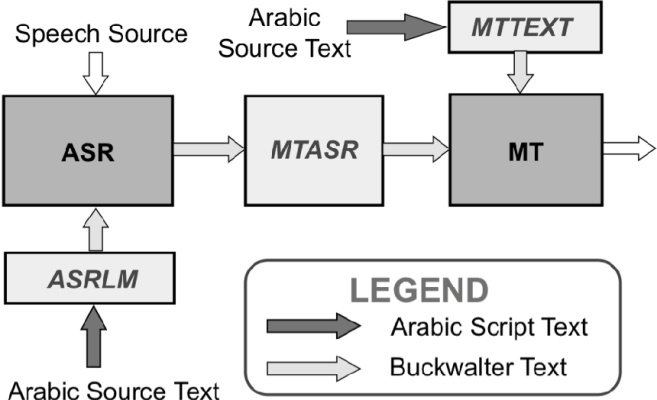


Figure 1.14: Example of an ASR/MT pipeline. MADA and TOKAN are used in the ASRLM, MTASR, and MTTEXT subsystems.

In the following example we show how MADA+TOKAN can be incorporated into an MT project that is based loosely on the SRI GALE project Nightingale. Figure 1.14 shows the overall architecture of the project. The MT process needs to make use of data from both text sources and audio sources via automatic speech recognition (ASR). The ASR process processes audio files, but needs to build reliable language models from text sources first. The subsystems ASRLM and MTTEXT process raw Arabic script before passing important information to the ASR and MT components, respectively. The MTASR subsystem processes ASR output for use in MT. These three subsystems use specific instantiations of MADA+TOKAN with different input and output specifications.

The ASRLM subsystem cleans the raw data and converts the UTF-8 encoding into Buckwalter transliteration. A separate utility is used to convert numeric digits to words (Habash and Roth, 2008), as is required for ASR. The subsystem then runs MADA and uses the toolkit’s stem orthographic normalization tool to remove spelling variations. The subsystem consequently runs TOKAN to produce an output suitable for ASR; here, TOKAN uses a basic read-off scheme that produces only the fully-diacritized words without further tokenization. Finally, punctuation is removed. This provides the ASR system with nicely formatted, fully diacritized data, which is what the acoustic component of the ASR component produces.

The MTASR subsystem takes the output of ASR (which originally came from the audio files), cleans it, and runs MADA+TOKAN, using a tokenization scheme that splits off conjunctions and particles, i.e. D2 (see Table 1). Stem orthographic normalization is also used. The same numerical utility used in AS-RLM is also used here to tag numerical expressions (which may be digital or expressed as words). The Buckwalter-transliterated data is converted back to UTF-8 prior to sending the data to the MT system.

The MTTEXT subsystem processes text for MT. It cleans the raw data and converts UTF-8 to Buckwalter transliteration. MADA+TOKAN (with stem orthographic normalization, number tagging and UTF-8 conversion) are used here to produce the same tokenization (D2) as the MTASR subsystem, making the output of MTTEXT and MTASR identical. The result is that the MT system can draw on similarly formatted ASR-derived and text-derived data for training and development.

# Chinese Statistical Parsing

Authors: Mary Harper and Zhongqiang Huang

#### Introduction

This paper describes several issues that are fundamental to achieving accurate Chinese parsing given available Chinese resources and the challenges of the Gale processing pipeline. For Gale, our parsing algorithm was expected to accurately parse various different materials, ranging from newswire text, which tends to be grammatically well formed, to -best ASR outputs, many of which are poorly formed sentences. To address this challenge, we have re-implemented and enhanced the Berkeley parser to handle unknown Chinese words efficiently, parse difficult sentences robustly, and operate more efficiently. We also address issues related to training the parser for several different genres given a limited number of available training trees, the importance of matching word segmentation to the treebank segmentation standard to support accurate parsing, and the need for standardized tokenization for managing the types of things that will appear as input to the parser. Understanding and handling these issues is a prerequisite for achieving adequate parsing performance levels. We also investigate self-training with automatically labeled in-domain data to enhance parsing performance given the limited number of trees in the Chinese treebanks.

There have been several attempts to develop high quality parsers for Chinese (Bikel and Chiang, 2000; Levy and Manning, 2003; Petrov and Klein, 2007b), but the state-of-the-art performance, around 83% F measure on Penn Chinese Treebank, achieved by the Berkeley parser (Petrov et al., 2006; Petrov and Klein, 2007b) falls far short when compared to English[[25]](#footnote-25). As pointed out in (Levy and Manning, 2003), there are many linguistic differences between Chinese and English, as well as structural differences between their corresponding treebanks, and some of these make it a harder task to parse Chinese. Additionally, the fact that the available treebanked Chinese materials are far more limited than those for English also increases the difficulty of building high quality Chinese parsers. While there have been some investigations using treebanks to determine what makes Chinese hard to parse, here we attempt to learn about and control the factors that challenge Chinese parsers as applied to materials that are less controlled than a treebank.

Research that involves only parsing treebank test sets in some sense masks the challenges of providing high quality syntactic analyses for Chinese text and speech. First, Chinese documents are not segmented into words that conform to the word segmentation standard used by the Chinese treebank. Second, the Chinese treebanks contain punctuation, letters, and digits that are typically full-width; whereas, it is not uncommon for materials to contain mixtures of full- and half-width representations. It is in fact unlikely that the treebank would cover the combination of possible character types that may be observed in Chinese documents. Third, the treebank data is quite limited compared to the resources available for English parsing of newswire and speech genre, and so we must develop strategies to effectively parse materials in the text and speech genre covered by the GALE program given the resources at hand: Chinese Treebank 6 (CTB6) with parsed newswire and broadcast news sentences and Chinese Broadcast Conversation Treebank (BCTB) with parsed broadcast conversation sentences.

We will focus our parsing investigations in this section on the Chinese newswire and broadcast conversation genres because of the availability of tree-banks representative of these genres. Unfortunately, the broadcast news treebank does not fully capture the spoken aspects it should represent in that it contains symbolic expressions that can be verbalized in a variety of ways. Although we can map this treebank to a verbalized form, this mapping is unfortunately less variable than an exact transcription of the corresponding speech would be. To maximize parsing performance, we process the treebanks differently for the newswire and broadcast conversation genres and train the models to match genre conditions. We evaluate genre matched parsers on a test set drawn from the respective newswire and broadcast conversation treebank.

#### Experimental Data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data Split** | | | | |  | |
| **Genre** | **Train** | **Dev** | **Test** | **Unlabeled** | |  | |
| NW | CTB5  14,925 (~405k) | CTB5  1,904  (~51k) | CTB5  1,975  (~52k) | 210k  (~6255k) | |  | |
|  | |
| CTB5+CTB6BN  24,416 (~679k) |  | |
|  | |
|  | |
| BC | BC  8,149 (~110k) | BC  1,297  (~14k) | BC  1,497  (~17k) | 210k  (~2128k) | |  | |
|  | |
| CTB5+CTB6BN  24,412 (~584k) |  | |
|  | |
|  | |
| BC+CTB5+CTB6BN  32,561 (~695k) |  | |
|  | |

Table 1.. Number of sentences and words (in parentheses) of the data splits used in our experiments.

We used the Penn Chinese Treebank 6.0 (CTB6) and the Chinese Broadcast Conversation Treebank (BCTB) in our parsing studies, as well as a set of unlabeled sentences to support the use of semi-supervised self-training (see Table 1.16).

CTB6 contains 28k parsed sentences, including news articles from Xinhua news agency (China-Mainland), Information Services Department of HKSAR (Hong Kong), and Sinorama magazine (Taiwan), as well as broadcast news from ACE evaluation (which we call CTB6BN). The news articles from the first three sources, with a total of 19k sentences, constitute the former Penn Chinese Treebank 5 (CTB5) and are the primary source of labeled data used for newswire experiments. Since the CTB5 corpus was collected during different time periods from different sources with a diversity of topics, in order to obtain a representative split of train-test-development sets, we divide it into blocks of 10 files in sorted order and for each block use the first file for development, the second for test, and the remaining for training. Although CTB6BN consists of parses for broadcast news transcriptions, it exhibits many of the characteristics of newswire text (it contains many nonverbal expressions, e.g., numbers and symbols, and is fully punctuated). Because of this similarity, we evaluate using this corpus as additional labeled training data for training the newswire model in Section 1.4.5.7.

BCTB contains 11k parsed sentences from three broadcast conversation sources (i.e., CCTV, CNN with Chinese translations, and Phoenix TV). In contrast to the CTB6BN data, the words in this treebank are fully verbal; however, most of the sentences in the treebank are punctuated. We manually selected the train-test-development split of the files to balance the number of sentences in each set. Since BCTB may be too small to train an accurate grammar for parsing speech, we also augment the training set with a verbalized version of CTB6, as discussed in Section 1.4.5.7.

In addition to the above two labeled treebanks, we also utilize a greater number of unlabeled sentences for investigating the use of semi-supervised self-training in Section 1.4.5.8. For the newswire experiments, 210k unlabeled sentences are selected from three newswire sources covering materials from China Mainland, Taiwan, and Hong Kong. For broadcast conversation experiments, 210k unlabeled sentences are selected from transcribed broadcast conversations. We have developed a set of scripts to clean up the treebank trees used in our investigations. We remove comments and then delete empty nodes and non-terminal-yield unary rules, e.g., NP→VP, using tsurgeon (Levy and Andrew, 2006). As non-terminal-yield unary rules are less likely to be posited by a statistical parser, it is common for parsers trained on the standard Chinese treebank to have substantially higher precision than recall. This gap between bracket recall and precision is alleviated without loss of parse accuracy by deleting the non-terminal-yield unary rules.

#### SParseval Scoring

The SParseval tool (Harper et al., 2005; Roark et al., 2006) was originally designed to support English speech-based bracket and head dependency scoring (recall, precision, and F-score) at the level of a demarcated chunk of speech such as a conversation side, while also supporting more traditional text-based scoring methods that require the input to the parser to match perfectly in words and sentence segments to the gold standard. The tool was developed to address the fact that output from an automatic speech recognition system is likely to contain word errors and the automatic segmentation of these words into sentences is likely to differ from those in the gold standard parses (Harper et al., 2005). To calculate the bracket scores in the face of word and sentence segmentation errors, the tool utilizes information from a word-level alignment between the yields of the system parses and reference parses in a transcript chunk (e.g., a conversation side or story)[[26]](#footnote-26).

We extended the functionality of this tool to support the scoring of Chinese parses when the word segmentation of the input to the parser differs from the gold standard word segmentation in order to investigate the effect of word segmentation algorithms on parse quality. We use bracket scores with word alignment as the metric for evaluating word segmentation impact. To score parses with different word segmentations using this tool, we align the words in a gold standard file with the words in a test file and score the brackets with punctuation removed[[27]](#footnote-27).

#### Parsing Model

The Berkeley parser (Petrov et al., 2006; Petrov and Klein, 2007b) is an efficient and effective parser that introduces latent annotations (Matsuzaki et al., 2005; Prescher, 2005) to refine the syntactic categories and learns PCFG grammars based on these latent annotations. We have re-implemented[[28]](#footnote-28) and enhanced the Berkeley parser to handle unknown Chinese words efficiently, parse difficult sentences robustly, and operate more efficiently. We will describe our enhancements in detail in the remainder of this section.

#### Unknown Word Handling

The Chinese word formation process can be quite complex (Packard, 2000), and it differs substantially from the English process. Although Chinese morphology is generally considered less informative of the part-of-speech (POS) type of the entire word, its characters do reflect some information about a Chinese word. The last characters in a Chinese word are, in some cases, most informative of the POS type, while for others, it is the characters at the beginning or the middle of a word.

The Berkeley parser has some built-in ability to handle certain classes of unknown Chinese words such as digits and dates. For words outside of these classes, the parser ignores character-level information and relies only on the rare-word part-of-speech tag statistics. In our approach, which is similar to (Huang et al., 2007), we employ a rather simple but effective method to estimate the word emission probability, , of an unknown word given the latent state of some syntactic category. We use the geometric average of the emission probability of the characters in the word, i.e., with being the -th character of the word, to estimate the word emission probability:

where

.

Characters unseen in the training data are ignored in the computation of the geometric average. In case there is any character in the word that was previously seen in the training data but only with other latent annotations, then we back off by using the rare word statistics from the state regardless of the word.

As we can see in Table 1.17, the character based unknown word handling method improves performance on both recall and precision when evaluated on CTB5.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Treebank** | **Unk** | **Recall** | **Precision** | **F** |  |
| CTB5 | default | 82.12 | 82.88 | 82.50 |  |
| character | 82.84 | 83.22 | 83.03 |  |
|  |

Table 1.. The effect of the character-based unknown word handling method.

#### Robustness

Although the grammars trained by the Berkeley parser are compact, parsing can still be computationally expensive because of the many fine-grained latent states to consider in the computation of inside-outside probabilities in the chart. Fortunately, Petrov and Klein (Petrov and Klein, 2007b) developed an efficient coarse-to-fine parsing strategy that starts from simpler grammars and prunes away unlikely chart items before parsing using the more complex grammars. In practice, the pruning thresholds are chosen to balance the parsing accuracy and speed.

Setting pruning thresholds to support robust but efficient parsing can be challenging over the range of parsing tasks we support in the GALE project; our parser is expected to parse various different materials, ranging from newswire text, which tends to be grammatically well formed, to -best ASR outputs, many of which are poorly formed sentences. It is not uncommon for thresholds that work well for parsing treebank materials to fail to parse less well-formed sentences we are expected to process. To support more robust parsing, we incrementally lower pruning thresholds when the current thresholds fail to parse a sentence and then restore the default thresholds after finishing parsing that sentence. With this strategy, we were able to more fully parse sentences in the GALE processing pipeline. We balance this with a parsing time threshold to address cases when sentences are too long to parse in a reasonable amount of time, returning an empty parse when the time threshold is exceeded.

In addition to variable pruning thresholds, we made several other minor changes to the parser to improve robustness. For example, we added smoothing for the word emission probabilities associated with a word tag pair , to eliminate zeros when a frequently observed word does not co-occur with tag in the training data.

#### Parallelization

The time needed to train a single-threaded Berkeley parser is acceptable for a corpus of moderate size. For example, it takes roughly one day to train a grammar on CTB5+CTB6BN. However, the training speed becomes a bigger issue when applying self-training strategies. For example, in the self-training experiments described in Section 1.4.5.8, the automatically labeled training data is an order of magnitude larger than the treebank training set, and it takes several weeks to finish training a more complex grammar. In our re-implementation of the parser, we parallelized the Expectation-Maximization step, the most computationally intensive component of the training code to take advantage of the computing power of multi-core/multi-CPU machines. The resulting training code is about 7 times faster with 10 threads on a 16-core machine than the single threaded version. We parallelized the parsing code as well to improve the decoding speed. The MapReduce framework (Dean and Ghemawat, 2005) is also naturally applicable to both grammar training and sentence parsing, and this framework will be explored in future work.

#### Word Segmentation and Parsing

Written Chinese text consists of sequences of characters with no delimiters between words, and yet for word-based NLP applications (tagging, parsing, MT), word segmentation is a prerequisite. The output of word segmentation algorithms may vary depending on their different definitions of words and system engineering requirements. While word segmentation, in and of itself, is worthy of study, as shown in the efforts in the SIGHAN Chinese Word Segmentation Bakeoffs (Sproat and Emerson, 2003), in this section we focus on the effect of word segmentation on parse accuracy. Discrepancies in word segmentation compared to the gold standard are likely to be the source of significant error when comparing the parse of those words to the gold standard parse of the gold standard word segmentation. Consistency between the words in the gold standard trees and the input to the parser is likely to be an important factor for achieving greater parse accuracies.

Here we will use the Penn Chinese Treebank (CTB) Segmentation Standard (Xia, 2000) because that is the resource we use to train our Chinese parser (Xue et al., 2002; Xue et al., 2005). In the rest of this subsection, we will first describe the word segmentation algorithms we use and then evaluate the effect of those algorithms on parse accuracy. As described in Section 1.4.5.3, the Sparseval tool with alignment will be used to score parses because there can be mismatches between the yields of the test parses and gold standard trees.

#### Word Segmentation Algorithms

Parse accuracy can be substantially affected by the word segmentation algorithm and how it is trained. A word segmentation algorithm that is trained on treebank data is likely to produce a word segmentation that is more consistent with the treebank, and so is likely to result in parses with greater parse accuracy. We evaluated three different segmenters.

The first is Fair Issac’s extension of the LDC segmenter (LNUplus) with an expanded dictionary (it adds words from the NYU name dictionary and the lexicon used by the University of Washington word segmenter), and instead of using a greedy left-to-right, longest dictionary entry match at each point, it uses dynamic programming to find a segmentation that maximizes mean token length. This is strictly a dictionary-driven segmenter, and the dictionaries it uses were not optimized to conform with CTB6.

The University of Washington word segmenter (UW) (Hwang et al., 2006) uses a very large segmentation dictionary that includes frequent words extracted from GALE acoustic data transcripts, Chinese Gigaword2, and the Chinese treebank 6.0. It then uses the longest-first-match algorithm to segment the training data comprised of GALE acoustic data transcripts and Chinese Gigaword, and then trains an -gram LM on these sources and the Chinese treebank 6.0 data duplicated 5 times to weight the source more heavily. The language model uses a vocabulary of 70K words, with the remaining words mapped to a garbage word. This -gram LM is then used to provide the maximum likelihood segmentation for a sentence. This word segmenter uses information provided by CTB6, but it is not optimized for the treebank.

The Stanford Chinese Word Segmenter (Tseng et al., 2005; Chang et al., 2008) uses conditional random fields with character identity, morphological, and character reduplication features extracted from the training data. The current segmenter also uses external lexicon features (Chang et al., 2008) to segment more consistently. We trained several versions of the Stanford segmenter:

* **Stanford (all)** is trained using the entire Chinese Penn Treebank. This represents an upper bound on the parsing accuracy for the parser given this word segmentation approach (since the training set contains the test sentences).
* **Stanford (parser)** is trained using the sametraining data split of the Chinese Penn Treebank as the parser, i.e., the CTB5+CTB6BN training set. The same dev set in Table 1.16 is used for development.
* **Stanford (parser+LDC)** is trained under thesame conditions as Stanford (parser) except that the training data is augmented with an additional file that was segmented by LDC and contains 16,448 sentences.

• **Stanford (parser+LDC+self-labeled)** is trained under the same conditions as Stanford (parser+LDC) except that the training data is augmented with Gale data files that were segmented using Stanford (parser). Hence, this segmenter was trained on 175,852 sentences in total (24,416 from CTB5+CTB6BN training, 16,448 LDC hand-segmented, and 134,988 automatically word segmented).

• **Stanford (parser+LDC+UW-labeled)** is trained under the same conditions as Stanford (parser+LDC) except that the training data is augmented with the same Gale data files as in Stanford (parser+LDC self-labeled) segmented using the UW segmenter.

#### Results and Discussion

In Table 1.18, we compare the word segmentation and parsing performance of each of the segmentation algorithms and training conditions described in Section [1.4.5.5.1](#_Word_Segmentation_Algorithms). The parser is our re-implementation of the Berkeley parser described in Section 1.4.5.4. The parser was trained on the CTB5+CTB6BN training set (i.e., treebank data only), and this parser was used to parse the CTB5 test set re-segmented by each of the word segmentation models. The resulting parses were then compared with the gold standard using aligned bracket scoring from the Sparseval tool as described in Section 1.4.5.3.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Models** |  | **Word Segmentation** | | |  | **Parser** | | |  |
|  | **Recall** | **Precision** | **F** |  | **Recall** | **Precision** | **F** |  |
| Treebank Segmentation |  | 100 | 100 | 100 |  | 83.58 | 84.00 | 83.79 |  |
| LNUplus |  | 81.70 | 82.50 | 82.10 |  | 67.46 | 68.29 | 67.87 |  |
| UW |  | 86.60 | 91.60 | 89.00 |  | 71.15 | 76.71 | 73.83 |  |
| Stanford (all) |  | 99.40 | 99.40 | 99.40 |  | 83.12 | 83.45 | 83.28 |  |
| Stanford (parser) |  | 95.70 | 96.60 | 96.20 |  | 80.49 | 81.65 | 81.07 |  |
| Stanford (parser+LDC) |  | 96.10 | 96.90 | 96.50 |  | 80.59 | 81.75 | 81.16 |  |
| Stanford (parser+LDC+self-labeled) |  | 97.50 | 97.60 | 97.50 |  | 81.84 | 82.27 | 82.05 |  |
| Stanford (parser+LDC+UW-labeled) |  | 89.00 | 90.50 | 89.80 |  | 74.13 | 75.98 | 75.04 |  |

Table 1.: Results on the Chinese Treebank 6.0 test set for different models and training configurations.

It is not surprising that the best parsing performance is obtained using the treebank’s gold standard word segmentation. Parses of the sentences processed by the LNUplus and UW segmenters had significantly[[29]](#footnote-29) lower parse scores than was obtained given the gold standard word segmentation. Since neither segmenter was tuned to the treebank’s word segmentation standard, the errors in word segmentation harm the parse accuracies greatly. The UW segmenter’s use of the treebank data improves its word segmentation accuracy relative to the LNUplus segmenter; however, the use of training data that is inconsistent with the treebank’s standard plays a role in the 10% drop in parse accuracy.

To get at the upper bound parsing performance of the Stanford segmenter, we evaluated parsing accuracy when it was trained on the entire treebank. Although parse accuracy in this case is slightly lower than that obtained with gold standard word segmentation, it is clear that match to the gold standard words is an important factor for obtaining accurate Chinese parses. The Stanford segmenter was able to produce fairly accurate word segmentations when trained on the same treebank data as the parser and achieved parse F-scores within 3% of those obtained using gold standard word segmentations. It is notable that retraining the Stanford segmenter using additional LDC-segmented data and self-labeled data improves both word segmentation and parsing scores. When adding automatically labeled data to the training set, it is important that it is consistent with the treebank word segmentation standard, as can be observed by the decline in performance obtained by adding data segmented by the UW segmenter. The use of consistently self-labeled data to re-train the segmenter improves parse performance by 1% F-measure over using the treebank training data alone. If parsing accuracy is important for downstream applications, then using a word segmenter that is tuned to the treebank standard is vital for achieving performance levels that are within 1-2% of those obtained with perfect word segmentation.

#### Text Tokenization

The UW Decatur (Zhang and Kahn, 2008) text normalization process was developed to standardize the text pipeline stream for the Nightingale team. The value of this normalization for parsing lies with the fact that a parser that is trained to match its input conditions will perform better than a parser that must process highly varying data that is harder to model. For example, if we train the parser on the treebank, which largely contains full-width punctuation, but the input to the parser contains only half-width punctuation, then the parsing performance would be reduced due to the fact that the treebank contains only a few half-width punctuation tokens[[30]](#footnote-30). While it is a simple matter to create a second set of training trees with all punctuation converted to half-width and mix it with the original treebank training, it may not cover all the ways in which full and half-width punctuation occur in real-world data. However, it is a simple matter to apply the Decatur normalization to our treebank by converting all full-width letters, digits, and punctuation marks to their half-width equivalents and train on the normalized trees. The Decatur tokenizer would then be applied to all input coming through the Nightingale pipeline to ensure that we have a good match between training and test conditions.

To ensure that the Decatur tokenization does not harm parsing performance significantly, we compare a parser trained on the original treebank CTB5 training set and tested on the original CTB5 test set with one trained based on Decatur normalized training and tested on the Decatur normalized CTB5 test set. As can be seen in Table 1.19, the parsing performance is slightly degraded by the normalization process, largely due to the fact that some punctuation distinctions are collapsed. However, because Decatur normalization will be carried out on all input to the parser, it eliminates a lot of the variability in the input that it would be hard to anticipate to adequately train the parser for the Gale pipeline.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CTB5** |  | **Recall** | **Precision** | **F** |
| Original |  | 82.84 | 83.22 | 83.03 |
| Decatur |  | 82.66 | 83.14 | 82.90 |

Table 1.. Performance of the parser on CTB5 before and after Decatur normalization.

#### Genre Mapping

Newswire stories contain a wide variety of textual phenomena, including words, symbolic expressions, and punctuation, using both full and half-width representations. When parsing newswire text, it is beneficial to normalize the input to the parser and map the training trees to the same normalized style to match the conditions in which the parser will be used. Additionally, it is important to fully utilize existing treebank resources to train the parser, given the limited amount of training data. Hence, we investigate the effect of combining trees from the broadcast news genre with the newswire training trees when training a newswire model. Table 1.20 shows the benefit of adding text normalized broadcast news data for training a better grammar for newswire. As we can see, the grammar trained on the combination of CTB5 and CTB6BN has a significantly better performance when evaluated on the CTB5 test set, due to the increase in domain matched training data. In an attempt to further boost performance, we will investigate the effect of combining automatically parsed newswire genre-matched trees with treebank training trees to further improve the newswire model in subsection 1.4.5.8.

| **Training Data** |  | **Recall** | **Precision** | **F** |
| --- | --- | --- | --- | --- |
| CTB5 |  | 82.66 | 83.14 | 82.90 |
| CTB5+CTB6BN |  | 83.36 | 83.95 | 83.65 |

Table 1.. Results on the Chinese CTB5 test set for grammars trained on different configurations.

When parsing automatically transcribed broadcast conversation speech, the parser will need to process speech transcripts that are segmented into SUs using automatic sentence boundary detection; hence, words will be fully verbal (i.e., no symbolic expressions) and there will be no punctuation. While there is a small treebank that matches this genre, access to a greater diversity of speech-based trees is important for training as accurate a model as possible.

Because the annotated BC treebank is quite limited in size, a grammar trained only on BC treebank alone is quite poor as can be observed in. It is a natural idea to use CTB6 to enhance the training corpora for BC grammars. However, the textual nature of the CTB5 sub-corpus presents a challenge. One obvious difference is that the transcripts of the BC treebank are verbalized, i.e., all numbers, dates, and other phenomena are orthographically transcribed while CTB5 contains many symbolic forms. Although CTB6BN should be orthographically transcribed as speech, it contains digits and other symbolic expressions that make it inconsistent with the BC treebank. To address the mismatch between the CTB6 and BC treebanks in order to use CTB6 as additional training material for BC grammars, we have developed scripts to speech normalize (i.e., verbalize) the CTB6 treebank. The normalization procedure considers the part-of-speech tag of a word, as well as its context to determine how to verbalize the word. The major normalization operations include:

* Verbalize all sequences of digits based on their context. If the context suggests that a sequence of digits represents a number, this number is verbalized according to how it is pronounced, e.g., 12.4% is verbalized to百分之十二点四(twelve point four percent). Otherwise each digit is verbalized as it is pronounced individually, e.g., 03012 is verbalized to 零三零一二(zero three zero one two).
* Special care is taken to verbalize digits in temporal nouns. For example, 1998 is verbalized to 一九九八(year nineteen ninety eight), 5.4 to 五四(The May 4th Movement), 10:12 to 十分十二秒(ten minutes twelve seconds), etc.
* Split foreign names with first and last names delimited by a dot or hyphen into two names. Web and email addresses are also split into separate tokens with ‘.’ and ‘@’ verbalized.
* Remove all punctuation.

Table 1.21 shows the benefit of adding the speech normalized CTB6 for training a better grammar for conversational speech. As we can see, although the grammar trained on speech normalized[[31]](#footnote-31) CTB6 has a much lower performance when evaluated on the BC test set, its combination with the small amount of in-domain BC training data boosts the performance significantly, outperforming the grammar trained on BC training data alone. We will also investigate the use of additional automatically parsed BC genre-matched sentences in the self-training manner to further improve parsing performance on BC data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Training Data** |  | **Recall** | **Precision** | **F** |
| BC |  | 76.89 | 76.48 | 76.69 |
| CTB5+CTB6BN |  | 72.85 | 71.66 | 72.25 |
| BC+CTB5+CTB6BN |  | 78.95 | 78.96 | 78.96 |

Table 1.. Results on the Chinese BC test set for grammars trained on different configurations.

#### Parser Self-training

To support self-training in a semi-supervised setting, manually labeled training data is used to train an initial parser, which is then used to parse additional unlabeled data to combine with the manually labeled data to retrain a new parser. Early investigations using self-training for parser training were fairly unsuccessful at improving in-domain parsing. Charniak (1997) reported that self-training does not improve the performance of a context-free grammar trained on the WSJ training set. Steedman et al. (2003) reported some degradation using a lexicalized tree adjoining grammar parser (Sarkar, 2001) and a minor gain using Collins lexicalized PCFG parser (Collins, 1999); however, this gain was obtained using a poor parser trained on few sentences and was expected to level out quickly on a larger training set. One might conclude from these investigations that either the self-labeled data does not provide useful information or the training algorithm used for the parsers does not learn useful information from the self-labeled data.

Recently, McClosky et al. (2006) used semi-supervised training of a reranking parser (Charniak and Johnson, 2005) and obtained exciting positive results on WSJ over a strong baseline. Their state-of-the-art reranking parser consists of two components, a lexicalized probabilistic 50-best parser and a discriminative reranker, which re-orders and selects the best of the 50-best parses returned by the first generative parser by utilizing a rich set of features that could not be feasibly used by the first parser. They used this two-stage parser to parse millions of unlabeled news article sentences and retrained the first parser using the combination of the original labeled data and the reranker-labeled data. Although they did not fully explain why this two stage method works, they suggest that the parse trees selected by the superior reranker provides guidance for retraining the first parser, which in turn produces better 50-best parses for the reranker. It should be noted that they also reported that no improvement was obtained from self-training their generative parser.

We have investigated the self-training capability of our parser on both newswire and broadcast conversation by utilizing moderately large amount of unlabeled in-domain data. As described in Section 1.4.5.2, 210k unlabeled sentences are used for both the newswire and broadcast conversation genres. The results are presented in Table 1.22. For the NW task, self-training gains one point on F measure over the grammar trained using the treebank CTB6 alone. This improvement is even greater than the benefit of adding CTB6BN to CTB5. The improvement on the BC task is smaller but consistent across different random runs. Given that the number of words for self-training BC is much smaller than for the NW task, use of additional genre-matched data is an important next step. All the improvements are statistically significant. We also tried self-training using a Chinese port of Charniak’s generative parser on our data set but obtained no significant improvement.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Genre** |  | **Training Data** | **Recall** | **Precision** | **F** |  |  |
|  | NW | | CTB6 | 83.36 | 83.95 | 83.65 |  |  |
|  | +unlabeled | 84.28 | 85.28 | 84.78 |  |  |
|  | BC |  | BCTB+CTB6 | 78.95 | 78.96 | 78.96 |  |  |
|  |  | +unlabeled | 79.54 | 79.44 | 79.49 |  |  |

Table 1.. The performance of self-training on both NW and BC.

We observed that many of the rule parameters of the grammar trained on CTB6 alone have zero probability. On one hand, this is what we want because the grammar should learn about impossible rule expansions. On the other hand, this might also be a sign of over-fitting. In contrast, the grammar obtained using self-training contains much more non-zero rules than the grammar trained on CTB6 alone. This suggests that one of the benefits of using automatically labeled data is smoothing. The nature of the EM algorithm aims to adjust the model parameters to increase the likelihood of the training data. The greater the number of free parameters, the more power EM has to learn from and fit the training data. Resulting grammars may not generalize well when the training data is too small in size. We believe that the addition of automatically labeled data helps to prevent the EM algorithm from over-fitting the correctly labeled training data while learning with more latent states. This hypothesis will be investigated further in future work.

#### Conclusions

In this section, we have explored issues such as unknown word handling and word segmentation in parsing Chinese, and conducted experiments that highlight the fact that greater newswire parse accuracy can be achieved by training on the combination of newswire and broadcast news parses, than by training on newswire parses alone. Additionally, we have found that self-training with a large amount of unlabeled data further improves parsing performance. We conjecture based on our analyses that the EM training algorithm is able to exploit the in-formation available in both gold and automatically labeled data with more complex grammars while being less affected by over-fitting the treebank. Self-training should also benefit other discriminatively trained parsers with latent annotations (Petrov and Klein, 2007a; Petrov and Klein, 2008), although training would be much slower compared to using generative models, as in our case.

In future work, we will evaluate the impact of using larger quantities of self-labeled data. As the amount of data increases, it will also be important to investigate the impact of weighting the self-labeled data. It is quite possible that the errors in the automatically labeled data could limit the accuracy of the self-trained model, especially when there is a much greater quantity of automatically labeled data than the gold standard training data, so by weighting the posterior probabilities computed for the gold and automatically labeled data the information provided by the gold trees will not be swamped by the errors in the automatically labeled trees. Another approach would be to introduce the automatically labeled data at different stages of training rather than at the outset, because it is possible that a later introduction of the automatically labeled data, after well-founded annotations are learned from the treebank, would result in more effective learning. We will also investigate methods for automatic data selection of the automatically labeled sentences in order to choose those that would be most helpful for self-training.

# Evaluating the Impact of Word Segmentation on Chinese Parsing

Author Nianwen Xue

#### Introduction

Because Chinese does not have conventional word delimiters like white spaces, it is generally assumed that a prerequisite in Chinese language processing is to perform automatic word segmentation. Despite the enormous literature on Chinese word segmentation and syntactic parsing, to our knowledge, there has not been a thorough evaluation on the impact of word segmentation on Chinese parsing. As a result, there are often conflicting views on how much of a problem Chinese word segmentation is for later processing stages.

Word segmentation (or tokenization) has long been recognized as the first step in Chinese language processing and it is a problem that has generated a lot of research interest in the NLP community in the last few decades (Chen and Liu, 1992; Sproat et al., 1996; Xue, 2003; Gao et al., 2005; Low et al., 2005; Zhao and Kit, 2008; Huang et al., 2008). There has been a steady improvement in word segmentation accuracy as the research shifted from early dictionary lookup approaches to statistical approaches (Sproat and Shih, 1990; Sproat et al., 1996) and machine learning-based character-tagging approaches (Xue, 2003; Gao et al., 2005; Zhao et al., 2006; Zhao and Kit, 2008). It is still a very much active area of research in the Chinese language processing community, and the primary forum has been the successive SIGHAN Chinese word segmentation bakeoffs (Sproat and Emerson, 2003; Emerson, 2005; Levow, 2006; Jin and Chen, 2008).[[32]](#footnote-32) Early successes came with the Maximum Match (Maxmatch) algorithm, a greedy search algorithm based on dictionary-lookup. Given a text string, the algorithm looks for the longest word in the dictionary that matches the text from the current position. If there is a match, the current position is advanced past each character in that word. If there is no match, the algorithm starts again at the next character. There are some inherent problems with the Maxmatch algorithm. The performance of the Maxmatch algorithm is tied closely to the completeness of the dictionary. Since a complete dictionary is impossible to come by, given that new words such as names constantly find their way into the language, Chinese word segmentation is insolvable by the Maxmatch algorithm alone. In addition to new words, some character sequences are inherently ambiguous and the Maxmatch algorithm would consistently get it wrong in contexts where shorter words are the correct segmentation. For example, there are two perfectly legitimate segmentations for the character sequence “个-人”, depending on the context, e.g., “一/one 个/CL 人/person” vs. “个人/individual 所得税/income tax”. The Maxmatch algorithm would consistently get the latter but miss the former. To address the weaknesses of the Maxmatch algorithm, statistical approaches such as mutual information were proposed to capture the strength of the internal binding between the two characters within a word (Sproat and Shih, 1990). However, mutual information based approaches have their own limitations. Mutual information for words longer than two characters is hard to define, and it is not straightforward to incorporate a dictionary in this framework, thus losing the benefit of a dictionary, in spite of the fact that no dictionary is complete. More recently, Chinese word segmentation has been formulated as a character tagging problem where each character in the text is labeled with a tag indicating its position within a word (Xue, 2003). Combined with supervised machine learning methods this approach has consistently outperformed other approaches in the SIGHAN Bakeoffs (Xue and Shen, 2003; Low et al., 2005; Zhao and Kit, 2008) and produces state-of-the-art results. Top performing systems have consistently achieved accuracy that are in the high nineties (above 95%) using a wide range of benchmarks.

As large-scale hand-parsed Chinese corpora such as the Chinese Treebank (Xue et al., 2005) became available, there as has been considerable work done on Chinese parsing as well (Levy and Manning, 2003; Bikel, 2004). However, Chinese parsers, at least by North America based researchers, are mostly extensions of English parsers. The same basic assumptions and methodologies are carried over from English to Chinese. With rare exceptions such as Luo (2003), most Chinese parsers assume word-segmented sentences as input and parsing accuracy is reported assuming gold standard word segmentation. In real-world applications, however, Chinese text has to be automatically segmented and there is inevitable degradation in parser performance caused by segmentation errors. As far as we know, no formal evaluation on the effect of automatic word segmentation on parsing accuracy has been performed, although in informal settings in the GALE community there has been talk that parsing accuracy drops substantially when automatic word segmentation is used as input. The goal of this paper is to elucidate this issue by performing a formal analysis and see how much impact word segmentation has on syntactic parsing.

#### Methodology

To perform such an analysis we obviously need i) training and test data, ii) a Chinese word segmenter, iii) a Chinese parser, and iv) an evaluation metric. For training and test data we use the Chinese Treebank 6.0 (CTB6.0)[[33]](#footnote-33). CTB 6.0 has three main sources: Xinhua newswire, Sinorama magazine articles, and broadcast news.[[34]](#footnote-34) The LDC release of CTB 6.0 has a recommended training set, development set, and test set and the data split can be found in CTB 6.0 documentation. The test set can be further divided into three subsets: the Xinhua newswire, the Sinorama and the broadcast news test sets. The training set has 639,697 words, consisting of roughly equal amount of Xinhua newswire, Sinorama and broadcast news data. The test sets also have roughly the same size: the Xinhua test set has 24,801 words, the Sinorama test set 28,879 words and the broadcast news test set 27,744 words. Both our word segmenter and parser are trained on the training set as a whole and tested separately on each of the three test sets. The Chinese parser we use is the Bikel parser (Bikel, 2004) retrained on this training set without any further optimization, so the parsing results we present in this paper may not be the best possible performance it can achieve. However, we believe that the performance is good enough so that whatever conclusion we draw from our experiments here will not be affected by the accuracy of the parser. The word segmenter is a Maxent-based Chinese word segmenter we developed in-house. The test data is first segmented by the Maxent word segmenter, and then parsed by the Bikel parser, and evaluated with the SParseval metric (Roark et al., 2006) and the CParseval metric we developed proposed. It is reasonable to assume that word segmentation accuracy will have a great impact on parsing accuracy as the latter takes the former as input, so it is important to use a state-of-the-art word segmenter so as to evaluate the impact of word segmentation on Chinese parsing given the current state of the art. It is also important to look at the evaluation metric very closely and examine the assumptions based on which the evaluation metrics are formulated. In the next two sections, we will look at these two aspects in greater detail.

#### Word Segmentation

The word segmenter we used for our experiment is an improved version of the word segmenter described in (Xue, 2003). The word segmenter formulates Chinese word segmentation as a character-tagging problem. The idea is to label each character in a sentence with its position in a word, and then reconstitutes the words in a sentence based on the character position labels. There are four possible positions for a character: in the beginning (LL), in the middle (MM), in the end (RR), or a word by itself (LR). Based on this formulation, the word sequence in (1a) would be labeled as (1b). The training data for the segmenter is derived from the training set in CTB 6.0 and a Maxent classifier is trained to tag un-seen data. The labeled data is then converted back to word sequences.

1. 上海计到本世纪末实现人均国内生产总值五千美元
2. 上/LL 海/RR 计/LL 划/RR 到/LR 本/LR

世/LL 纪/RR 末/LR 实/LL 现/RR 人/LL

均/RR 国/LL 内/RR 生/LL 产/RR 总/LL

值/RR 五/LL 千/RR 美/LL 元/RR

1. Shanghai plans to reach the goal of 5,000 dollars in per capita GDP by the end of the century.

The features used in the (Xue, 2003) are fairly simple and they are character unigrams and bigrams extracted from a 5-character window, consisting of the current character (*C*0), the previous two characters (*C***−**1 and *C***−**2), and the next two characters (*C*1, *C*2), as well as the tags of the previous two characters (*T***−**1, *T***−**2 ). The full list of features is described below:

1. Character unigrams: current character (*C*0), the previous (next) two characters (*C***−**1 , *C***−**2 , *C*0, *C*1,*C*2)
2. Character bigrams: the conjunction of the previous (next) character and the current character (*C***−**1*C*0 , *C*0 *C*1) , the previous two characters (*C***−**1*C***−**2 ) and the next two characters (*C*1 *C*2), the previous and the next character (*C***−**1*C*1 )
3. Tags: the tags of the previous two characters (*T***−**1 and *T***−**2 )

This simple tagset is effective in predicting shorter words but often makes mistakes with longer words such as numbers (e.g. 一百五十六点七八/1256.78), time expressions (e.g. 二零零九年/2009), idioms (e.g. 一目了然/clear at first sight)and names (e.g. 阿尔及利亚/Algeria), which often consist of four or more characters. To address these shortcomings, the improved segmenter used in our experiments also uses regular expression-based features, features extracted from a name list and an idiom list. These features are described below:

1. The position of the current character (*C*0 ) in a number, where the values for the position are “ln” (left of a number),“rn” (right of a number) or “mn” (middle of a number). For example, if the current position is 点, then its position is “mn” in一百五十六点七八
2. The position of the current character in a time expression, where the values for the position are “lt” (left of a time expression), “rt” (right of a time expression), or “mt” (middle of a time expression). For example, the position of 二 is “lt” in 二零零九年 if it is the current character.
3. The position of the current character in an id-iom, where the values of the position are “li” (left of an idiom), “ri” ( the right of an idiom), or “mi”, (the middle of an idiom). For example, the position of 然 is “ri” in the idiom 一目了然
4. Whether the current character starts or ends a name. For example, if the character sequence *C*0*C*1matches the first two characters of aname, the feature “StartName” is extracted. If the current and the previous character match the last two characters of a name, then the feature “EndName” is added. For example, if *C*0=阿and*C*1=尔, then the feature “Start-Name” is extracted given 阿尔及利亚 is in the name list.

Adding these features to the classifier cleaned up the errors with numbers, time expressions, names and idioms and improved the overall word segmentation accuracy significantly. Table 1.23 presents the word segmentation accuracy for each of the three test sets described in Section 1.4.6.2 when trained on the entire training set. For reasons that are not entirely clear, the accuracy for the Sinorama data is significantly lower than the other two sources, the Xinhua newswire and the broadcast news. The accuracy for the broadcast news is slightly lower, as expected because it is transcribed from speech data and is slightly noisier. Overall, the accuracy is comparable to that of the best systems in the SIGHAN bakeoffs. This allows us to measure the impact of word segmentation on syntactic parsing given the current state of the art.

|  |  |  |  |
| --- | --- | --- | --- |
| Genre | precision | recall | f-score |
| Bn | 95.36 | 95.51 | 95.44 |
| Xinhua | 96.80 | 96.25 | 96.52 |
| Sinorama | 93.18 | 92.46 | 92.81 |

Table 1.: Word Segmentation Accuracy.

#### Evaluation Metric: SParseval vs. CParseval

In over a decade, most of results on parsing are reported with the Evalb software using the Parseval metric, which measures the precision (the number of matched constituents between the treebank parse and the parser output to the total number of constituents in the parser output) and recall (the number of matched constituents to the total number of constituents in the treebank) and the F1 measure, which is a harmonic mean between precision and recall.

A key assumption for the Parseval metric is that there is a complete match between word tokens in a treebank parse and those in the parser output for a given sentence, an assumption that does not hold when parsing Chinese with automatic word segmentation as input.[[35]](#footnote-35) One can imagine a trick that uses the characters instead of words as terminals in the parse tree. However, such a metric is essentially a combined metric of word segmentation and parsing, and given the reasonable assumption that word segmentation is substantially easier than syntactic parsing because the latter builds larger structures, the resulting accuracy is inflated and is incomparable with parsing accuracy reported for other languages using the Parseval metric.

The SParseval metric (Roark et al., 2006), which is developed to evaluate parsing accuracy on transcribed speech data and does not assume total match between word tokens in the gold standard parse and the parser output, seems like a natural alternative. So SParseval is the first evaluation metric we tried. When examining the evaluation results, we noticed that the metric penalizes word segmentation errors severely, even though word segmentation is not the target of the evaluation. For example, the two parses in the example below only differ in the segmentation of 全球(“the whole world”). However, when the two parse trees are compared, the SParseval metric returns a recall (precision as well since there is the same number of constituents in the two parse trees) of 11/13 = 0.84. Apparently, both layers of NP that dominate this character sequence are considered to be wrong by the SParseval metric due to the different segmentations of this character string. The SParseval seems to be overly harsh, especially considering that there are cases in Chinese where reasonable people differ as to what the correct word segmentations are.

To address this problem, we developed an alternative *CParseval* metric that assumes *a constituent is correctly parsed if it spans over the same character sequence as a constituent in the gold standard parse*. This metric factors out word segmentation errors when evaluating syntactic parses. With this metric, there is a complete match between (a) and (b), below.

1. (IP (NP-SBJ (NP (**NN**全球**)**)

(QP (OD 第五)

(CLP (M 个)))

(NP-PN (NR 迪斯尼)

(NN 乐园)))

(VP (ADVP (AD 即将))

(PP-LOC (P 在)

(NP (PN 这里)))

(PP-DIR(P 向)

(NP (NN 公众 )))

(VP (VV 开放)))

(PU 。))

1. (IP (NP-SBJ (NP **( NN** 全**)( NN** 球**)**)

(QP (O**D**第五)

(CLP (**M** 个)))

(NP-PN(NR迪斯尼)

(NN乐园)))

(VP (ADVP (AD 即将))

(PP-LOC (P 在)

(NP (PN 这里)))

(PP-DIR(P 向)

(NP (NN 公众 )))

(VP (VV 开放)))

(PU 。))

Translation: The fifth Disney World around the globe is going to be open here to the public soon.

#### Experimental Results

With the evaluation metrics in place, we are ready to report parsing evaluation results. The parsing results are reported for the Bikel parser retrained on the training set and tested on the three test sets described in the Section 1.4.6.2. The accuracy is reported in terms of labeled F1 measure and all nodes in the tree are counted, including unary nodes. There are three points of comparison in the results presented in Table 1.24. The results are for all three sources (Xinhua, Sinorama and Broadcast news) of CTB 6.0, and the parser takes both gold standard and automatic word segmentations as input, and the parser output is evaluated against both SParseval and CParseval metrics.

When gold standard word segmentation is used as input, the CParseval metric consistently returns slightly lower parsing accuracy, indicating CParseval is not generally a more lenient metric than SParseval. However when automatic word segmentation is used as input to the parser, the accuracy returned by the CParseval metric is consistently higher, indicating a significant number of errors reported with the SParseval metric are due to incorrect word segmentation. The constituents built on top of wrongly segmented words may not be incorrect in that they still span over the correct character strings.

The third point to notice is that Sinorama data is substantially harder to parse than the other two sources, and there is also a larger divide between the parsing accuracy for gold standard and automatic word segmentations and presumably this is due to the lower word segmentation accuracy for this data set. Over-all, the parsing performance degradation caused by using automatic word segmentation ranges from 2% to 4% using the CParseval metric, and from 4% to 8% using the SParseval metric.

|  |  |  |  |
| --- | --- | --- | --- |
| Genre | Seg | SParseval (f1) | CParseval (f1) |
| Xinhua | Gold | 80.37 | 79.52 |
| Xinhua | Auto | 75.93 | 77.13 |
| Sinorama | Gold | 71.37 | 70.20 |
| Sinorama | Auto | 63.41 | 65.96 |
| BN | Gold | 79.39 | 78.51 |
| BN | Auto | 74.56 | 76.44 |

Table 1.: Comparing SParseval and CParseval.

Table 1.25 presents results with unary nodes included or excluded, using just the CParseval metric. When unary nodes are excluded from calculation, the accuracy is consistently higher for all three sources and for both experimental conditions (when gold standard word segmentation or automatic word segmentation is used as input to the Bikel parser). It is also interesting to note that the difference is around 2% by F1 measure for all experimental conditions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Genre |  | Seg | F1 (w/o unary) | F1 (w/ unary) |
| Xinhua |  | Gold | 81.47 | 79.52 |
| Xinhua |  | Auto | 79.04 | 77.13 |
| Sinorama | | Gold | 72.72 | 70.20 |
| Sinorama | | auto | 68.36 | 65.96 |
| BN |  | gold | 80.60 | 78.51 |
| BN |  | auto | 78.47 | 76.44 |

Table 1.: CParseval labeled accuracy with and without unary nodes.

#### Conclusion

This paper reports an attempt to formally evaluate the impact of automatic word segmentation on syntactic parsing. Given a state-of-the-art word segmenter, the degradation in parsing performance when using the automatic word segmentation as input ranges from 4% to 8% in F1 score by the SParseval metric, depending on the source. We also described the CParseval metric that factors out the word segmentation errors from the parsing accuracy calculation. By this metric the parsing accuracy degradation is 2% to 4% in F1 score. The substantial drop in ac-curacy floated around informally in the GALE community is likely a result of lower word segmentation accuracy and overly harsh evaluation metrics.

# Combining Discriminative Re-ranking and Co-training for Parsing Mandarin Speech Transcripts

Author: Wen Wang

#### Introduction

Discriminative reranking has significantly improved parsing performance, and co-training has proven to be an effective weakly supervised learning algorithm to bootstrap parsers from a small in-domain seed labeled corpus using a large amount of unlabeled in-domain data. In this section, we present systematic investigations on combining discriminative reranking and co-training, including co-training reranked parsers and co-training rerankers. We show that combining discriminative reranking and co-training could improve the F-measure by 1.8%–2% absolute compared to co-training two state-of-the-art Chinese parsers without reranking for parsing Mandarin broadcast news and conversation transcripts.

Parsing aims at resolving structural ambiguity. State-of-the-art statistical parsers require treebanks to estimate their parameters, but their performance degrades when there is mismatch on genres/domains between the training treebank and the data to parse. Furthermore, creating a high-quality in-genre/in-domain treebank for the data to parse is expensive and difficult. However, under the GALE program, there are new genres besides newswire text, namely, broadcast news (BN), broadcast conversation (BC), newsgroup (NG), and web log (WL). Generating high-quality parse trees for Chinese data in these genres can be useful for various tasks within GALE, including syntax-guided translation and reordering models for Chinese-to-English ma-chine translation (MT), name entity detection, and structured language modeling for automatic speech recognition (ASR) on Mandarin BN and BC audio.

In our earlier research (Wang, 2008), we employed the weakly supervised co-training technique on two state-of-the-art parsers, Charniak's parser and the Berkeley parser, to bootstrap them from a newswire Chinese treebank and a small amount of BN and BC seed annotated treebank with a large amount of unlabeled BN and BC transcripts, in order to achieve high parsing accuracy on Mandarin BN and BC transcripts. By employing co-training, we obtained 2.2% – 2.6% absolute improvement on F-measure for parsing BN and BC transcripts. F-measure is based on labeled Precision (LP) and labeled Recall (LR). LP is the number of correct constituents divided by the number of constituents found by the parser, and LR is the number of correct constituents divided by the number of constituents in the gold parse. F -measure is defined as F1 . On the other hand, discriminative reranking for parsers (Collins and Koo, 2005; Charniak and Johnson, 2005) has produced significant improvement on parsing accuracy. In this paper, we explore the effectiveness of combining discriminative reranking and co-training to further improve parsing performance on Mandarin BN and BC transcripts.

#### Discriminative Re-ranking

We first describe our use of the RankBoost-based discriminative reranking approach that was originally developed by Collins and Koo (Collins and Koo, 2005) for parsing. This approach allows us to investigate the impact of various features on Mandarin parsing performance. The reranking algorithm takes as input a list of candidates produced by a Chinese parser and reranks these candidates based on a set of features. For training the reranker for the parsing task, there are *n* sentences , each with candidates along with the log-probability produced by the parser. Each parsing candidate in the training data has a score that measures the similarity between the candidate and the gold reference. For parsing, we use parse accuracy as the similarity measure. Without loss of generality, we assume that has the highest score, i.e. for . A set of indicator functions {hk : k = 1, ···, m } is used to extract binary features, on each example . Each indicator function is associated with a weight parameter that is real valued. In addition, a weight parameter is associated with the log-probability . The ranking function of candidates is defined as . The objective of the training process is to set the parameters to minimize the loss function (which is an upper bound on the training error), as where is the weight function that gives the importance of each example, and is the margin (Collins and Koo, 2005). All the 's are initially set to zero. Then a greedy sequential optimization method is used in each boosting round to select the feature that has the most impact on reducing the loss function and then update its weight parameter accordingly.

Collins' method allows multiple updates to the weight of a feature. Huang et al. (Huang et al., 2007) found that for those strong features, Collins' weight update formula can increase their weight (in absolute value) in only one direction. Although these features are strong and useful, setting weights too large can be harmful in that it limits the use of other features for reducing the loss. Based on this analysis, Huang et al. (Huang et al., 2007) have developed an update-once method, in which the weight update is limited so that once a feature is selected in a certain iteration and its weight parameter is updated, no update will be conducted on it again. In this way, the weights of the strong features will not be allowed to prevent other features from being considered during the training procedure. Huang et al. observed that the update-once method could select significantly more features compared to Collins' original method and produce better reranking performance. In this paper, we employed this update-once strategy for updating feature weights

For the work described in this paper, we employed the features described in (Collins and Koo, 2005). Note that before generating these features, we applied headword percolation on the trees that are the output of parsers, as employed in (Collins and Koo, 2005). Features include rules (all context-free rules in the tree), bigrams (adjacent pairs of non-terminals to the left and right of the head), grand-parent rules (same as rules, but also including the non-terminal above the rule), head-modifiers (all head-modifier pairs, also including the grandparent non-terminal), and PPs (lexical trigrams involving the heads of arguments of prepositional phrases) and so on. More details appear in (Collins and Koo, 2005).

#### Co-training

Co-training was first introduced by Blum and Mitchell (Blum and Mitchell, 1998) as a weakly supervised learning method and can be used for boot-strapping a model from a seed corpus of labeled examples, which is typically quite small, augmented with a much larger amount of unlabeled examples, by exploiting redundancy among multiple statistical models that generate different *views* of the data. Informally, co-training can be described as picking multiple classifiers (“views”) of a classification problem, building models for each view and training these models on a small set of labeled data, then on a large set of unlabeled data, sampling a subset, labeling the sampled subset using the models, selecting examples from the labeled results, adding them to the training pool, and iterating this procedure until the unlabeled set is all labeled.

|  |  |
| --- | --- |
| **Input**: *S* is a seed set of labeled data. | |
| is labeled training data for *h*1. | |
| is labeled training data for *h*2. | |
| *U* is the unlabeled data set. | |
| *C* is the cache holding a small subset of *U*. | |
| **1** *S* | |
| **2** *S* | |
| **3** Train classified *h*1 on | |
| **4** Train classified *h*2 on | |
| **5 repeat** | |
| **6** | Randomly partition *U* into *C* where and |
| **7** | Apply *h*1, *h*2 to assign labels for all examples in *C* |
| **8** | Select examples labeled by *h*1 and add to |
| **9** | Train *h*2 on |
| **10** | Select examples labeled by *h*2 and add to |
| **11** | Train *h*1 on |
| **12** |  |
| **13 until** *U is empty* | |

Figure 1.: General co-training algorithm.

In Figure 1.15, when calling the classifier that provides additional training data for the opposite classifier the *teacher* and the opposite classifier the *student*, since the labeled output from both classifiers *h*1 and *h*2 is noisy, an important question is which newly labeled examples from the teacher should be added to the training data pool of the student. This issue of example selection plays an important role in the learning rate of co-training and the performance of resulting classifiers. In (Wang, 2008), we investigate four example selection approaches, namely, *naive co-training*, *agreement-based co-training*, *max-score*, and *max-t-min-s*. We developed the *max-score* and *max-t-min-s* approaches in (Wang et al., 2007). Wealso compare the performance of co-training to self-training. Self-training in this work simply adds all examples in the labeled cache to the training pool in each iteration (Nigram and Ghani, 2000).

In (Wang, 2008), we systematically investigated applying weakly supervised co-training approaches to improve parsing performance for parsing Mandarin BN and BC transcripts, by iteratively retraining two competitive Chinese parsers, Charniak's reranking parser (Charniak and Johnson, 2005) and the Berkeley parser (Petrov et al., 2006), from a small set of treebanked data and a large set of unlabeled data. Compared to parsers trained only on the small in-domain seed labeled corpus, the parsers resulting from co-training could gain 6.8% absolute on BN and 7.3% absolute on BC based on the F-measure.

Overall, compared to parsers trained on all available treebank data including in-domain and out-of-domain treebanks, co-training yields a 2.2%– 2.6% absolute gain on BN and 2.4% – 2.5% absolute gain on BC based on the F-measure (and 1.5% – 1.9% absolute gain on BN and 1.7% – 2.0% absolute gain on BC over self-training (Wang, 2008)). In this paper, we investigate the combination of discriminative reranking and co-training on Charniak's maximum-entropy inspired parser (Charniak, 2000) (i.e., the parser without reranking compared to the parser in (Charniak and Johnson, 2005)) and the Berkeley parser (also originally without reranking). For co-training parsers, we employed the *max-t-min-s* example selection approach, as it is computation-ally inexpensive and also produced the best performance (Wang, 2008).

#### Combining Re-ranking and Co-training

Both Charniak and Berkeley parsers support generating N-best parses for reranking purposes. In fact, Charniak and Johnson have implemented a discriminative reranker using a MaxEnt estimator to find the feature weights and when using the reranker to rerank 50-best parses from Charniak's Maximum-entropy inspired parser, it improved F-measure by 1.3% absolute on sentences of length less than 100 words in Wall Street Journal Penn Treebank section 23 (Charniak and Johnson, 2005).

In this work, we adopted this reranker for Charniak's parser, implemented the RankBoost-based reranking algorithm described in the second section to rerank 50-best from the Berkeley parser, and then investigated two ways to combine reranking and co-training. The direct combination approach is for each iteration of co-training, instead of generating 1-best parse directly from the no-reranking, standard Charniak and Berkeley parsers, 50-best parses are generated from each parser and then reordered by their cor-responding rerankers, respectively. Then the 1-best parses after reranking for the unlabeled data are selected and added to the training pool of the parsers. In this paper this approach is denoted **co-training** **reranked parsers**. Note that for this approach thefeatures and feature weight parameters for rerankers remain the same during the co-training procedure.

Different from the original binary classification problems on which co-training was developed, parsing contains a number of smaller decisions about which constituents are probable, and inherently each parser includes good and bad decisions on how to create/attach different constituents. On the other hand, reranking is closer to binary classification than parsing, as it tries to decide whether or not a parse hypothesis is the best parse for the sentence, so it is explicit to maximize agreement between rerankers, as the principled agreement-based example selection approach could be applied here, which could guarantee co-training to improve parsing accuracy. Hence, we hypothesize that co-training rerankers could better fit the co-training algorithm. For effectiveness, rerankers can consider features that span the entire tree of a parse (while parsers generally consider only local features). For efficiency, co-training rerankers requests unlabeled data to be parsed just once, compared to multiple parsing iterations for co-training reranked parsers. The out-put will be reranked many times but this is much more efficient than training and running parsers. Hence, in this work, we also investigated co-training our RankBoost-based reranker with Charniak's and Johnson's MaxEnt reranker and applied the co-trained rerankers to the two standard parsers. This second approach is denoted **co-training rerankers**.

#### Data

For selecting parsers and also contributing to training parsers, we used Chinese Treebank 5.2 released by LDC (denoted as CTB). Chinese Treebank 5.2 contains 500K words, 800K characters, 18K sentences, and 900 data files. Under the GALE pro-gram, the BN genre follows its tradition and consists of “talking head” style broadcasts, i.e., generally one person reading a news script. The BC genre, by contrast, is more conversational and spontaneous, consisting of talk shows, interviews, call-in programs, and round-tables.

The evaluation of co-training for parsing Mandarin BN and BC transcripts is conducted on the GALE OntoNotes released Mandarin BN and BC treebanks. The BN treebank is from the Mandarin TDT4 collection, and the BC treebank is from GALE Mandarin BC data and translations from English BC data. The Mandarin BN treebank includes 300K words and 814 data files, and the BC treebank 100K words and 16 data files.

To create a seed corpus and a test set for evaluating parsing accuracy, for BN and BC respectively, we divided the whole BN/BC treebank into blocks of 10 files by sorted order. Within each block, the first file is used for co-training development and the second for testing parsing accuracy. The remaining eight files are used as part of the seed annotated corpus for co-training. The resulting BN test set is denoted **BN-test** and the seed annotated corpus **BN-seed**. The BC test set is de-noted **BC-test** and the BC seed annotated corpus **BC-seed**. BN-test includes 31K words and 1,565sentences. BC-test includes 11K words and 1,482 sentences. The large set of unlabeled data for BN parsing includes Hub4 1997 Mandarin BN acoustic transcripts, LDC Chinese TDT {2,3,4} corpora, Chinese Gigaword 3.0, and all GALE released BN audio transcripts, denoted **BN-unlabeled**. For BC parsing, we add all GALE released BC audio transcripts denoted **BC-unlabeled**. After word segmentation, **BN-unlabeled** comprises around 1.4 billion words and **BC-unlabeled** around 11 million words.

#### Selecting Parsers for Co-training

To select the two parsers for co-training, we investigated four publicly available parsers, namely, Charniak's maximum-entropy inspired parser (Charniak,2000), the Stanford unlexicalized parser (Klein and Manning, 2003), Berkeley parser (Petrov et al., 2006), and Dan Bikel's reimplementation of Michael Collins' Model 2 parser (Collins, 1999). To select two from them in our co-training setup, we considered two important factors: accuracy and mutual complementariness. To evaluate parser ac-curacy, we consider the F-measure. Using the train/dev/test split described in the fifth section, Table 1.26 shows the F-measure of all four parsers on the test set.

|  |  |  |
| --- | --- | --- |
| Parser |  | F-measure |
| Charniak |  | 83.2% |
| Stanford |  | 82.0% |
| Berkeley |  | 83.5% |
| Bikel |  | 82.9% |

Table 1.. F-measure of all four parsers on the CTB test set. The train/dev/test split is described in the fifth section.

The co-training principle requires the two views to be conditionally independent or weakly conditionally independent. This means that we need to select parsers that are complementary on their learning patterns and error types. To measure the structural complementariness between parsers, we adapted the measure of structural consistency between parsers and modified the objective function for maximizing the structural complementariness between parsers to be selecting parsers with the minimal structural consistency.

Note that to measure the structural consistency between the bracketing parses from parsers and gold standard parses, Black, Garside, and Leech (Black et al., 1993) defined the metric *average crossing brackets* (ACB), the mean number of times per sentence that a bracketed sequence from one parser overlaps with the gold standard from the treebank such that neither is properly contained in the other. Although ACB does not account for all types of conflicting constituency, it is a practical measure for the structural consistency between two sets of parse trees.

By using the output from one parser *B* as the gold set, we can calculate the pair-wise ACBAB of parser *A* on parser *B*. The *ACB* values on the CTB test set among all six pairs from the four parsers are ordered as {Charniak, Stanford}, 2.11; {Berkeley, Stanford}, 2.09; {Charniak, Bikel}, 2.05; {Berkeley, Bikel}, 2.01; {Charniak, Berkeley}, 1.99; {Bikel, Stanford}, 1.87. Since we need to achieve the best combination of maximizing parsers' accuracy and maximizing their mutual complementariness (i.e., maximizing their pair-wise ACB), we selected Charniak's maximum-entropy inspired parser and the Berkeley parser for co-training.

#### Experimental Results and Discussions

Table 1.27 shows the parsing accuracy F-measure (%) on BN-test under various parser training conditions on Charniak's parser and the Berkeley parser with-out reranking. As can be seen from the table, training Charniak's parser and the Berkeley parser using only the small training set of BN treebank, i.e., BN-seed, resulted in relatively poor parsing performance, at 75.1% F1 for Charniak's parser and 75.2% for the Berkeley parser. Using the larger full CTB corpus for training improves parsing performance significantly and adding BN-seed to CTB brought additional gain. However, co-training using CTB plus BN-seed as the initial training pool significantly improved the performance of the two parsers over directly training on CTB plus BN-seed, with 1.9% absolute and 2.1% absolute improvement on F-measure for Charniak's parser and the Berkeley parser, respectively.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Training Condition** | |  | **F-measure (%)** | |
|  |  |  | Charniak | Berkeley |
| **1.** | **BN-seed** |  | 75.1 | 75.2 |
| **2.** | **CTB** |  | 79.1 | 79.1 |
| **3.** | **CTB+BN-seed** |  | 80.4 | 80.5 |
| **4.** | **co-training initialized** |  | **82.3** | **82.6** |
| **as Condition 3, max-t-min-s** | |  |

Table 1.. Overall parsing accuracy F-measure (%) on the Mandarin BN treebank test set, BN-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without reranking.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Training Condition** | |  | **F-measure (%)** | |
|  |  |  | Charniak | Berkeley |
| **1.** | **BC-seed** |  | 72.0 | 72.8 |
| **2.** | **CTB** |  | 73.4 | 73.7 |
| **3.** | **BN-co-trained** |  | 74.7 | 74.8 |
| **4.** | **(CTB+BN)-co-trained** |  | 75.6 | 75.7 |
| **5.** | **(CTB+BN)-co-trained+BC-seed** |  | 76.8 | 77.0 |
| **6.** | **co-training initialized** |  | **79.3** | **79.5** |
| **as Condition 5, max-t-min-s** | |  |

Table 1.. Overall parsing accuracy F-measure (%) on the Mandarin BC treebank test set, BC-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without reranking.

For co-training carried out in these experiments, we used cache size as 10K sentences. Table 1.28 shows the F-measure from the two no-reranking parsers on BC-test under various training conditions. The condition **BN-co-trained** denotes the BN-seed treebank and thefinal annotated BN-unlabeled data after applying *m*ax-t-min-s co-training to the two parsers initialized on the BN-seed treebank. **BN-co-trained** significantly outperforms **CTB**, indicating greater similarity between the two speech genres compared to CTB vs. BC. Using CTB and BN-seed to initialize the two parsers and then co-training on the BN-unlabeled data achieved further gain on parsing performance, denoted by **(CTB+BN)-co-trained**. Consistent with Table 1.27, it is always helpful to add the small in-genre seed treebank into training, as **(CTB+BN)-co-trained+BC-seed** out-performs **(CTB+BN)-co-trained**. Co-training on BC-unlabeled also produced consistent improvement on F-measures.

Overall, we gained 2.5% absolute on F-measure on BC-test over the two parsers from co-training. Using the same BC-unlabeled data for co-training, we also compared initializing the two parsers with the condition of CTB only and the condition of adding the small BN-seed and BC-seed corpora, and observed that adding this small in-genre seed corpus always outperforms initializing with CTB only, by 1% on BN and 1.4% on BC.

|  |  |  |  |
| --- | --- | --- | --- |
| **Training Condition** |  | **BN-test F-measure (%)** | |
|  |  | Charniak | Berkeley |
| **co-training standard parsers** |  | 82.3 | 82.6 |
| **co-training reranked parsers** |  | 83.8 | 84.0 |
| **co-training rerankers** |  | **84.0** | **84.4** |

Table 1.. Overall parsing accuracy F-measure (%) on BN-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without reranking and with reranking.

|  |  |  |  |
| --- | --- | --- | --- |
| **Training Condition** |  | **BC-test F-measure (%)** | |
|  |  | Charniak | Berkeley |
| **co-training standard parsers** |  | 79.3 | 79.5 |
| **co-training reranked parsers** |  | 81.0 | 81.1 |
| **co-training rerankers** |  | **81.5** | **81.5** |

Table 1.. Overall parsing accuracy F-measure (%) on BC-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without reranking and with reranking.

The results from the two approaches of combining discriminative reranking and co-training, as pro-posed in Section 1.4.7.4, are shown in Tables 1.29 1.30. The results of **co-training standard parsers** are the last rows in Tables 1.27 and 1.28. When co-training reranked parsers, the rerankers were trained on CTB+BN-seed for BN and CTB+BN-seed+BC-seed for BC and remained the same during co-training. When co-training rerankers, the rerankers were initialized on CTB+BN-seed for BN and CTB+BN-seed+BC-seed for BC and updated during co-training. For both combination approaches, co-training explored BN-unlabeled for BN and BC-unlabeled for BC as unlabeled data, respectively.

As can be seen, **co-training reranked parsers** (using the*max-t-min-s* example selection approach) significantly outperforms co-training without reranking, by 1.5% absolute and 1.4% absolute gain on F-measure on the two parsers on BN-test, and 1.7% absolute and 1.6% absolute gain on F-measure on the two parsers on BC-test. For **co-training rerankers**, as discussed in the fourth section, it is feasible now for us to employ the more principled agreement-based example selection approach during co-training since we can simply train each reranker multiple times on different subsets of the automatically labeled data and examine which partition of the data produced the maximum agreement among the rerankers.

As a re-minder, for **co-training reranked parsers**, we still used the *max-t-min-s* approach as it is computation-ally efficient and also proved to be very effective for co-training parsers (Wang, 2008). As can be seen from the tables, **co-training rerankers** produced a small yet consistent gain over **co-training** **reranked parsers**, by 0.2% – 0.4% absolute improvement on BN-test and 0.4% – 0.5% absolute improvement on BC-test, raising the absolute improvement on F-measure up to 1.8% on BN-test and 2% on BC-test, from combining discriminative reranking and co-training compared to co-training only.

1. For Chinese data, 1.5 characters were counted as 1 word. [↑](#footnote-ref-1)
2. LDC’s web collection efforts are described in detail in Lee et al., this volume. [↑](#footnote-ref-2)
3. LDC2008E61 - Arabic Treebank Part 1 v 4.0 [↑](#footnote-ref-3)
4. LDC2008E62 - Arabic Treebank Part 2 v 3.0 [↑](#footnote-ref-4)
5. LDC2008E22 - Arabic Treebank Part 3 v 3.1 [↑](#footnote-ref-5)
6. As far as possible within the current time constraints and without a full-scale entirely manual review. [↑](#footnote-ref-6)
7. For now we have chosen to retain the noun and adjective labels for active participles, passive participles and gerunds/masdars, rather than switching to an entirely traditional representation of these categories. [↑](#footnote-ref-7)
8. <http://projects.ldc.upenn.edu/ArabicTreebank/>. [↑](#footnote-ref-8)
9. Future new annotation will use the newly revised LDC Standard Arabic Morphological Analyzer (SAMA) Version 3.1. LDC Catalog Number: LDC2009E73. [↑](#footnote-ref-9)
10. We found this version to be so useful internally that as of ATB5, ATB releases will also include a version of the corpus showing this mapping between the source text tokens and the Treebank tokens. [↑](#footnote-ref-10)
11. http://nlp.stanford.edu/software/parser-arabic-data-splits.shtml [↑](#footnote-ref-11)
12. http://www.cis.upenn.edu/~dbikel/software.html#stat-parser [↑](#footnote-ref-12)
13. Note, however, that due to differences in data and style of annotation, a comparison of scores across Treebanks is at best a rough indication of performance differences. [↑](#footnote-ref-13)
14. Note that this right branching structure is also used for tamyiz in some constructions. See the ATB syntactic guidelines for details on those constructions. [↑](#footnote-ref-14)
15. Personal Communication with Mohamed Maamouri. [↑](#footnote-ref-15)
16. http://www.nist.gov/speech/tests/rt/rt2007/ [↑](#footnote-ref-16)
17. http://www.nist.gov/speech/tests/rt/2004-fall/ [↑](#footnote-ref-17)
18. A previous publication about ALMORGEANA focused on the generation component of the system which was named Ara-gen (Habash, 2004). [↑](#footnote-ref-18)
19. Arabic transliterations are in the Habash-Soudi-Buckwalter transliteration scheme (Habash et al., 2007). [↑](#footnote-ref-19)
20. http://chasen.org/ taku/software/yamcha/ [↑](#footnote-ref-20)
21. For example, the definite article preceding sun letters in Arabic. [↑](#footnote-ref-21)
22. A clear demarcation of the definite article Al is present in the fully vowelized version of the PATB. [↑](#footnote-ref-22)
23. Prepositions and definite articles, by definition, do not affix onto verbs and similarly, future marker and verbal clitic È l clitic do not prefix onto nominals and do not co-occur. [↑](#footnote-ref-23)
24. <http://ilk.uvt.nl/sabine/chunklink> [↑](#footnote-ref-24)
25. In some cases, the parse results do not take into account the brackets of gold trees for sentences that do not produce a final parse in the score, and so the reported results are more optimistic than they should be. [↑](#footnote-ref-25)
26. The tool can also provide scores based on all of the head dependencies extracted from the system and reference trees, as well as a more focused set of open class dependencies, which involve open class content words. [↑](#footnote-ref-26)
27. When evaluating test parses that have the same word segmentation as the gold standard parses, SParseval provides scores on a sentence-by-sentence basis, just like EVALB. [↑](#footnote-ref-27)
28. The major motivation for re-implementation is to allow more generic and flexible state-tying operations that are important for some algorithms we are developing. [↑](#footnote-ref-28)
29. We used Bikel’s randomized parsing evaluation comparator to determine the significance (*p* < 0.05) of difference between two parsers’ output. [↑](#footnote-ref-29)
30. If we accidentally parse sentences using a model trained with a mismatching representation of punctuation, say the sentence contains full-width punctuation but the parser was trained with parses containing half-width punctuation, then the parse accuracy would be quite low, dropping from 82.90 to 72.16 F score. This reduction in parse accuracy results because the punctuation marks are not recognized as such. [↑](#footnote-ref-30)
31. If we had not normalized the treebank data for speech, the parsing accuracy would have been far worse due to a mismatch between the training and testing conditions. [↑](#footnote-ref-31)
32. SIGHAN is a special interest group under ACL focusing on Chinese language processing. Website: www.sighan.org [↑](#footnote-ref-32)
33. Available from the Linguistic Data Consortium. Website: www.ldc.upenn.edu [↑](#footnote-ref-33)
34. A small amount of Hong Kong news has been folded into Xinhua newswire section in CTB 6.0. [↑](#footnote-ref-34)
35. http://nlp.cs.nyu.edu/evalb [↑](#footnote-ref-35)