Part 6: Operational Engines

Editors: Douglas W. Oard, Carl Madson, Joseph Olive, John McCary, and Caitlin Christianson

Chapter 6.1 Introduction

The goal of the GALE Program is to empower the warfighter through the use of language technologies. In order to accomplish this goal, the technologies must be integrated into operational engines that meet the needs of both warfighters and those who support them. That integration imperative leads directly to the two key parts of this chapter: how to perform that integration, and assessing how well we have met operational needs.

The chapter begins with a look at the deployment history over the past decade and a half for speech and language technologies in operational environments. As Kathleen Egan and Allen Sears (Section 6.2.1) explain, this has been an evolutionary process driven by an increasingly rich understanding of warfighter needs. The second contribution provides details on the design of an operational engine that has recently been deployed in operational environments around the world, including in direct support of operational forces in the field. As Daniel Kiecza and his colleagues (Section 6.2.2) explain, meeting the need for responsive and accurate performance by integrating systems that each have a different development heritage can be a substantial undertaking, sometimes requiring development of new technologies. As an example of that process, Amit Srivastava and Daniel Kiecza (Section 6.2.3) describe the development of a new technique for establishing sentence boundaries during speech recognition that substantially improves the latency for subsequent translation of the recognized speech.

The integration challenge in GALE extends well beyond pairing components, however. John Pitrelli and his colleagues (Section 6.2.4) introduce the full scope of the challenge by describing the GALE Interoperability Demonstration (IOD) system. Coordinating real-time processing by eleven different types of language technologies would be a substantial undertaking under any circumstances; doing so in a distributed manner using the facilities of seven research groups spread over thousands of miles in three countries raises the challenge to an entirely new level. But, it is exactly that level at which we must learn to work if we are to learn how best to integrate the diverse language technologies that we are now able to create.

The core technology behind the GALE IOD system is the Unstructured Information Management Architecture (UIMA), a flexible integration platform designed to meet the unique demands of distributed language processing. Eric Nyberg (Section 6.2.5) and his colleagues describe how UIMA-enabled components can be shared using GALE's UIMA Component Repository (UCR), and how individual researchers can flexibly configure entire UIMA processing environments using GALE's UMIA Component Container (UCC).

The unprecedented complexity of the systems that we can now assemble poses substantial challenges for evaluation. We know from intrinsic evaluation how well individual components, and even specific types of component combinations, meet our expectations for their behavior. However, calibrating our expectations for parts of a complex system against the true needs of users of the entire system is another matter altogether. No one approach can provide a comprehensive answer to every question we must answer about the utility of the systems that we build, so in the second part of this chapter we look at three ways that GALE researchers have gained such insights. Daqing He and his colleagues begin that process by describing a series of user studies to evaluate the utility of specific ways of integrating language technologies by using the Evaluation Dataset for Integration Exploration (EDIE) that they developed for GALE to create a controlled evaluation environment. Studies of this type by each GALE team offered a principled basis for system design decisions.

To bring GALE systems from the research lab to the warfighter requires that we augment these initial controlled user studies with additional user studies in representative environments and with representative users. Douglas Oard and his colleagues tackle the challenge of evaluating a fully integrated system in a representative environment by running a rich set of formative evaluation studies using live content, designing tasks based on real-world events as they were unfolding. Connie Fournelle and her colleagues then put experienced intelligence analysts in front of fully integrated GALE systems and challenging them with realistic tasks. Never before has it been possible to deliver this level of integrated language to real analysts, and the results of this important study have served to ground resource allocation decisions throughout the remainder of the GALE program.

As with many difficult challenges, robust investment in system integration is, perhaps, more often preached than practiced. We have forces deployed in harm's way in the service of our nation, however, and those brave men and women demand and deserve our best effort to create the robust and effective integrated systems that they accomplish what we have asked of them. Unprecedented challenges call for innovative approaches, and our goal in this chapter is to describe how the GALE research teams have risen to meet that challenge.

Chapter 6.2 Implementation of Operational Engines

6.2.1. Bridging the Gap between Research Advances and Operational Users

Authors: Kathleen Egan and Allen Sears

6.2.1.1. Introduction

While the First Gulf War in the 1990s marked the rise of CNN and global television as the mass communication media of preference, Western Media were dominant, with English as the principal language of communication. The Internet was still regarded as a research tool, with little to no consumer penetration. Al Jazeera, the television channel

most widely associated with Operation Enduring Freedom (OEF) and Operation Iraqi Freedom (OIF), would not be founded for another six years.

The efforts in bringing new technologies to bear on the operational setting started with the first meaningful Arabic data collection in 1998, including a variety of Arabic media sources, such as Al Jazeera. Established in 1996 with a grant from the Emir of Qatar, Al Jazeera's BBC-trained staff brought the styles and techniques of Western media to the Arabic-speaking world for the first time. Covering the politics and conflicts of the region with a relatively apolitical eye, Al Jazeera accomplished the feat of appearing to be both anti-American (from the point of view of the U.S.) and anti-establishment (from the point of view of regional governments and powers). It was Al Jazeera's airing of Al Qaeda's home-produced video statements that brought the station to worldwide attention.

Because there were so many media outlets to monitor (including a rapidly increasing number of Web pages and Blogs), by 2000 it became clear that U.S. personnel could no longer keep up with the inflow of media at all levels. No automation existed to stem the flow, although various organizations attempted to handle it by force of human effort. In addition, the vast majority of the content was in languages outside the Cold-War stalwarts of Russian, German, and the like, more than doubling the challenge to keeping up using human translators.

The Department of Defense (DoD) graduates approximately 2500 linguists each year, of which about 750 specialize in Middle Eastern languages. Today these numbers satisfy only 10-15% of the military's requirements for interpreters and translators. The current requirement is for about 10,000 linguists. With a rough annual cost of training each individual at approximately $250,000, the military cannot afford to train enough qualified linguists to support today's needs (Gilmore 2007). Moreover, even if enough linguists in all the priority languages were available, the amount of data to be analyzed is growing exponentially, and humans alone cannot monitor all of the available data in a timely fashion.

Language is at the heart of human communication and having timely and accurate translations can save lives. Words are powerful. They have infuriated populations, created misunderstandings, fueled hatred, and built barriers across cultures and civilizations. At the same time, words have opened doors, reconciled groups, gained peace, and created laws for justice, order, democracy and freedom. The imperative with the sea of words from around the world is for accuracy, but more importantly, the meaning of each word in context.

In the past ten years, human language technology (HLT) insertion in user-friendly solutions stepped in to bridge the two major capabilities gaps afflicting U.S. operations: **media triage** and **foreign language comprehension**. Various government agencies are now leveraging GALE advances and building on the on-going work of the Technical Supporting Working Group (TSWG). TSWG is an interagency research and development arm of the Office of the Secretary of Defense that is funding media monitoring development to enable the warfighter to gain immediate actionable intelligence from open source media.

6.2.1.2. Pre-GALE Efforts: Early Applications of HLT in the Intelligence Community (IC) and DoD

Throughout the 1990s, DARPA funded basic research in speech recognition (sometimes known as speech-to-text) and machine translation. While incremental progress had been made in the accuracy of transcription and translation during that decade, transition of these technologies to operational use had occurred only on a piecemeal basis. Users of these technologies were largely responsible for envisioning a concept of operations that would be introduced into their existing business processes, all of which assumed manual, human-intensive approaches. As such, technology transition was limited, and early adopters were not fully supported in the insertion of advanced language processing technologies into operational settings. Most user groups did not have strong information technology support, and did not want to depend on unreliable machines in support of operational tasks. Progress in transition has improved over the years through the adoption of a strategy of testing technologies with proxy users and then engaging them to define and improve on the proposed concept of operations. However, it is only when moving technology from an experiment to an operational prototype and later to a fully integrated system in the workplace that full transition and technology maturity occurs.

Before 2000, the overall technical goal for automated language processing was to reduce word error rates for speech, and achieve fluency and accuracy measures for machine translation quality. A shift started around 2000, with a goal of supporting users in the language processing tasks with which they were faced on a daily basis. From 2000 to 2003, three major efforts in the IC and DoD laid the groundwork for bridging the gap between a community of users who relied heavily on human capital for all of its analysis, and a research community that focused on the hard scientific problems of speech recognition and machine translation.

OASIS, TAP projects, and MAPS were the predecessors to the broadcast monitoring and Web monitoring capabilities that now support language needs and are currently deployed in a growing number of operational sites.

6.2.1.3. OASIS

In 2000, the Intelligence Community wished to broaden and accelerate the coverage of open-source broadcast media to Government decision makers. The Foreign Broadcast Information Service (FBIS, now known as the Open Source Center) was responsible for media processing that was extremely labor-intensive. The sources were non-English, and native speakers performed selection, transcription, and translation of the audio sources. Selection was the process of locating media that was "newsworthy" for dissemination. Transcription was the process of transforming the media into typewritten form in the source language, and translation converted the non-English transcript into English for all selected excerpts.

FBIS was interested in automating the processing pipeline, starting with the selection process, and considered using automatic speech recognition as a solution that would turn a listening problem into a reading or skimming problem, and eventually into a searching problem. Furthermore, having a transcript as a starting point would speed up the eventual

translation of the media. To test this theory, BBN was awarded a contract to develop the OASIS system and use technology that BBN had been developing for DARPA called "Rough and Ready" (Kubala *et al.* 2000). OASIS allowed the scheduled recording and transcription of broadcast news quality audio, initially in South Asian-accented English, and later also in Arabic and Mandarin Chinese. OASIS was deployed to several international customer locations for use by local media analysis staff.

OASIS proved to be highly successful at this task. While initially regarded as a threat by the local translators, the system acted as an additional set of "digital ears", monitoring stations that the customer wished to cover, but for which they lacked staff to conduct real-time reviews. OASIS recorded and transcribed stations on a time-shift basis, then allowed rapid review of the captured content for intelligence value. In a 2000 Weekly Activity Report (WAR) to the Secretary of Defense, DARPA reported that automatic language processing technology had been licensed by the commercial sector and would be used for automatic indexing of audio information, such as radio broadcasts. Basic speech recognition and information extraction components came from previous DARPA investment in the Human Language Systems program.

The users concluded that being able to view the text while gisting the most important excerpts of the news made it possible for analysts to cover a larger number of hours of captured media in substantially less time. During these early OASIS prototypes testing, analysts reported that the word error rate did not bother them as long as they could corroborate the accuracy of the transcription by listening to the audio. This was important feedback from users that helped shape technology development priorities and user interface.

OASIS marked the first major successful use of human language technology to bridge the media triage gap in an operational context for an operational government customer (Shepard *et al.* 2002).

6.2.1.4. Text and Audio Processing (TAP) experiments

In late 2003, the DARPA TIDES program was focused on using HLT to enable native English speakers to access and manipulate non-English media with the same ease as English media. The TIDES program held regular pseudo-operational evaluations known as Integrated Feasibility Experiments, or IFEs, allowing proxy users to work with the technology developed by the research community in a controlled environment. Three organizations participated in the TAP experiments and helped shape the technology from their own perspectives. MITRE developed the MiTAP system that focused on information management of language reports on the Web regarding identifying and tracking disease outbreaks. Both Virage and BBN developed experimental systems (ViTAP and OnTAP respectively) to harvest and translate open-source Web content from news sites. ViTAP and OnTAP allowed analysts users to write reports based on the content processed by the system. Both systems were given operational prototype stress-tests in 2004 at the Strong Angel II humanitarian response technology demonstration.

In September 2001, the world's attention shifted to the 9/11 attacks by terrorists on the U.S. The DARPA HLT programs, which had been developing English transcription and translation capability, immediately focused all their effort on Arabic. Terrific progress was made on automated Arabic language processing in 2002 and 2003, and by

the end of 2003, DARPA was ready to conduct an operational prototype experiment with an Arabic system. In December 2003, DARPA management approved an "experimental Text Information Retrieval (eTIR)" project that was based on the emerging TAP technologies.

DARPA, with support from the National Virtual Translation Center (NVTC), developed an eTAP initiative with the J2 Open Source Cell at United States Central Command (CENTCOM) in Tampa, Florida. The Open Source (OSINT) Cell was charged with collecting and reporting on Middle Eastern media coverage of U.S. actions in Iraq and Afghanistan for the CENTCOM commander. Their product was known as the "Foreign Media Perceptions Digest" and was published three times a week. The technology at their disposal consisted of a set of America Online accounts, which they used to review English Web sites reporting on news in the CENTCOM theater. The OnTAP technology was selected to be modified and applied to the OSINT task for the eTIR project.

With support from DARPA, BBN developed an operational prototype variant of OnTAP called eTAP, for CENTCOM's exclusive use. eTAP collected Web pages from Arabic language news Web sites, translated them using a machine translation engine from Language Weaver, Inc., and made them searchable in English through a simple user interface. The system was built to provide a platform on which to develop a CONOPS for automated language processing in support of the Enduring Freedom effort in Iraq, and to measure the contribution of the technology in response to the operational needs. Even though the machine translation capability was far from perfect, the hypothesis was that the current state of the art in machine translation was of sufficient quality to allow accurate gisting of open-source content.

The eTAP system allowed the CENTCOM users to set up personalized search terms, called a "watchlist," and review them at will. Using the rough machine translation to judge the relevance of an article, the users could click an on-screen button, which would task a human linguist at the NVTC in Washington, D.C., to create a more accurate, human-vetted translation. This was performed first overnight, then in less than two hours if deemed critical, and returned to the analysts at CENTCOM for use in their reports.

The eTAP capability proved to be extremely successful. During the first six months of operational use, the system harvested over 1.5 million web pages. CENTCOM staff selected over 1,300 automated translations for human gisting, and used over 99% of them in their final products, rejecting only two translations because they were either not relevant or redundant. As a direct result of their new level of success, CENTCOM increased its publication rate from three news roundup products per week to multiple daily products and expanded its staff to more than ten analysts and linguists. Growth in demand occurred in part due to the increased visibility brought about by the technological assist; instead of reducing overall labor, the system helped to create a capability. The use of eTAP at CENTCOM marked the first automated system that provided foreign language comprehension to the active warfighter.

6.2.1.5. MAPS

In 2003, TSWG responded to a requirement from the Deputy Directorate for Information Operations (DDIO), a component of the Joint Chiefs of Staff J2/J3

Operations Directorate based in the Pentagon. The DDIO was focused on winning the international information campaign against terrorism. Their primary function was to monitor, collect, and resolve conflicting information from domestic and international broadcasts that was relevant to U.S. military interests. Timeliness of the collected information was of the essence.

The DDIO had little automation support; staff members kept televisions on all the time, waiting for relevant broadcasts to air. When an important story appeared on the screen, staff members and linguists would run into a media room equipped with a set of digital video recorders to attempt to translate the news program. The DDIO was seeking automated methods to allow them to monitor an increasing amount of media with the same or fewer staff.

IC and DoD entities have always monitored foreign media using language analysts and/or native speakers to watch the news and report on critical events. This human approach was time and resource intensive, leading to a strong requirement to add support and go beyond the human effort. TSWG responded to DDIO need by capitalizing on the research gains realized by DARPA and moving them into an operational context, with real users who would rely on the technology for their day-to-day work. The practice of humans watching television news and reporting the highlights at the end of the day was deemed insufficient and unpractical when timely responsiveness and documented evidence of facts were paramount in gaining situational awareness and combating terrorism. The realization that human-machine collaboration was a better approach for timely handling of the volume of data motivated the subsequent development, architecture, configuration, integration and deployment of systems in operational environments.

TSWG leveraged OASIS and funded a team led by BBN to develop and deploy the Multimedia Alert Processing System (MAPS), delivered to the DDIO as a prototype in two phases. The initial prototype supported monitoring of two English and three Arabic television channels. The second delivery incorporated upgrades and features requested by the user community. MAPS was the first turnkey application that was physically deployed in two operational settings (DDIO from 2003 to 2004 and CENTCOM from 2003 to the present) combining audio, text, video, and translingual search into one application.

TSWG funded extensions and improvements in response to specific requirements from users at the CENTCOM J2 OSINT section targeted at increasing its analysts' ability to access Arabic-language media. These requirements included expanding its MAPS system to two channels and enhancing the eTAP Web harvesting system.

At the same time, three overseas deployments of MAPS took place in support of Operation Iraqi Freedom: one with the Department of State Multi National Forces - Iraq (MNFI), one with PsyOps, and one with Special Forces.

6.2.1.6. TALES (Translingual Advanced Language Exploitation System)

In 2005, the Joint Intelligence Center PACIFIC (JICPAC) and DARPA planned a field test experiment to exploit web and broadcast information coming from the Pacific area of operations. DARPA selected and funded IBM for the experiment and decided that the focus should be on Chinese transcription and translation. IBM was to develop

and insert a prototype system to be called TALES. The hardware and software were to be packaged in a "deployment unit" installed at JICPAC in Hawaii. Open source processing personnel at JICPAC were given the task of developing analytical models to predict the possible impact of automated language exploitation capability. In addition, the DARPA/GALE Leading Edge Environment (LEE) effort would demonstrate capabilities through the use of IBM's UIMA (Unstructured Information Management Architecture) framework. The objective was to test data management and technical integration point approaches in order to discover both strengths and weaknesses relevant to future JICPAC operations.

The plan was for DARPA and JICPAC to develop a strategic relationship based on leveraging current language processing components within the UIMA framework to offer PACCOM an open-source processing capability that could be expanded to support requirements for an increasing volume of Chinese translation while reducing turn-around time and improving open source products. However, one lesson learned from the TALES insertion effort was that there was a delicate balance in opportune timing between capability maturity and technology insertion, as highlighted by the relative lack of maturity and capability of Chinese audio and text processing. At the time, the research capabilities for Chinese language processing were not sufficient to be of assistance to operational users. DARPA learned about the shortcomings in Chinese processing techniques from TALES and from metrics-based testing. This led to increased efforts to find new approaches aimed at improving automated translation. Chinese translation techniques are still a high priority for DARPA/GALE today.

6.2.1.7. BMS and WMS

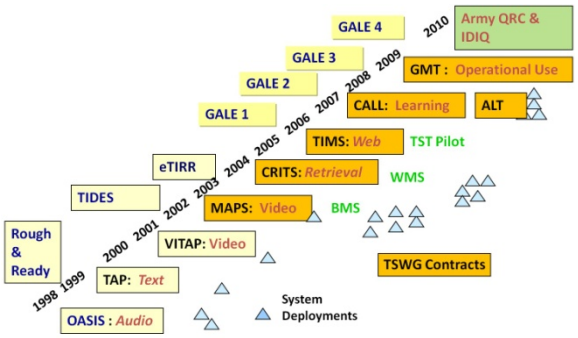

Figure 6.1: Timeline of HLT R&D efforts

The approach adopted in these efforts was to mitigate machine errors and focus on robust retrieval methods, maximizing the users' expertise by providing intuitive, user-friendly interfaces and productivity tools that supported the users' day-to-day operations. TSWG insisted that development be closely linked with efforts to observe, listen to, and respond as a team to the unique operational needs of users. A key objective was to exploit open-source media in order to efficiently and effectively monitor the growing impact of

that media on military and intelligence analysis and actions. Both the BMS and WMS helped to moderate the compounding challenges of access to the growing volume of data and the limited number of analysts and hours in the day to complete operational tasks.

Since 2005, the BMS and WMS have been available as fully-functional, turnkey systems that can be deployed to customer sites in a leave-behind mode for operational use. Since these two systems have provided the most successful insertions for GALE technology to date, a short review of the applications follows:

The BBN Broadcast Monitoring System creates a continuous, searchable archive of foreign language television broadcasts. For a complete description of BMS, see Section 6.2.2.6.

The BMS allows users with no foreign language skills to get the gist of a broadcast and triage enormous volumes of media, allowing linguists to focus on translation tasks. Users can quickly find spoken content in the video archive through keyword queries — in English or the source language — and play back video from any point in the system's one-year internal cache. Searches can be saved in a watch-list and are automatically populated by the system as new matches are found.

Figure 6.2: WMS Article View

The BBN Web Monitoring System (WMS) is an end-to-end system providing the capability of collecting, organizing, and translating open-source content from the World Wide Web. The WMS integrates and manages the workflow of the media analysis process from beginning to end — from data collection and processing, to automated triage and retrieval, to machine-assisted translation and support for human translation, to publication and dissemination.

WMS continuously captures content from user-selected Web sites and stories it in an archive that can be shared by multiple, distributed user groups. The captured site is archived and versioned for later use.

WMS supports over 30 input languages. Pages are automatically translated into English using machine translation software from Language Weaver. English speakers can use the machine translation to get the gist of an article, and then reach through the network for high quality human translation support.

In addition to the insertion of GALE advances in speech and machine translation through system upgrades, we have found that developing assistive tools to improve workflow is critical. Users differ in their needs for the output of the data. By working with them directly, we have been able to adjust the applications to fit the users' tasks and facilitate user acceptance of the technology.

6.2.1.8. User Feedback

Technology transition to a community of operational users involves a delicate balancing act. The natural tendency for technology providers is to attempt to field the most aggressive, highest performance application that represents the fruit of the labor of the research community. Yet, what leads to rapid adoption tends to be much different. TSWG insertion experiments have demonstrated that rapid adoption is best achieved if applications are simple to use and integrate seamlessly with users' tasks. Moreover, prototype systems must be robust and require little in the way of maintenance. The success of the TSWG/DARPA partnership lies in the ability to address both of these seemingly conflicting goals, resulting in applications that can be fielded to environments as harsh as a dusty machine room in Iraq, and still result in measurable improvements in military operations.

Over the past ten years, in successive deployment of automated language technologies, the key lessons learned have been that users care about quality and give the highest priority to applications that support their operations. The successes of the TSWG/DARPA partnership can be attributed to the rapid transition of research engines improved by GALE algorithms into useful applications. Human language technologies, such as speech-to-text and machine translation, by themselves are not sufficient to empower users to perform analytical tasks. Rather, these technologies are critical enablers of knowledge discovery, data exploitation, trend analysis, and product creation. Military and other Government users, including both language professionals and operators with no foreign language skills, must find an application to be user friendly and to match their operational workflow.

Starting in 2004, the CENTCOM Open Source Intelligence Cell became the true test bed for a full operational insertion of HLT. The experience deploying the WMS and BMS at CENTCOM represents a case study of this effect. Since 2004, the CENTCOM J2 Open Source Cell has grown from three users creating a thrice-weekly handcrafted product to a 24x7 operation with more than 25 analysts, creating multiple daily products and servicing in excess of 2,500 requests for information annually. Functioning as a stateside laboratory, both the WMS and BMS evolved based on requirements that were expressed by the CENTCOM user community and then rolled into the technology baseline.

Examples of products enabled by this technology include: responses to requests for information (RFIs), daily Foreign Media Trackers, weekly Pandemic Threat Updates, weekly Friday Sermon Analyses, weekly Threat Network Bulletins, and many more that

are sensitive but critical to operations. As an indicator of success, CENTCOM started to anticipate RFIs, and started producing CIPs (Critical Information Products) to answer critical information needs before they had even been requested.

CENTCOM management has stated that:

> *Our whole operation is based around these [foreign language processing] technologies and without them we would have to rebuild our operation from scratch."*

Figure 6.3 shows the quantity of RFIs and CIPs satisfied by CENTCOM staff from 2005 through September 2009.

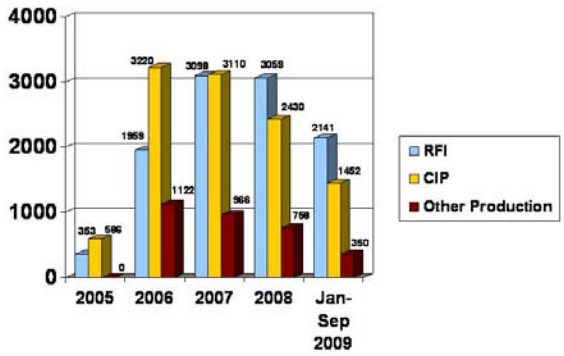

Figure 6.3: CENTCOM RFI and CIP Production, 2005-09

Working hand-in-hand with users in domestic command center environments, such as that found in Tampa at CENTCOM, helped shape capabilities and CONOPS, which in turn increased the odds of success in more hostile environments. For example, deployments to Iraq were successful as a result of the intense insertion experiments in Tampa. In fact, four deployments of BMS and WMS to units in Iraq occurred from 2005 through 2009. In all cases, deployments were handled by contractors with minimal pre-deployment contact with the fielding units. Hostile conditions are taxing and challenging. For example, reliable power is rare, with abrupt power failures regularly damaging components. In addition, audio/visual and internet capabilities are obtained by receiving units through local companies in neighboring towns in Iraq. Another serious problem is the fact that units rotate back to the United States every six to twelve months, with virtually no handover to the incoming unit.

Despite these challenges, user acceptance has been extremely high for the BMS and WMS. One site in Baghdad reports that:

> *BMS was a true labor saver. With dozens of Pan Arab media outlets to monitor, BMS provided a 24/7 eye on what was happening in the information space. Countless times, BMS captured a piece of news that our human monitors did not catch. With the information battle space a key component of war fighting, we had a responsibility to not only be first with our*

information, but also to ensure media reporting was accurate and provided the right context and characterization. BMS was a key enabler for the communication team.

The technical transfer and integration of operational engines from DARPA/GALE as described in Operational Integration of STT and MT for Rich Transcription and Translation of Speech (Chapters 1-3) became essential to demonstrate the value of research in responding to critical national security and military needs (Section 6.2.2).

On a yearly basis, TSWG and BBN bring together the users of the BMS and WMS. Lessoned that have been learned from real users include:

> *It WORKS! It is an integral piece to our daily operations. It speeds intelligence action; it doesn't need systems personnel to work it. It is straightforward and intuitive. It only takes 1-2 hours to train someone to use it.*

Positive comments do not come only from early adopters, but also from the latest deployments in Iraq. Starting in 2009, the users' meeting agenda included review of the Foreign Media Monitoring System funded by the Army Machine Foreign Language Translation Software (MFLTS) to gather feedback and determine new requirements. MFLTS acquired systems that fused together BMS and WMS capabilities. The Multi-National Coalition in Iraq called in to the 2009 users' meeting to share their feedback, stating:

> *Our Open Source Team fills in some expected changes in reporting that are a result of having U.S. forces out of the Iraqi cities since June. The reliability of Open Source reporting can be questionable so having multiple sources to review allows us an opportunity to get different perspectives on the same story—[the system] has given us eyes on events that we do not ordinarily have access to due to the June Security Agreement that removed U.S. Forces from the Iraqi cities. Immediate unfiltered access to reporting in local and national Iraqi media allows for immediate validation of other intelligence reporting, i.e., validate claims of attacks, confirm events.*

The BMS has become an essential tool for intelligence analysts as showcased by Bemish (2008) and by user testimonials like those above.

6.2.1.9. Expanding the Impact of Automated Language Processing

From 2003 to 2009, TSWG and other IC and DoD groups have funded various versions of the BMS and WMS. All of these efforts have helped move language processing technology from the research lab to the hands of the warfighter. Like any extremely large organization, the DoD has its own procedures for deploying new systems and capabilities, ensuring that they are sustained in the field, that users are adequately

trained, that system use is incorporated into doctrine, and that systems continue to be enriched to meet evolving needs.

Two DoD organizations have stepped in to organize the set of requirements coming from the field and to transform them into a cohesive program: 1) The Army Machine Foreign Language Translation Software (MFLTS) Program of Record has taken the lead in fusing the WMS and BMS into a single Foreign Media Monitoring (FMM) system. 2) The Defense Intelligence Agency (DIA) has taken the lead in supporting intelligence capabilities for the combatant commands at the joint level. These interagency efforts will benefit all the users of the deployed applications.

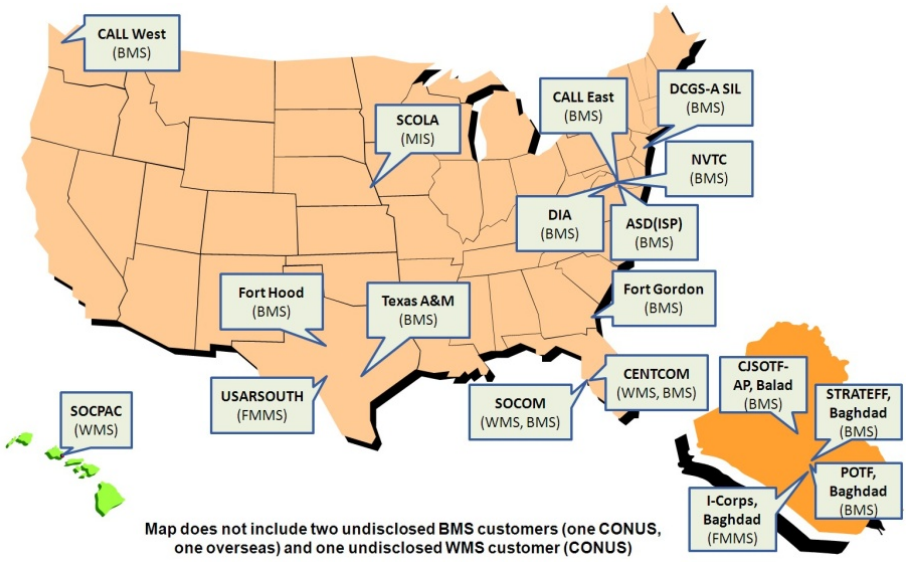

Figure 6.4: Current BBN Media Monitoring Deployments, January 2010

Starting with a few experimental efforts around the year 2000, the scope and maturity of deployed systems have increased significantly. Currently, there are 24 sites with 86 channels in the U.S. and overseas, with support in languages beyond the GALE core languages of Arabic and Chinese, including Farsi, Indonesian, Urdu, and Hindi. These language capabilities have been funded by TSWG, using GALE methodology to create and field operational systems. As these systems are put into service-wide use by the Army, DIA, and other organizations, language requirements will undoubtedly be addressed at the joint level.

These interagency efforts will benefit all the users of the deployed applications. New requirements are emerging to extend capabilities to more sources, such as radio, online video sharing sites, and social networks.

6.2.1.10. Dual Use of Technology

While the main users of the applications that have been described are language analysts and operators in the field, the BMS has been modified for dual use in the cultural awareness and language learning arenas. In every single operational deployment,

language analysts have stated that they wished they had had these technologies available to them when they were learning a foreign language. Access to authentic materials, especially video content, is highly valued among instructors and language learners who aim to sustain their skills and stay up to date on cultural issues. In this role, the BMS provides instructors and students with access to authentic material that can be used in the creation of language-learning content and activities. This material is authentic, topical, and current, and contains pronunciations by a variety of native speakers. Accessing video cuts is a labor intensive task for instructors, and in the words of one of the instructors using the BMS: *The BMS saves me time and makes it easy to find the right videos to insert in my lessons.*

While analysts may need 24x7 monitoring of each source, the language learning community prefers to focus on key shows and times of the day to gather data of interest. This requirement led to the development of a DVR-like scheduler that an administrator can customize to capture a variety of programming around the clock, multiplexing many sources into a single BMS channel. This multiplexing capability reduces the overall system cost and exposes users to a greater variety of sources.

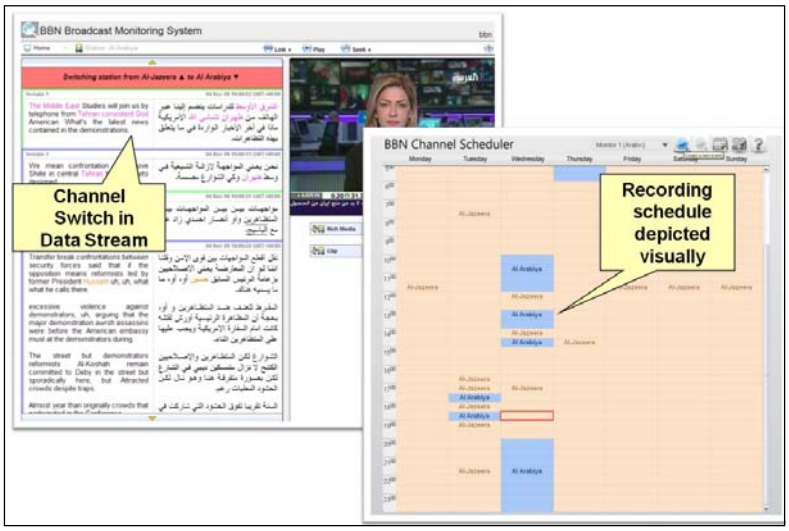

Figure 6.5: BMS scheduler

Three current deployments of the BMS to language learning centers provide instructors the ability to search video to create language activities based on authentic clips with full synchrony of video, audio, and text. This rich media is making it easy for teachers to create materials, and for learners to improve and sustain their language skills. In addition to making it possible for users to edit both the transcription and translation, third-party tools were integrated to enhance the learning process.

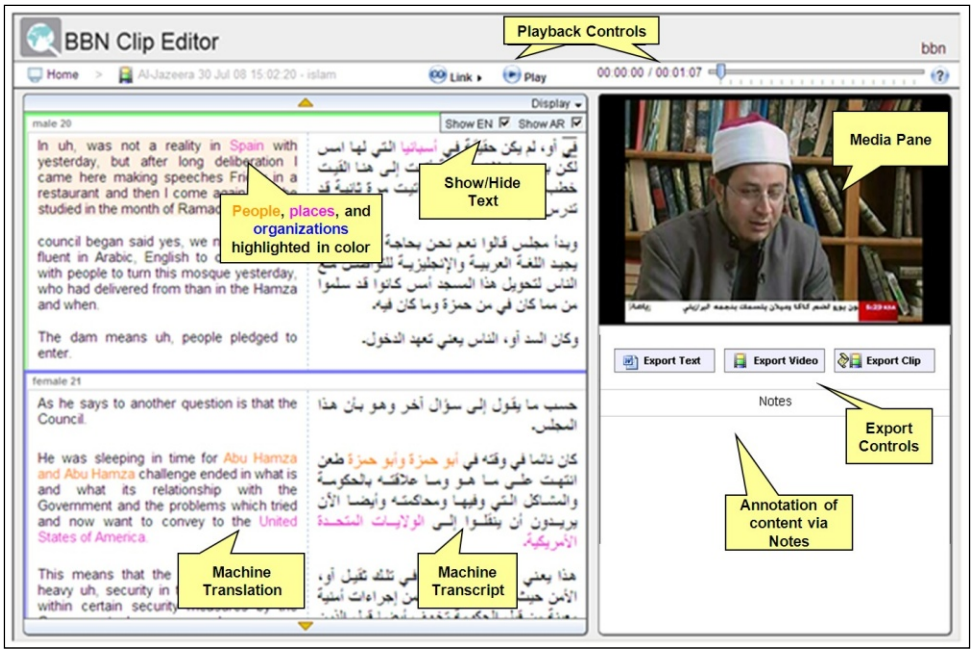

Figure 6.6: Clip Editor

Figure 6.7: Building Rapid Rote

The latest additional tool to be integrated provides the ability to create flashcards (for use in Rapid Rote from Transparent Language) with terms and expressions for further practice. Users can add notes to highlight cultural and linguistic features of a word, sentence and paragraph.

The adaptation of the BMS to language learning requirements has not only supported the needs of this new community of users, but also promoted development new functionalities and user interfaces that can be offered to all BMS users.

6.2.1.11. Conclusion

The improvements to operational engines made under the GALE program have made it feasible for the Combating Terrorism Technical Support Office/TSWG to insert state of the art human language technologies into capabilities that have been developed and deployed in operational settings. These technologies are not intended to be a replacement for human language analysts, but rather an extension to support their task. The heavy dependence of HLT training algorithms on language, training domain, sources, and channels is a problem that will need to be resolved in order to make fuller operational implementation of HLT possible. While machine-extracted content metadata hold great promise for effective data triage and operational adoption has been successful in assisting open-source analysis missions, world conditions and mission objectives are constantly changing, and the static nature of current HLT algorithms will need to evolve and adapt to the dynamic and unpredictable content of operational data. It is critical to emphasize the importance of human-machine collaboration in achieving successful deployment of HLT. An excellent approach is to focus on the tasks of the users and respond to requirements by soliciting the expertise of humans who engage the automated systems.

TSWG's mission is to facilitate the emergence of innovation based on user requirements and build effective solutions to support specific missions. TSWG will continue to rely on the research community to advance and improve the quality of automated speech-to-text and machine translation algorithms and expect to see those advances integrated into systems that fill operational requirements.

Addressing both ends of the research and development pipeline is vital to achieving success in providing functional solutions to the warfighter. For both DARPA/GALE and TSWG, success in bridging the gap between research and operational reality has been due to a combination of factors—vision, teamwork, listening to users, taking risks on new ways of doing business, and integrating top-of-the-line operational engines into user-friendly applications.

Acknowledgements

Since 2003 to present, David Herrington of DoD/CTTSO/TSWG has provided valuable resources and guidance. We also would like to acknowledge and thanks all of the users, -the early adopters- as well as the researchers and developers on the team who have made this DARPA/TSWG leveraging a success story. We would also like to thank Charles Wayne for his early contributions to the Pre-GALE efforts.

6.2.2. Operational Integration of STT and MT for Rich Transcription and Translation of Speech

Authors: Daniel Kiecza, Amit Srivastava, Guruprasad Saikumar, Sean Colbath, Martha Lillie, Prem Natarajan, Abdessamad Echihabi, Daniel Marcu, and William Wong

6.2.2.1. Introduction

In this section, we describe the joint engineering effort between BBN Technologies and Language Weaver to produce a tighter and richer coupling between their respective Speech-to-Text (STT) and Machine Translation (MT) systems. The coupling includes the sharing of orthographic normalization rules, synchronization of lexicons between the two systems, and definition of an extensible interface that supports increased information

flow between the engines. We show that tighter operational integration leads to better overall system performance. We demonstrate that the integrated rich transcription engine with automatic translation has great practical utility for intelligence analysis operations as the primary enabler of the BBN commercial off-the-shelf (COTS) Broadcast Monitoring System (BMS).

Goals

State-of-the-art STT capability at BBN can automatically transcribe foreign language speech at real-time throughput. The Language Weaver MT system offers automatic translation of text in many foreign languages into English. Both systems have been available as part of various commercial product solutions from BBN and Language Weaver, respectively, for some time.

The goal of the Operational Integration effort under the GALE program is to bring the two systems together into a joint tightly-integrated product offering that can produce a rich transcript of foreign language audio and video data with automatic translation of the transcribed speech content to English. The integrated system must run on a single commodity compute server and must be capable of continuous, real-time, low-latency processing of an audio signal.

Beyond the immediate benefits of an integrated operational system we anticipate that the integrated system should match or exceed overall system performance in comparison to operating a loosely-connected pipeline of the two individual systems.

Challenges

In our attempt to integrate the BBN Audio Monitoring Component (AMC) and Language Weaver MT systems we faced several formidable challenges.

BBN and Language Weaver (LW) have invested significant resources for many years to productize their respective systems. In the process, each system was tuned to perform optimally on state-of-the-art commodity hardware. Can the two systems run on a single server and still meet their respective operational requirements? Or do we need to scale each system down significantly to make the integrated system work and will this cause significant degradation in performance?

The BBN and LW systems have evolved independently, each defining its own input and output data formats without regard for the other. Naturally, these formats are not compatible as is. Moreover, there are no available standardized representations for rich transcription output of STT or MT systems. Can we define a joint data format that supports the goal of the effort and that that can be made compatible with both systems?

Approach

Product-quality operational integration of STT and MT involves two major aspects: tight software-level system integration and data format specification, i.e. first integrate the systems into a single data pipeline through which data flows seamlessly, and then specify the data format in which the systems exchange information.

Our integration approach consisted of the following steps:

1. Identify and resolve significant incompatibilities between the systems, including mismatched orthographic normalization rules, unsynchronized system lexicons, and assumptions on acceptable durations of sentences.
2. The AMC product already emitted rich transcription according to a formal XML-based data format specification. We chose to leverage it for the development of an STT/MT information exchange format used to communicate between AMC and the MT system.
3. While the primary mode of using the MT system is via a SOAP interface, it also already supported other ways of integration. In particular, the MT system can be linked as a library into C++ applications. AMC was already a pipeline of C++ components, so we chose to integrate the MT system via this library. We developed a wrapper component to embed the MT system into the AMC pipeline as just another C++ application.
4. Testing of the integrated system on real world data such as live satellite television feeds for extended periods of time to ensure that the system operates stably and that it meets the functional requirements.

6.2.2.2. BBN COTS Rich Transcription System

The BBN Audio Monitoring Component (AMC), which is used for operational integration of STT and MT, is a software solution for creating rich transcriptions of speech in audio recordings. The transcript contains metadata beyond the raw STT annotation to enable improved human readability of the speech content. All elements in the transcript are indexed to the time at which they occur in the source audio stream. The following figure illustrates the richness of the AMC transcript: it is segmented into sentences and proper names are highlighted with color.

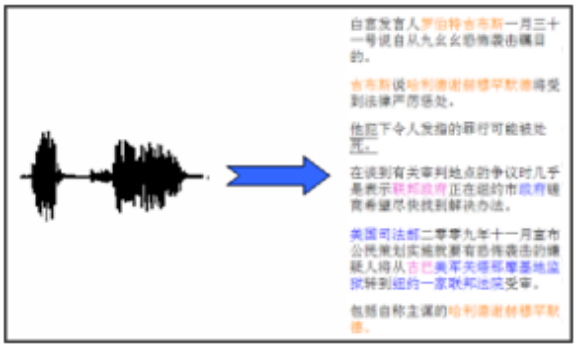

Figure 6.8: AMC – audio signal to rich transcription

System Description

The AMC integrates a suite of computationally demanding signal processing and pattern recognition techniques to produce a rich transcription of an audio input signal. The following figure shows an overview of the AMC system architecture.

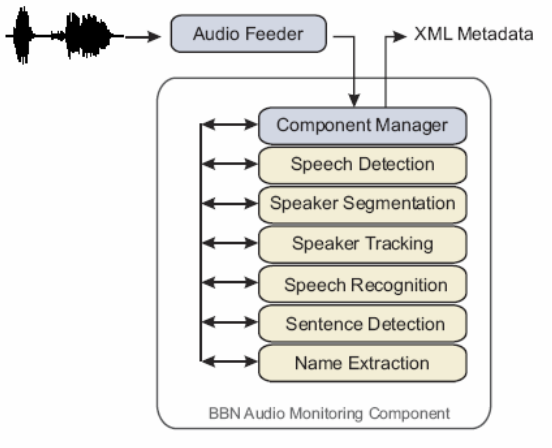

Figure 6.9: Integrated AMC component pipeline

The AMC components are arranged in a simple data pipeline. The final output of the AMC system is an XML-based transcript of the input audio signal. Included in the transcript are the following features:

1. Automatic word transcription
2. Named entities: people, locations, and organizations
3. Speaker turns and gender
4. Sentence boundaries
5. Non-speech audio events such as music or silence

At the time of this writing, the AMC product supports the following languages for rich transcription:

1. Modern Standard Arabic – with integrated Iraqi dialect
2. Mandarin Chinese
3. Persian/Farsi
4. Western Hemisphere Spanish
5. North American English

The system is tuned to perform at an average throughput time of ½-times real-time, so a one hour audio file is processed in approximately 30 minutes. The AMC provides an HTTP input and output interface. The system accepts audio data and returns the rich transcription and translation through this interface. The transcription and translation output is represented using an XML data format. BBN has a fully documented API to support integration into other applications.

6.2.2.3. LW COTS Machine Translation System

This section describes the commercially available Language Weaver Machine Translation system used for operational integration of STT and MT. The Language

Weaver COTS MT product is a statistical system that translates text from one language to another while preserving any phrase annotations.

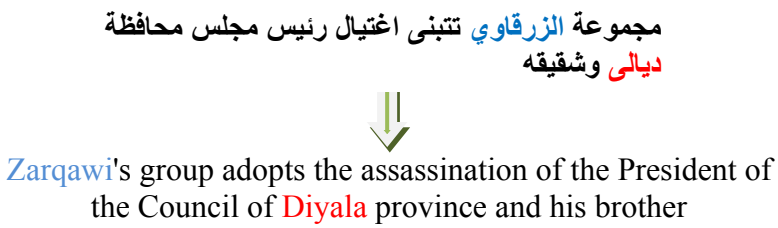

Figure 6.10: MT with linked name annotations.

The LW MT product was engineered to enable organizations to meet the volume, speed, and accuracy requirements for on-going translation demands. It is deployed in over 100 installations and has been integrated into over 20 complimentary applications. Key features are:

- Client-Server architecture for on-premise, large-scale and distributed deployment
- Industry-standard Web services interface
- Flexible input and output formats
- MS Office support via Open Document Format
- Customizable terminology translation
- Phrase annotations linked across translation

The LW Machine translation system supports two translation strategies: phrase-based and syntax-based translation.

The phrase-based system is based on translating phrases that are extracted automatically from word-aligned parallel text. The system uses a log-linear combination of translation models and an n-gram language model. The weights of these models are discriminatively tuned on a held-out development set. The system requires 2GB of RAM and delivers translation speeds ranging from 1000 wpm to 5000 wpm depending on the desired quality.

The syntax-based system is based on tree transducer rules that are learned automatically from a set of tuples of the type (English parse trees; foreign sentences; word alignment). The system employs a CKY-style decoder that uses a log-linear combination of translation models and an n-gram language model. The system requires 4GB of RAM and delivers translation speeds ranging from 100 wpm to 1000 wpm depending on the desired quality.

6.2.2.4. Operational Integration

This section describes the technical details of integrating BBN AMC and Language Weaver MT.

Desired Capabilities

When this effort started, the AMC served already as the rich transcription engine in the BBN BMS. Our operational integration work was guided by the goal of enhancing the BMS with automatic translation of the transcribed foreign language speech into English. The integrated system should run on a single commodity compute server, support continuous, real-time, low-latency processing of audio data and operate stably for months at a time with no need for human maintenance.

The STT/MT interface and the associated data exchange format should provide the following capabilities:

1. Structured representation: The data should be formatted as XML which is capable of encoding all the indexed features by STT, named-entity recognition, and MT. XML's Unicode support allows unambiguous encoding of source language and English translation text. In addition, the representation can be easily extended as needed.
2. Sentence-by-sentence translation: AMC transcribes speech content in the form of sentences. The MT system is required to translate the source language transcript one sentence at a time.
3. Crosslingual name linkage: AMC annotates names of type person, location and organization in each source language sentence. The MT system must preserve the identical set of names in the English translation. The data format must encode name annotations in both the source language and the English translation and it must preserve an unambiguous link between a source name and its translation.
4. Output constraints specification: In order to improve human readability of the source transcript and its translation, the data format must define constraints on the lexical representation of both the source transcript and the translation, including controlling the range of legal characters, regulating usage of capitalization and punctuation, and removing typographical anomalies.

Integration Issues

This section summarizes the issues that we addressed to achieve effective operational integration of AMC and the LW MT system. The overarching theme of these issues is that the two systems were developed independently, and so various engineering design choices involved in developing each independent system led to inherent incompatibilities between the systems.

First of all, the systems were mismatched as to the domains in which they were designed to operate. AMC was optimized to handle audio signals that contain broadcast news recordings from television and radio sources, while the intended use of LW MT is to translate written text such as Web pages and documents. While there are various potential problems associated with this mismatch, the most important one turned out to be the length of a translation unit. Whereas LW MT was designed to perform optimally on sentence units as is typical for written text, but AMC was not yet capable of producing sentence-like STT output with average lengths similar to text, and a maximum length not to exceed a practical hard upper limit. AMC routinely emitted token sequences of

lengths that would cause the MT throughput performance to degrade noticeably. To solve this issue, we developed and integrated a discriminative sentence boundary detection component into AMC that rendered the system capable of emitting well-formed sentence units. Details are provided in the article entitled *Discriminative Sentence Boundary Detection for Robust Operational Engines* in this chapter.

Differing system lexicons were another source of incompatibility between the systems. The LW Arabic-to-English MT lexicon was three times the size of the AMC Arabic STT lexicon; yet it covered only about 63% of the STT words. More than one third of all tokens emitted by the STT system were unknown to the MT system! We addressed this issue by first analyzing and categorizing the unknown words. While our analysis showed that a good portion of the unknown words consisted of proper names and of legal morphological variations, we discovered that a large proportion of these had one simple cause: the two systems used different sets of orthographic normalization rules during the lexicon design process. We carefully synchronized the rule sets and found that the MT lexicon coverage of the STT lexicon grew by an absolute amount of over 24% to a value of 87%.

A final source of incompatibility was found in the runtime characteristics of the two systems. Specifically, each system was tuned to make optimal use of the then best available commodity server. Co-locating the systems on a single server caused the joint system to run very slowly – it exhibited end-to-end latency peaks above 300 seconds. This was significantly larger than a maximum latency of 100 seconds that BBN requires of the AMC system. We resolved this problem by jointly tuning the runtime characteristics of the two systems.

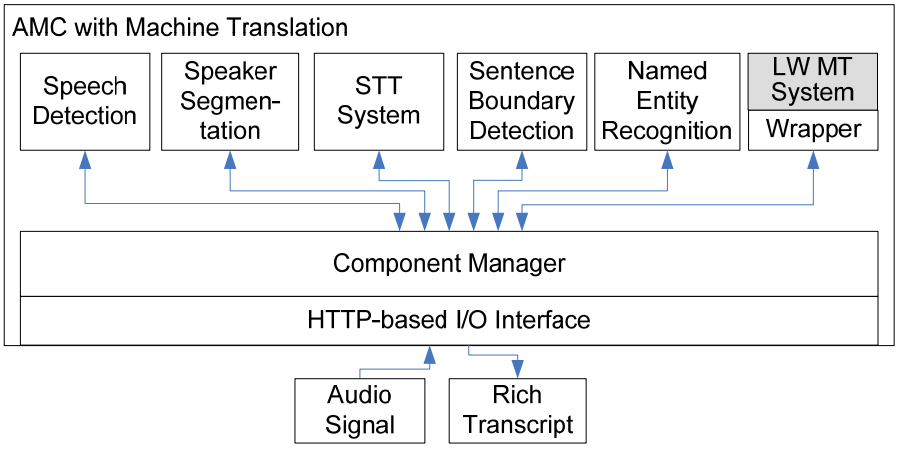

Figure 6.11: Component-level integration of AMC and MT

Software Integration

A significant part of our effort to integrate the AMC and MT systems was spent on designing how the two systems would interoperate at a software component level. Originally, LW recommended integrating with its MT system via its SOAP I/O interface. While we considered this path for operational integration we ultimately selected a different approach: the MT system is also available as a C++ link-library. We, thus,

integrated it into a C++ wrapper application to access a function-level interface to the MT engine. As shown in Figure 6.11 the wrapper application contains the MT system (gray box).

The C++-based approach proved desirable for a number of reasons. Firstly, the wrapper application was easy to develop using the AMC Component Manager (CM) client library and it yielded tight component integration into the AMC processing pipeline via function calls into the MT library. Secondly, this design placed the MT component under system state management of the CM. By design, the CM integrates the individual component states into an overall AMC system state. Now, the AMC system state also reflected the state of the MT system. In particular, in the event that MT encountered a fatal error, the AMC system state could immediately reflect this information. Further, several AMC components require large in-memory models to operate, including the MT system. To minimize system startup time and to protect the system hard drive the CM starts the pipeline components in a staggered fashion. Under CM control the MT system automatically participated in the staggered startup procedure.

Data Format Specification

We have jointly developed a formal specification of the STT/MT data format. As described in the previous sections, this format is derived from the pre-existing specification of the AMC rich transcription format.

Specifically, the following features of the AMC rich transcription output are preserved in the STT/MT format:

5. Lexeme tokens: The source language lexeme tokens of a sentence form the basis on which MT produces an English translation.
6. Sentence boundaries: The MT system implicitly utilizes the sentence boundary information since translation is done one sentence at a time.
7. Named entities: Named entity attributes including name boundaries, name types, and unique name id are provided as input to MT which, in turn, is required to preserve each name in the MT output.

Benefits of Integration

The joint BBN—Language Weaver effort under GALE succeeded in creating an operational system that tightly integrates the BBN AMC and LW MT systems. The resulting *AMC with MT* system produces a rich transcript of foreign language speech contained in broadcast news recordings with automatic translations into English of the speech transcript.

AMC with MT provides several benefits over loosely combining the individual systems that were available before the integration effort. Firstly, as a result of our extensive synchronization work the translation performance of *AMC with MT* on speech data is demonstrably better than that of the loosely connected individual systems. Secondly, the integrated system produces an improved rich transcription that preserves linkage of names across translation. And finally, by design *AMC with MT* runs on a single commodity server resulting in better scalability. Prior to integration, each of the two individual systems required its own server for proper operation.

We have tested *AMC with MT* extensively to ensure runtime stability within the specified operational constraints for extended periods of time. Since then *AMC with MT* has been available as part of the BBN suite of commercial product offerings. In particular, *AMC with MT* is a critical enabler of one of BBN's flagship products: the Broadcast Monitoring System (BMS). The BMS creates a continuous searchable archive of international television broadcasts in real-time.

Finally and most importantly for the GALE program, after many operational deployments of systems and solutions that integrate *AMC with MT,* it is clear that a system such as this has become a much appreciated and essential tool for intelligence analysis operations on open-source broadcast media sources. In effect, the system enables English-only speaking analysts to reach across the language barrier and to monitor and gist broadcast media in languages that they themselves do not understand.

6.2.2.5. Performance Evaluation: the BBN AMC, LW MT, and Integrated STT/MT

This section describes the STT performance of BBN AMC, the MT performance of LW MT and the performance of the integrated STT/MT system over the course of the GALE program.

Technology Integration

STT Technology Integration
1. Heteroscedastic Linear Discriminant Analysis
2. Perceptual Linear Predictive (PLP) features
3. LM interpolation and new backoff estimation
4. Discriminative STT training
5. Log-domain probability computation
6. Support for large corpus training – Arabic: 1000+ hours, Chinese: 500 hours
7. Discriminative named entity recognition
8. Perceptron-based sentence boundary detection

Table 6.1: Technology integration into BBN AMC

MT Technology Integration
1. 5-gram language models
2. Language models trained on very large corpora
3. Better rule-based components
4. More accurate word alignment models
5. Better feature weights tuning
6. Improved translation models
7. Syntax-based MT models

Table 6.2: Technology integration into LW MT

One goal of the GALE program is the transfer of proven research advancements into systems that can be deployed for operational use. Largely independent of each other,

BBN and LW continue to integrate new research improvements into their operational systems. Table 6.1 summarizes the specific research improvements that have been integrated into BBN AMC during GALE.

Table 6.2 summarizes the specific research improvements that have been integrated into the LW MT system during GALE.

AMC System Performance

Figure 6.12 tracks the performance of AMC Arabic during the first three years of GALE. Performance is measured using the Word Error Rate (WER) metric on the GALE-2006 speech development sets – split into the broadcast conversation (BC) and broadcast news (BN) portions of the set.

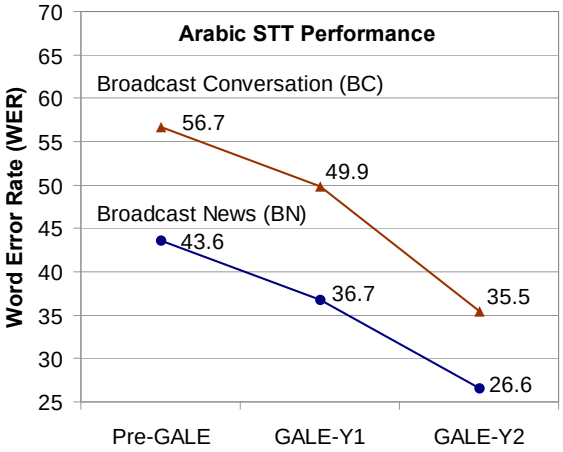

Figure 6.12: Arabic STT performance during GALE

Figure 6.13 tracks the performance of AMC Chinese over the course of the first three years of the GALE program. Performance is measured on the GALE-2006 speech development sets – broken up into the broadcast conversation and broadcast news portions of the set. The GALE-improved Chinese system achieves a dramatic 50% relative reduction in WER on the Chinese conversational data compared to the pre-GALE AMC system. We have achieved a remarkably consistent and large WER reduction for both Arabic and Chinese.

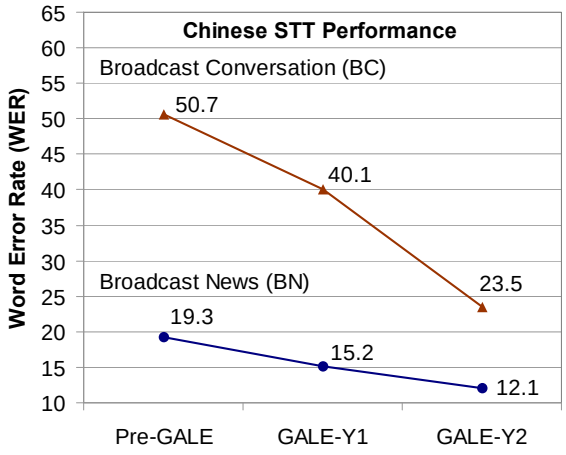

Figure 6.13: Chinese STT performance during GALE

MT System Performance

Figure 6.14 tracks the performance of the phrase-based Arabic-to-English MT system during the first three years of GALE. Performance is measured using the Bilingual Evaluation Understudy (BLEU) metric on a 1,000-sentence newswire test set with 4 translation references.

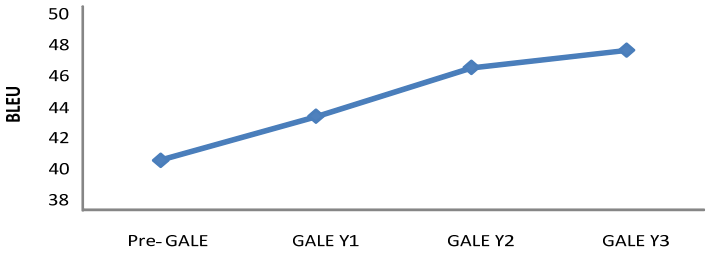

Figure 6.14: Arabic-to-English phrase-based translation performance during GALE

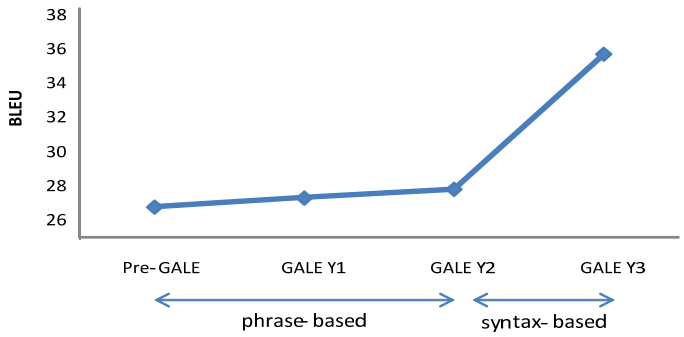

Figure 6.15: Chinese-to-English translation performance during GALE

Figure 6.15 tracks the performance of the Chinese-to-English MT system during the first three years of GALE. Note that LW introduced Syntax-based MT in the product. Performance is measured using the BLEU metric on a 1,000-sentence newswire test set with 4 translation references.

AMC-with-MT System Performance

In addition to tracking performance improvements in the individual engines we also track the cumulative effect – translation performance of LW MT on AMC STT output. We score the *MT on speech* performance using the Translation Error Rate (TER) metric with a single reference and with mixed case and punctuation. Performance is measured on the GALE-2006 speech development sets. All results are produced by COTS engines in standard configuration. In particular, all systems run at real-time with an average end-to-end latency of less than one minute. Figure 6.16 tracks the translation performance of the integrated system on Arabic speech data during GALE.

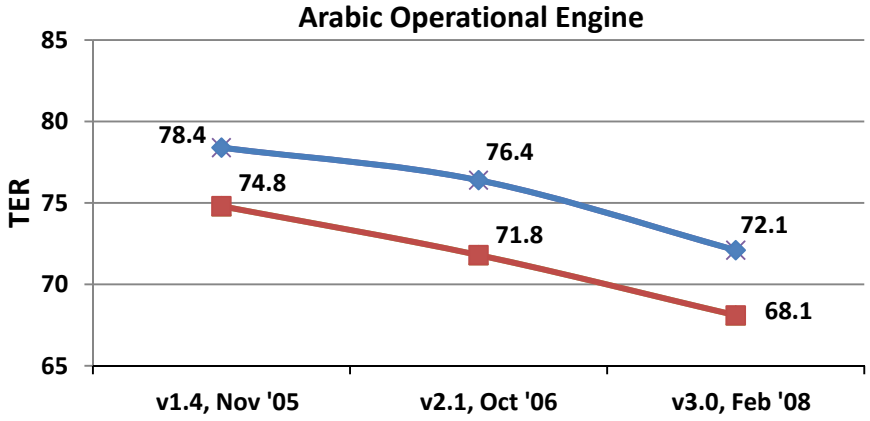

Figure 6.16: MT performance on Arabic speech during GALE

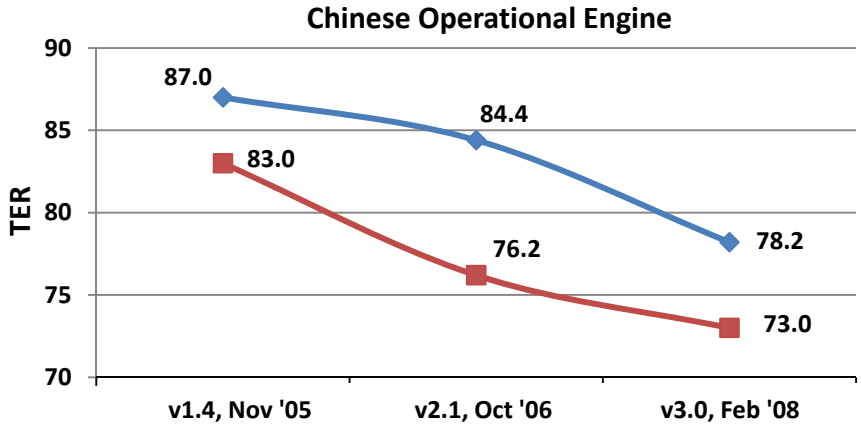

Figure 6.17: MT performance on Arabic speech during GALE

Figure 6.17 tracks the translation performance of the integrated system on Chinese speech data during GALE.

As the figures demonstrate, we have achieved consistent performance gains across languages and genres. The translation performance on speech data of the GALE-enabled integrated system has improved significantly over the pre-GALE performance baseline.

6.2.2.6. The BBN Broadcast Monitoring System

BBN's commercial Broadcast Monitoring System (BMS) was developed in partnership with the Technical Support Working Group (TSWG), a research division under the Assistant Secretary of Defense for Special Operations/Low Intensity Conflict. TSWG focuses on rapid prototyping of new solutions with the ultimate goal of transitioning robust products to end users. In late 2001 when TSWG was keen on providing soldiers with the capability to access non-English media, specifically international television, they surveyed the technological landscape and selected BBN's speech and language processing technologies (Shepard 2002). Since its inception, the BMS has integrated the immense advances accomplished under DARPA-funded research efforts and in particular the current GALE effort.

The BMS is a turnkey COTS product (hardware and software) that creates a continuous searchable archive of international television broadcasts. Real-time broadcast streams from satellite or cable are captured by the system and automatically maintained in a one-year archive. As video is ingested by the BMS, the audio stream is automatically transcribed by the AMC and translated into English by the Language Weaver MT system. The browser-based interface on the BMS lets users access the BMS from any PC or laptop via a common high-speed IP network. The only required client software is Microsoft® Internet Explorer and Windows Media® Player.

The following figure shows the home screen of the BMS – the Channel Overview. It contains a search box, a list of persistent queries (the watchlist), and a thumbnail overview of the available channels.

Figure 6.18: BBN BMS channel overview

The BMS enables its users to quickly find specific spoken content in the video archive through keyword queries — in English or the source language — and to jump directly to

the streaming video from the search results. Searches can be scoped by channel, language, and time.

Figure 6.19: BMS channel view

Figure 6.19 shows the BMS Channel View, which displays the rich transcript and automatic translation side-by-side with the original video signal. The video can be accessed and played back from any point in the archive by clicking on a word in the transcript. As video plays back, the spoken words and translation are highlighted. The transcript and translation are enriched by highlighting the names of people, places, and organizations in color, and by separating the speakers in the audio file. This integrated view helps users to converge on the meaning of the broadcast.

Users can save collections of searches in a watchlist that continuously monitors incoming video for matches to search criteria. Video segments are automatically added to the watchlist as they are broadcast and the system alerts the user as new matches occur. Watchlists can be filtered by date, user, and topic, so users view only watchlist items of interest to them.

Users can extract video segments or still photos for collaboration, presentation, and reports. Rich media — integrated video, transcript, and translation packages — can be exported to HTML files for playback in a browser or to BBN Translator's Aide, an application that improves the efficiency of human translation from video sources.

The BMS is available as a turnkey solution, requiring no hardware setup or software installation by the customer. The system is designed to run as an appliance without needing any onsite system administration. The BMS turnkey solution is designed to comply with the Defense Information Systems Agency (DISA) Gold standard to support deployment security requirements. The BMS has been deployed operationally since August 2004 for open source intelligence collection and information operations at many locations around the world.

The groundbreaking capability offered by the BMS enables users with no foreign language skills to get the gist of international television and radio broadcasts and to triage enormous volumes of media, allowing skilled linguists to focus on translation tasks. Realizing this capability required both fundamental research in human language

technology, provided by DARPA, along with applied technology transition and user focus, provided by TSWG. For more information, see Section 6.2.1. More information on the BMS is available on the BBN website.[1]

6.2.2.7. Conclusion

With the commercial availability of AMC with MT, for example as part of the BBN Broadcast Monitoring System product, we have achieved the goal of this effort to demonstrate a practical integrated system that creates a rich transcript of foreign language audio data with automatic translation into English. The MT performance on speech is significantly improved over the pre-GALE system which consisted of a loose connection of the individual systems. AMC with MT also provides additional metadata: annotated and linked name phrases in both the source language and the corresponding English transcript.

Over the course of the GALE program, we have continually transferred practical research advancements into both AMC and the LW MT system and are, thus, able to offer the resulting performance improvements in our deployable systems. In fact, we managed to demonstrate that cutting edge research advancements achieved during GALE can be transitioned into deployable systems in as little as 12 to 18 months.

We continue to transition our systems into operational deployments and we have found that these systems are rapidly becoming a much appreciated and essential tool to assist intelligence analysis operations on open-source broadcast media sources. In effect, the systems enable English-only speaking analysts to reach across the language barrier and to monitor and gist broadcast media in languages that they themselves do not understand.

6.2.3. Discriminative Sentence Boundary Detection for Robust Operational Engines

Authors: Amit Srivastava, Daniel Kiecza

6.2.3.1. Introduction

We present a discriminative Sentence Boundary Detection (SBD) component for speech that not only improves the accuracy of sentence segmentation; it also improves the overall robustness of the system, enabling real-world deployments of GALE operational engines. In this section, we describe the combination of two SBD subsystems in the BBN operational engine: a discriminative Neural Network (NN) model built over prosodic features estimated from the underlying speech signal, and a Perceptron model that uses linguistic and prosodic information jointly. We also describe the analytical model used to estimate a length-based penalty during decoding to constrain the output sentence lengths. The discriminative SBD component outperforms the pre-GALE SBD component in both, the sentence segmentation performance on Arabic and Chinese speech, as well as the end-to-end system latency of the operational engine when processing live audio.

[1] http://www.bbn.com/products_and_services/bbn_broadcast_monitoring_system

As speech recognition, machine translation, and natural language processing technologies mature, there is an urgent need to transition the proven algorithms and techniques into the field to provide tools to analysts for effective distillation of information from large volumes of foreign speech. In addition to integrating and demonstrating the latest research algorithms and models, the GALE operational engines need to demonstrate a practical capability that can do useful work in the real world. In particular, we defined the core capabilities of the integrated operational engine as follows. The system would have to:

- Run integrated speech-to-text (STT) and machine translation (MT) on a single commodity compute server to produce sentence-based translation of the input foreign speech,
- Support continuous, real-time,[2] low-latency processing of audio data and,
- Operate stably for months at a time with no need for human maintenance

The BBN Audio Monitoring Component (AMC) is an example of a practical, integrated operational engine that produces automatically and in real-time a rich transcript of foreign language speech. Sentence boundaries are important structural metadata that enhance speech recognition transcripts in terms of human readability and natural-language processing. Parsers, information extraction systems, and machine translation systems rely on sentential inputs from speech-to-text systems for optimal performance. The Sentence Boundary Detection (SBD) component in the AMC is critical to providing high-level linguistic structure in the output rich transcription, enhancing the accuracy and readability of the English translations, and enabling operational viability of the integrated operational engine on real world data in the field.

Most SBD systems are statistical engines that learn linguistic and acoustic cues from human-annotated data to hypothesize sentence boundaries in speech transcripts. Limitations in the SBD algorithms and models as well as unspecified constraints can lead to very long sentence hypothesis that has many adverse effects on the operational engines:

- Degraded readability and comprehension – long sentences are difficult to parse and understand especially in the presence of speech-to-text and machine translation errors
- Inefficient MT – long sentences increase the potential search space for statistical machine translation engines leading to reduced speed and accuracy of translations
- Increased end-to-end system latency – in pipelined, integrated operational engines that produce sentence-based translations of the input foreign speech, such as the BBN AMC, the length of the sentence

[2] We use the term real-time to denote a throughput data rate that is the same or faster than the data rate of a live audio source.

directly influences the system latency and hence the practical usability of the system in the field.

Previous research on SBD has focused on combining lexical and prosodic cues in separate models or in a single model-based framework to hypothesize sentence boundaries between words in the STT output (Stolcke and Shriberg 1996 and Liu *et al.* 2004). Model combination has been shown to significantly improve segmentation performance over individual model outputs as in our pre-GALE approach (Stolcke *et al.* 1998). However, most of these approaches lack sentence-length constraints unlike the SBD approach described in this section, which has been developed specifically for integration in the BBN AMC to enable sentential segmentation in high-throughput, low-latency speech transcription for readability and translation.

The approach described in by Matusov *et al*. (2006) focused on improving MT performance from speech and included a sentence-length constraint similar to our approach. However the decoding algorithm in their approach uses an HMM-style search over the complete document whereas our approach uses a tagging framework to reduce end-to-end system latency.

6.2.3.2. Previous Approach and Limitations

Prior to GALE, the operational engine included a SBD system (Srivastava and Kubala 2003) that consisted of a combination of two subsystems: an Acoustic subsystem that consisted of a discriminative Neural Network model over prosodic features, and a linguistic subsystem that consisted of a generative N-gram language model over word-based trigram features. The scores from the two subsystems were combined in a Maximum-Likelihood framework in a Viterbi decoder to hypothesize sentence boundaries between words from each automatically segmented speaker-turn in speech.

Prosodic Feature Extraction

Figure 6.20 shows the conceptual diagram of our feature extraction method. We define putative boundaries (PB) as the intervals between words in the speech transcripts. The SBD system can hypothesize sentence boundaries in each of these PB. We extract a variety of prosodic features from the audio at each PB.

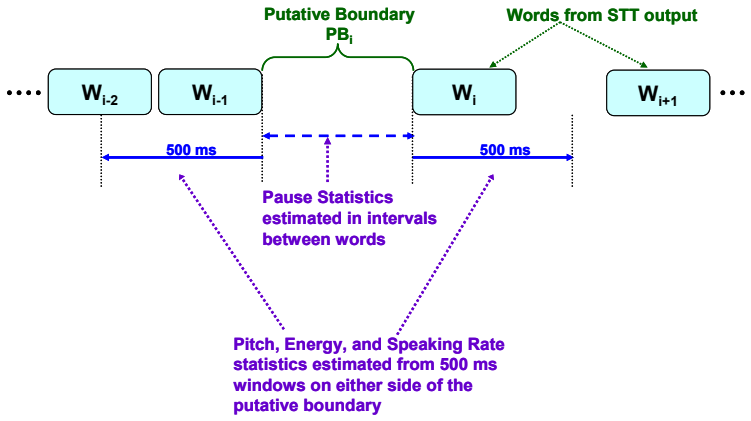

Figure 6.20: Prosodic feature extraction method

Pause statistics are estimated from the intervals between the words, which may be absent altogether when there is no pause between the hypothesized words. Pitch, Energy, and Speaking Rate statistics are estimated from 500 ms windows on either side of PB_i extending from its edges. The estimated prosodic features are functions of these statistics. We used these four fundamental prosodic attributes to generate a total of 47 features.

Table 6.3 shows the distribution of the prosodic features estimated, as well as, the types of statistics used to estimate these features. There were 9 Pause related features, 2 Speaking Rate features, 6 Energy based features, and 30 Pitch related features estimated at the putative boundaries. The prosodic features estimated during this effort were a diverse set of features with some being continuous, others being discrete, and yet others were Boolean features.

Feature Type	Feature Description	#Features
Pause	•Pause Duration •Pause Attribute (Filler, Breath, etc.) •Time since last Pause •Normalized Pause Duration	9
Speaking Rate	•Absolute Value •Difference across Putative Boundary	2
Energy	•Absolute Values •Difference across Putative Boundary •First Difference of Energy	6
Pitch	•Discontinuous Chains in Voiced Regions •Interpolated Continuous Pitch •First-Order Pitch Differences •Tone Estimates (Legendre Polynomials)	30

Table 6.3: Distribution of prosodic feature attributes

Acoustic Subsystem

The NN model consists of a 2-layer feed-forward Neural Network that was trained on the 47 prosodic features described previously. Standard back-propagation training was applied with the Minimum Squared-Error (MSE) criterion to the NN with configuration: 47 input nodes, 4000 hidden nodes, and one output node (Srivastava and Kubala 2003). Sentence boundary classes are hypothesized by comparing the NN output score to a threshold as shown in the equation below. The output of the NN can also be viewed as the MSE estimate of the posterior probability of the class, sentence boundary. We will use this property of the Neural Network during model combination.

$$\boldsymbol{c_i} = \begin{cases} \boldsymbol{sentence-boundary} & \textbf{if } \boldsymbol{\varphi(\tilde{F}_i) \geq \tau} \\ \boldsymbol{non-boundary} & \textbf{if } \boldsymbol{\varphi(\tilde{F}_i) < \tau} \end{cases} \tag{6.1}$$

Here, $\varphi(\tilde{\mathrm{F}}_i)$ denotes the Neural Network output when presented with the input 47-dimensional *prosodic* feature vector $\tilde{\mathrm{F}}_i$, and τ is an empirical threshold.

Linguistic Subsystem

For the Linguistic subsystem, we inserted the *sentence-boundary class* as the token, $< s >$, in the word sequence for the transcripts in the training data and estimated a trigram language model (LM) using the BBN Byblos™ Speech Recognition LM toolkit. The resulting LM is very similar to the hidden segment model proposed by Stolcke and Shriberg (1996). Viterbi decoding is used to find the most likely class sequence given a sequence of words and the trained language model. The most likely class sequence is estimated as:

$$\hat{\boldsymbol{c}}_{[1:n]} = \mathbf{argmax}_{\boldsymbol{c}_{[1:n]}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \mathbf{log}\left(\mathbf{p}^{\boldsymbol{L}}\left({}^{\boldsymbol{w_i}}/_{\boldsymbol{w_{i-1}, w_{i-2}, c_{i-1}, c_{i-2}}}\right)\right) \tag{6.2}$$

where p^L denotes the trigram Language Model probability for the context that includes the current word w_i and two preceding tokens from the set $\{w_{i-1},\ w_{i-2},\ c_{i-1},\ c_{i-2}\}$.

ML Combination Framework

In the *Acoustic* subsystem, the NN produces scores, $\varphi(\mathrm{c}_i/\tilde{\mathrm{F}}_i)$ that are MSE estimates of the posterior probability of the boundary classes. These scores are transformed into likelihoods by scaling them with the prior probability, $\mathrm{P}(\mathrm{c}_i)$, of the boundary classes, estimated from the same training data.

The *Combined* system computes an exponentially-weighted product of the likelihoods from the two subsystems in a Maximum-Likelihood framework. The optimal sequence of boundary classes $\hat{c}_1^n$, given the sequence of words w_1^n, and the sequence of prosodic feature vectors $\tilde{\mathrm{F}}_1^n$ in a speaker turn with *n* words, is determined using the Viterbi algorithm as follows:

$$\hat{\boldsymbol{c}}_{[1:n]} = \mathbf{argmax}_{\boldsymbol{c}_{[1:n]}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{\lambda} \mathbf{log}\left(\frac{\boldsymbol{\varphi(c_i/\tilde{F}_i)}}{\mathbf{P}(\boldsymbol{c_i})}\right) + \boldsymbol{\delta}\ \mathbf{log}\left(\mathbf{p}^{\boldsymbol{L}}\left({}^{\boldsymbol{w_{i+1}}}/_{\boldsymbol{w_i, w_{i-1}, c_i, c_{i-1}}}\right)\right) \tag{6.3}$$

Powell's minimization technique was used to search for the optimal weights, λ and δ that correspond to the minimum SBD error over the development test set.

Impact on Sentence Length

The Viterbi decoding framework needs well defined end-points for optimal search. Automatic speaker turns hypothesized by the speaker segmentation component in the AMC (Liu and Kubala 1999) provide these end-points for sentential decoding. However, speaker turns in speech can be very long. In the above approach it is very difficult to integrate sentence length constraints. This effectively implies that a sentence could span the entire speaker turn, which can lead to adverse effects on the operational engine.

In the next section we will describe the new formulation for SBD that will allow more diverse and discriminative features to be integrated into a statistical modeling framework and also allow sentence-length constraints to directly influence the placement of the boundaries.

6.2.3.3. Discriminative Tagging Framework

We formulate the Sentence Boundary Detection task as a tagging problem where the goal is to tag each word in the hypothesized transcription with one of the two classes: *sentence-boundary* (SB) class, and the *non-boundary* (NB) class. The hypothesized transcription is produced by the automatic speech recognition engine in the AMC. The sentence boundary tags are hypothesized independently at each word boundary by examining a fixed context around the word in each hypothesized speaker turn.

Approach

The SBD component in the AMC consists of an incremental decoder that accumulates speech utterances from the output of the Speech Recognition system in an input buffer. Each utterance consists of acoustic features, the top-1 word and phonetic transcripts. Acoustic and Indicator-function features are estimated and used by the Acoustic and Perceptron subsystems to produce scores for the two sentence classes: sentence-boundary and non-boundary. These two scores are combined with a sentence-length-based penalty score based on the comparison of the current sentence length with the average sentence length for each language. A sentence boundary is hypothesized at the putative boundary if the score for the sentence-boundary class is greater than the score for the non-boundary class. At this point, all but the last sentence are output from the AMC and the last sentence is retained in the input buffer for the next pass of decoding.

Feature Extraction

Features for sentence boundary detection are functional representations of eight fundamental attributes of the transcription and the underlying speech signal. There are four linguistic attributes: the word identity, the POS tag associated with the word, the word prefixes, and the word suffixes. There are also four prosodic attributes: pause, pitch, energy, and speaking rate. We used these attributes to create two types of features: Prosodic and Indicator-function. The prosodic features, described in detail in Section 2.1, were used in a NN model as in the previous approach.

Indicator-function Features

Attribute Type	Symbol	Attribute Values
Linguistic	W	• word identity • word suffices up to length 4 • word prefixes up to length 4
Linguistic	P	associated POS tag
Prosodic	F	• Pause duration quantized to 0.5 secs

Table 6.4: Attributes used to define history at a word

Similar to the formulation of Huang and Zweig (2002), we denote H as the set of all possible contexts, or *histories*, of the word that is to be tagged, and C as the set of all possible classes or tags. In our case, the size of the set C is 2. The transformation function Φ, where Φ: H x C → R^M maps a history-class pair to an M-dimensional feature vector. Each component of the feature vector is represented as $\Phi_s(h_i, c_i)$ and denotes a particular indicator function over the history-class pair (h_i, c_i) at the word w_i.

We defined the history over a context of 5 words around the word that is to be tagged. The history includes all four linguistic attributes and one prosodic attribute. These attributes are defined in Table 6.4. The history h_i at a word w_i is then defined as:

$$\begin{aligned} h_i \; = \; < & \; P_{i-2}, F_{i-2}, W_{i-2}, \\ & P_{i-1}, F_{i-1}, W_{i-1}, \\ & P_i, F_i, W_i, \\ & P_{i+1}, F_{i+1}, W_{i+1}, \\ & \mathbf{P_{i+2}, F_{i+2}, W_{i+2}} \; > \end{aligned} \tag{6.4}$$

These indicator-function features encode co-occurrences of histories and classes. Effective features can denote that certain histories are strong predictors of the presence or absence of a sentence boundary class at a particular word in the transcription.

Perceptron Subsystem

The Perceptron model is a linear-discriminative model over the *indicator-function* features trained with the Averaged Perceptron algorithm (Collins 2002). The score of each boundary class from the Perceptron model at each word, given a fixed context around it, is given by:

$$\mathbf{S(c_i) = \sum_{s=1}^{M} (\alpha_S \Phi_S (h_i, c_i))} \tag{6.5}$$

Here, α_s denotes the weight attributed to the indicator function Φ_s. Thus, given parameter values, α_s and a history h_i in the context of word w_i, the highest scoring sentence class can be determined using the score from Equation 6.5 Training the Perceptron model consists of hypothesizing the sentence class at each putative boundary using the current parameter values α_s, followed by simple additive updates to α_s based on the true sentence classes, as explained in detail by Collins (2002).

Combined Decoder with Sentence Length Constraint

The scores from the Acoustic subsystem and the Perceptron subsystem are combined linearly with a sentence length penalty that discourages the combined decoder from hypothesizing sentence boundaries resulting in output sentence lengths that are shorter or longer than a target sentence length. An analytical sentence duration model is used to calculate the penalty score as a function of the difference in hypothesized and target sentence lengths as shown in the equation below.

$$\mathbf{P^D}(SB/d) = \frac{1}{1+\mathbf{e}^{(\bar{\mathbf{d}}-\mathbf{d})}}$$
$$\mathbf{P^D}(NB/d) = \mathbf{1.0} - \mathbf{P^D}(SB/d) \tag{6.6}$$

where, $\bar{\mathrm{d}}$ is the target sentence length estimated simply as the average sentence length from SBD training data for each language.

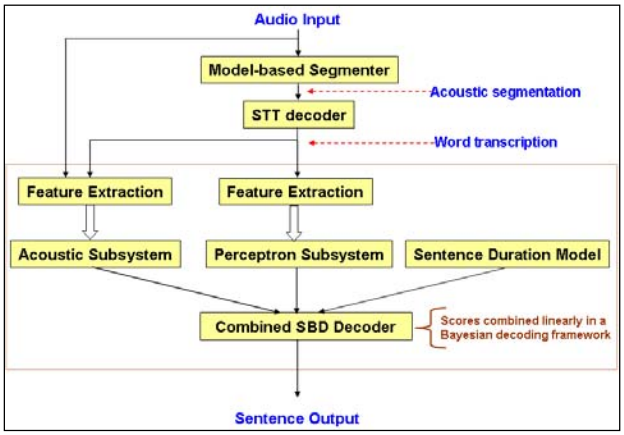

Figure 6.21: Sentence Boundary Detection in the BBN AMC

The Combined decoder, in Figure 6.21, computes the score for each sentence class at each putative boundary i using the equation:

$$\tilde{\mathbf{S}}(\mathbf{c}_i) = \boldsymbol{\mu}\left(\sum_{\mathbf{s}=\mathbf{1}}^{\mathbf{M}}(\boldsymbol{\alpha}_{\mathbf{S}}\boldsymbol{\Phi}_{\mathbf{S}}(\mathbf{h}_i, \mathbf{c}_i))\right) + \boldsymbol{\theta}\mathbf{log}\left(\frac{\varphi(\mathbf{c}_i/\tilde{\mathbf{F}}_i)}{\mathbf{P}(\mathbf{c}_i)}\right) + \boldsymbol{\gamma}\,\mathbf{log}\left(\mathbf{p^D}\left(\mathbf{c}_i/\mathbf{d}\right)\right) \tag{6.7}$$

where:

- $\varphi(\mathrm{c}_i/\tilde{\mathrm{F}}_i)$ is the Neural Network output for sentence class c_i given the feature vector $\tilde{\mathrm{F}}_i$
- 'd' represents the current sentence length given current position *i* and the last hypothesized sentence boundary in the turn
- $\mathrm{P}(\mathrm{c}_i)$ is the prior probability of sentence class c_i

- μ, θ, and γ are empirical weights that are estimated by tuning on the development test data

Sentence boundaries are then hypothesized using the Bayesian framework:

$$\mathbf{c}_i = \begin{cases} \boldsymbol{SB}, & \textbf{if}(\tilde{\mathbf{S}}_i(\boldsymbol{SB}) \geq \tilde{\mathbf{S}}_i(\boldsymbol{NB}) \\ \boldsymbol{NB}, & \textbf{otherwise} \end{cases} \tag{6.8}$$

6.2.3.4. Performance Evaluation

Arabic Experimental Data

We used Arabic Broadcast News (BN) acoustic data consisting of approximately 103 hours of spoken modern-standard Arabic, transcribed without diacritic markings, for our experiments (Billa *et al.* 2002). This data is drawn from 10 different sources, which include Egyptian, Syrian, and Saudi broadcast radio and TV, and the Al-Jazeera TV network. The transcriptions in this corpus are annotated with sentence boundaries. Approximately 96 hours of audio and transcriptions were drawn from this corpus to comprise the acoustic training set. We also used text articles drawn from Al-Jazeera online website as additional text training data.

Set	Duration (hours)	# Words	# Sentences	%WER
Acoustic Training	96	600K	20K	-
Text Training	-	370K	21K	-
Dev	12	52K	2018	17.0
Eval	11	47K	1911	19.4

Table 6.5: Arabic Experimental Data Summary

The remaining seven hours of audio from the Arabic BN corpus were divided into two sets, the SBD development (Dev) set and the SBD evaluation (Eval) set. Approximately sixteen hours of Arabic test data, both development and evaluation sets, from the DARPA EARS Rich Transcription Evaluations in 2003 and 2004 were manually transcribed with sentence boundaries at BBN and added to the SBD Dev and Eval sets respectively. All the data used in our experiments is summarized in Table 6.5. The AMC Speech-to-Text performance, in terms of Word Error Rate (WER), on the Arabic speech in the two test data sets is also shown in this table.

Chinese Experimental Data

Training data for Chinese SBD consisted of approximately 50 hours of speech transcriptions from the FBIS and LDC STT training corpus that contains manual annotation of sentence boundaries and punctuation.

The development and evaluation test sets were created with data drawn from five separate test data collections.

1. LDC Mandarin Development set for BN Hub4 Speech Recognition Evaluation 1997.

2. LDC Mandarin Evaluation set for BN Hub4 Speech Recognition Evaluation 1997.
3. LDC Mandarin Evaluation set for BN Hub4 Speech Recognition Evaluation 1998.
4. Mandarin EARS RT03+RT04 Development test set.
5. Mandarin EARS RT03+RT04 Evaluation test set.

Sets 1 to 3 contain manually annotated sentence boundaries from LDC while some files in sets 4 and 5 were manually annotated with sentence boundaries at BBN. Files without manual annotation in sets 4 and 5 were excluded from the development and evaluation test sets. Table 6.6 shows the composition of the data sets used for Chinese SBD experiments. The AMC STT performance, in terms of Character Error Rate (CER), is also shown for the test data sets.

Set	Duration (hours)	# Words	# Sentences	%CER
Acoustic Training	50	420K	24K	-
Dev	4	41139	1362	12.5
Eval	5	50863	1673	14.5

Table 6.6: Chinese Experimental Data Summary

Evaluation Metric

The reference and the hypothesized Rich Transcription are represented as an ordered sequence of generalized tokens that includes the word identity and the sentence boundary information. The two sequences of generalized tokens are aligned with respect to a generalized edit-distance function that preserves the Speech-to-Text (STT) Word Error Rate (WER) while optimizing the SBD error. Sentence Boundary Detection performance is computed from this alignment by examining the sentence boundary attributes of each pair of aligned tokens. We used the Sentence Boundary Detection Slot Error Rate (SER) in the equation below as our primary evaluation metric (Makhoul *et al.* 1999). For all our experiments, we will present the system SER and the *Correct Rate*, which is 1.0 minus the *Deletion Rate*.

$$\mathbf{SER} = \frac{\#\mathbf{Insertion}+\#\mathbf{Deletions}}{\#\mathbf{References}} \tag{6.9}$$

Results

System	Arabic Dev		Arabic Eval	
	% Corr.	% SER	%Corr.	% SER
Pre-GALE	75.6	53.6	75.4	49.3
Discriminative SBD without length constraints	79.6	50.8	79.8	46.9
Discriminative SBD with length constraints	79.9	51.4	80.3	46.0

Table 6.7: Arabic SBD Results

System	Chinese Dev		Chinese Eval	
	% Corr.	% SER	%Corr.	% SER
Pre-GALE	70.3	48.4	66.5	59.7
Discriminative SBD without length constraints	76.0	42.5	71.9	56.7
Discriminative SBD with length constraints	79.4	44.5	76.6	54.6

Table 6.8: Chinese SBD Results

For each of the two languages, Arabic and Chinese, we evaluated three SBD systems: the Pre-GALE SBD system, the new discriminative SB tagging system without length constraints, and finally the new discriminative SB tagging system with length constraints. The empirical weights used in the combination framework were estimated using the Powell's method to minimize the SBD SER on the development test set. The optimal weights were then used to hypothesize sentence boundaries in the speech transcripts of the evaluation test set. The SBD performance on the Arabic and Chinese Dev and Eval sets is detailed in Tables 6.7 and 6.8.

These results show that the new discriminative SBD system with local Bayesian decoding outperforms the Pre-GALE SBD system. There is a significant improvement in the correct percentage as well as in the SER performance for the new approach even with the restricted context imposed by the local Bayesian decoding framework. We expected the performance to degrade with the addition of the length constraints. However, as seen from the result tables, there is an insignificant change in performance for Arabic SBD even when the length-based penalty is added to the decoding equation although, surprisingly, Chinese SBD shows a modest improvement. We suspect that this improvement is related to the property of the Chinese language whereby Chinese sentences are much more well-defined structurally and are also much shorter, in average, than Arabic sentences. Imposing a sentence-penalty then that is influenced by the average sentence length turns out to be much more beneficial for Chinese than for Arabic.

Engine Runtime Performance

We compared the end-to-end system latency of the AMC engine with respect to the pre-GALE SBD and the new discriminative SBD systems. In both cases, the AMC was run continuously for five days on live audio feed captured from the Al-Jazeera Arabic television channel via satellite. The end-to-end system latency is defined as the lag (in seconds) between the time at which a sentence is output from the AMC, and the time at which the audio corresponding to the beginning of the sentence was received at the input of the AMC.

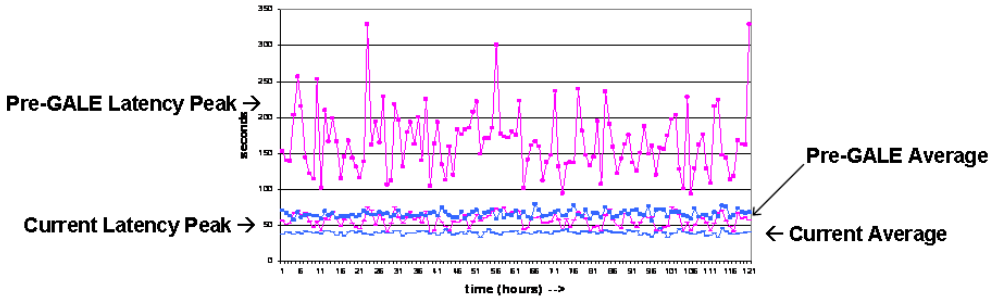

Figure 6.22: Improvement in end-to-end system latency

Figure 6.22 shows the improvements in both peak and average latency (over one hour sessions) due to the new SBD system with explicit length constraints. Peak latency dropped from over 300 seconds to below 80 seconds. The plot also shows the reduction in latency variation in the discriminative SBD-based AMC engine, which leads to robust operation in the field.

We believe that the improvement in system latency is due to the following reasons:

- An upper limit on the length of the hypothesized sentences,
- An explicit sentence-length constraint in the SBD algorithm results in well-behaved sentence length on average, and
- Incremental Bayesian decoding that does not require completed speaker turns before hypothesizing sentence boundaries.

6.2.3.5. Conclusion

We have demonstrated that the constrained, discriminative SBD system outperforms the previous system, on Arabic and Chinese speech, in terms of sentence segmentation performance. Adding sentence-length constraints has minimal effect on the sentence segmentation performance. However, the peak system latency during live audio processing is significantly reduced.

Perceptron models have a simple formulation, are easy to train, and perform surprisingly well. The Perceptron model we used for SBD was impoverished since we did not use all the available prosodic features. It would be interesting to learn if adding more prosodic features helps the Perceptron model. Model combination has always been beneficial for SBD. We need to examine more discriminative models that might combine effectively with the Perceptron model. Finally, we used a very simple logistic regression model to compute the sentence-length-based penalty score. Further research could examine a machine learning approach to compute the parameters of a similar model or propose new models for imposing length-based constraints in SBD systems.

The Sentence Boundary Detection research described in this section was implemented in the AMC engine, which is offered as a turnkey product by BBN. The length of sentences in the new system and the quality of the segmentation has far-reaching implications for solutions that integrate operational engines in GALE or beyond. These improvements have been partially responsible for the operational viability and

robustness of the BBN operational engine, the AMC. Sentences hypothesized in Arabic and Chinese speech by this new discriminative SBD system were used as the common segmentation across the components for the GALE Interoperability Demo (IOD) (Pitrelli *et al.* 2008a). The AMC is also the core Rich Transcription engine for the BBN Broadcast Monitoring System, which is a turnkey product developed in partnership with the Technial Support Working Group (TSWG), which allows intelligence analysts to capture, monitor, and triage information from foreign television stations in real-time with low latency. The BBN BMS is deployed at multiple U.S. Government installations around the world and has been instrumental in transitioning GALE technology and research improvements into the field to aid in counter-terrorism, intelligence analysis, and language learning.

6.2.4. System Combination of Distributed Speech-to-Text, Translation, and Information Extraction Engines: The GALE Interoperability Demo (IOD)

Authors: John F. Pitrelli, Salim Roukos, Edward A. Epstein, David Ferrucci, Greg Hanneman, Daniel Kiecza, Alon Lavie, Burn L. Lewis, Amit Srivastava, Paola Virga, and Kenneth Heafield

6.2.4.1. Introduction

Speech- and text-processing engines, such as speech-to-text and machine translation, are being assembled into ever-larger groups of engines, to achieve several goals:

- to perform increasingly-complex tasks, such as transcription, translation into English, and extraction of information from foreign-language news broadcasts,
- to facilitate system-combination techniques (Jayaraman *et al.* 2005; Fiscus 1997) to achieve higher accuracy, and
- to provide fault-tolerance in case of failure of an engine.

GALE's Language Exploitation Environment (LEE) working group has embarked on several activities to achieve interoperation of GALE engines using the Unstructured Information Management Architecture (UIMA) software framework (Ferrucci and Lally 2004). By "interoperation", we refer to single-point invocation of a group of engines, which we refer to as an "aggregate". The GALE Type System (GTS) (Pitrelli *et al.* 2008b) has been developed as a common set of data formats for such UIMA aggregates. The GALE Interoperability Demo (IOD) system (Pitrelli *et al.* 2008a) employs UIMA and GTS to interoperate 15 engines from seven remote sites spanning all three GALE consortia, to make Arabic broadcast news and Web sites browsable as English text grouped and summarized by topic, and playable as synthesized English speech. IOD inter-operates engines performing 11 distinct functions:

- language/dialect identification,
- gender determination / speaker detection,
- speech-to-text (STT),

- entity detection (ED),
- machine translation (MT),
- translation system combination (multi-engine machine translation or MEMT),
- story-boundary detection,
- topic-clustering of stories,
- summarization of topic-clusters of stories,
- headline generation, and
- text-to-speech synthesis.

These engines run on diverse computing platforms physically distributed across multiple states and countries at their home sites:

- IBM, New York,
- Carnegie Mellon University, Pennsylvania,
- BBN, Massachusetts,
- RWTH Aachen University, Germany,
- University of Massachusetts at Amherst,
- Columbia University, New York, and
- Systran, France.

Several principles guide the design of IOD, as more-fully detailed previously (Pitrelli *et al.* 2008a):

1. Fault-tolerance: When practical, multiple engines performing the same function are employed, to yield robustness against failure of individual engines.
2. Complementarity of engines: Different engines performing the same function may each in turn be best suited to process different types of input data, and so interoperating them followed by a system-combination engine can yield improved accuracy.
3. Exploitation of multiple engine functions: An engine should take advantage of functions preceding it in the aggregate, by making use of their results when appropriate.
4. Well-delineated functions: Some combinations of engines were tightly coupled and mutually optimized, e.g., STT and MT integrated to process audio. Mutual optimization of engines should be exploited for accuracy when practical, however, not in cases in which such coupling impedes interoperability with other engines, as IOD's goal is to achieve accuracy and functionality through interoperation.

In short, IOD's goal is to use UIMA to interoperate engines, to exploit the complementary performance and function of engines by imposing minimal constraints upon each engine, such as input-output specifications to delineate functionality.

For three years, the IOD aggregate of GALE engines has been invoked nightly, processing the audio from three to six hours of broadcast news from two Arabic news networks, Al-Arabiya and Al-Jazeera. A simpler "web aggregate" is also being run to process text from Arabic news Web sites. In addition, the UIMA Component Repository (UCR) and UIMA Component Container (UCC), described in Section 6.2.5, have been established at Carnegie Mellon University to provide GALE community members with Web-based access to configure their own aggregates of these and other engines to process audio and text data of their choosing.

6.2.4.2. UIMA: Managing Engine Interoperation

The widely-distributed engines that make up the IOD operate under the management of UIMA (Ferrucci and Lally 2004). UIMA is an architecture and software framework for composing and deploying multi-modal analytics, adding structure to unstructured data such as text, audio, and video. The architecture is undergoing a standardization effort, referred to as the UIMA specification, by a technical committee within the Organization for the Advancement of Structured Information Standards (OASIS).[3] IOD uses Apache UIMA, an open-source implementation of the UIMA Architecture.[4]

Capabilities

Some of the features Apache UIMA provides are:

- An extensible type-system mechanism used to formalize the format for all input and output data,
- A common analysis structure to organize and maintain a segment of data, (e.g., text, audio) and all the analysis results,
- An extensible component-based framework that simplifies the integration and deployment of UIMA-compliant analytics,
- Ability to create UIMA-compliant components from analytics written in Java, C++, Perl, Python and Tcl,
- Support for Linux, Windows and other Unix-based OS,
- Tools for developing and testing components individually,
- Ability to run components as shared services on the Internet,
- Ability to create custom analysis flow, with complex error-handling options,
- Ability to process multiple data segments simultaneously through different stages in the aggregate, for increased throughput,
- Ability to process the same segment through multiple engines in parallel to reduce latency, and
- Ability to re-segment data.

UIMA's capabilities are employed by IOD as follows.

[3] UIMA Architecture at OASIS: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uima
[4] Apache UIMA: http://incubator.apache.org/uima

Data and views

Data is passed among UIMA components in the Common Analysis Structure (CAS) with the format and structure of the data defined by an application-specific type system. The GALE Type System (GTS) (Pitrelli *et al.* 2008b) has been created for aggregates of speech- and natural-language-processing engines, such as IOD. It is important to note that agreeing on a type system does not constitute a complete data-flow design for an aggregate such as IOD. Rather, it must be combined with usage conventions, such as, which types or attributes of types are optional and which required for each engine.

Each CAS contains a "view" of the data being analyzed – for the IOD, a two-minute segment of audio from a news video, or a Web page. For multi-modal applications, UIMA supports multiple views within a CAS, each representing a different way to view the same basic data. In the IOD we create additional views as the CAS progresses through the aggregate – a source-language text view for the Arabic text from each STT engine, and a single target-language view for the English text comprising the merged translation. Each view initially contains only the data to be analyzed; engines add their results as annotations on the data.

Most engines need access to only one type of view, and are not dependent on the existence of other views. STT uses just the audio view, MT and MEMT the source-language views, and monolingual engines such as summarization use the target-language view.

Data re-segmentation capability

UIMA supports a powerful type of component, the CAS Multiplier, which has the ability to produce new CASes. For each input CAS processed, a CAS Multiplier can emit zero or more new CASes, effectively splitting, merging, delaying, or arbitrarily re-segmenting the input CASes. If an analytic needs more contextual information than is available in a single CAS, it can be implemented as a CAS Multiplier, releasing delayed CASes when sufficient context has been received. IOD initially segments audio data into manageable but arbitrary segments, and later uses a CAS Multiplier to re-segment the data into complete stories for the convenience of the analytics that follow.

Scale-out, load balancing, and parallelism

Apache UIMA provides a flexible and powerful scale-out capability enabling UIMA components to be deployed as shared services, running in separate processes on the same or different machines. Apache UIMA uses the Apache ActiveMQ implementation of Java Message Service (JMS) to provide asynchronous connectivity between an application and its services. Multiple instances of a UIMA service can be deployed on the same or different machines, all processing CASes from the same input queue, hence providing dynamic load balancing. All components, the dedicated local ones as well as the shared remote ones, have their own input queue and processing thread, so multiple CASes can be active simultaneously, increasing throughput. Since STT is by far the slowest engine function in IOD, this scale-out capability has been exploited by deploying up to five instances of the slowest engines, so achieving throughput close to real-time.

If an application has a set of engines with no processing interdependences, UIMA can invoke them in parallel, reducing latency. The same CAS is sent to all, and UIMA merges the results returned from each. The engines must only add new entries to the CAS, leaving all existing data unmodified. In the IOD this is done for all of the STT engines, reducing their contribution to the latency to that of the slowest.

Data-reorganization components

IOD requires data-reorganization components for a variety of purposes, including:

- the creation of CASes at the beginning of the aggregate, and extraction of information from CASes at the end,
- creating, from the early engines' annotations, new views suitable for processing by subsequent engines, along with cross-references back to the original annotations,
- creating data structures within views to prepare for processing by subsequent engines, and
- re-segmenting data into new CASes as needed by subsequent engines.

From a research perspective, these components are very distinct from the engines in the high-level design of IOD. They perform relatively "mechanical" functions interfacing among the engines and between them and the outside application. In contrast, "engines" are being improved experimentally on an ongoing basis in distributed research labs, and so are most conveniently hosted at their home sites. Despite this distinction, from the perspective of the UIMA framework, engines and data-reorganization components are all simply "UIMA components", simplifying development, debugging, and deployment. Specific details of how engines and data-reorganization components interact in the IOD aggregate are described in order of process flow in the following section.

Flow control and error handling

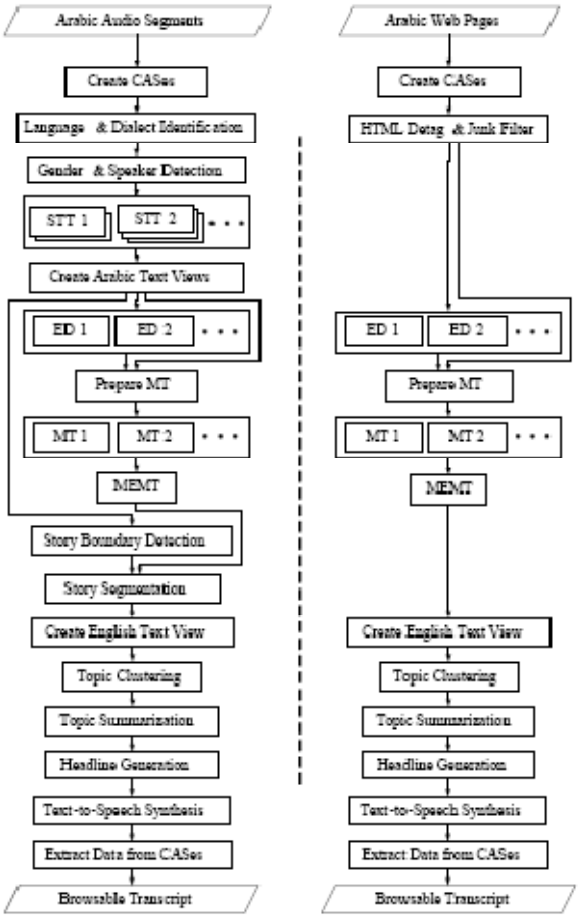

Figure 6.23: IOD system diagram. Bold boxes indicate engines; other rectangles represent data-reorganization components. Arrows represent data dependences; processing flow is top-to-bottom.

UIMA supports a customizable flow controller that manages the progress of each CAS through the sequence of components. It can specify which components may be run in parallel, and can change the flow depending on CAS content or on engine failure. UIMA supports a number of error-handling operations for each CAS sent to a remote service, e.g. a limit on processing time, the number of retry attempts, the number of errors that will trigger removal of the component or termination of the application, and also if a CAS can continue in the flow after the failure. If the failure of an engine does not affect any succeeding engines or the overall application, or if there are other engines performing the same function, then error handling should let the application continue processing that CAS. In this way, multiple engines provide fault tolerance. In the extreme case when all engines in the set fail, and the function is critical, the flow controller terminates the application.

For the configuration depicted on the left side of Figure 6.23, a pre-processing step extracts two-minute audio segments from Arabic broadcast news video. The news show

is later re-segmented into story segments as described below. The configuration on the right side of the figure uses many of the same components, along with an HTML detagger, to process Arabic news Web pages output by a Web crawler. Engines are operated as remote UIMA services, deployed, maintained and upgraded by their home sites in order to keep current with latest research developments. IOD users operate as UIMA clients in accessing these engines. One user is IBM, which runs the audio aggregate every night on current news sources and the Web aggregate on demand. Another is the UCC (Section 6.2.5) at CMU, which allows users to configure their own aggregates from the IOD engine services, and to supply them with audio or text files.

The following paragraphs detail the operation of the engines and the data-reorganization components interleaved among them to facilitate their interoperation.

CAS-creation data-reorganization components (UIMA "collection readers") populate each CAS with the data to be analyzed. The "audio collection reader" creates a CAS for each two-minute audio segment, initializing it with an audio view and an `AudioDocID`[5] containing attributes which represent data about the entire segment, such as that this is the *N*th out of *M* segments in a show. The "web collection reader" creates a CAS for each news Web page, and a following HTML detagger strips out HTML tags and other material to create a source-language text view, representing the segment as an Arabic plain-text string.

The *Language/dialect identification* engine creates `AudioLanguageID` objects in the audio view, identifying the language and dialect between two time points in the audio.

The *Gender/speaker detection* engine creates `SpeakerID` objects in the audio view, each labeling the span between two time points as containing speech by a specified gender, or by a speaker represented in the engine's models.

Speech-to-text (STT) engines add objects to the audio view as follows:

`AudioToken`s, which label a word of speech onto a span between two time points (mandatory)

`SU`s (sentence units), which label a span of time as containing one sentence of speech (preferred),

`SpeakerID`s (optional), and

`AudioLanguageID`s (optional).

STT engines are normally the longest running components in IOD. Therefore, they are a priority for deploying multiple instances of an engine, as depicted by stacked boxes in Figure 6.23, currently ranging up to five instances per engine.

Create source-language views is a data reorganization component creating an Arabic text view corresponding to each STT engine. For each STT, the strings in its `AudioToken`s are concatenated into a single Arabic text string, which serves as a new representation of the content of the segment, to prepare for text only engines such as ED

[5] Typewriter font will be used for names of data types defined by the GALE Type System; for more details, please consult the GTS paper (Pitrelli *et al.* 2008b).

and MT. This component then creates in each new view a set of `AudioXref`s, each associating a span of characters in the new view with a span of time in the audio view. It also creates `Sentence` annotations, which indicate that a span of string indices in the view corresponds to a sentence. These are derived by taking the `SU` boundary times from a selected STT engine and mapping them across the view alignment onto positions in the new view's text string.

Entity-Detection (ED) engines process each source-language view and create `EntityMention` objects associating an entity type such as "person" with a span of characters in the view. Optionally, ED engines may produce `Entity` objects reflecting coreference analysis associating a set of mentions with a single underlying entity. Also optionally, ED engines may produce `EventMention`s and `Events` analogous to the corresponding entity types, and `RelationMention` and `Relation` objects relating pairs of entities. In Web aggregates, an ED engine is responsible for creating `Sentence` objects.

Prepare MT is a data-reorganization component which creates objects needed by MT engines, specifically, `WordToken` annotations which identify that a span of characters in the view string is a word, and `Translatable` annotations which serve to specify a partitioning of text in each source-language view into units to be processed by MT engines. Currently, one `Translatable` is created for each `Sentence.`

Machine Translation (MT) engines place target-language text strings in the CAS within one `TranslationResult` for each `Translatable` in each source-language view. In addition, MT engines create `Alignment` structures linking target-language words back to source-language words.

The *Multi-Engine Machine Translation (MEMT)* engine creates a single translation of the content by mining the *N* x *M* translations resulting from the *N* STT engines and *M* MT engines. MEMT formats its output translation as `TranslationResult`s and `Alignments` just as MT engines do. As MEMT's function is inherently related to multiple-engine processing, it is therefore particularly relevant to interoperation, and so MEMT is described in further detail below.

The *Story-Boundary Detection (SBD)* engine annotates each sentence boundary in a source language view with a score indicating how likely it is to be a story boundary. It requires two minutes of audio context before and after each boundary, so is implemented as a CAS Multiplier in order to delay the processing of each segment until the next one has been received.

Story Segmentation is a data-reorganization component which takes in two-minute segments and emits story segments, and is therefore a CAS Multiplier. Boundaries are chosen by applying a threshold to the scores produced by SBD. Re-segmentation enables subsequent engines, such as topic clustering, to operate directly on the appropriate data, i.e. a complete story. Note: Re-segmentation is not necessary in Web applications since each Web page is assumed to comprise a story.

Create target-language view is a data reorganization component creating an English text view in each video and Web segment by concatenating the `TranslationResult` strings corresponding to a selected translation, typically that output by MEMT. This view enables monolingual-text functions like summarization and headline generation to focus

on English text as their natural input. It also creates entity-related objects in the new view corresponding to any it finds in the source-language view, and creates cross-reference objects linking the new view and the appropriate source-language view.

The *Topic Clustering* engine creates a single `Topic` object in each video-story segment and Web segment. `Topic` contains an identifier which represents a cluster of stories it determines to be about the same topic. For each client, the topic-clustering engine maintains a history about past segments in order to label the current segment.

The *Summarization* engine also maintains a history and gives each CAS a `Summary` object containing a string providing an English summary of all stories in history to date sharing topic identifiers with the current CAS.

The *Headline Generation* engine creates two objects in a typical CAS: a `SummaryHeadline` containing a headline derived from the CAS's topic summary, and a `StoryHeadline` derived from the story in the CAS.

The *Text-to-Speech Synthesis* engine creates an audio rendition of the target-language view string, matching detected gender in the case of a video segment. This engine makes use of multiple views, as it follows cross-reference links between the views to associate each target-language sentence with the gender annotations applied to the corresponding audio span.

Extracting Data from CASes are UIMA CAS consumers which end the aggregates by extracting the results from each CAS, saving the information in an XML structure enabling users of a custom Web application to browse the topics, stories, and parallel translations.

Sliding results window

Currently, IOD is run every night and maintains a four-day results window. The intermediate results prior to the first history-maintaining engine, topic clustering, are saved for three days, so that four days worth of data can be run through the final stages to produce the daily results.

6.2.4.3. Multi-Engine Machine Translation

Of particular note among IOD's engines is the translation-combination engine, Carnegie Mellon University's multi-engine machine translation (MEMT), because it particularly exploits interoperation of multiple engines. Specifically, IOD typically runs two or three STT and two or three MT engines; each resulting STT–MT system combination provides another possible target-language translation for any given segment of source-language audio. MEMT collects the translations resulting from each STT–MT combination in the IOD aggregate, and assembles a new synthetic translation that tends to be better than that from any single STT–MT output. To carry out fast online synthetic combination within the aggregate, it relies on minimal information from the incoming translations, requiring only the text string produced by each STT–MT combination and (optionally) alignment information tracing each target-language word back to the source-language word that produced it.

Combination approach

The translations for each source sentence are first word-aligned to each other based on exact string match, stem match, or synonymy match using WordNet synsets. MEMT then iteratively builds up collections of partial synthetic hypotheses of increasing length by selecting an unused word from one of the original inputs and adding it to the end of a hypothesis currently under construction. When a hypothesis is completed, it is scored and added to a list of final hypotheses. At the end of the process, the highest scoring hypothesis is selected as the MEMT output and is passed to the remainder of the IOD aggregate. Special constraints can be enforced for the creation of atomic phrases, or high quality contiguous sequences of words in an original input that we would like to keep together as a single unit in the synthetic MEMT output.

Although we have preferred the synthetic combination method from the beginning (Jayaraman *et al.* 2005) as a way to produce a more flexible word ordering in the MEMT output, the MEMT system can be configured to perform hypothesis selection or fixed-word-order combination in the style of a confusion network (Rosti *et al.* 2007).

Integration into the IOD

The online and distributed nature of the IOD environment enforces some key operating constraints on the MEMT system, while the richness of the data defined within the GALE Type System (GTS) allows the algorithm to take additional information into account when constructing synthetic combinations.

MEMT is designed to be robust to variation in number of translations of a sentence, and can handle cases in which some of the input translation alternatives are empty, due to the possibility of STT/MT engines being lost due to system or network failure.

GTS includes detailed alignment information for mapping from audio segments to recognized source language words to translated targetlanguage words, which is maintained throughout the IOD's processing. This presents a particular challenge for MEMT, as it is simultaneously dealing with translations that were produced using different text views of the same audio, each of which has a different set of alignments. To solve this problem, we augment the output of the MEMT combination to additionally specify, for each word in its synthetic output, which word instance from which MT output was chosen. Our system is then able to assemble a consistent and correct set of alignments for the MEMT output by recovering the original alignment information for those words.

6.2.4.4. Adapting Existing Engine to UIMA Interoperation: Case Study, BBN STT

BBN's Audio Monitoring Component (AMC) product[6] produces in real-time a continuous rich transcript of foreign-language broadcasts. The transcript includes speaker information, sentence boundaries, and named entities. The integration of AMC into the IOD/UIMA framework was completed rapidly for several reasons: (1) AMC is a mature product producing XML-formatted transcripts which adhere to a strict schema contract, (2) GTS required only a subset of the AMC schema types and mapping these types was

[6] BBN BMS system: Hhttp://bbn.com/products_and_services/bbn_broadcast_monitoring_system/H (the BMS includes BBN's AMC STT engine)

straightforward, and (3) UIMA provided the necessary infrastructure for quick integration, including simple framework installation, an API to generate annotations, and scripting language support.

The MEMT component in IOD requires common sentence segmentation for all STT and MT engines. The sentence segmentation produced by AMC was adopted as a common segmentation for the demo for several reasons: The AMC sentence segmentation is produced by a tightly-coupled, rich transcription engine. Also, and more importantly, the AMC sentence segmentation is specifically tuned for human readability. Its sentences will, for example, not exceed a fixed upper-duration threshold. This behavior proved critical to ensure reliable performance of the MEMT component. AMC for IOD is hosted at BBN Technologies. Incorporation of AMC's ED remains work in progress, due to IOD's categorization of ED as a text-annotating rather than a speech-annotating function. However, all other AMC functionality has been part of the entire history of IOD.

6.2.4.5. Experiences and Observation

Constraints on Engines

Running an engine in the context of an aggregate imposes constraints which do not apply to running in isolation. Adding a new engine to an aggregate often creates new constraints on several other engines. Many of these constraints are low-level in nature; discussion here will be limited to illustrative examples.

One example is that an engine's standard output often includes special codes for exceptional situations, such as one MT's "unk" for an unknown word. Isolated evaluations of engines account explicitly for such codes. However, in a growing aggregate, requiring engines to handle others' codes becomes impractical. In this example, subsequent engines may mistreat all appearances of "unk" as multiple occurrences of a single popular word, degrading the accuracy of algorithms dependent on word frequency, so MT needs another scheme for handling unknown words, such as attempting a transliteration.

Another example is that interoperation can magnify the consequences of a malfunctioning engine. Outputting malformed data can cause a compounding of malfunction from the downstream engines. Therefore, interoperation imposes heightened requirements for engines to produce valid data.

Performance Evaluation

Rigorous evaluation of IOD accuracy is impeded by

- the large number of interacting engines,
- the complexity in obtaining ground-truth data against which to assess aggregates' outputs, and
- the presence of several engine functions whose results' accuracy remains subjective, such as some of the text-processing engines.

We have begun to define a formal framework for evaluation of such aggregates (Murthy *et al.* 2008).

Upgrading engines

Frequently, an engine somewhere in the aggregate is improved, and the improvement is validated by a unit test; however, care must be taken in the presence of a downstream engine whose classifier is trained on output data from the engine to be upgraded, often necessitating ripple-forward re-training.

6.2.4.6. Conclusions

The IOD has demonstrated that it is possible to build a useful application from the combination of many powerful analytic engines deployed on a variety of platforms and languages at sites around the world. The UIMA development environment facilitated the independent development and testing of each component, making it easy for this diverse mixture of companies and universities to collaborate efficiently.

Natural language processing technologies have advanced to the point that large aggregates of NLP engines can provide useful output on tasks requiring such combinations of engines. In spite of their computing-platform variations and distributed locations, UIMA enabled the IOD team of seven GALE sites to collaborate on interoperating 15 engines to transcribe, translate, and extract information from Arabic news video and Web content daily. Results are available[7] showing English text headlines hyper-linked to topic summaries, story headlines, and story translations.

The type system and data reorganization components developed for IOD could be used as an experimental test bed for a collaborative effort evaluating the effectiveness of various combinations of engines developed by separate groups. Many research areas involving the interoperation of engines could be explored, e.g. the effectiveness of system-combination techniques on transcriptions and entities as well as on translations, the benefits to MT of using the source-language entities, the value of independent gender and/or speaker detection to STT, etc.

Acknowledgements

The authors thank James Allan, Bob Armstrong, Shilpa Arora, Manuel Bardea, Oliver Bender, Eric Brown, Jaroslaw Cwiklik, Bret Ehlert, Gopi Flaherty, Martin Franz, Andre Gauthier, Isaac Harris, Abe Ittycheriah, Nanda Kambhatla, Francis Kubala, Daben Liu, Evgeny Matusov, Kathy McKeown, Justin Merrill, Uma Murthy, Eric Nyberg, Leiming Qian, Jerome Quinn, Ganesh Ramaswamy, Eric Riebling, George Saon, Barry Schiffman, Jean Senellart, Sergey Sigelman, Olivier Siohan, Sara Stolbach, David Svoboda, Alan Watkins, William Wong, and Jian-Ming Xu for their contributions to IOD.

6.2.5. The UIMA Component Repository (UCR) and UIMA Component Container (UCC): A Facility for Flexible

[7] Video: http://rosetta.watson.ibm.com:8888/iod/
Web: http://rosetta.watson.ibm.com:8888/iod/?web=1

Exchange and End-User Configuration of Operational NLP Engines

Authors: Eric Nyberg, Eric Riebling, Burn L. Lewis, and David Svoboda

6.2.5.1. Introduction

As part of our work on the GALE Language Exploitation Environment (LEE), the LEE working group specified a common annotation type system for speech- and text processing engines (Pitrelli *et al.* 2008b) using the Unstructured Information Management Architecture framework (UIMA)[8]. This common type system has been used to integrate a variety of different components from different organizations into the GALE Interoperability Demo (IOD), an aggregate analysis engine that supports daily processing of emerging news in foreign-language video (Pitrelli *et al.* 2008a). Although the development of the shared type system and IOD was motivated primarily by the requirements of the GALE program, the shared type system and aggregate component architecture support a degree of modularity and interoperability which in turn imply a significant opportunity for low-cost component and aggregate reuse.

This section describes two additional capabilities developed by the GALE LEE team: the UIMA Component Repository (UCR), a Web-based portal designed to support efficient component dissemination and reuse; and the UIMA Component Container (UCC), a Web-based portal for run-time execution of aggregate analysis engines.

6.2.5.2. UIMA Component Repository (UCR)

The UIMA Component Repository (UCR) consists of a Web-based user interface to a relational database which stores UIMA components (Collection Readers, Annotators, CAS [Common Analysis Structure] Consumers) along with associated metadata about each component. Users may submit their own component in one of two ways. They may publish only the component's descriptor (an XML file which indicates how the component can be invoked as a service), or if they wish to disseminate the binary (and optionally source) code for their component, users can upload it to the UCR in a special .zip format (UIMA Processing Engine ARchive [PEAR] format). There are two types of users: registered and unregistered. All users may browse and download components or descriptors, but only registered users may upload components. The specific use cases supported by the UCR are listed in the following subsection.

Capabilities

The UCR supports the following user actions:

- *Register User*. New users register by selecting a username and password, and providing a valid email address.

[8] The latest version of UIMA is currently being developed as an Apache Incubator project: Hhttp://incubator.apache.org/uima

- *Join Mailing List*. The site provides an interface for subscription to a LISTSERV mailing list for UCR users.
- *Browse Components*. The site supports a variety of browsing options for displaying components: All Components, Components by Category, Components by Submitter, and Commercial (costs money) vs. Noncommercial (free) Components.
- *Search Components*. The site provides a simple keyword-based search capability that matches the user's keywords with the component titles and descriptions; an advanced search capability allows the user to specify search terms to be matched against particular component metadata fields (name, description, developer name, input and output annotation types).
- *Download Component or Descriptor*. The user may download either the component code (PEAR file) or XML component descriptor for the component being viewed.
- *View Component Type Tree*. When the annotation type system is provided with the Component (a PEAR file is present), the system provides a capability for viewing the annotation type hierarchy within the Web browser.
- *Rate Component*. A simple 1-5 stars rating system is provided to registered users so that they may provide feedback on individual components; the average rating is displayed when a user displays a component.
- *Announce Component*. The user provides the metadata and descriptor for the component, but does not provide the PEAR file (component code).
- *Publish Component*. The user provides the metadata, descriptor and code for the component, encoded in a UIMA PEAR file.

The UCR also provides additional resources for the user, such as site help, a UIMA FAQ, tutorial videos, and community links to additional off-site resources.

Design

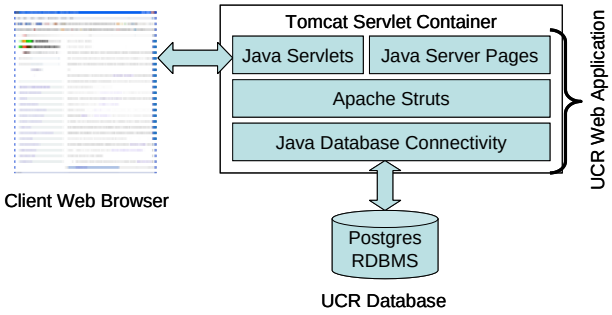

Figure 6.24: UCR Architecture

The UCR is designed as a Web application based on a typical three-tier Model-View-Controller Web application architecture. The underlying data model is a relational database schema which persists information about components and their metadata, users,

user roles, type systems, licensing information, and runtime requirements. The primary interface to the UCR is a set of Java servlets, which access static page content via HTML and Java Server Pages, and dynamic content (e.g., results of component searching and browsing) via a set of actions defined in Apache Struts. Struts actions are implemented to use a specialized data access object (DAO), which utilizes the Java Database Connectivity (JDBC) library to access the underlying relational model. The current architecture is shown in Figure 6.24.

Deployment

The UCR was first deployed on the Web in July 2006.[9] UCR currently runs on Red Hat Linux servers using Apache Tomcat as the servlet container and PostgreSQL as the relational database. In addition to the public deployment, we have also deployed an access-restricted enterprise instance of UCR to support dissemination of components within the GALE community. As of January 2009 the public deployment supports 15 unique component submitters, and 43 distinct components have been uploaded; the broader community includes 87 users who have subscribed to the UCR mailing list.

6.2.5.3. UIMA Component Container

The UIMA Component Container (UCC) was created to support Web deployment of completed UIMA analysis engines, including the aggregate analysis engine, which integrates the GALE IOD. The UCC allows any user with access to a Web browser and the Internet to upload data and process it with the IOD pipeline, eliminating the need to pre-install and manually configure the UIMA SDK and the IOD components, which are accessed as remote Web services. Users may upload speech and/or text data, select components to run, and queue processing jobs which run asynchronously; as jobs are completed, the resulting output is placed in a directory for later retrieval.

Capabilities

The UCC supports the following user actions:

- *Upload Data*. The user may upload data files (text or PCM audio) to be processed.
- *Configure Pipeline*. Based on the input data which has been selected, the user may choose from the available components to be run by the pipeline.
- *Create Job*. The user selects a particular set of input files to be processed by the configured analysis engine.
- *Queue Job*. The user queues the configured job for processing.
- *View Jobs*. The user may view the contents of the job queue, as well as the status of their submitted job(s).
- *Cancel Job*. The user may cancel a running job, which is terminated and removed from the job queue.

[9] Hhttp://uima.lti.cs.cmu.edu

- *View Output*. The user may view the text and/or speech output from any completed job.
- *View Log.* The user may view the log for completed as well as in-progress jobs.

Design

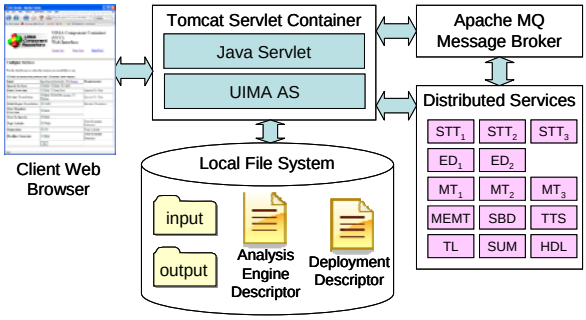

Figure 6.25: UCC Architecture

The UCC is designed as a servlet Web application that stores job input and output on the local file system (see Figure 6.25). For the IOD pipeline, input may be Arabic text or Arabic audio (no other languages), and output is English text and optionally synthetic speech audio. Based on the options selected by the user, an analysis engine descriptor and deployment descriptor are created for the configured pipeline. Configured jobs are queued and executed within an UIMA Asynchronous Engine (AE). The AE runs a local Collection Reader and CAS Consumer to read and write data from/to the local input and output directories; the components in the pipeline are accessed as distributed services. An Apache MQ message broker instance provides a mapping from the component namespace used in the deployment descriptor to the actual IP addresses of the nodes running the deployed services. This lightweight design does not require that any component code be delivered to the UCC or run locally, simplifying the integration process and reducing the level of processing resources required to deploy the UCC.

The UCC Web interface uses a simple string protocol to transmit the names of selected services to the UCC Web application. Services are named based on provider and service type, and pipelines are specified by appending these name strings together, e.g.:

```
IOD.artxt.IBM-ED.BAS-ED.IBM-MT.RWTH-MT.CMU-MEMT
```

There can be more than one provider for a given service. For example, three different organizations provide Speech-to-Text engines for the IOD pipeline. When more than one provider of the same service is selected, the services are run in parallel. Different services are presented to the user for selection, based on whether the chosen input data is text or audio.

Deployment

The UCC runs as an Apache Tomcat Web application. For each queued job, an analysis engine descriptor (specifying the components) and a deployment descriptor

(specifying how to locate the components as Web services and run the pipeline) are created from the string description of the pipeline provided by the HTML forms interface. These descriptors are used to create an instance of a UIMA Asynchronous Engine, which reads the input data and connects to multiple remote Asynchronous Engines that provide various NLP services via the ApacheMQ Message Broker. The current set of services and providers integrated in the UCC IOD pipeline (see Figure 6.25) includes:

- **STT**: Speech-to-Text (BBN, IBM, CMU)
- **ED**: Entity Detection (IBM, BasisTech)
- **MT**: Machine Translation (IBM, RWTH Aachen, Systran)
- **MEMT**: Multi-Engine Machine Translation (CMU)
- **SBD**: Story Boundary Detection (IBM)
- **TTS**: Text-to-Speech (IBM)
- **TL**: Topic Labeler (UMA)
- **SUM**: Summarizer (CU)
- **HDL**: Headline Generator (IBM)

6.2.5.4. Experiences and Observations

When components are uploaded to the UCR as PEAR files, they are validated using built-in library code that is part of the UIMA SDK. In practice, some submitted UCR components failed to validate. In addressing this issue, we learned that the validation process is sensitive to the version of Java used to compile the component code, which must be identical to or earlier than the version of Java used to compile and run the UCR. In our case this required upgrading the UCR and the enclosing Tomcat container to run under Java 6.

We also learned that components submitted to the UCC should not depend on external code, services, file structure, or environment variables. We revised the UCC help documentation to encourage developers to test their components using the UIMA PEAR installer before submitting them to the UCR. This mirrors the UCC run-time environment, and has helped developers to create more environmentally independent components.

Incremental revisions to the UCC design were required as more services were added to the IOD pipeline. The original design required developers to run a Perl script to generate Analysis Engine Descriptors and Deployment Descriptors for each possible combination of services, and then run a separate script to remove files for ill-formed service combinations. This process proved unwieldy as the number of possible descriptors grew to several hundred. The revised design allows the UCC to generate custom descriptors based on the services chosen by the user. A file containing Analysis Engine and Deployment Descriptor 'snippets' for each remote AE is used to generate the custom descriptor for each job's specific pipeline. This approach is much more efficient, significantly reducing the number of descriptors that comprise the UCC Web application.

The mechanism for adding new services can still be improved. Although the use of pre-defined descriptor snippets helps, the names of services referenced within the snippets file must be coordinated with service references within the Java code. We plan

to refactor the code to use service names which are defined in an easily-maintained central location.

It is difficult to report progress on the processing of individual documents (UIMA CASes) within pipelines configured using the UCC. Because of the presence of CAS Multiplier components (to support parallel execution of services), the number of output CASes can exceed the number of input documents. Although the UCC currently presents "M of N" CASes processed as part of its status display, these numbers do not make sense to the user if M exceeds N. Better approaches may be to simply report the number of CASes currently being processed or the percentage of work completed, based on seconds of audio or characters of text.

Because the Asynchronous Engine instance runs in a separate thread from the UCC servlet (which returns immediately after a job is queued), UCC does not provide dynamic status information during execution; the user must manually refresh a "View Job Status" page. This solution is less than satisfactory for long-running jobs which require constant monitoring (e.g., during debugging). A better solution would be to implement some form of "push" technology (e.g., AJAX) to improve usability.

Due to the nature of the GALE Interoperability Demo, some services (such as the Topic Labeler and Summarizer) are required to save state as streams of documents are processed. Since each remote service receives documents to process as separate service requests, an additional mechanism is required to indicate that a job is a continuation of an earlier job from the same user, so that components know whether to re-initialize any state variables. To address this requirement, we added a new capability to the UCC that allows the user to specify whether a job represents a new session, or whether it is a continuation of a previous session from that user. Another component (Story Boundary Detection) requires that all documents in a job are to be processed in order, and therefore, saves state between documents, as well as, any out-of-order documents. To help SBD recognize overlapping or incomplete jobs UCC adds a unique job ID to each request.

6.2.5.5. Conclusions

We have described the UIMA Component Repository and UIMA Component Container, two capabilities which are designed to promote re-use and easy re-deployment of text and audio analysis components configured as UIMA asynchronous analysis engines. Although the current implementations of UCR and UCC meet the basic capability requirements, there are several areas which are targeted for ongoing improvement:

- The UCR does not explicitly manage or reason about the Java version level and UIMA SDK compliance of submitted components; currently developers must re-compile and re-submit their own components, and users who download components may be required to recompile them.
- The UCR cannot execute a single component on sample test data in order to determine is suitability for an intended task before it is downloaded.
- The UCC requires significant effort to deploy a new pipeline; it would be useful to provide a wizard capability for generation of snippets files for automatic compilation of descriptors.

- The UCC persistence mechanism (using the local file system of the server) will be difficult to scale unless it is re-implemented using a more flexible and powerful persistence mechanism (e.g., a relational database).
- Some users of the UCC may prefer a deployment which supports streaming data to/from the UCC server in real time, eliminating the requirement for manual data upload/selection and local data storage.

We plan to address these issues in ongoing refinements to the UCR and UCC packages.

Acknowledgements

The authors thank Shilpa Arora, Eric Brown, Luis Chavez, Yurdaer Dogonata, David Ferrucci, Tong-Haing Fin, Lei Fong, Greg Hanneman, Kimberly Kettner, Nanda Khambatla, Alon Lavie, Humberto Lezama, Justin Merrill, James Rankin, Hideki Shima, and Django Wexler for their contributions to the UCR and/or UCC.

Chapter 6.3 Evaluation of Operational Engines

6.3.1. The for User-centered Evaluation of GALE Systems

Authors: Daqing He, Peter Brusilovsky, Jonathan Grady, Jaewook Ahn, Yiming Yang, and Monica Rogati

6.3.1.1. Introduction

As part of our participation in the development of evaluation tools for GALE systems, we developed the "Exploration and Discovery Information Evaluation" (EDIE) framework to assist with the development of the Rosetta system and its components.[10] Along with the series of experiments testing the Rosetta system, we also developed associated evaluation methodologies and metrics, which make EDIE a complete framework for user-centered utility evaluation.

The GALE program aims to develop and apply technologies to deliver pertinent and consolidated multilingual/multimedia information in easy-to-understand forms for military personnel and monolingual English-speaking analysts to use in response to direct or implicit requests. Therefore, utility evaluation, which specifically examines how well the technologies can facilitate such delivery, is vital to the development of GALE systems. Our EDIE work concentrated on informing the design and development of GALE systems through user-centered utility evaluation. Our primary application focused on the Rosetta system and its CAFÉ adaptive engine. Our experience shows that EDIE itself is a valuable contribution to the GALE program. The goal of this section, therefore, is to introduce EDIE and the accompanying evaluation design principles to the GALE community.

[10] See section 6.3.2.2 for details on the design of the Rosetta system.

6.3.1.2. EDIE for Evaluating GALE Systems

EDIE was designed at the beginning of the GALE program to enhance the development of the GALE systems by providing robust utility evaluation. At that point, the utility evaluation issues in the GALE systems were still at the discussion stage. The fundamental challenges that EDIE, as an evaluation framework, was designed to address are: can any existing information access tasks be borrowed to model GALE user context? Can a set of task characteristics be proposed to simulate a least one type of GALE tasks? How should a performance-based user evaluation of GALE systems be organized?

6.3.1.2.1. GALE Analyst's Activity as Task-based Information Exploration

When interacting with a GALE engine, the information needs and corresponding search processes of military personnel and intelligence analysts are often heavily influenced by the complex tasks assigned. These types of search activities can be modeled as task-based information exploration (TBIE), which is a primary research topic in information access (White *et al.* 2006). Existing evaluation frameworks such as Text REtrieval Conference (TREC) are limited in evaluating TBIE, because the metrics and the methodologies are not adequate in the context of highly interactive exploratory search.

6.3.1.2.2. The Simulated Tasks

In order to achieve realistic user-centered utility evaluation of GALE systems, the EDIE framework simulates the tasks that an analyst performs while working with this class of systems. The following characteristics were assumed:

- *EDIE tasks resemble real-world work assigned to analysts*. All EDIE tasks start with a topic statement that simulates a Request For Information (RFI), and result in a short summary that resembles a two page "point paper." A subject has to explore large volumes of data from various sources to generate such a summary.
- *EDIE tasks are complex*. Each task requests information about a seminal event and its associated aspects. Therefore, these tasks usually include several subtasks that are connected to the overall task.
- *EDIE tasks are dynamic.* Events naturally evolve over time, so an EDIE task has to track the development of the events over several sessions before the final point paper can be generated.
- *EDIE tasks collect information at sub-document (snippet) level.* Although documents still play important roles in GALE searches, the useful information is typically contained in sentences or paragraphs (called snippets). With their small size, but high density of useful information, these snippets are perfect for investigation or report writing.

Among the existing test collections, we chose the TDT4 corpus developed for the Topic Detection and Tracking evaluation (Allan 2002) as the initial dataset in EDIE. We redesigned 18 EDIE tasks based on TDT topics using the assumed characteristics noted above as the guidelines.

G40001 – Galapagos Oil Spill

Background:
San Cristobal is a place near the Galapagos Islands in the Pacific Ocean.

Short description of the task:
An Ecuadorian oil tanker has spilled fuel near the Galapagos Islands, affecting the local fishing industry and threatening local beaches and wildlife. Your tasks are to suggest 1) how and where ocean currents may spread the oil, and 2) what actions should to be taken after the oil leak event caused by the oil tanker Jessica near the Galapagos Island. You should investigate possible support that can be provided by the U.S,, as well as potential legal actions against the responsible persons.

From the documents, find snippets of text that contain answers to each of the following questions:

1. On what date did the oil spill occur?
2. Where is the location of the oil spill?
3. What type of vehicle was involved in the event?
4. How many gallons of fuel were spilled?
5. What is the cause of the spill?
6. Who is responsible for the spill?
7. What animals have been affected by the leaked fuel?
8. Who put efforts in relieving the pollution?
9. Any support from other countries?
10. What is the impact on the local people, environment, animals and plant life?
11. What are the penalties/lawsuits for those responsible?

Figure 6.26: An EDIE Task G40001

In addition to the description of the task and subtasks, each EDIE task also contains brief background information, including a seed story (i.e., a document relevant to the scenario). This helps to establish some consistency in the minimum background knowledge. Figure 6.26 shows a real EDIE task.

Human annotators were recruited to mark up the "ground truth" for each EDIE task (i.e., to mark-up the documents relevant to each task). To reduce subjectivity, at least two annotators worked on each task.

The annotations were collected based on using a much higher level of detail than a traditional document-level ground truth:

- annotations were made at the snippet level;
- annotations were independently collected under three aspects: topical relevance, novelty, and utility. We ask the annotator to provide utility annotation directly to us so that we may study the utility as a function of combining topic relevance and novelty.

- annotations were collected at three levels of judgment: highly relevant/novel/useful, slightly relevant/novel/ useful, and not relevant/novel/useful;

In total, we annotated snippets for all 18 tasks. The annotation files are available at http://amber.exp.sis.pitt.edu/gale/GALE-resources.html

6.3.1.2.3. Human-Centered, Multi-Stage, Multi-Aspect Utility Evaluation

EDIE assumes controlled laboratory experiments to assess the GALE systems. As the GALE systems are highly interactive, evaluations using EDIE do require that human subjects conduct task-based exploration and interact with the information system. Human subjects' interaction behaviors and interaction processes are important aspects for observation and evaluation.

More importantly, EDIE assumes that analysts' information exploration behaviors contain two major stages (or loops): *information foraging* and *sense-making* (Pirolli and Card 2005). During the foraging stage, analysts use their domain and search knowledge to collect potentially useful information from various media and sources. In the sense-making stage, analysts then process, organize, and distill the collected information into a coherent form so that it can be integrated into their state of knowledge. Because of the different nature of foraging and sense-making, we believe that the evaluation of systems, which targets information analysis, should consider both stages.

As indicated by the annotation work presented in Section 6.3.1.2.2, EDIE focuses on examining multiple aspects of the utility of the selected snippets. Both the topical relevance and the novelty of the snippets are critical. In addition, we examine the final utility of the snippets by tracking whether and how the snippets are used by the analysts in their final reports. To capture the event development and the need to handle duplicate information, EDIE also simulates the multiple sessions in utility search.

6.3.1.2.4. Utility Evaluation Metrics

EDIE adopted several evaluation measures. The first type of measures focused on *system retrieval effectiveness*. This includes measuring the accuracy and coverage of identified information (i.e., precision and recall) and measuring the utility of the obtained information for a specific task. All these measures concentrate on snippet level examination, rather than document level.

An example of the proposed measures of system performance is a snippet level precision that takes advantage of the fact that two human annotators generated the ground truth. The formula (1), which is derived from the work of Allan (2003), calculates the precision of a snippet *i* selected by a human subject, where oll_i is the character length of the common text chunk between the snippet *i* and the corresponding ground truth; w_i is the weight of the ground truth combining the mark-ups of both annotators. The value of the weight w_i can be one of five ad hoc assigned levels: 0, 0.25, 0.5, 1, 1.25, 2. nml_i is the character length of the part of the snippet *i* that has no overlap with the ground truth. Here the 0.5 associated with nml_i is the penalty weight.

(6.10)

The second type of measures focused on *formal user performance*, on *user activity patterns* mined from transaction logs, and on *subjective evaluation* obtained via questionnaires and interviews. Our idea of formal user performance is similar to that of Marchionini and Shneiderman (1988), where the measures examine the paths and decisions taken by the users during the process of completing their tasks. All these measures have been expanded to evaluate the information foraging and sense-making stages separately.

The utility measure in EDIE also deserves special mention (Yang *et al.* 2007). To be most precise, utility should be judged on the point paper, which is the final product of the task. However, the varying abilities of the subjects to produce a coherent report are beyond the scope of evaluating a GALE system. Therefore, EDIE's utility measure examines both: the process of snippet selection (i.e., foraging) and organization (i.e., sense-making). Together, the quality of snippet selection and organization reflect the system's utility. The measure also imposes a word limit on the final selected snippets so that the final snippet selections more closely resemble the point paper in size. At the same time, the word limit brings in a cost function for tradeoff - should it include as much available relevant information to a specific question or should it cover as many required questions within the specified word limit?

6.3.1.3. Applications of EDIE in Evaluating GALE Systems

In this section, we will briefly present the involvement of EDIE in evaluating several GALE systems and components. The goal is to reveal EDIE's usages in the context of the GALE systems.

6.3.1.3.1. CAFÉ Utility Evaluation

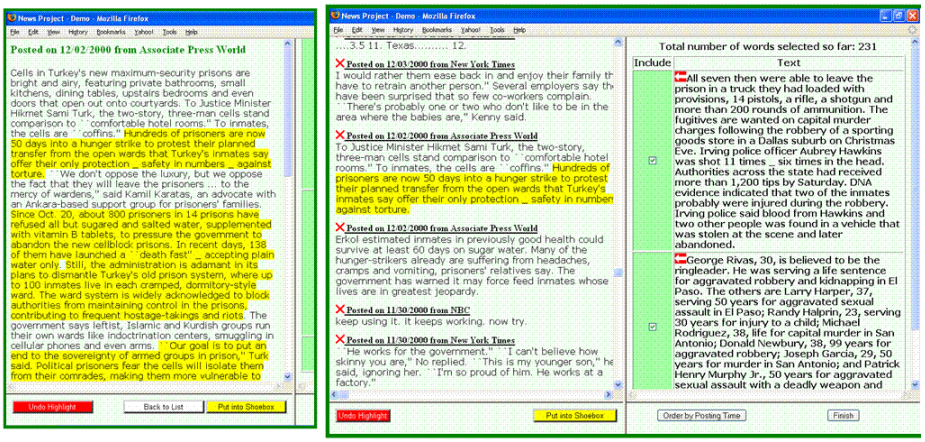

Figure 6.27: Information foraging in CAFÉ system: assembling text fragments in the shoebox

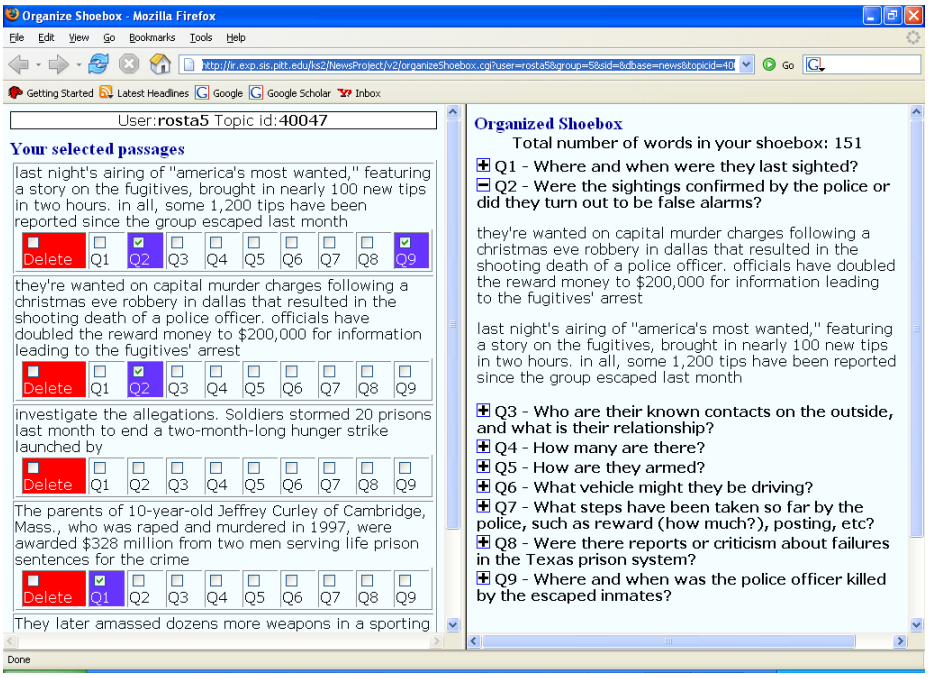

Figure 6.28: Sense-making in CAFÉ: selecting and organizing fragments in the shoebox

This evaluation examined an innovative adaptive filtering engine known as the Carnegie Mellon Adaptive Filtering Engine or CAFÉ, which is one of the major engines supporting Rosetta (Yang *et al.* 2007). Figure 6.27 shows the information foraging interface in CAFÉ. The left side of the right window shows the snippets recommended for examination. Analysts can select any part of the snippets and add them to the right side of the right window. The right side is called "the shoebox," which is a container for holding the snippets that the analysts find potentially useful for the final results. At any time, the analysts can click on a snippet to see the corresponding full document in a pop-up window, which is shown as the left window in Figure 6.27. Figure 6.28 shows the sense-making interface for CAFÉ. Here, all the snippets in the shoebox that were collected by the analysts during the information foraging stage are displayed at the left side of the screen. The analysts can delete any snippet, or label the snippet as related to one of the sub-task questions. Once the snippet is labeled, it will appear at the right side of the screen which shows the content associated with each sub-task question.

The goal of this evaluation was to confirm our hypothesis that adaptive filtering provides value in both stages of GALE searches (at least in comparison with a traditional Google-like search) and to measure the effect of CAFÉ. The study was conducted with eight subjects in July and August, 2006. The baseline system was a state-of-the-art snippet retrieval engine based on Indri (He and Demner-Fushman, 2003). Eight tasks were selected among the 18 possible scenarios.

Our study confirmed the usefulness of the CAFÉ engine, and also confirmed that there is a strong need to have adaptive engines in GALE systems. The results demonstrated that CAFÉ can generate ranked lists of snippets of significantly higher

precision than the baseline system. CAFÉ users waste significantly less effort finding useful information and could find more useful information in various stages of their exploration. The details of this CAFÉ study were published by He *et al*. (2008).

6.3.1.3.2. Evaluation of CAFÉ's Profile Update Model

Figure 6.29: CAFÉ engine in Rosetta System

The goal of this study was to assess the value of keeping the list of recommended items up-to-date in the information foraging stage of a GALE search. To make the appropriate design decision for CAFÉ and similar GALE systems, it was important to know how frequently the user's profile and the list of recommended items should be updated in GALE. The literature analysis revealed two popular approaches: some systems do it once per session (thus, it is called the *between-session* update strategy), whereas, others update whenever there is feedback entered into the system (called the *instant update* strategy).

To choose the approach which works best for the GALE system, we designed a controlled experiment with human users performing realistic tasks using two versions of the Rosetta CAFÉ engine that have the same underlying adaptive filtering engine, but with two different update strategies. Figure 6.27 and Figure 6.28 show the Rosetta CAFÉ engine with the between-session update strategy. Figure 6.29 shows the Rosetta CAFÉ engine with the instant update strategy.

The study was conducted in the summer and winter of 2006. We recruited 15 subjects and used two task scenarios from EDIE.

Our results demonstrated that the optimal update strategy to achieve both adaptability and stability for GALE system lies between the instant update and the between-session strategies. The between-session strategy helped to find information with better utility, and received better responses from subjects about its usefulness and usability. However, it prolonged the selection of useful snippets, whereas the instant update strategy helped

subjects to obtain almost all of their selected snippets (> 98%) within the first five minutes of the search sessions. The detail of this study was published in by He, Brusilovsky, Grady, Li, and Ahn (2007).

6.3.1.3.3. Evaluation of Usages of Task Models

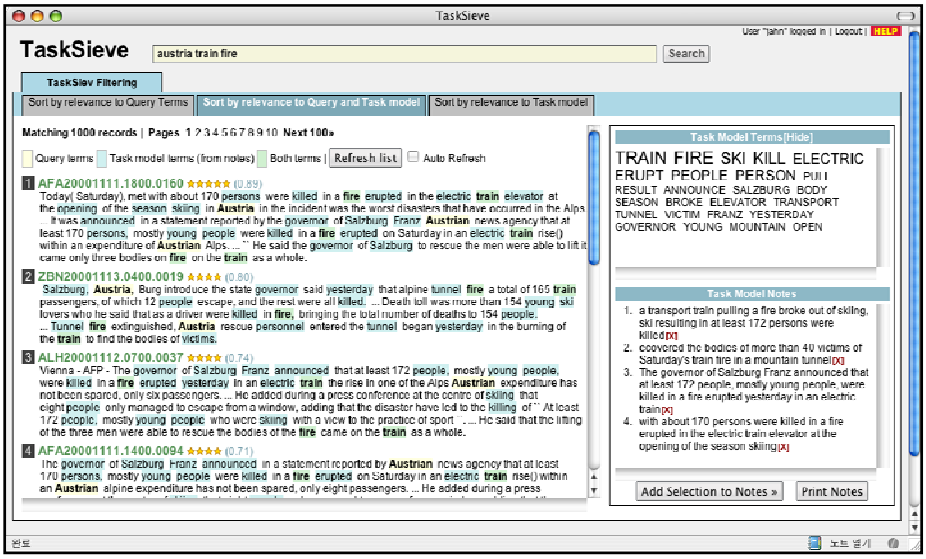

Figure 6.30: TaskSieve System

The goal of the third study was to explore the value of the task model in the information foraging stage of an analyst's work. In order to better support the analyst during a specific task, GALE systems should maintain a task model, which represents the task-related information obtained through interacting with the analyst. TaskSieve was developed as a prototype for GALE systems to explore how to design such a task model and how to employ it to improve user performance.

Ten subjects participated in the TaskSieve study in October, 2007. Each subject was randomly assigned to four tasks selected from EDIE tasks; they each performed two tasks on TaskSieve and the other two on a traditional search system. The sequence of tasks was randomized as well.

The results showed that it is important to integrate a task model for GALE searches. The task model enables a foraging system to keep more relevant items at the top of ranked lists, as compared to Google-like traditional search systems. At the same time, analysts can conduct more productive searches: the average precision values of TaskSieve's ranked lists gathered in the first ten minutes – when users were still relatively unfamiliar with the assigned task - were higher than those of the traditional search system's final ten minutes. Thus, TaskSieve users could select more relevant information earlier in their search sessions.

The results of this study were published by Ahn *et al*. (2008).

6.3.1.3.4. Evaluation of Usages of Named Entities

The last study was designed to assess the value of named entities (NE) in supporting both the information foraging and sense-making stages of the analyst's work. In GALE, named entities which refer to people, locations, organizations, events etc. can deliver important information. NameSieve is a GALE prototype system that supports named entity-based information exploration (Figure 6.31).

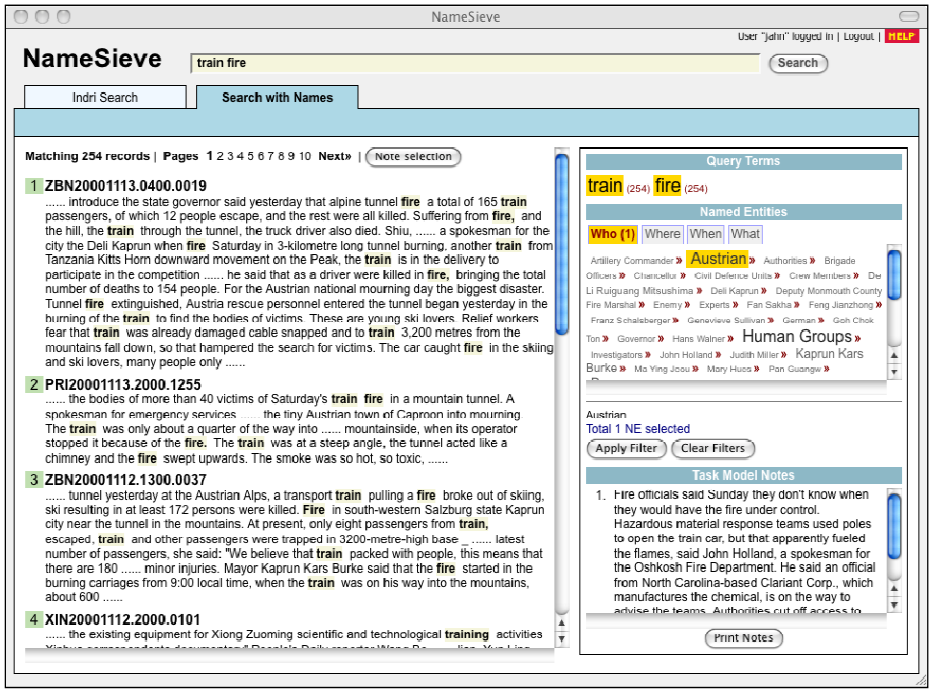

Figure 6.31: NameSieve System

NameSieve provides a simple and intuitive interface to manipulate NEs. Augmenting the conventional ranked list returned by a search engine, NameSieve extracts NEs from the set of retrieved documents and organizes them into tabs corresponding to the "editor's four-Ws" (Who, What, Where, and When). More frequently occurring names are displayed in larger sizes. Each name can be chosen to serve as a filter via a single mouse click. It allows users to post-filter initial search results and, thus, to focus on information relevant to their tasks.

A named entity tagger developed by IBM was used for the name extraction from the search results. Some heuristics, including Wikipedia article title matching, were used for noise elimination.

Ten subjects participated in the study (from December 2007 to March 2008). The study design and measures were the same as in the TaskSieve study. The experimental results showed that the NE-based filtering was actively used by the subjects, frequently switching to look for the Who, What, and Where information. The active usage of the NEs led to improvements in the information filtering results. The system performance

measured by the precision at ranks five and ten was significantly higher in NameSieve with its NE filters than in the baseline system. Subjects were able to make better annotations for the topic using the NameSieve system. In their feedback, the subjects expressed positive opinions about NameSieve, especially as they accumulated more experience on the search task and had more opportunity to compare the NameSieve and the baseline systems.

6.3.1.4. Discussion and Conclusion

In this contribution, we presented EDIE, an evaluation framework designed for assessing and comparing technologies created for GALE systems. EDIE includes a human-centered, multi-stage, multi-aspect, evaluation methodology; 18 realistic, complex, and dynamic information exploration tasks; and a set of evaluation metrics. A set of EDIE tasks built on the basis of the TDT4 test collection offers passage-level ground truth data annotated on multiple aspects of relevance. All components feature valuable innovations.

Our work brought to light some interesting and important insights related to the evaluation side of the GALE systems. First, the usage of multiple sessions in EDIE was important for capturing the dynamic development of the event and the tasks. According to the results collected from our experiments, this feature indeed showed the complexity of the tasks, the changes in systems performance, and the changes in analysts' behaviors over different sessions.

Second, it was also very helpful to assess separately the information foraging stage and the sense-making stage in GALE searches. This helped to assess how GALE systems are able to support each of these two stages.

Third, our work provided important insights on the evaluation of information exploration systems. Existing information retrieval systems are designed to work well within existing evaluation frameworks - i.e., produce strong topical relevance. However, as our studies hint, topical relevance does not equal better user support. Therefore, this gives us a basis to argue that evaluation involving human users is essential for evaluating GALE systems in order to obtain truly meaningful and useful testing results.

Finally, we observed that the GALE systems' performance may vary a great deal from one task to another. This phenomenon was observed for every type of performance measure explored. Not only do specific performance measures depend upon the specifics of the individual task, but the performance between systems focusing on different information access approaches may fluctuate. This data allows us to argue that instead of perfecting a single specific information access approach such as the Rosetta information filtering, GALE utility systems should use a combination of approaches and engines. Given the large variety of tasks, one engine's advantages can cover the other's weakness in a specific group of tasks.

We hope that this work will contribute to the establishment of evaluation approaches for GALE systems and other similar TBIE systems. We also hope that the EDIE framework will be further utilized to explore various innovations related to user study design and utility evaluation parameters.

6.3.2. Formative Evaluation for Multilingual Multimedia Search and Sense-Making

Authors: Douglas W. Oard, Judith Klavans, Dagobert Soergel, Pengyi Zhang, Peter Brusilovsky, Daqing He, Tomasz Loboda, and Leiming Qian

6.3.2.1. Introduction

This section describes the iterative development of Rosetta, a distillation system for multilingual (Arabic, Chinese, English, Spanish) multimedia (television, Web) streaming content. The process involved close collaboration between the IBM T.J. Watson Research Center (IBM), the University of Maryland (UMD), the University of Pittsburgh (Pitt), and Carnegie Mellon University (CMU). The fundamental challenge was process-system co-design: as new technical capabilities were introduced, new work processes were sometimes needed to best leverage those capabilities; those new work processes in turn help to identify new technical requirements. The result was a virtuous cycle of innovation.

Rosetta integrates six key technologies to support search and sense-making: Automatic Speech Recognition (ASR), Machine Translation (MT), Information Extraction (IE), Information Retrieval (IR), User Modeling (UM), answer pinpointing for Question Answering (QA), and summary generation from structured knowledge representations. The focus of the work reported in this section was on design innovation and process innovation for integrated architectures (e.g., ASR→MT→IR→IE→QA), in which GALE technologies are used together to accomplish challenging and realistic tasks.

6.3.2.2. The Rosetta System

The Rosetta System consists of three major components: the data collection sub-system, the data processing pipeline, which integrates the ASR, MT and IE components, and the Web application, which integrates IR, UM, QA and summarization components.

Rosetta captures foreign language news broadcasts from Dish Network and performs daily crawl of foreign language Web sites. Rosetta's data processing pipeline is developed on top of IBM's Unstructured Information Management Architecture (UIMA) platform and features a series of data processing components (or "annotators" in UIMA terminology) analyzing input data sequentially. The list of major data annotators in Rosetta includes: ASR, MT, and IE. The Rosetta Web application consists of a browser-based end-user client GUI, and a J2EE back-end server Web application. A browser-based client design was chosen for maximum cross-platform compatibility. Extensive use of Web 2.0 technologies such as AJAX (Asynchronous JavaScript and XML) gives the client a desktop-like feel and enables it to support advanced features not commonly found in conventional Web applications.

Rosetta provides the user with four major modes for information access, each of which can be selected using a tab in the upper left of the screen shots in the following figures. These major modes are organized from left to right in an order that we think of as going from most focused to most general.

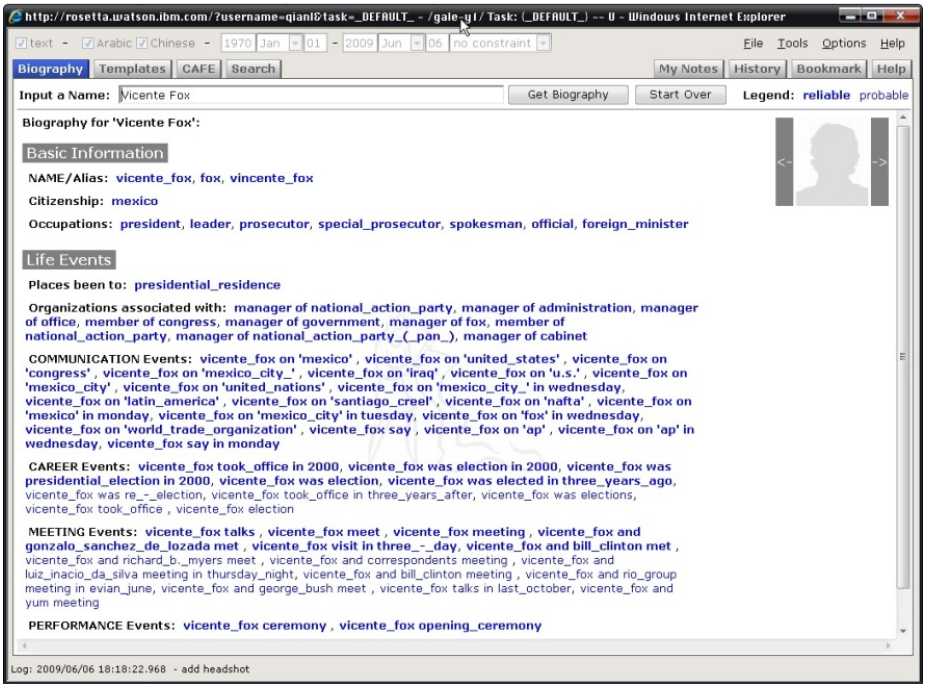

Figure 6.32: The biography mode

The **Biography** and **Templates** modes are driven by the full ASR, MT, IE, QA pipeline. As Figure 6.32 shows, the Biography mode produces an extensive structured display of all information about a person that is known to the system, with drill-down available to a supporting document for each item. In the Template mode, the user can choose from a set of 15 predefined *question templates*, filling in required arguments. The output is a list of snippets that are selected and ordered, based on their likelihood of providing the answer. The **CAFÉ** mode exposes the user to GALE user modeling technology (Yang *et al.* 2007). Rosetta allows each user to create multiple "tasks," each corresponding to a project that the user is working on over time. The search history, document bookmarks, and notes are all task-specific. When users access documents in Rosetta, their actions are tracked and made visible by the system as "footprints" that can act as reminders. For example, a document that has been visited will be highlighted differently when it shows up again, and users can also select a span of text and add it to the "Notes" tab. CMU's Adaptive Filtering Engine (CAFÉ) also uses this evidence to iteratively improve search results. The user first issues a simple query to obtain a list of result snippets from the CAFÉ engine. He can then provide explicit feedback to the CAFÉ engine by rating individual passages as useful (which results in adding it to the notes), irrelevant (which immediately removes it from the view), or redundant (by marking it as "not new"). The CAFÉ engine also accumulates implicit feedback when the user is working in other modes. For example, if a user finds a document using the Search mode and adds some text from this document to his notes, which will be interpreted as "relevant" feedback by CAFÉ.

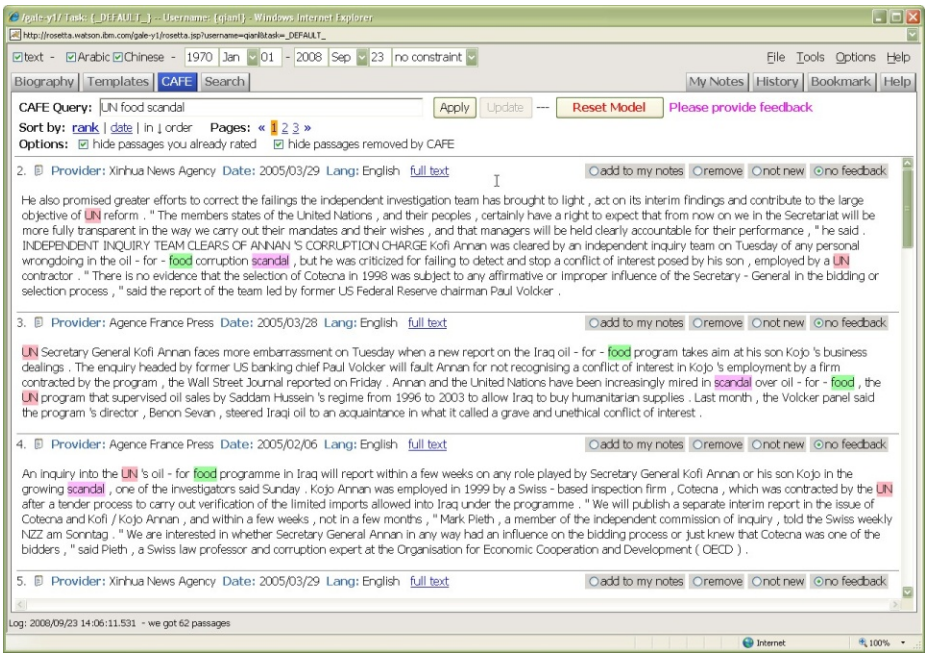

Figure 6.33: The CAFÉ Mode

The feedback helps CAFÉ refine the system's internal model of the user's task, which can then result in identification of additional useful passages. Figure 6.33 shows the output after just a few feedback iterations. The original query was "UN food scandal;" after adaptive filtering, most of the top passages retrieved are about the food scandal and the role of U.N. secretary Kofi Annan and his son, which in this case are right on target.

The **Search** mode utilizes the GALE ASR, MT, and IR technologies. It allows the user to do conventional keyword search using English queries to find foreign-language multimedia content. Among the four tabs, Search is probably most familiar to most users. For video results, the user can stream the video online with English closed caption, look at the document in a "Storyboard" type of view, and download the video file together with its caption for local playback. For Web results, the user can look at the non-structured textual content, or submit the cached Web page for on-demand translation.

Rosetta has an elaborate event logging mechanism, able to capture nearly every user action (down to mouse movement and keyboard strokes) and log those events and the associated event context at the server. The events are logged asynchronously in the background so that they do not impact the system's performance. This functionality makes Rosetta well suited for user studies.

One of the most popular features is the ability to create a Microsoft Word report directly from inside Rosetta. The user can select a span of text or an image and click a button to create a new report in Word, he can further add more material to the report and edit the report freely, and each snippet of information contained in the report also links back to the original document. The user never has to leave Rosetta to compile a report.

6.3.2.3. Formative Evaluation

We conducted 52 formative user study sessions at UMD and Pitt between June 2006 and May 2007. In this section we describe those studies in four clusters.

Initial Eye-tracking Experiments

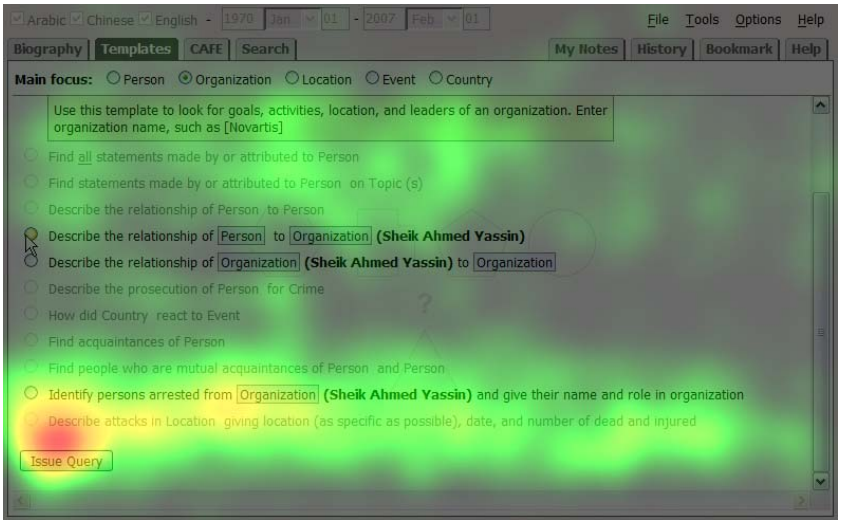

Figure 6.34: Heat map showing gaze locations for early version of Templates mode.

In an attempt to gain insight into the way users employed the CAFÉ and Templates modes, we used eye tracking to collect user gaze data in several user studies. We employed the Tobii 1750 remote eye-tracking device (http://www.tobii.com), which introduced virtually no intrusion in the subject's working environment. The exploratory insights we gained resonated with the more complete and convincing body of evidence described in the following subjections. For example, Figure 6.34 depicts a fixation-count "heat map" for an early version of Templates mode, illustrating a case in which the user may have paid little attention to the control widgets near the top of the screen (in the dark grey "cold" region) because they were placed too far from the focal region (around the Issue Query button in the lower left, where the circular red "hot" region is).

Formative Evaluation for Template QA

In order to evaluate the utility of the Templates mode we ran a controlled experiment that compared it with a Search mode based on the Indri retrieval engine. The study was run at the University of Pittsburgh and involved ten graduate students. As surrogates for intelligence analysts, we recruited study participants who were pursuing advanced degrees in Information Sciences, were native English speakers, and had either 1) taken a masters-level course in information retrieval, or 2) had an extensive background in journalism. Each participant was asked to perform two 40-minute search tasks with Rosetta in a counterbalanced design, one task using only the Search mode and one task using only the Templates mode. Prior to starting the tasks, each subject received 15

minutes of training on the Templates mode. All subjects were familiar with generic search systems that are similar to the Search mode. Each task required the participant to interact with a large collection of documents to find answers to several questions that had been provided to them in written form. The outcome of their work was a collection of text passages that they had marked as relevant. An analyst could later use these passages to compile a report, but that was not done as a part of this study.

We found that the rate of accumulation of putatively relevant text passages for subjects using Templates was significantly higher than for subjects using Search. That supports the view that it was easier to find relevant pieces of information when using Templates. Further, we found that subjects issued significantly more queries when using Search. As a result, the inter-query time was significantly longer for the Templates mode, which provides evidence that at average results of Template search brought more interesting information for users to analyze. Query log analysis indicated that this difference reached a statistically reliable level after 20 minutes. Therefore, for similar studies, we recommend not employing short sessions. We hired three annotators to judge the topical relevance of each text passage collected by our participants. Our comparison of the two modes with respect to the relevance of the results also favored the Templates mode.

Our early version of the Templates mode offered a choice between 16 question types. We found that four question types accounted for 77% of all questions (and that six question types accounted for 89% of all questions). This is an instance of the well-known Pareto (80/20) principle. Of course, the specific questions involved surely depended on the specific tasks that we assigned. Nevertheless, this serves to remind us that identifying the most heavily used functions can help to focus optimization effort.

Two additional formative evaluation studies that were focused on CAFÉ are described elsewhere in this chapter.

Scenario-based Integrated Evaluation

Well designed studies with static collections are particularly useful for comparing the utility of alternative components, but for understanding how complete systems will actually be used we wanted to get as close to the real setting as possible. In our case, this called for additional user studies using live content. We therefore chose a study design in which tasks were designed in near real-time and executed simultaneously by all participants (i.e., without counterbalancing). Our principal goal was to explore process system co-design: the coupled iteration between new system capabilities and new ways of using the resulting system.

A cohort of information studies graduate students at the University of Maryland was trained as surrogates for intelligence analysts. An initial training session was supplemented with brief lectures at the start of each session on aspects of an analyst's tasks and methods that were relevant to that day's scenario. A second cohort of graduate students performed observational data collection and analysis. Observation notes were automatically integrated with system logs in real time; complementary data were collected from post-session interviews, and both structured questionnaires and brief free-

style reaction papers from participants. Sessions were generally scheduled about two weeks apart, thus allowing some time to implement system improvements. Summary reports from each session, a requirements tracking database, and weekly teleconferences facilitated information flow between the development and evaluation teams. These studies provided important guidance for user interface refinement, which has been previously reported (Zhang *et al.* 2007). Here, we focus on what was learned about how users actually employ integrated GALE technologies.

Every session that we ran focused principally on Arabic and/or Chinese content (although during training we did make some use of English content). In some cases, the translations were based on ASR transcripts; in other cases the translations were of Web content. We can, therefore, think holistically of our system as a device for assessing the utility of ASR, MT, or ASR-MT cascades for certain tasks. Consistent with previously reported results, we found that cascading present ASR and MT systems often proved to be adequate for tasks that involved identifying topics and sometimes proved to be adequate for tasks that involved detection of specific factual content, and rarely proved to be adequate for tasks that require detection of nuance. We also found that our participants had more difficulty making use of present Chinese MT systems than present Arabic MT systems, which corresponds well to the reported differences in Translation Error Rate (TER) results in machine transition evaluations.

Remarkably often, our participants proved to be adept at using context and (when present) multiple media (e.g., photos or videos) to infer the correct meaning of misrecognized and/or mistranslated terms. For example, several spellings of the same name could be recognized if they appeared next to pictures of the same person. Moreover, our participants learned that using systematically misrecognized and/or mistranslated terms as query terms could sometimes improve results in the Search mode. For example, in a task scenario about Iran's nuclear program, a participant recognized the mistranslated expression “nuclear file” as actually meaning “nuclear program” and used the misrecognized term with good results in subsequent queries. Similarly, a participant recognized “berating women” as the intended meaning of what had been misrecognized as “brainstorming women.”

Our participants were repeatedly observed to spontaneously post-edit incorrect translation results, either using the Notes facility or the Microsoft Word editor in Rosetta. Moreover, when asked to suggest system enhancements, they repeatedly requested the ability to provide feedback to the system about mistranslations. Of course, user study participants may be more focused on the system design, so we do not know whether this preference would also be present in operational users of such systems. Even if the translation system were not adapted, some benefit to future searchers might also accrue from simply memorizing the edits to individual documents. Rosetta now includes this capability.

Earlier user studies had shown that fatigue could become a problem in complex tasks when translations were not easily readable. For this reason, Rosetta includes several features that are designed to help focus the user on specific information (e.g., the snippets in the CAFÉ and Templates mode, and highlighting query terms in Search mode result summaries). These generally proved to be useful, and we believe that investigating additional techniques for helping to guide the user’s focus would be useful.

One unexpected and potentially important result was that we repeatedly observed participants engaged in a behavior akin to what journalists might think of as "fact checking from multiple sources." In this case, they were not seeking to verify the correctness of the report, but rather the correctness of the translation. Our participants quickly learned that the same pre-translation input typically resulted in the same post-translation output, so they typically dismissed exactly identical output as unhelpful duplication. When they saw the same fact paraphrased in a different way, however, they tended to develop greater confidence in both translations. We are now exploring how we might use this type of cross-source evidence as a basis for unsupervised estimation of translation quality.

We generally left our participants free to use whichever system modes they found most useful, although in some cases we did ask them to use specific modes in order to gain insight into specific usability issues. The Search mode was initially most familiar to our participants, and not surprisingly they initially chose to use it most often. This persisted over time, and an interesting trend emerged: the further to the right the tab was (i.e., the more general the tool), the more often it was used. Such an outcome would not have been predicted from what we had seen earlier in more controlled settings, where Templates and CAFÉ had both shown substantial advantages. Interview and self-report data suggest that the cause was not a reluctance to use new tools; indeed, our participants were all volunteers, and hence could reasonably be described as early adopters of new technologies. Rather, the cause of this effect seems to be twofold. First, more specialized tools are naturally useful in fewer situations. For example, we had a template asking about the relationship between two people (which might be an event), but no template asking about the relationship between two events (which might be a person). Second, our least sophisticated tool (Search) had to expose the most context to the user (for the simple reason that the user had to do more of the job). As our tools became progressively more sophisticated, we adjusted the focus-vs. context tradeoff more in favor of focus (although with the full context still available on demand). This became most extreme in the biography mode, where the initial display included a large number of very short results. With the context less easily accessible, our participants seemed to have more trouble assessing the quality of the system's results.

Because information seeking is often an iterative process, this difficulty with assessing results may have had the additional effect of making exploratory use of new capabilities somewhat more challenging as well. Indeed, we saw some evidence of that from the way in which participants used CAFÉ. Positive feedback was often provided, and our participants often seemed to be able to develop some degree of confidence from seeing the results of positive feedback. Negative feedback was used far less often, and our users expressed serious reservations about using it in interviews and self-report data. Their reasoning was generally that when they used positive feedback they could see what they were then getting as future results, but when they used negative feedback they had no way of seeing what they then did not get as future results. In this case, the context that they would need extends beyond individual documents – they would also need some way of skimming documents that would have been displayed to them had the negative feedback not been provided.

6.3.2.4. Conclusion and Future Work

Looking back over our full set of studies, we can draw several broad conclusions. More obviously, and most importantly, there are things that we could learn through user studies that we could not have seen as easily (if at all) in "batch-mode" experiments. Just as we use TER as a predictor of Human-assessed Translation Error Rate (HTER) during development, we should be using HTER as a predictor of actual utility during development, but then we should check that prediction from time to time using actual user studies. User studies are more expensive than HTER (which is in turn far more expensive than automatic TER scoring), so we need not do user studies every year. But unless we do user studies periodically, it would be hard to know for sure whether we're headed in the right direction.

Another conclusion that we can draw is that user studies are not all created equal. Highly structured user studies are useful early in a development process when the questions focus on capability, but studies situated in setting that are as representative as possible of the envisioned application are also needed at some point if we are to iterate between system design and discovery of the most effective way to use the resulting systems. Indeed, a sequence of increasingly realistic studies may be needed if we are to optimally balance cost and insight. For example, when actual intelligence analysts later used our systems, we learned that some of the behaviors that we had observed with students (e.g., seeking background information) were less common with actual analysis (who of course started with more background).

Ultimately, the most important benefit of our formative evaluation process may have been the bridges that we built between research communities. Just as GALE has brought speech and translation researchers together to study how best to translate spoken content, we have brought component developers together with system developers, and system developers together with process developers (i.e., real users). After all, the ultimate cascade does not end with transcription, translation, retrieval, extraction, or summarization; it ends with use.

6.3.3. Utility Assessment for the GALE Program

Authors: Connie Fournelle, Jon Holbrook Dan Hunter, and William Salter

6.3.3.1. Introduction

The systems developed under GALE aim to improve timeliness and quality of information available to users via a comprehensive system that supports transcription, translation, and distillation of multiple languages in text and audio formats. The purpose of the Utility Evaluation is to qualitatively and quantitatively assess the performance and utility of the GALE systems in providing these capabilities to the user as he or she interacts with the system. We describe the task definition and environment for the Utility Evaluation, the scenarios used for testing, and the requests for information (RFI's) issued to support them. We further describe how we captured user feedback through written and group feedback sessions, and user performance based on evaluation of final products generated by analysts using GALE systems.

The findings of the user-centric Utility Evaluation complement the findings of the technology-centric Evaluation, which measures the amount of on-target information a GALE system is able to produce in response to a specific, formatted query, and compares GALE system performance to humans completing the same task without GALE tools. The utility of an automated system can be evaluated in terms of the performance of a human user when paired with that system to perform realistic tasks, which may in turn be influenced by a number of factors, including the ease with which that system can be used, as well as, users' desire and willingness to use the system.

6.3.3.2. Utility Evaluation Goals and Methodology

Evaluation Goals

In the context of this evaluation, the term "user" refers to both DoD intelligence analysts and operations personnel. System users were DIA Lead Analysts employed by BAE-Information Technology (BAE -IT) – operational analysts with at least five years of experience with independent project-level responsibility under general supervision – who used the GALE systems during the testing sessions of the evaluation, as well as, the baseline system. The Utility Evaluation was not intended to establish a performance baseline of these analysts, but rather to provide users with exposure to GALE technology, and create a venue for them to give early feedback on the effectiveness of the technology; this feedback can consequently have a direct impact on the research and development activity as it is underway, rather than waiting for the operational needs and research direction to potentially diverge as the program progresses.

Utility Methodology

There were two types of roles for participants in the evaluation: system users and judges. System users were DIA Lead Analysts (mean age = 33 years, SD = 10.0) employed by BAE Systems– operational analysts with an average of 8.7 years of experience (SD = 6.8) performing intelligence-related work – who interacted with the GALE systems during the evaluation. Sixteen system users participated in the Utility Evaluation each day. Thirteen of the system users had previous military experience. All system users were fluent speakers of English; none had any familiarity with spoken or written Arabic or Chinese. The first day of the evaluation familiarized the system users with the goals of the GALE program and the systems under evaluation; on days two-through-five, users employed the GALE systems and a baseline system to generate reports. The reports were evaluated by judges drawn from two different pools of people: BAE Systems principal analysts – analysts with at least eight years of technical leadership including assisting in decision-making in major program operations and supervisory responsibilities – and senior Operations Specialists – operations officers with recent experience in Iraq/Afghanistan. Five judges (mean age = 46 years, SD = 12) participated in the Utility Evaluation (three Principal Analysts and two Operations Specialists).

During the Utility Evaluation, system users were provided with scenarios that described the context, scope, and requirements for the tasks they were to perform using the provided systems. All scenarios were generated in conjunction with or under the guidance of intelligence analysts.

All-source intelligence analysts typically operate within specific domains of expertise (e.g., geo-graphic region, politics, economics, etc.). Because the intelligence analysts serving as system users in the Utility Evaluation were presented with scenarios potentially outside their domains of expertise, each scenario included a section that outlined information related to the scenario topic that the analysts might be assumed to already know if operating in their real working environment.

Each scenario included one or more Requests for Information (RFIs) that define the task(s) that the analysts were asked to perform and the product(s) they were asked to generate. These products, based on those intelligence analysts produce in their everyday jobs, involved writing short (½- to 2- page) prose briefings (e.g., "executive summaries").

All scenarios were limited to the 1.5-hour testing sessions. That is, a single scenario – with multiple RFIs – was to be completed during each testing session.

Each system user interacted with one system per day over four days. In addition to the three GALE systems under evaluation, users employed a baseline system developed using commercial-off-the-shelf (COTS) technologies, including the desktop search tool Copernic. Users' work stations included Microsoft Word for the generation of work products. Each of the work stations was connected via a remote connection to offsite software and databases. Connection to other remote sites (e.g., Google, Wikipedia, etc.) was not permitted at any point during the testing sessions. The TDT5 and GALE Y1 corpora (LDC2004E41 and LDC2006E21, respectively), available from the Linguistic Data Consortium, were used in the testing sessions of the Utility Evaluation.

6.3.3.3. Evaluation Measures

Measuring the utility of any system requires identifying ways in which that system aids the user performing her real-world job. Because human behavior in real-world settings is complex, we evaluated a range of quantitative and qualitative measures of performance. The performance measures can be categorized into three broad classes:

- *System-based measures:* these include total time spent in product generation, proportion of time spent in System versus Text Editor (e.g., MS Word), frequency of switches between System and Text Editor, and changes in the ratio of time spent in System versus Text Editor over the time course of product generation.
- *Self-report measures:* All system users participated in debriefing sessions each day of the evaluation that included structured interviews and questionnaires concerning their experience with the system they used that day.
- *Observer-based measures:* Human factors specialists observed users' interactions with the systems and identified issues with users' interactions. Judges (described above in Section 2.2) rated each work product on Likert-scale measures of whether the work product was supported by evidence, thorough, organized, relevant, compelling, and actionable. Judges were not provided with information about which users or which GALE systems contributed to a particular work product.

6.3.3.4. Utility Evaluation Results

Product Evaluations

Five judges (i.e., three senior analysts and two military operations specialists) rated each work product using a seven-item instrument that measured the degree to which the work products were supported by evidence, thorough, organized, relevant, compelling, and actionable, as well overall quality. These judges were also asked to rank order all of the products for each RFI from best to worst. All ratings/rankings were performed independently of the other judges, and were performed without any knowledge of which analyst generated the work product or which system the analyst used.

A statistical analysis of the ratings revealed no statistically significant difference (at the level $p < 0.05$) among the systems used for either the simple mean or weighted mean of the ratings of the work products. (The weighted mean combined ratings on highly correlated variables.)

System users were divided into two groups based on their years spent doing intelligence-related work. A median split along this dimension yielded a low experience group (m = 3.9 years of intel experience, SD = 1.5) and a high experience group (m = 12.9 years of intel experience, SD = 6.9). Comparisons of overall product ratings for products generated using the Baseline system revealed a marginal effect of system user experience, $F(1,28) = 3.9$, $p = .058$, with products from high experience users (4.23) rated higher than products from low experience users (3.48). However, among GALE systems, no differences were found between ratings of products from high and low experience users, $F < 1$. Given that the Baseline system is similar to the search tools that analysts regularly use, it is not surprising that high experience users were better able to utilize the Baseline system than low experience users. Among the GALE systems, the finding of no effect of user experience on product ratings may indicate a) that the GALE systems function in a way that is sufficiently different from the search tools that analysts regularly use that user experience had little impact of performance in the evaluation, and/or b) the GALE systems provided sufficient support to the less experienced user to boost performance closer to that of more experienced users. This issue is one that may warrant further study in future evaluations.

Usability Evaluations

After completion of each scenario, system users completed a 14-item questionnaire designed to address various aspects of system usability, or the ease with which users felt they were able to interact with the systems. For each item on the questionnaire, system users were asked to rate on a 7-point scale the system they had just used to complete that scenario.

Usefulness Evaluations

After each testing session, participants completed a 12-item questionnaire designed to assess the usefulness of the systems, that is, the degree to which system features would be valuable to the users' job performance. Users were asked to rate the usefulness on 7-point scales of various aspects of the system they had just used. Analysis of overall usefulness ratings revealed a significant effect of System, $F(3,56) = 7.2$, $p < .001$. Two of the GALE

systems were rater higher than Baseline (3.26) with scores of (5.39) and (4.64), p = .001 and p = .02, respectively.

Overall Results

Question	Low Anchor (I)	Mean	High Anchor (7)
Would be helpful in doing my job	Not helpful	5.77[a]	Very helpful
Ease of finding info for products	Baseline easier	5.46[a]	GALE easier
Confidence in finding all available info	Not confident	4.40	Very confident
Ease drill-down/ Follow-up search results	Not easy	5.03[a]	Very easy
Ease of using results to prepare products	Not easy	5.63[a]	Very easy
Easier to use to find information	Current system	4.23	GALE
Creates higher workload	Current system	4.07	GALE
Enables finding more relevant info	Baseline	5.54[a]	GALE
Enables higher quality Product generation	Baseline	5.64[a]	GALE

Table 6.9: User evaluation of GALE systems (a p <.05 in comparison with Midpoint of scale).

At the conclusion of testing, system users were asked nine questions designed to capture users' impressions of how the GALE systems might help intelligence analysts perform their jobs, as well as, comparisons of the GALE systems with users' current search tools and with the Baseline system. For the purpose of answering these questions, users were asked to think of the GALE systems in terms of the most effective of the three systems they used during the evaluation.

The results are shown in Table 6.9, below. Responses were on a scale from one-to-seven inclusive, with the low and high anchors indicated for each questions. These results indicated that users believed that GALE technologies would be helpful in doing their jobs, were generally easy to use, and were more effective than the Baseline system. Users rated the GALE systems equivalent to their current systems in terms of ease of use and overall work-load. Users' confidence in the GALE systems to find all available information did not differ from the scale midpoint, which is consistent with findings for this same question from the System Trust instrument, on which the GALE systems were rated as equivalent to or below Baseline. Users' comments about the overall potential of the GALE technologies were generally positive:

> "*The concept of the systems are really good and should help the open-source community when it's done.*"

"All of the systems have features that should be part of the final product."

A feature of the GALE systems that was frequently mentioned as a valuable feature was the ability to do template directed search in addition to key-word search. However, users also indicated areas for improvement, including expanding the flexibility of the search options, and designing the interface to maximize efficiency of use.

6.3.3.5. Related Work

Gonzalo and Oard (2004) describe a set of experiments conducted by five teams using a common experimental design to test hypotheses about the utility of cross-lingual information retrieval systems. The experiments bear some similarity to the utility evaluation described in this section, but there are important differences.

The experiments are similar to the one described here in that each experiment involved the use of different IR systems by a group of subjects and the answers produced by the subjects were evaluated by an independent group of judges (or "assessors"). Significant differences are that the queries were very specific (e.g., "What year was Thomas Mann awarded the Nobel Prize?") rather than open-ended as were the queries used in the GALE Utility Evaluation; the systems compared were in general variations on automated IR systems so that a baseline for standard IR retrieval tools such as Google was not established; and finally, the answers produced by the subjects were evaluated on the basis of whether they were correct (valid and supported by documentation), un-supported (valid, but not supported by documentation), or incorrect. The GALE utility evaluation in contrast allowed the judges to additionally express judgments about the degree to which a work product is relevant, well organized, and actionable, and it collected evaluations from the users themselves on their experience with the different systems.

6.3.3.6. Conclusions and Recommendations

We have described the testing framework for the GALE utility evaluation and some of the salient results of that evaluation. In general, the results reflect favorably on the GALE systems, especially in light of users' initial unfamiliarity with the systems and the limited training time available. We conclude with some recommendations for tool design and future evaluations that are suggested by our analysis of the evaluation results.

In terms of tool design, it would be beneficial to combine key word search capabilities present in current search tools used by analysts with the template-based search capabilities of the GALE systems, since the two kinds of search capabilities turned out to have complementary strengths and weaknesses. Template-based search showed strength when there was a large amount of information relevant to a given query since it allowed for more focused search. When there was little information relevant to a query, however, it proved difficult for users to discover that with template-based search because they could not be sure that all possibilities for filling in template parameters had been exhausted. Having both key word and template-based search capabilities in the same tool would enable analysts to search more efficiently without wasting time on futile searches.

Another design issue that emerged from user feedback was that of tuning the system to adjust the level of irrelevant information returned. Many users expressed distrust in a system if it returned too much irrelevant information. One solution to this problem would be to make the sensitivity of the system an adjustable parameter under the control of the user. Another would be for the system to associate a relevance level with returned information; if the relevance level is strongly correlated with the user's own judgments of relevance, which would increase the user's trust in the system.

Even at the current state of development, analysts rated the GALE systems as useful overall. However, users do not perceive technologies in the same way as technology developers. Specifically, it is not always apparent to system users which aspects of the system are the most difficult to engineer or which aspects of the system are the primary focus of the developer's attention and resources. Rather, users' experience with a system is primarily through its user interface, and fundamental under-the-hood technologies may go completely unnoticed and/or unappreciated by system users. This being the case, there may be a number of low-cost and low-effort features that could be implemented in a GALE tool that, while not the primary focus of the GALE program, can dramatically impact users' satisfaction with and willingness to use the system, which will facilitate transition of the systems into operational settings.

Similarly, when users are evaluating a new technology designed to support or extend their current tasks, to the extent that the tool provides functionality that overlaps with their current tools, users expect the new technology to provide at a minimum all of the functionality of their existing tools in addition to any new features. While this may seem somewhat unreasonable to the developer of the new technology, user satisfaction with any technology designed to replace existing technology requires that the new technology not be viewed as a step backward. Specifically, in the Utility Evaluation users expressed dissatisfaction with the keyword search capabilities of the GALE tools (e.g., no or limited ability to use wildcards). Although keyword search is not the focus of the GALE program, if GALE technologies are intended to replace current keyword search tools, The GALE technologies must be able to perform all of the functions of current keyword search tools. An important distinction to make is that new technologies do not have to instantiate this functionality in the same way as existing technology, but providing that functionality is an important key to successful transition.

Finally, with regard to possible future utility evaluations, user comments on training time and training content were offered frequently and training was mentioned in all four broad categories of the post-scenario surveys (e.g., Workload, Usability, Usefulness, and Trust), as well as, the End-of-Evaluation survey, Daily Debriefs, and Group Debrief. These comments, as well as observations by the evaluation team, indicate clearly that analysts thought they might have been more successful in answering the RFIs had they received more training.

To facilitate training in future utility evaluations, we suggest a more specified approach to the training. Training should include a brief lecture-style introduction, but the majority of the training session should comprise supervised hands-on interaction for the analysts.

References

Ahn, J., P. Brusilovsky, D. He, J. Grady, and Q. Li. 2008. Personalized Web Exploration with Task Models. *Proceedings of the International World Wide Web Conference*. Beijing, China.

Allan, J., ed. 2002. *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers.

Allan, J. 2003. HARD Track Overview in TREC 2003 High Accuracy Retrieval from Documents. *Proceedings of the 12th Text Retrieval Conference.* Gaithersburg, MD.

Bemish, N. 2008. Broadcast Monitoring Tools Open Doors to Other Cultures, Information. *Communiqué - A Defense Intelligence Agency Publication* 21(2).

Billa, J., M. Noamany, A. Srivastava, D. Liu, R. Stone, J. Xu, J. Makhoul, and F. Kubala. 2002. Audio Indexing of Arabic Broadcast News. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* pp. 15-18. Orlando, FL.

Chandra, A. K., D. C. Kozen, and L. J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery* 28(1)pp. 114-133.

Colbath, S., F. Kubala, D. Liu, and A. Srivastava. 2000. Spoken Documents: Creating Searchable Archives from Continuous Audio. *Proceedings of the 33rd Hawaii International Conference on System Sciences* vol.3. Maui, HI.

Collins, M. 2002. Discriminative Training Methods For Hidden Markov Models: Theory And Experiments With Perceptron Algorithms. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) 2002* pp.1–8. Philadelphia, PA.

Ferrucci, D. and A. Lally. 2004. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Natural Language Engineering* 10(3-4): pp. 327–348.

Fiscus, J. G. 1997. A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). *Proceedings of the Institute of Electrical and Electronics Engineers Automatic Speech Recognition and Understanding Workshop* pp. 347–354. Santa Barbara, CA.

Gilmore, G. J. 2007. Modern Missions Rely on Languages, Cultural Awareness, Official Says. *American Forces Press Service*, April 27.

Gonzalo, J, and D. W. Oard. 2004. iCLEF 2004 Track Overview: Pilot Experiments in Interactive Cross-Language Question Answering. *Proceedings of*

the Multilingual Information Access for Text, Speech, and Images, Fifth Workshop of the Cross-Language Evaluation Forum pp. 310-322. Revised Selected Papers Series, Springer-Verlag, LNCS (3491) pp. 310-322. Bath, UK.

He, D., and D. Demner-Fushman. 2003. HARD Experiment at Maryland: from Need Negotiation to Automated HARD Process. *Proceedings of the 12th Text Retrieval Conference* pp. 707-714. Gaithersburg, MD.

He, D., P. Brusilovsky, J. Ahn., J. Grady, R. Farzan, Y. Peng, Y. Yang, and M. Rogati. 2008. An Evaluation of Adaptive Filtering in the Context of Realistic Task-Based Information Exploration. *Information Processing and Management* 44(2): pp. 511-533.

He, D., P. Brusiloviksy, J. Grady, Q. Li, and J-W. Ahn. 2007. How Up-to-date Should It Be? The Value of Instant Profiling and Adaptation in Information Filtering. *Proceedings of the International Conference on Web Intelligence* pp.699-705. Silicon Valley, CA.

He, D., P. Brusilovsky, J-W. Ahn, J. Grady, R. Farzan, Y. Peng, Y. Yang, and M. Rogati. 2008. An Evaluation of Adaptive Filtering in the Context of Realistic Task-Based Information Exploration. *Information Processing and Management* 44(2): pp. 511-533.

Huang, J., and G. Zweig. 2002. Maximum Entropy Model for Punctuation Annotation from Speech. *Proceedings of the International Conference on Speech and Language Processing(ICSLP)* pp. 917-920.

Jayaraman, S., and A. Lavie. 2005. Multi-engine Machine Translation Guided by Explicit Word Matching. *Proceedings of the 10th Annual Conference of the European Association for Machine Translation* pp.143–152. Budapest, Hungary.

Jones , D. A., and W. Shen. 2006. Two New Experiments for ILR-Based MT Evaluation. *Proceedings of the Association for Machine Translation in the Americas*. Boston, MA.

Kubala, F., S. Colbath, D. Liu, A. Srivastava, and J. Makhoul. 2000. Integrated Technologies for Indexing Spoken Language. *Communications of the Association for Computing Machinery* 43: pp. 48–56.

Kubala, F., S. Colbath, D. Liu, and J. Makhoul. 1999. Rough'n'Ready: a Meeting Recorder and Browser. *Association for Computing Machinery Computing Surveys* 31(2es).

Liu, D., and F. Kubala. 1999. Fast Speaker Change Detection for Broadcast News Transcription and Indexing. *Proceedings of the 6th European Conference on Speech Communication and Technology (EuroSpeech*) 3: pp. 1031-1034.

Liu, D., D. Kiecza, A. Srivastava, and F. Kubala. 2005. Online Speaker Adaptation and Tracking for Real-Time Speech Recognition. *Proceedings of the International Speech Communication Association Conference (Interspeech)* pp. 281-284.

Liu, D., J. Ma, D. Xu, A.Srivastava, and F. Kubala. 2002. Real-Time Rich-Content Transcription Of Chinese Broadcast News. *Proceedings of the International Conference on Speech and Language Processing (ICSLP)* pp. 1981-1984.

Liu, Y., E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, B. Peskin, and M. Harper. 2004. The ICSI-SRIUW Metadata Extraction System. *Proceedings of the International Conference on Spoken Language Processing* pp. 577-580. Jeju Island, Korea.

Makhoul, J., F. Kubala, F., T. Leek, D. Liu, L. Nguyen, R. Schwartz, and A. Srivastava. 2000. Speech and Language Technologies For Audio Indexing. *Proceedings of the IEEE* 88: pp. 1338–1353.

Makhoul, J., F. Kubala, R. Schwartz, and R. Weischedel. 1999. Performance Measures for Information Extraction. *Proceedings of the DARPA Broadcast News Workshop* pp. 249-252. Herndon, VA.

Marchionini, G., and B. Shneiderman. 1988. Finding Facts Vs. Browsing Knowledge in Hypertext Systems. *IEEE Computer* 21(1): pp. 70-79.

Matusov, E., A. Mauser, and H. Ney. 2006. Automatic Sentence Segmentation and Punctuation Prediction for Spoken Language Translation. *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)* pp. 158-165. Kyoto, Japan.

Murthy, U., J. F. Pitrelli, G. Ramaswamy, M. Franz, and B. L. Lewis. 2008. A Methodology and Tool Suite for Evaluation of Accuracy of Interoperating Statistical Natural Language Processing Engines. *Proceedings of The International Speech Communication Association Conference (Interspeech)* pp. 2066–2069. Brisbane, Australia.

Nyberg, E., E. Riebling, R. C. Wang, and R. Frederking. 2008. Integrating a Natural Language Message Pre-processor with UIMA. *Proceedings of LREC Workshop on Enhanced Operability for Large HLT Systems*. Marrakech, Morocco.

Pirolli, P., and S. Card. 2005. The Sensemaking Process and Leverage Points for Analyst Technology as Identified through Cognitive Task Analysis. *Proceedings of the International Conference on Intelligence Analysis.* McLean, VA.

Pitrelli, J. F., B. L. Lewis, E. A. Epstein, D. Kiecza, M. Franz, J. L. Quinn, G. Ramaswamy, A. Srivastava, and P. Virga. 2008a. Aggregating Distributed STT, MT, and Information Extraction Engines: The GALE Interoperability Demo System. *Proceedings of The International Speech Communication Association Conference (INTERSPEECH)* pp. 2743–2746. Brisbane, Australia.

Pitrelli, J. F., B. L. Lewis, E. A. Epstein, J. L. Quinn, and G. Ramaswamy. 2008b. A Data Format Enabling Interoperation of Speech Recognition, Translation, and Information Extraction Engines: The GALE Type System. *Proceedings of The International Speech Communication Association Conference (Interspeech)* pp. 1654–1657. Brisbane, Australia.

Rosti, A., N. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. 2007. Combining Outputs from Multiple Machine Translation Systems. *Proceedings of The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)* pp. 228–235. Rochester, NY.

Shepard, S., S. Colbath, K. Egan, and F. Kubala. 2002. Oasis Translator's Aide. *Proceedings of the Second international Conference on Human Language Technology Research* pp. 389-390. San Diego, CA.

Srivastava, A., and F. Kubala. 2003. Sentence Boundary Detection in Arabic Speech. *Proceedings of the European Conference on Speech Communication and Technology (EuroSpeech*) pp. 949-952. Geneva, Switzerland.

Stolcke, A., and E. Shriberg. 1996. Automatic Linguistic Segmentation Of Conversational Speech. *Proceedings of the International Conference on Spoken Language Processing* pp. 1005–1008. Philadelphia, PA.

Stolcke, A., E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, M., G. Tur, and Y. Lu. 1998. Automatic Detection Of Sentence Boundaries And Disfluencies Based On Recognized Words. *Proceedings of International Conference on Spoken Language Processing (ICLP)* pp. 2247-2250. Sydney, Australia.

White, R. W., G. Muresan, and G. Marchionini. 2006. Evaluating Exploratory Search Systems. *Proceedings of the Association for Computing Machinery's Special Interest Group on Information Retrieval Conference.* Seattle, WA.

Yang, Y., A. Lad., N. Lao, A. Harpale, B. Kisiel, M. Rogati. 2007. Utility-based Information Distillation Over Temporally Sequenced Documents. *Proceedings of the International Association for Computing Machinery Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval.* pp. 31-38. Amsterdam, Netherlands.

Zhang, P., E. L. Plettenberg, J. L. Klavans, D. W. Oard, and D. Soergel. 2007. Task-based Interaction with an Integrated Multilingual, Multimedia Information System: A Formative Evaluation. *Proceedings of the Joint Conference on Digital Libraries*. Vancouver, Canada.

Contents