

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

LABORATORY *for*
 COMPUTER
 SCIENCE
July 2000
 SPOKEN
 LANGUAGE
 SYSTEMS



©2000 Massachusetts Institute of Technology

For information or copies of this report, please contact:

Victoria L. Palay
MIT Laboratory for Computer Science
545 Technology Square, NE43-601
Cambridge, MA 02139 USA
palay@mit.edu

Please visit the Spoken Language Systems Group on the
World Wide Web at <http://www.sls.lcs.mit.edu>

SUMMARY *of*

RESEARCH

covering

July 1, 1999 - June 30, 2000

Table of Contents

1

Summary of Research

Research, Technical, Administrative and Support Staff	vii-x
Graduate Students	xi-xiv
Visitors, Undergraduate Students, Transitions	xiv-xvi
Research Sponsorship	xvii

2

Research Highlights

Research Highlights, 1999-2000	
<i>Victor Zue</i>	3

3

Research Projects

MERCURY	
<i>Stephanie Seneff, and Joseph Polifroni</i>	9
Data Collection and Performance Evaluation of MERCURY	
<i>Joseph Polifroni, Stephanie Seneff, James Glass and Victor Zue</i>	15
Acoustic Modeling Improvements in SUMMIT	
<i>Nikko Ström, Lee Hetherington, T.J. Hazen, James Glass and Eric Sandness</i>	20
Confidence Scoring for Use in Speech Understanding Systems	
<i>T.J. Hazen</i>	22
MOKUSEI: A Japanese, Telephone-based Conversational System in the Weather Domain	
<i>Victor Zue, Stephanie Seneff, Joseph Polifroni, Yasuhiro Minami, Mikio Nakano, T.J. Hazen and James Glass</i>	25
MUXING	
<i>Chao Wang, Victor Zue, Joseph Polifroni, Stephanie Seneff, Jon Yi and T.J. Hazen</i>	27
ORION	
<i>Stephanie Seneff, Chian Chuu and Scott Cyphers</i>	29
Facilitating Spoken Language System Creation with SPEECHBUILDER	
<i>James Glass, Jef Pearlman and Eugene Weinstein</i>	31
Finite-State Transducer-Based Unit Selection for Concatenative Speech Synthesis	
<i>Jon Yi, James Glass and Lee Hetherington</i>	35
Barge-in	
<i>Nikko Ström</i>	39

4

Thesis Research

GENESIS-II

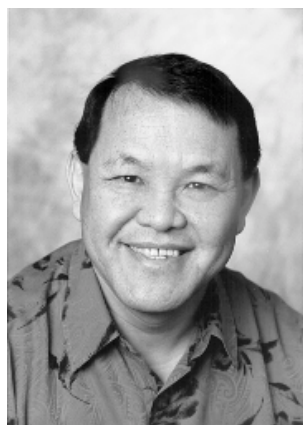
<i>Lauren Baptist</i>	43
Modeling Out-of-Vocabulary Words for Robust Speech Recognition	
<i>Issam Bazzi</i>	45
Using Support Vector Machines for Spoken Digit Recognition	
<i>Issam Bazzi</i>	48
Building a Speech Understanding System using Word Spotting Techniques	
<i>Theresa Burianek and Timothy J. Hazen</i>	50
A Three-Stage Solution to Flexible Vocabulary Speech Understanding	
<i>Grace Chung</i>	52
Lexical Modeling of Non-Native Speech for Automatic Speech Recognition	
<i>Karen Livescu</i>	55
The Use of Dynamic Reliability Scoring in Speech Recognition	
<i>Xiaolong Mou</i>	58
Subword-based Approaches to Spoken Document Retrieval	
<i>Kenney Ng</i>	60
Enabling Spoken Language Systems Design for Non-Experts	
<i>Jef Pearlman</i>	64
Discriminative Training of Acoustic Models in a Segment-Based Speech Recognizer	
<i>Eric Sandness</i>	65
Analysis and Transcription of General Audio Data	
<i>Michelle Spina</i>	67
Framework for Joint Recognition of Pronounced and Spelled Proper Names	
<i>Atiwong Suchato</i>	70
Robust Pitch tracking for Prosodic Modeling in Telephone Speech	
<i>Chao Wang</i>	72
Methods and Tools for Speech Synthesizer Design and Deployment	
<i>Jon Yi</i>	75

5

Theses, Publications, Presentations and Seminars

Ph.D. and Masters Theses	79
Publications	80
Presentations	81
SLS Seminar Series	82

Research Staff



VICTOR ZUE

Victor Zue is a Senior Research Scientist, the head of the Spoken Language Systems Group, and an Associate Director of the Laboratory for Computer Science. His main research interest is in the development of spoken language interfaces to facilitate natural human/computer interactions, and he has taught many courses and lectured extensively on this subject. He is best known for his acoustic phonetic knowledge, and for leading his research group in developing conversational systems that allow users to access information using spoken dialogue. In 1994, Zue was elected Distinguished Lecturer by the IEEE Signal Processing Society. In 1999, he received the "Sustained Excellence" Award from DARPA-DoD. Zue is a Fellow of the Acoustical Society of America. He received his Sc.D. in Electrical Engineering from MIT in 1976.



JAMES GLASS

James Glass is a Principal Research Scientist and Associate Head of the SLS group. He received his Ph.D. in Electrical Engineering and Computer Science from MIT in 1988. His research interests include acoustic-phonetic modeling, speech recognition and understanding in the context of conversational systems, and corpus-based speech synthesis. In addition to publishing extensively in these areas, he has supervised S.M. and Ph.D. students, and co-taught courses in spectrogram reading and speech recognition. He served as a member of the IEEE Acoustics, Speech, and Signal Processing, Speech Technical Committee from 1992-1995. From 1997-1999, he served as an associate editor for the IEEE Transactions on Speech and Audio Processing.



SCOTT CYPHERS

D. Scott Cyphers received his S.B. at MIT in 1982, and S.M. in 1985. While at MIT, he worked on program understanding, speech recognition, and systems to support speech recognition research (Spire). In 1986 he left MIT for Symbolics, where he worked on many parts of the operating system, compiler, and implemented the CLOS object system. Scott switched to the Object-Oriented database world at Object Design in 1991, where he worked with schema, Windows operating system interactions, COM integration, and XML. In 1999 Scott returned to the speech recognition world at MIT. Scott's professional interests include distributed objects, natural language processing, problem solving and knowledge representation. Scott is also interested in mathematics, logic, meta-mathematics, astronomy, and cosmology.



T.J. HAZEN

Timothy James (T. J.) Hazen arrived at MIT in 1987 where he received his S.B. degree in 1991, S.M. degree in 1993 and PhD in 1998, all in Electrical Engineering. T.J. joined the SLS group as an undergraduate in 1991 and has been with the group ever since. He is currently working as a research scientist in the group. His primary research interests include acoustic modeling, speaker adaptation, automatic language identification, and phonological modeling.

Research Staff



LEE HETHERINGTON

Lee Hetherington received his S.B., S.M., and Ph.D. degrees from MIT's Department of Electrical Engineering and Computer Science. He completed his doctoral thesis, "A Characterization of the the Problem of New, Out-of-Vocabulary Words in Continuous-Speech Recognition and Understanding," and joined the SLS group in October 1994. His research interests include many aspects of speech recognition, including search techniques, acoustic measurement discovery, and recently the use of weighted finite-state transduction for context-dependent phonetic models, phonological rules, lexicons, and language models in an integrated search.



RAYMOND LAU

Raymond Lau received the B.S. in Computer Science and Engineering, the M.S. degree in Electrical Engineering and Computer Science, and the Ph.D. degree in Computer Science, all from the Massachusetts Institute of Technology in 1993, 1994, and 1998, respectively. He was a National Science Foundation fellow and is a member of Eta Kappa Nu. His current research interests are in the area of speech recognition and spoken language systems with a particular focus on subword modelling, search strategies and language modelling.



JOSEPH POLIFRONI

Joseph Polifroni's interests include language generation, human-computer interaction, and multilingual systems. He has worked on the back-end components of many of the SLS systems, including GALAXY and DINEX in addition to his work on GENESIS, the natural language generation system that is part of the overall GALAXY architecture. He has also contributed to the Spanish and Mandarin Chinese systems. Before joining SLS, Joe worked in the Speech Group at Carnegie Mellon University and was also a consultant for Carnegie Group Inc. in Pittsburgh. In addition, Joe spent two years living in China, teaching English at Shandong University in Jinan.



STEPHANIE SENEFF

Stephanie Seneff has a B.S. degree in Biophysics and M.S., E.E., and Ph.D. degrees in Electrical Engineering and Computer Science from MIT. Her research interests span a wide spectrum of topics related to conversational systems, including phonological modelling, auditory modelling, computer speech recognition, statistical language modelling, natural language understanding and generation, discourse and dialogue modelling, and prosodic analysis. She has published numerous papers in these areas, and she is currently supervising several students at both master's and doctoral levels.



NIKKO STRÖM

Nikko Ström received the Master of Science, (Engineering Physics) degree in 1991, and the Ph.D. degree in Electrical Engineering (Department of Speech, Music, and Hearing) in 1997 at the Royal Institute of Technology (KTH), Stockholm, Sweden. He joined SLS in May 1998 as a Postdoctoral Associate, and was appointed Research Scientist in January 1999. His main areas of interest are human/machine dialogue, lexical search in automatic speech recognition, and acoustic/phonetic modeling. At KTH, he developed a continuous automatic speech recognition system that is still in use in several dialogue systems. He is also the author of the NICO tool kit for Artificial Neural Networks. The tool kit is publicly available, and has been downloaded from more than 1,000 different sites in 55 countries (September 1998).

Technical Staff



CHRISTINE PAO

Christine Pao has been a member of the technical research staff since 1992. She is primarily involved in the development and maintenance of the GALAXY system. Her research interests are in discourse and dialog, systems integration with a focus on multilingual systems and language learning, and open microphone issues such as rejection and channel normalization. Christine has a bachelor's degree in Physics from MIT.

Administrative & Support Staff



VICTORIA PALAY

Victoria Palay has been a member of the Spoken Language Systems group since 1988. As SLS program administrator, she manages personnel, fiscal, publication and contractual matters as well as space and other group resources. In addition, she supports Victor Zue's duties as LCS Associate Director by coordinating EPOCH-IT activities and distributing equipment donations made to the laboratory. Victoria has a B.A. in Government and French Studies from Smith College.



SALLY LEE

Sally Lee joined the Spoken Language Systems group as senior secretary in 1993. She received a B.A. in Studio Art/ Art History from Colby College in 1984. She also studied at the Art Institute of Boston and the New York Studio School. In addition to her secretarial duties, Sally has made many of the animated and still icons for SLS programs including GALAXY and JUPITER. She also is responsible for transcribing sentences that are recorded from people calling into the JUPITER system.

Graduate Students



LAUREN BAPTIST

Lauren Baptist joined the Spoken Language Systems Group in September 1999, with a fellowship from NTT. She received the S.B. from Dartmouth College in May 1999. She expects to receive the S.M. degree from MIT in August 2000. Her thesis is entitled "Genesis-II: A Language Generation Module for Conversational Systems."

Advisor: Stephanie Seneff



ISSAM BAZZI

Issam Bazzi's research interest is in the area of subword and language modeling for automatic speech recognition. He received his B.E. in computer and communication engineering from the American University of Beirut, Beirut, Lebanon in 1993 and his S.M. from MIT in 1997. Between 1993 and 1995, He did research on networked multimedia systems at the Center for Educational Computing Initiatives at MIT. Before joining SLS in 1998, Issam was a member of the Speech and Language Department at BBN Technologies, GTE Internetworking where he did research on language-independent character recognition.

Advisor: James Glass



THERESA BURIANEK

Theresa Burianek joined the Spoken Language Systems group in September 1999 after receiving the S.B. from MIT in Computer Science and Engineering in June 1999. She is researching the use of word-spotting understanding to ease the creation of spoken language systems and implemented verbal responses for direction queries in VOYAGER, an interactive city guide that uses verbal human to computer interaction to process and retrieve queries between locations in Boston. She expects to receive the M.Eng. degree in July 2000.

Advisor: T.J. Hazen



GRACE GHUNG

Grace Chung graduated in Electrical Engineering and Mathematics from the University of New South Wales, Sydney, Australia. She earned a Fulbright scholarship to attend MIT and completed her master's degree in June 1997. Her interests are in acoustic modelling and prosodic modelling for speech recognition.

Advisor: Stephanie Seneff

Graduate Students



ATTILA KONDACS

Attila Kondacs earned his BA and MSc in mathematics at Eotvos University, Hungary in 1995. He researched Quantum Computation, an emerging area within complexity theory in computer science. Attila went on studying at the University of Cambridge in England where he received a Diploma in Economics in 1996. Before starting his graduate studies at MIT, he worked for 3 years first in a telecommunication startup called TeleMedia, followed by the product management branch of a larger telecom, Matav in Hungary where he was a product manager and later the staff assistant to the director.

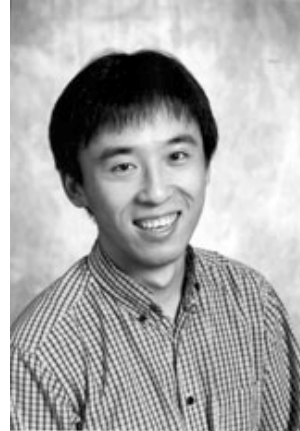
Advisor: Victor Zue



KAREN LIVESCU

Karen Livescu received her B.A. in Physics at Princeton University in 1996. She spent the following year at the Technion in Haifa, Israel, as a visiting student in the Electrical Engineering department. Karen started graduate study as a National Science Foundation fellow in the SLS group in September 1997 and completed her Master's thesis on improved speech recognition for non-native speakers in September 1999. Her research interests include statistical modeling and lexical feature design for speech recognition.

Advisor: James Glass



XIAOLONG MOU

Xiaolong Mou received his bachelor's degrees in Computer Science and Enterprises Management from Tsinghua University, Beijing, China in 1996. He completed his Master's thesis on continuous speech recognition systems and received his master's degree in Computer Science from Tsinghua University in 1998. He is currently pursuing a Ph.D. in the SLS group. Xiaolong's research interest includes conversational speech systems, automatic speech recognition and speech understanding.

Advisor: Victor Zue



KENNEY NG

Kenney Ng's current research interest is in the area of information retrieval of spoken documents, which is the task of identifying those speech messages stored in a large collection that are relevant to a query provided by a user. Prior to his return to MIT in 1995, Kenney was a member of the Speech and Language Department at BBN Systems and Technologies where he did research on large vocabulary recognition of conversational speech, word spotting, topic spotting, probabilistic segmental speech models, and noise compensation. He received his B.S. and M.S. degrees in EECS from MIT in 1990 and completed his Ph.D. in February 2000.

Advisor: Victor Zue



JEF PEARLMAN

Jef Pearlman joined the Spoken Language Systems Group in September 1999. He expects to complete the M.Eng. degree in August 2000. His thesis is entitled "SLS-Lite: Enabling Spoken Language Systems Design for Non-Experts."

Advisor: James Glass



ERIC SANDNESS

Eric Sandness joined the SLS Group in June 1999 as a participant in the Undergraduate Research Opportunities Program. His summer project was entitled "Real-Time Speech Recognition." Eric was appointed as a Research Assistant in September 1999. He received the M.Eng. degree in June 2000. His thesis is entitled "Discriminative Training of Acoustic Models in a Segment-based Speech Recognizer."

Advisor: Lee Hetherington



MICHELLE SPINA

Michelle received the B.S. degree in electrical engineering from the Rochester Institute of Technology in 1991, and the S.M. degree in electrical engineering from the Massachusetts Institute of Technology in 1994. She completed her Ph.D. degree in May of 2000 in the Spoken Language Systems Group at the MIT Laboratory for Computer Science. Michelle's research interests include automatic indexing of audio content, speech recognition and understanding, and biomedical issues of speech processing as they relate to automatic speech recognition. Her Ph.D. thesis involved general sound understanding and orthographic analysis of general audio data. Michelle was a 1995 Intel Foundation Graduate Fellow, and is a member of Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi.

Advisor: Victor Zue



ATIWONG SUCHATO

Atiwong Suchato joined the Spoken Language Systems group in the fall of 1999. He graduated from Engineering department at Chulalongkorn University, Bangkok Thailand with B. Eng., major in Electrical Engineering in 1998. His Master's thesis is entitled "Framework for Joint Recognition between Spelled and Pronounced Proper Names" and is expected to be completed in September 2000.

Advisor: Stephanie Seneff

Graduate Students



CHAO WANG

Chao Wang received her bachelor's degree in Biomedical Engineering, with a minor in Computer Science from Tsinghua University, Beijing, China in 1994. She started her graduate study in MIT in September 1995 and joined the SLS group in April 1996. Chao's master's degree, completed in June 1997, worked on porting the GALAXY system to Mandarin Chinese.

Advisor: Stephanie Seneff



EUGENE WEINSTEIN

Eugene joined the Spoken Language Systems Group in June 2000 as a Research Assistant. He is working on SPEECHBUILDER and expects to receive his MEng degree in 2001.

Advisor: James Glass



JON YI

Jon Yi received the S.B. and the M.Eng. degrees in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology in 1997 and 1998, respectively. He also graduated in 1997 with a minor in Music. At SLS he has worked on developing a Mandarin Chinese concatenative speech synthesizer and a UNICODE/Java World Wide Web interface for the GALAXY system. His research interests include speech synthesis, communications systems, and multilingual speech understanding systems.

Advisor: James Glass

Visitors



YI-CHUNG LIN

Yi-Chung Lin received his Ph.D. in Electrical Engineering from the Electrical Engineering Institute of National Tsing Hua University, Taiwan, in July 1995. In his Ph.D. thesis he investigated the use of "A Level Synchronous Approach to Ill-formed Sentence Parsing and Error Recovery". He joined SLS in March 1999 as a Visiting Scientist. He is interested in building Chinese conversational systems for real users. His main focus has been in the area of language understanding, language generation and dialogue management. He has been working on Chinese JUPITER, the weather information system using spoken Mandarin Chinese.



YASUHIRO MINAMI

Yasuhiro Minami received the M. Eng. Degree in Electrical Engineering and the Ph.D. in Electrical Engineering from the Keio University, in 1988 and 1991, respectively. He joined NTT in 1991. He had worked in robust speech recognition. He joined SLS in 1999 as a visiting researcher. He is interested in modeling for robust speech recognition and building conversational systems. He has been working on porting JUPITER to Japanese.



MIKIO NAKANO

Mikio Nakano obtained his M.S. degree in Coordinated Sciences and Sc.D. degree in Information Science from the University of Tokyo respectively in 1990 and 1998. In 1990 he joined NTT, where he had worked on unification-based parsing, spoken language analysis, and real-time spoken dialogue systems. He joined the SLS group in February 2000 as a visiting scientist. His current research interests include spoken language understanding and generation as well as building integrated spoken dialogue systems. He is now working on multilingual conversational interface through the development of MOKUSEI, the Japanese version of the JUPITER weather information system.



**ALEJANDRA OLIVIER
MERINO**

Alejandra Olivier received her S.B degree in Computer Science from the Universidad de las Americas-Puebla, Mexico, in 1999. She joined Tlatoa, a Speech Technology Research Group, the same year. She had worked in automatic transcriptions and she is interested in building conversational systems. She joined SLS in June 2000 as a visiting scientist, where she has been working on porting JUPITER to Spanish.

**ALCIRA VARGAS
GONZALEZ**

Alcira Vargas obtained her S.B degree in Computer Science from the Universidad de las Americas-Puebla, Mexico in 1998 and joined the speech technology research group TLATOA. As a visiting scientist, she joined the SLS group in January 2000 to work on the Spanish version of the JUPITER weather information system. Her current research interests include language modelling, language understanding and spoken dialogue systems.

Undergraduate Students

Chian Chuu
Michael Freedman
Mark Knobel
Vivian Ma
Matthew Mishrikey

Transitions

Christine Pao, departed August 1999
Simo Kamppari, received M.Eng. September 1999
Yi-Chung Lin, departed September 1999
Scott Cyphers joined October 1999
Ray Lau, departed January 2000
Kenney Ng, received Ph.D. February 2000
Yasu Minami, departed March 2000
Michelle Spina, received Ph.D. June 2000
Eric Sandness, received M.Eng. June 2000

Research Sponsorship

Defense Advanced Research Projects Agency¹
BellSouth Intelliventures
Industrial Technology Research Institute
National Science Foundation²
Nippon Telegraph & Telephone

In addition, discretionary funds for research are provided by ATR Interpreting Telecommunications Research Laboratories, Hewlett-Packard Laboratories -Bristol, Hughes Research Laboratories, Science Applications International Corporation, Telcordia Technologies, Inc. and VoiceIn VoiceOut Corporation.

1. Contract Nos. N66001-96-C-8526 and N66001-99-1-8904, from the Information Technology Office, monitored by the Naval Command Control, and Ocean Surveillance Center and contract no. DAAN02-98-K0003, monitored through US Army Soldier Systems Command.

2. NSF grant no. IRI-9618731.

RESEARCH

HIGHLIGHTS

Research Highlights

Victor Zue

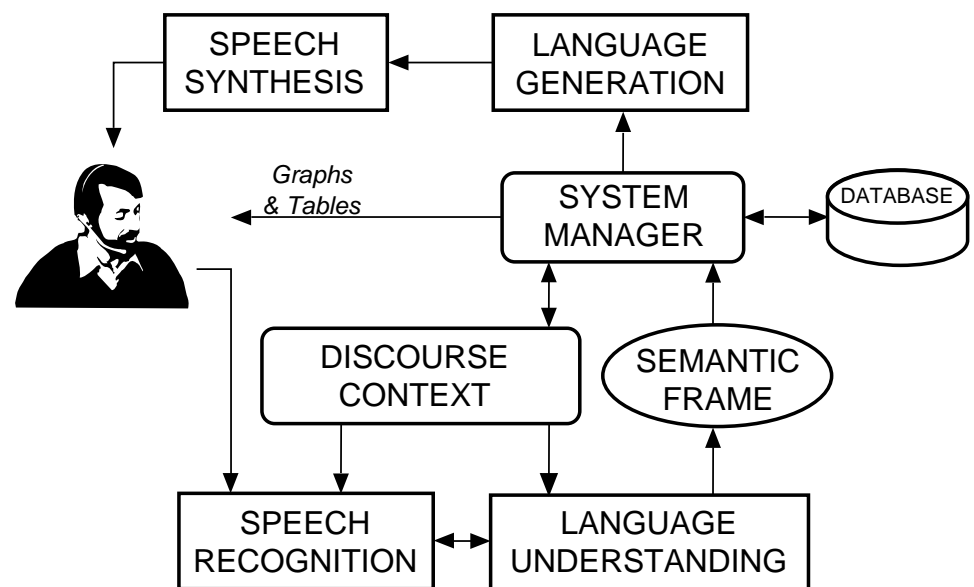
During this reporting period, we continued our research and development of human language technologies and conversational systems in several directions. Considerable efforts have been devoted to improving our speech recognition system's performance. These include the use of discriminative training for acoustic models (c.f. page 65) and better phonological modeling for non-native speakers (c.f. page 55). To make the recognition system more robust, we have extended the utterance-based confidence measures to word-based ones (c.f. page 22). We also introduced the notion of dynamic reliability scoring during recognition that adjusts the partial path score while the recognizer searches through the lexical and acoustic-phonetic network (c.f. page 58). Related work deals with speech understanding using simple word- and phrase-spotting methods (c.f. page 50).

Almost all speech recognition systems are formulated as a problem of recognizing the underlying word sequence. As a result, these systems require the availability of

domain-specific training data to derive the statistical language models. To separate the use of domain-independent and -dependent knowledge, we have started a promising line of investigation on how to develop a domain-independent recognition kernel based on sub-word units such as the syllable (c.f. page 45) and an entire sub-word phonological hierarchy, including morphs, syllables and other constituents based on ANGIE (c.f. page 52). Related work deals with the recognition of pronounced and spelled proper names (c.f. page 70).

Much of our group's research over the years has been treating speech as an "interface" problem. With the increase of speech as a data type on the World Wide Web, speech is fast becoming a "content" problem. Two of our students completed their Ph.D. theses in this general area. In one case, mixed-speaker and -environment broadcast data were analyzed and classified (c.f. page 67). In another case, speech-based information retrieval is being pursued by representing the signal as a concatenation of

Figure 1. A generic block diagram of a typical MIT conversational system.



sub-word units (c.f. page 60).

For the past three years, we have been pursuing a concatenative approach to speech synthesis in our ENVOICE system. While the technique is domain-dependent, requiring a certain amount of domain-specific data, the resulting synthetic speech is quite natural. Besides, it has always been our intention to relax the domain-dependency, so that ENVOICE can eventually produce natural output without the need for domain-specific data. As a first step in this direction, we have begun to build a sub-word based version of ENVOICE using finite-state transducers (c.f. pages 35 and 75).

One active area of research for us this year has been the development of multilingual conversational systems. Our approach to developing multilingual conversational systems is predicated on the assumption that it is possible to extract a common, language-independent semantic representation from the input, similar to the “interlingua” approach to machine translation. Whether such an approach can be effective for unconstrained machine translation remains to be seen. However, we suspect that the probability of success is high for spoken language systems operating in restricted domains, since the input queries will be goal-oriented and therefore more constrained. In addition, the semantic frame may not need to capture all the nuances associated with human-human communication, since one of the participants in the conversation is a computer. Thus far, we have applied this formalism successfully across several languages and domains. Our most recent effort involves

the development of a Japanese version of our JUPITER weather information system called MOKUSEI (c.f. page 25). This is a collaborative project with NTT. We have also developed a Mandarin version of JUPITER called MUXING, partly funded by ITRI (c.f. page 27). Work related to this latter project deals with the modeling of Mandarin tones (c.f. page 72). We also have an ongoing collaboration with the Universidad de las Americas Puebla in Mexico on the development of a Spanish version of JUPITER.

The development of multilingual systems, especially for Asian languages, exposed some of the shortcomings of our language generation component, GENESIS. As a consequence, considerable effort has been devoted to rewriting GENESIS (c.f. page 43). The resulting system, GENESIS-II, can provide very straightforward methods for simple generation tasks across multiple languages, while also supporting the capability of handling more challenging generation requirements, such as movement phenomena, propagation of linguistic features, and the context-dependent specification of word sense.

At the system development level, most of our effort has been devoted to the development of MERCURY, which can provide real flight schedules and fare information. MERCURY provides a platform for us to investigate conversational interfaces that require language generation (c.f. 9) and dialogue management (c.f. 10) that is considerably more complex than in previous systems that we have developed. We have also collected hundreds of dialogues from real users interacting with MERCURY to book

real flights, and the data have enabled us to propose some dialogue related metrics to evaluate system performance (c.f. page 15). In related work, we have also begun to investigate the use of intelligent barge-in, which can be overridden under certain circumstances, for example, when the system is describing a disclaimer or providing instructions (c.f. page 39).

This reporting period saw the beginning of two new projects that are motivated by a lab-wide five year initiative called Oxygen. The goal of the Oxygen project is to create a system that will bring an abundance of computation and communication to users through natural spoken and visual interfaces, making it easy for them to collaborate, access knowledge, and automate repetitive tasks. An important aspect of Oxygen is the ability to rapidly deploy speech-based interfaces for naïve users. In this regard, we started the development of a research infrastructure called SPEECHBUILDER that will enable novice users to develop conversational systems with little or no knowledge of the underlying human language technologies (c.f. pages 31 and 64). Another goal of Oxygen is the need to automate some of the information-related tasks (e.g., “Call me when United flight thirty-four lands.”) We have begun the development of a conversational agent called ORION, which can accept tasks specified verbally and deliver the information requested at a later time (c.f. page 29). We expect both of these areas to flourish in the coming years.

Finally, the GALAXY Communicator architecture continues to serve as the reference architecture for the DARPA Communicator Program. Recently, we have decided to make this architecture open source to promote standard setting. Users can freely download this architecture, and the associated libraries, for their own use, unencumbered by IP issues.

RESEARCH

REPORTS

MERCURY

Stephanie Seneff and Joseph Polifroni

MERCURY is a spoken conversational system that allows users to make flight reservations among several hundred cities worldwide. MERCURY engages the user in a mixed-initiative dialogue to plan an itinerary. Once the itinerary is fully specified, MERCURY then prices the itinerary and sends the user an e-mail message containing a detailed record of the flights and fares. MERCURY runs in both a displayful and a displayless mode. In displayful mode, MERCURY provides, in addition to the spoken output, a paraphrase of the original user query in a paraphrase window, as well as a text of the verbal response and a table of the set of flights retrieved from the database. MERCURY is configured as a suite of servers that communicate with one another via a central programmable hub using the GALAXY Communicator architecture. In this section, we will highlight two different aspects of MERCURY where we have spent considerable research effort: (1) language generation and (2) dialogue control.

Language Generation

The generation server plays an important, multi-faceted role in MERCURY, serving all generation needs for both natural and formal languages, as indicated in Figure 2 and described more fully in [3]. We have recently developed a new generation server, which we call GENESIS-II [1] (c.f. page 43), which is significantly improved over the original system, in terms of both its capabilities and ease of use. One of the

most important roles is to generate a natural language response for the user. GENESIS-II uses a common grammar file but distinct lexicons to convert the response frame into a well-formed English text to display in the graphical interface and into a marked-up text format for further processing by the ENVOICE speech synthesizer (c.f. pages 35 and 75), to produce the response speech waveform. Often, just before going to the database, MERCURY paraphrases back to the user the topic of their question, as in, “Okay, United flights from Boston to Dallas on March third. One moment please.” In displayful mode, the full paraphrase of the user query is displayed in a special paraphrase window. These mechanisms serve to inform the user the degree to which MERCURY understood the question. The full paraphrase is especially difficult for wh-queries, where a trace mechanism is necessary to move the wh-marked noun phrase to the front of the surface form string.

GENESIS-II also handles several instances of paraphrases into formal languages. The first is to convert the linguistic frame (in context) that the NL component produces into a flattened electronic form (E-form), a representation that is more convenient for the dialogue manager to interpret. In this case, GENESIS-II produces a string in a simple mark-up language, which is then converted into an E-form frame by a GALAXY library routine. Another formal language task is to convert the E-form into the database query appropriate for database retrieval. Finally, in displayful mode, it converts the retrieved list of flights into a hyperlinked HTML table for graphical display. The user can click on an item in the table and refer to it verbally in a follow-up question.

Figure 2. Various generation needs in Mercury that are handled by the Genesis-II server.

input query frame	→	English paraphrase
		→ Electronic Form
database frame	→	Database query string
response frame	→	English text
		→ synthesized speech
flight list	→	html table

Dialogue Management

Dialogue modeling is a critical and challenging aspect of conversational systems, particularly when users are permitted flexibility with regard to defining the constraints of the task. For systems that adopt a strict system-initiated approach, it is feasible to define a set of states and state transitions depending on the usually small number of possible user actions at each state. However, MERCURY follows a mixed-initiative dialogue strategy, as do all of our other systems, where the system may make specific requests or suggestions, but the user is not required to be compliant. When the user is permitted to say anything within the scope of the recognizer at any time, a finite-state solution becomes unwieldy.

As described more fully in [2], MERCURY's dialogue is controlled by a turn manager that manipulates linguistic and world knowledge represented in the form of semantic frames. At each turn it begins with an E-form representing the constraints of the current query as a set of [key: value] pairs. This E-form provides the initial values in the dialogue state, which evolves over the course of the turn as rules are executed. The turn manager consults a dialogue control table to decide which operations to perform, and typically engages in a module-to-module subdialogue to retrieve tables from the database. Finally, it prepares a response frame, which may or may not include tabular entries.

In addition to the dialogue state, MERCURY also makes use of several other internal representations which retain state. These are packaged up into a turn-manager history frame which is passed to the hub for safe-keeping at the end of each turn, and returned with each subsequent turn. The

user model contains a record of the system's current belief about the state of the user's flight plan, as well as any user preferences specified at enrollment time and all of the details of the current flight plan as they become available. The turn manager also retains internally a list of all the flights that matched the user's constraint specifications, as well as a list of the subset of the flights that it has spoken about (e.g., the nonstop flights or the earliest flight).

Given MERCURY's complex dialogue state, and our desire to have a flexible mixed-initiative dialogue strategy, we have opted to use an "ordered rules" strategy for dialogue control. The activities for a given turn typically involve the sequential execution of a number of specialized routines, each of which performs a specific part of the dialogue requirements and alters the state variables in particular ways. To determine which of the operations should be performed, the system consults a dialogue control table, which is specified in a simple scripting language.

In addition to possibly altering the values of variables in the dialogue state, operations also have the responsibility to return, upon completion, one of three "move" states. The majority of the operations return the state "CONTINUE" which means simply to continue execution just beyond the rule that just fired. The two other possibilities are "STOP," i.e., exit from the dialogue control table, and "RESTART" which means to return to the top of the set of dialogue rules and start over. An (optional) distinguished operation, which handles activities that should occur routinely at every turn, is executed upon exiting the dialogue control process. The tests on variables can be binary, arithmetic, or string

clause request_keypad	—>	keypad_input
week day rel_date	—>	resolve_relative_date
flight_number & !airline	—>	need_airline
clause price & topic itinerary & truth_value deny	—>	dont_price_itinerary
clause book & num_found = 1	—>	add_flight_to_itinerary
num_nonstops > 1 & num_nonstops < 4 & arrival_time	—>	speak_three_arrivals

Figure 3. Selected rules from the MERCURY dialogue control table. “&” stands for “AND” and “|” stands for “OR”.

matching on values. Figure 3 shows several examples of actual rules in MERCURY’s dialogue control table.

There are currently a total of over 200 rules in MERCURY’s dialogue control table. These rules can be organized logically into three groups. The first of these deal with the input query representation, determining what type of information is being asked for and checking that sufficient information has been obtained for answering the query. The second group deals with obtaining the answer to the query, either from the database server (typically in a module-to-module subdialogue) or from a pre-existing flight list. The third group concerns preparing the reply frame, after the database retrieval has already taken place. The purpose of this last set of rules is to reduce the number of database tuples returned for a given query to a manageable set for a spoken response. In addition to these three broad categories, there are miscellaneous rules that have to do with updating the user model, preparing the intermediate reply, pricing or e-mailing the itinerary, preparing the database query, filtering flights, or updating the itinerary.

From the system developers’ point of view, this dialogue control strategy has made development of complex systems much easier. Although the number of rules can grow to be as large (or as small) as the developer wants, having one central file that

maps out the control flow through a system turn introduces a level of abstraction above the level of the code itself that facilitates both development and debugging. The operations specified by each rule are general functions, and it is up to the system developer to decide how to parcel up the computational requirements into the individual operations.

With experience, one acquires a set of guidelines to help formalize this process. As a general rule, it is preferable to limit the use of nested function calls. Instead, an operation can set a variable to indicate that another operation needs to be called, and then the intended subroutine gets promoted to the status of a dialogue control operation. This has the effect of exposing more of the internal activities of the system to the dialogue control table, which serves as a very useful outline of system control flow. Another general policy is that each operation should be concerned with a single well-defined task, again in order not to conceal complexity.

Figure 4 gives an example dialogue between MERCURY and a user illustrating some of the complexities that can arise in a mixed-initiative interaction, including early specification of the return date, the rejection of an offered airline, and a final adjustment to the original itinerary to improve the fare.

Figure 4. Example illustrative dialogue in the MERCURY domain involving an initial itinerary that is later adjusted for a cheaper fare.

S: Welcome to the initial development version of the MIT air travel information system. This call is being recorded for system development. You may hang up or ask for help at any time. For enrolled users please say your first and last name. Otherwise, say 'new user.'

U: Joe Polifroni

S: Please speak the month and day you registered as your password, as in 'May first.'

U: I want to key in my password.

S: Okay, Please enter your four digit date password... <keyed in>

S: Welcome, Joe. You have successfully logged onto mercury. How may I help you?

U: I want to fly to San Francisco the first Sunday in November and return the following Thursday.

S: Okay, flights from Boston to San Francisco on Sunday November 5. Can you provide the approximate departure time or airline preference?

U: Are there any flights in the morning?

S: Okay, flights from Boston to San Francisco departing between 5 a.m. and 12 p.m. on Sunday November 5. One moment please. ... Can you provide a departure or arrival time? I have flights on American, Frontier, America West, Nordic East Airways, Northwest, and United. The earliest nonstop United flight leaves at 7 a.m. and arrives at 10:20 a.m.

U: I don't want to fly on United.

S: I have a nonstop American flight leaving at 9 a.m. arriving at 12:40 p.m. Would that work?

U: Yes, I'll take that one.

S: American flight 195 has been added to your itinerary. You depart Boston at 9 a.m. and arrive in San Francisco at 12:40 p.m. Shall I help you with the return flight now?

U: Yes, please.

S: Okay, American flights from San Francisco to Boston on Thursday Nov. 9. One moment please. ... I have three nonstop flights: a flight leaving at 7:30 a.m., a flight leaving at 3:30 p.m. and a flight leaving at 10 p.m. Are any of these okay?

U: what time does the morning flight arrive?

S: American flight 196 arrives at 3:57 p.m. in Boston. Shall I add this flight to your itinerary?

U: Yes, please.

S: American flight 196 has been added to your itinerary. You depart San Francisco at 7:30 a.m. and arrive in Boston at 3:57 p.m. Shall I price your itinerary?

U: Yes, please.

S: Pricing your itinerary will take awhile. In the meantime I'll summarize your itinerary for you. On Sunday Nov. 5, on American flight 195, you depart from Boston at 9 a.m. and arrive in San Francisco at 12:40 p.m. On Thursday, Nov. 9, on American flight 196 you depart from San Francisco at 7:30 a.m. and arrive in Boston at 3:57 p.m. The cheapest price for your itinerary is \$2210.50. Shall I email your itinerary to you?

U: I want to change the first flight in my itinerary.

S: Okay, American flights from Boston to San Francisco on Sunday November 5. One moment please. ... I have three nonstop flights: a flight leaving at 9 a.m. a flight leaving at 3 p.m. and a flight leaving at 5:15 p.m. Would one of these work?

U: What if I leave on Saturday?

S: Okay, American flights from Boston to San Francisco on Saturday November 4. One moment please. ... I have three nonstop flights: a flight leaving at 9 a.m., a flight leaving at 3 p.m. and a flight leaving at 5:15 p.m. Do you like any of these?

U: I'll take the nine o'clock flight.

S: I have changed your itinerary. On Saturday, Nov. 4, on American flight 195, you depart from Boston at 9 a.m. and arrive in San Francisco at 12:40 p.m. Shall I price your itinerary?

U: Yes, please.

S: Pricing your itinerary will take awhile. In the meantime I'll summarize

your itinerary for you. On Saturday Nov. 4, on American flight 195, you depart from Boston at 9 a.m. and arrive in San Francisco at 12:40 p.m. On Thursday Nov. 9, on American flight 196, you depart from San Francisco at 7:30 a.m. and arrive in Boston at 3:57 p.m. ...The cheapest price for your itinerary is \$827.50. Shall I email your itinerary to you?

U: Yes, please

S: Your itinerary has been emailed to joe@lcs.mit.edu. Is there something else I can do for you?

U: No thank you. Goodbye.

S: Thanks for using Mercury. Have a nice day!

References

- [1] L. Baptist and S. Seneff, "GENESIS-II: A Versatile System for Language Generation in Conversational System Applications," to appear in *Proc. ICSLP 2000*, Beijing, China, Oct. 2000.
- [2] S. Seneff and J. Polifroni, "Formal and Natural Language Generation in the MERCURY Conversational System," to appear in *Proc. ICSLP 2000*, Beijing, China, Oct. 2000.
- [3] S. Seneff and J. Polifroni, "Dialogue Management in the MERCURY Flight Reservation System," *Proc. ANLP/NAACL 2000 Satellite Workshop*, Seattle, Washington, May, 2000.

Data Collection and Performance Evaluation of MERCURY

Joseph Polifroni, Stephanie Seneff, James Glass and Victor Zue

Data Collection

For just over three years, our group has been deploying mixed-initiative spoken dialogue systems on toll-free telephone numbers in North America. Our motivation was to perform a wider-scale data collection than we were able to do within our laboratory environment. In this regard we have been quite successful: in the past year alone, our JUPITER weather information line received 70,132 calls (out of approximately 97,000 total) accounting for nearly 516,000 utterances. The growth in the number of callers to JUPITER in the three years of its existence has been very gratifying, especially considering that we have done no overt recruitment for subjects. These data have been invaluable in improving the performance of our speech recognition and understanding components.

Over the past year, we have increased our efforts in collecting data for the MERCURY flight travel domain, moving beyond the scenario-based data collection that we used initially. We have made the system available to users outside the group and have also encouraged group members to arrange all business and personal travel through MERCURY. Although the totals for MERCURY for the past year, 4464 utterances from 385 callers, dwarfs in comparison with JUPITER, our data collection effort for MERCURY is just getting underway. (Calls to JUPITER have peaked in the last year after a comparably slow start.) Just as with JUPITER, our MERCURY system is designed, first and foremost, to provide useful information to users who have real problems to solve. We are currently looking into ways of recruiting subjects for MERCURY who have real, complex travel plans.

Evaluation Metrics

Over the past year, we have begun developing metrics for evaluating system performance at the dialogue level. Our previous work has focused on utterance-level evaluation, moving from word/sentence error rates (WER, SER, respectively) to concept/understanding error rate (CER, UER) [1]. We have found that the GALAXY Communicator architecture [2] has enabled us to instrument these metrics in a way that insures that off-line evaluation is performed on exactly the same configuration as the on-line system [3]. The GALAXY Communicator architecture has been particularly useful in helping us move beyond utterance-level evaluation to measuring dialogue performance. To do this, we have created two new measures which can quantify how effectively a user can provide novel information to a system, and how efficiently the system can understand information concepts from a user. The *query density* measures the mean number of *new* concepts introduced per user query, while the *concept efficiency* tabulates the average number of turns it took for a concept to be successfully understood. In addition to being useful for quantifying longitudinal improvements in a particular system, we also believe these measures may be useful for comparing different mixed-initiative systems.

Query density, QD , measures the mean number of new concepts introduced per user query,

$$QD = \frac{1}{N_d} \sum_{i=1}^{N_q} \frac{N_u(i)}{N_q(i)}$$

where N_d is the number of dialogues, $N_q(i)$ is the total number of user queries in the i^{th} dialogue, and $N_u(i)$ is the number of *unique*

Figure 5. Two dialogues illustrating QD. "U" refers to user utterances; "S" refers to system utterances.

<p>U: I would like to fly to Boston. S: To Boston from where? U: Denver. S: What date will you be travelling on?</p>	<p>U: I would like to fly from Boston to Denver. S: What date will you be travelling on?</p>
---	--

concepts understood by the system in the i^{th} dialogue. A concept in a dialogue is not counted in N_s if the system had already understood it from a previous utterance, and is only counted when it is correctly understood by the system. Thus, a concept that is never understood will not contribute to N_s .

To illustrate, consider the two dialogues in Figure 5. In the dialogue on the left, the user introduces one concept per turn, the destination in the first query and the source in the second. QD for this dialogue would be 1.0. In the second dialogue, the user introduces the same two concepts in one utterance, for a QD of 2. In a recent evaluation on over 200 dialogues from naive users of the MERCURY system (see below), we measured a QD of 1.47, showing that users are making use of our system's ability to understand more than one concept per turn.

Concept efficiency, CE, quantifies the average number of turns (expressed as a reciprocal) necessary for each concept to be understood by the system,

$$CE = \frac{1}{N_d} \sum_{i=1}^{N_d} \frac{N_u(i)}{N_c(i)}$$

where N_d is the total number of concepts in the i^{th} dialogue. A concept is counted whenever it was uttered by the user and was not already understood by the system.

Since, $N_c(i) \geq N_u(i)$, then $0 \leq CE \leq 1$.

Figure 6 illustrates two dialogues with contrasting values for CE. In the first

dialogue, there is a misrecognition on the first user utterance, leading to a repair in the second utterance. It took two turns to introduce one concept, making CE .5 for this dialogue. In the second dialogue, the destination is understood correctly in one turn, making CE 1.0 for that exchange. Measuring CE on the same subset of the MERCURY data used for QD above (see below), we obtained a value of .92, meaning that approximately one in ten concepts had to be repeated by users.

The criteria for incrementing the counts for N_s and N_c are based on monitoring concepts derived from the reference transcriptions. Increments only take place when an input concept, A , is not present (i.e., $A \notin S$) in the internal dialogue state, meaning that it had not been previously understood by the system. When this condition occurs, N_s is incremented, to indicate that the user was trying to convey a concept to the system. If the concept is also understood by the system (i.e., the hypothesis representation also contains concept A), then N_c is also incremented. We believe the QD metric is useful for quantifying the behavior of individual dialogue systems, as well as for making cross-system comparisons. The higher the density, the more effectively a user is able to communicate concepts to the system. Dialogues which prompt for specific concepts one-by-one may have lower QDs, as will those which contain more confirmations, or mis-recognitions. The CE is also a

U: I would like to fly to Boston. S: To Austin from where? U: No, to Boston. S: To Boston from where?	U: I would like to fly to Boston. S: To Boston from where?
--	--

Figure 6. Two dialogues illustrating CE. “U” refers to user utterances; “S” refers to system utterances.

useful diagnostic metric, as it is an indicator of recognition or understanding errors. The higher the efficiency, the fewer times a user had to repeat a concept.

Evaluation Infrastructure

The task of evaluating complex conversational systems is inherently difficult. In the first place, component technologies are interdependent. Typically, a spoken dialogue system is comprised of multiple modules, each of which performs its task within an overall framework, sometimes completely independently but most often with input from other modules. Secondly, once a mechanism is in place for running data through an off-line system, a simple reprocessing of data with a new version of any component can lead to an incoherent interaction, as only one side of a two-sided conversation has changed.

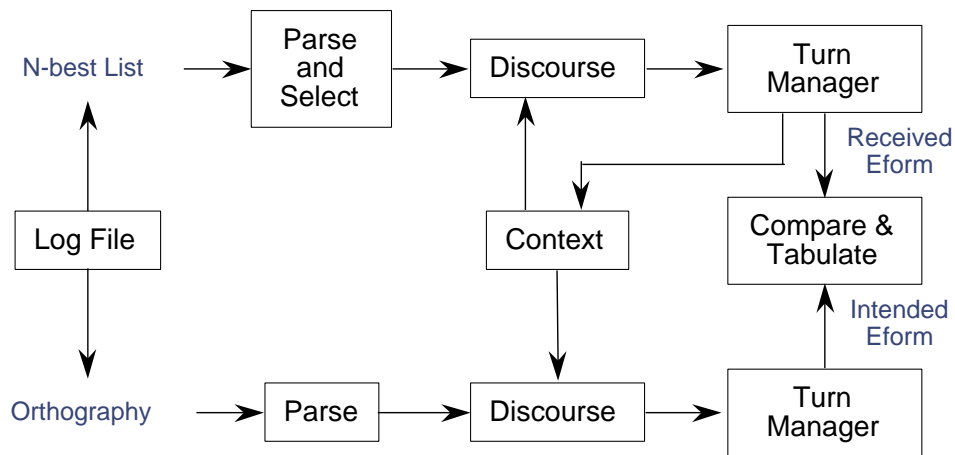
The GALAXY Communicator architecture, originally developed within our group, has enabled us to duplicate on-line behavior for both system development (e.g., new speech recognition or language understanding components) and off-line performance evaluation. This allows all components to interact in a consistent manner. We have achieved this goal by developing two new servers: a “Batchmode” server and an “Evaluate” server.

The batchmode server is used to generate inputs to a system during off-line processing (as opposed to an on-line system,

which would deploy an audio- and/or GUI-server instead). The batchmode server can be used to simulate the behavior of a system during actual data collection, or can be used in conjunction with the evaluation server to quantify the performance of pre-recorded data. The batchmode server can generate inputs from an existing log file (which was generated by the system at the time the interaction actually took place), or can directly process waveforms, orthographic transcriptions, word graphs, semantic frames, etc., and pass them to the appropriate server via the hub.

The evaluation server is used to gather performance statistics for an off-line evaluation. The server can measure both word- and concept-level error rates, and can be used to measure the *QD* and *CE* metrics described in the previous section. To compute the *QD* and *CE* metrics, we reprocess a log file after the orthographic transcription has been provided for the user queries. As illustrated in Figure 7, both the recognizer hypothesis and the original orthography are run through the system utterance by utterance, with the discourse and dialogue states being maintained exclusively by the recognizer branch. If the database content remains unchanged from when the dialogue originally occurred, then system responses derived in this way would be identical to those originally produced by the system. However, all of our systems make use of continually updated, dynamic data sources. It is therefore virtually

Figure 7. A flow graph illustrating the procedure for synchronizing discourse context during an offline logfile evaluation.



impossible to guarantee that the system responses which occur during a subsequent evaluation will be the same as the original ones. Thus, it is possible that dialogues could become incoherent during an evaluation. The only way to prevent this would be to freeze a version of the database used at the time, or store the answers retrieved during the dialogues.

Another problem has to do with the use of time expressions in user queries. If the database is static, it is usually just a matter of resetting the reference date of the dialogue so that relative expressions (e.g., “tomorrow”) can be correctly understood. Where the data are dynamic, and time-sensitive (e.g., short-range weather forecasts, flight schedule and fare information) the solution is more complex. In the case of MERCURY for example (described in the next section), we shift all dates in the dialogue to make all references to time occur in the future.

Experiments

Table 1 provides a breakdown of MERCURY performance on a subset of the data collected this year. This subset was selected

specifically to maximize for the number of naive users solving real problems (i.e., not scenario-based dialogues). The “All” condition reports results for the utterance-based and dialogue-based metrics we have described above for all 648 utterances in the test set. We then subdivide these data into three categories. “Accepted” utterances were those in which the reference orthography and recognition hypothesis were both parsable by our understanding component. “No Hyp” utterances were the ones in which the recognition hypothesis failed to parse, while “No Ref” utterances were the cases where the reference orthography could not be parsed. CER and UER were not available for the “No Ref” condition because we could not automatically produce a set of reference concepts. QD and CE are only reported for the “All” condition because they are intended to quantify the dialogue as a whole.

The WER on the accepted Mercury utterances is similar to our performance in JUPITER [4]. The CER and overall UER are higher however, which we attribute to the fact that MERCURY is in its early development stages, and because the air-travel domain is

Condition	# Utts.	WER	CER	UER	QD	CE
All	648	21.5	24.4	42.3	1.47	0.92
Accepted	551	12.7	22.5	-	N/A	N/A
No Hyp.	21	47.9	88.5	-	N/A	N/A
No Ref.	76	48.9	N/A	N/A	N/A	N/A

Table 1. Evaluation results for a set of recent MERCURY data.

inherently more challenging than weather, both in terms of number of concepts, and the complexity of the interaction. We plan to make QD and CE part of our regular evaluations.

References

- [1] Polifroni, J, S. Seneff, J. Glass, and T.J. Hazen, "Evaluation Methodology for a Telephone-based Conversational System," *Proc. LREC '98*, 43–50, Granada, Spain, 1998.
- [2] Seneff, S., E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue, "GALAXY-II: A Reference Architecture for Conversational System Development," *Proc. ICSLP '98*, 931–934, Sydney, Australia, 1998.
- [3] Polifroni, J. and S. Seneff. 2000. "GALAXY-II as an Architecture for Spoken Dialogue Evaluation," *Proc. LREC 2000*, 725–730, Athens, Greece, 2000.
- [4] Zue, V, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, and L. Hetherington, "JUPITER: A Telephone-based Conversational Interface for Weather Information," *Proc. IEEE Speech and Audio Processing*, 88(1), 85–96, 2000.

Acoustic Modeling Improvements in SUMMIT

Nikko Ström, Lee Hetherington, T.J. Hazen, Jim Glass, and Eric Sandness

We have recently made several improvements to our SUMMIT speech recognition system including the simultaneous use of both boundary and segment context-dependent acoustic models to improve accuracy and the use of Gaussian selection and Pentium-III optimizations to improve speed [1,2].

The SUMMIT segment-based speech recognition system is capable of handling two rather different types of acoustic models: segment models and boundary models. *Segment* models are intended to model hypothesized phonetic segments in an acoustic-phonetic graph, and can be context-independent or context-dependent. The observation vector for these models is of fixed dimensionality and is typically derived from spectral vectors spanning the segment. Thus we extract a single feature vector and compute a single likelihood for a phone regardless of its duration. In contrast, *boundary* models are intended to model transitions between phonetic units. The observation vector for these diphone models is also of fixed dimensionality and is centered at hypothesized phonetic boundary locations or landmarks. Since some hypothesized boundaries will in fact be internal to a phone, both internal and transition boundary models are utilized. Typically, the internal models are context-independent, and the transition models represent diphones.

Because the observation spaces of segment and boundary models differ significantly, they tend to contribute different information about the recognition search and ranking of hypotheses. The baseline JUPITER system uses boundary models only, and this work investigated the addition of triphone segment models and

their effect on accuracy and speed.

In the early development of the JUPITER system, context-independent segment models were used. As more data became available, context-dependent (diphone) boundary models were added. The log probability model scores of the boundary and segment models were linearly combined for every phone. Ideally, this combination of models should have been more accurate than either of the segment or boundary models separately. However, the boundary models, with their higher degree of context dependency, benefited more from the increased training data than the segment models, and thus as more data became available the context-independent segment models began to actually degrade performance as compared to boundary models only.

In our experiments, for boundary models we used 61 context-independent internal models and 715 diphone transition models. For segment models, we used 935 triphones, 1190 diphones, and 61 monophones. Examining the pronunciation network, we find that 71% of all arcs have triphone labels, the rest having diphone or monophone labels. For segment models, the selection of triphones and diphones was based on the number of occurrences in the training data by using a count threshold of 250.

SUMMIT now makes use of finite-state transducers (FSTs) [3] to represent context, phonological, lexical, and language model constraints. In particular, the first (Viterbi) pass of the recognition search makes use of a *single* FST composed of all these components. Such a formulation makes the correct application of cross-word context-dependent acoustic models very straightforward.

Because adding triphone segment models more than quadrupled the number of acoustic models evaluated on average, we incorporated the technique of Gaussian selection [4] to reduce the number of Gaussian mixture model components evaluated. The basic idea behind Gaussian selection is to apply vector quantization (VQ) to a feature vector and then evaluate only those mixture components that influence the mixture sum in the vicinity of a VQ codeword. Mixture components that are too far away will not contribute appreciably to the sum when other mixtures are significantly closer. The mapping from VQ codeword to mixtures to evaluate can be computed ahead of time, leaving only VQ and reduced mixture computation to do at runtime. In our system, we use binary VQ with 512 codewords. At the pruning levels used in this work, Gaussian selection evaluates approximately 20% of the mixtures that would otherwise be evaluated. This results in an overall system speedup factor of about 1.8.

Further speed gains were made by hand-optimizing the key parts of Gaussian code to take advantage of the Pentium III's floating-point SIMD assembly instructions. This optimization was good for a speedup factor of about 1.8 as well.

Putting all the pieces together, Figure 8 shows word error rate vs. speed for the boundary model-only system and the combined boundary and triphone segment model systems when running on a 500MHz Pentium III system. The horizontal axis shows the real-time factor (xRT), with 1.0 corresponding to "real-time", meaning that our pipelined recognizer can keep up with incoming speech. We see that the combined system can achieve better accuracies, but

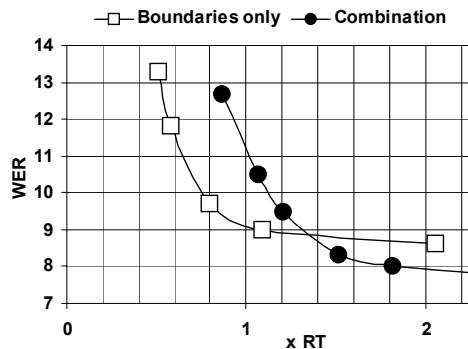


Figure 8. Word error rate (WER) vs. speed (xRT) for boundary-only and combination configurations.

only at greater computational effort. These results demonstrate that triphone segment models do offer accuracy gains compared to a boundary-only system, whereas previously monophone segment models were detrimental.

References

- [1] N. Ström, L. Hetherington, T.J. Hazen, E. Sandness, J. Glass, "Acoustic Modeling Improvements in a Segment-based Speech Recognizer," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, Keystone, December 1999.
- [2] J. Glass, T.J. Hazen, and L. Hetherington, "Real-time Telephone-based Speech Recognition in the JUPITER Domain," in *Proc. ICASSP*, pp. 61-64, Phoenix, March 1999.
- [3] E. Roche and Y. Schabes, ed., *Finite-State Language Processing*, MIT Press, 1997.
- [4] E. Bocchieri, "Vector Quantization for the Efficient Computation of Continuous Density Likelihoods," in *Proc. ICASSP*, pp. 692-695, Minneapolis, April 1993.

Confidence Scoring for Use in Speech Understanding Systems

T. J. Hazen, Theresa Burianek, Joseph Polifroni and Stephanie Seneff

Current speech recognition technology is still far from perfect and must be expected to make mistakes. A well designed speech understanding system should be able to recognize when speech recognition errors are likely to have occurred and take actions which gracefully recover from recognition errors in order to avoid frustrating the user. To develop such a system, two specific research goals must be addressed. First, a recognition confidence scoring technique which accurately determines when a recognizer's output hypothesis is reliable or unreliable must be developed. Second, confidence scores must be integrated into the back-end components of the system (e.g., language understanding and dialogue modeling) thereby enabling these components to make an informed decision about the action that should be taken when a confidence score indicates that a hypothesis may be incorrect. It is these two goals that we have strived to address in this research.

Over the last several years we have developed and refined a technique where confidence scores are computed at the word and utterance levels [1,2,3,4]. For each recognition hypothesis, a set of confidence measures are extracted from the computations performed during the recognition process and combined into a confidence feature vector. The feature vectors for each particular hypothesis are then passed through a confidence scoring model which produces a single confidence score based on the entire feature vector.

The same confidence scoring technique is used for both word and utterance level confidence scoring. To produce a single confidence score for a hypothesis, a simple linear discrimination projection vector is trained. This projection vector reduces the

multi-dimensional confidence feature vector from the hypothesis down to a single confidence score. Mathematically this is expressed as

(1)

where \vec{f} is the feature vector, \vec{w} is the projection vector, and c is the raw confidence score. The projection vector \vec{w} is trained using a *minimum classification error* (MCE) training technique.

Because the raw confidence score c is simply a linear combination of a set of features, a probabilistic score is generated from the raw score based on the following maximum *a posteriori* probability (MAP) classification expression:

$$c = \log \left(\frac{p(r|correct)P(correct)}{p(r|incorrect)P(incorrect)} \right) - t \quad (2)$$

Note that a constant decision threshold t is applied to the score. After the decision threshold t is subtracted, a negative score for c results in a rejection while a non-negative score results in an acceptance.

To test the confidence scoring techniques we utilize the recognizer for the JUPITER weather information system [5,6]. The recognizer is trained from over 70,000 utterances collected from live telephone calls to our publicly available system. The recognizer's vocabulary has 2,005 words. A test set of 2,388 JUPITER utterances is utilized for evaluation. The recognizer achieved a word error rate of 19.1% on this test set.

When evaluating the utterance level confidence scoring mechanism it is found that 13% of the test utterances were rejected. The word error rate on this 13% of the data was over 100% (e.g., there were more errors than actual words in the reference orthographies). By comparison,

the word error rate on the 87% of the utterances that were accepted was 14%. These results indicate that the utterance level confidence scoring mechanism is rejecting only the utterances which the recognizer has extreme difficulty recognizing.

To examine the performance of the word confidence scoring mechanism a receiver-operator characteristic (ROC) curve can be generated. Figure 9 shows the ROC characteristics of the three best individual confidence features and the confidence scoring method which combines the ten different individual features. A significant improvement is achieved by the combined model over all of the individual features.

The performance of the confidence scoring technique can also be examined in several other interesting ways. When examining accepted utterances only, the system correctly rejects 51% of the incorrectly hypothesized words while only falsely rejecting 4% of correct words. Furthermore, across all utterances the combination of utterance and word level scoring correctly detects 72% of the errors introduced by unknown words and 85% of the errors introduced by non-lexical artifacts.

During recognition an *N*-best list of sentence hypotheses is generated. A confidence score is computed for each word in each sentence hypothesis. A two-step process is utilized to integrate confidence scores into the TINA natural language understanding component [7,8]. First, if an utterance is rejected at the utterance level, the understanding component does not attempt to understand the utterance and assumes that no useful information for understanding can be extracted from the recognizer's output. If the utterance is

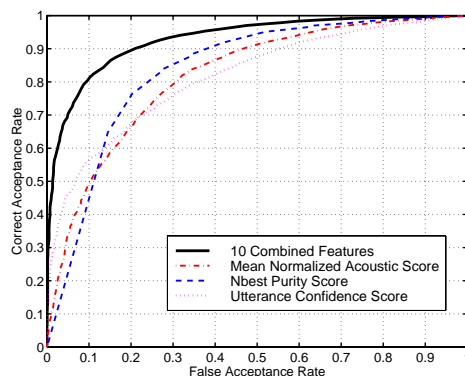


Figure 9. The ROC curves indicating relative word level confidence performance for three individual confidence features and the fully combined feature set.

accepted, the second step is to allow the natural language parser to try to interpret the utterance, given that some words may be misrecognized. Words with poor confidence scores can be “rejected” and replaced in the *N*-best with a rejected word marker. Either hard rejection (where rejection of poorly scoring words is required) or soft rejection (where rejection of poorly scoring words is optional) can be employed. In order to utilize an *N*-best list containing rejected words, the grammar must be augmented to incorporate rejected words.

To examine the effects of confidence scoring on language understanding, the JUPITER system can be evaluated on the test data under five different conditions: (1) using the original system which did not utilize word confidence scores, (2) using the new system which utilizes word confidence scores but does not perform any rejection, (2) using the new system with utterance rejection, (3) using the new system with utterance rejection and optional word rejection, and (4) using the new system with utterance rejection and hard word rejection. These conditions are investigated using key-value pair concept error rate [9]. The results are shown in Table 2 in terms of substitution, insertion, deletion, and total error rates for each condition. The table shows

Table 2. Understanding error rates as confidence scores and different levels of confidence rejection are added to the system.

Experimental Conditions	Error Rates (%)			
	Sub.	Ins.	Del.	Total
Original System	1.9	20.2	6.4	28.5
New system w/o rejection	2.1	18.2	16.1	26.3
+ utterance rejection	1.8	12.7	7.1	21.7
+ optional word rejection	1.3	9	8.4	18.7
+ hard word rejection	1	7.2	10.5	18.6

that concept error rate reduction of 35% has been achieved when confidence scoring with word and utterance level rejection is incorporated into the system.

This work has shown that confidence scoring can provide extremely useful information about the reliability of recognition hypotheses. This information can be used to guide the understanding and dialogue components of the system towards a useful interpretation and response to the user. Future work will focus on fully integrating these confidence scoring techniques into all of our publically available systems, especially MERCURY and VOYAGER. This will involve upgrading the grammars and dialogue modeling components of each system. As our understanding of the dialogue modeling issues matures, our systems should develop a range of dialogue actions based on confidence scoring including such actions as confirmation and clarification.

References

- [1] C. Pao, P. Schmid, and J. Glass, "Confidence Scoring for Speech Understanding," *Proc. of ICSLP*, Sydney, Australia, 1998.
- [2] S. Kamppari. *Word and Phone Level Acoustic Confidence Scoring for Speech Understanding Systems*. Master's thesis, MIT, 1999.
- [3] S. Kamppari and T. Hazen, "Word and Phone Level Acoustic Confidence Scoring," *Proc. of ICASSP*, Istanbul, Turkey, 2000.
- [4] T. Hazen, et al, "Recognition Confidence Scoring for Use in Speech Understanding Systems," *Proc. of ISCA ASR2000 Tutorial and Research Workshop*, Paris, France, 2000.
- [5] V. Zue, et al, "JUPITER: A Telephone-based Conversational Interface for Weather Information," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 1, January 2000.
- [6] J. Glass, T. Hazen and L. Hetherington, "Real-time Telephone-based Speech Recognition in the JUPITER Domain," *Proc. of ICASSP*, Phoenix, AZ, 1999.
- [7] S. Seneff, "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics*, vol. 18, no. 1, March 1992.
- [8] T. Hazen, et al, "Integrating Recognition Confidence Scoring with Language Understanding and Dialogue Modeling," to appear in *Proc. of ICSLP*, Beijing, China, 2000.
- [9] J. Polifroni, et al, "Evaluation Methodology for a Telephone-based Conversational System," *Proc. Int. Conf. on Language Resources and Evaluation*, Granada, Spain, 1998.

MOKUSEI: A Japanese, Telephone-based, Conversational System in the Weather Domain

Victor Zue, Stephanie Seneff, Joseph Polifroni, Yasuhiro Minami, Mikio Nakano, T.J. Hazen and James Glass

In 1997, we introduced the JUPITER weather information system in English [1]. Since it came on-line via a toll free telephone number, we have collected over 600,000 utterances from over 100,000 calls, which provide a rich corpus for training and refinement of system capabilities. Since JUPITER is our most mature conversational system to date, it has become the platform for our multilingual spoken language research effort. In 1998, we started the development of MOKUSEI¹, a conversational system that provides weather information in Japanese over the telephone [3]. This is a collaborative project between NTT and our group.

Like its English predecessor, MOKUSEI consists of a number of specialized servers that communicate with one another via a central programmable hub, using the GALAXY Communicator architecture [2]. An audio server captures the user's speech and transmits the waveform to the speech recognizer. The language understanding component parses a word graph produced by the recognizer and delivers a semantic frame, encoding the meaning of the utterance, to the discourse resolution component. The output of discourse resolution is the "frame-in-context," which is transformed into a flattened E-form (electronic form) by the generation server. This E-form is delivered to the dialogue manager, and provides the settings of the dialogue state. The dialogue manager consults a dialogue control table to decide which operations to perform, and typically engages in a module-to-module sub-dialogue to retrieve tables from the database. It prepares a response frame, containing

weather reports represented as semantic frames, which is sent to the generation component for translation into the target language². The speech synthesizer then translates the response text into a speech waveform, which it sends to the audio server. Finally, the audio server relays the spoken response to the user over the telephone.

MOKUSEI utilizes most of the same human language technology components as JUPITER, although some modifications were necessary to account for differences between English and Japanese. For speech recognition, alternative pronunciations of lexical items are generated using a set of phonological rules appropriate for Japanese. For example, one set of phonological rules accounts for the deletion of /i/ and /u/, resulting in the word sequence "desu ka" being pronounced as /d e s k a/. Acoustic models for the MOKUSEI recognizer were initially trained entirely from English utterances. As Japanese data became available, these models were retrained using a combination of read and spontaneous utterances. For language modeling, the class n-gram (consisting of some 50 classes) were either created by hand and trained, or derived automatically from the natural language grammar used by the understanding component of the system.

For language understanding, we were at first concerned that a top-down parser, TINA, might not be an appropriate choice for Japanese, which is a left-recursive language. The problem is that the system must propose the entire parse column above the first word before it has seen the rest of the sentence. The solution to this problem was

¹ Mokusei is the Japanese name for the planet Jupiter.

² Our English JUPITER system translates weather reports from English back into English.

found through TINA's trace mechanism. Using this approach each new content word is parsed first in a shallow parse tree, and then later moved to a position just after the subsequent particle that defines its role. The upper columns of the parse tree are not constructed until after the appropriate role has been identified. This has the intended effect of reordering the constructs to appear right-recursive. The current grammar for MOKUSEI has more than 900 categories and more than 2,000 vocabulary entries.

Perhaps the most difficult aspect of MOKUSEI is the translation of English weather reports into Japanese. In attempting to generate well-formed Japanese from the semantic frames for the weather reports, we encountered several situations where our language generation component, GENESIS, was unable to produce a natural Japanese translation. We found that more explicit control was needed over the choice of both the correct alternative translation of a given word and the correct ordering of certain constituents that were too deeply embedded. To solve these problems, we found it necessary to completely redesign the GENESIS system, thus providing far greater flexibility in translation tasks. Additional features of this GENESIS-II system (c.f. page 43) include better control over the ordering of constituents (e.g., noun phrase modifiers), and the capability of selecting for context-dependent lexical realization using the new "push" and "pull" capabilities. These additional features were essential for high quality generation in Japanese. For MOKUSEI, we have created nearly 500 generation rules, along with a generation vocabulary of about 3,000 entries. For Japanese synthesis, we made use of FLUET, a software synthesizer provided by NTT Cyber Space Laboratories.

The database for MOKUSEI is nearly identical to the one used by JUPITER. However, we expanded the number of Japanese cities from 6 to 46 for MOKUSEI. We also provided in a geography table information about the prefecture for each city. The weather information for the expanded Japanese set is obtained from a Web site.

The current status of MOKUSEI is that the entire system is operational at MIT, and we have used the system to collect some 2,000 read sentences, as well some 6,000 spontaneous sentences (in over 400 dialogues) from naïve Japanese users who called the system from Japan. The current word error rate, based on a 1,200 utterance test set containing no out-of-vocabulary words and with a test-set bigram perplexity of 22, is 15.4%. In the coming months, we plan to continue our data collection by installing a system in Japan, and to improve all components of MOKUSEI. We will also expand its domain of expertise, and evaluate its overall performance.

References

- [1] V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, and L. Hetherington, "JUPITER: A Telephone-Based Conversational Interface for Weather Information," *IEEE Trans. Speech and Audio Proc.*, 8(1), 85–96, 2000.
- [2] S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue, "GALAXY-II: A Reference Architecture for Conversational System Development," *Proc. ICSLP*, 1153–1156, Sydney, Australia, 1998.
- [3] V. Zue, S. Seneff, J. Polifroni, M. Nakano, Y. Minami, T. Hazen, and J. Glass, "From JUPITER to MOKUSEI: Multilingual Conversational System in the Weather Domain," to appear in *Proc. Workshop on Multi-lingual Speech Communication*, Kyoto, Japan, 2000.

MUXING

Chao Wang, Victor Zue, Joe Polifroni, Stephanie Seneff, Jon Yi, and T.J. Hazen

Over three years after deployment, our JUPITER weather information system has proven to be a valuable resource for data collection and system development. Recently, we have become interested in porting JUPITER to other languages besides English, mostly to address issues of language portability, but also with the hope that JUPITER could become a vehicle for foreign language learning. Over the past year, we have developed a Mandarin Chinese version of JUPITER, which we call “MUXING¹.” MUXING is now operational end-to-end, and we are currently collecting additional data from native speakers of Mandarin in order to begin a process of refinement modelled after our experiences with English. The system is integrated into our overall GALAXY Communicator architecture, configured as a central hub providing the communications links among a suite of servers, each specializing in different aspects of the overall system.

The JUPITER discourse and dialogue components, as well as the database server, are all language-transparent, operating from a meaning representation which we call a “semantic frame,” but which can also be viewed as an *interlingua*. For the language-dependent components such as speech recognition, natural language understanding, and language generation, the language-specific knowledge, such as acoustic models and grammar rules, is provided as control files external to the servers. Currently, the

system uses a software synthesizer provided by ITRI. However, we are in the process of developing our own Mandarin synthesizer based on the ENVOICE system (c.f. pages 35 and 75).

The recognizer is adapted from the segment-based SUMMIT system. The initial vocabulary is about 700 words, covering more than 400 place names and other common words used in weather queries. Chinese syllable initials and finals are used as acoustic model units. Tone models were included in post-processing the recognizer *N*-best output, as described in (c.f. page 72)

We were interested in modelling contextual effects through the use of boundary models. However, we have not been able to train boundary models for the Mandarin recognizer in the past, due to the difficulty in manually grouping cross-phone boundary classes based on phonological knowledge, especially when training data are very sparse. To address this problem, we have implemented a data-driven approach to derive boundary classes automatically, using a decision-tree based clustering technique. Altogether, forty one broad boundary classes were obtained in this way.

A hierarchical language model was developed which gave improved recognition accuracy over a class *N*-gram model. It uses *N*-gram probabilities to link “concepts,” each of which expands recursively into weighted sequences of concepts or words. The language model is trained by parsing a corpus using the TINA natural language formalism. Each grammar rule is decomposed into a set of trigrams linking left siblings with possible right siblings, in the context of the parent. A set of distinguished

Table 3. Summary of syllable error rate in percentage.

Configuration	Dev Set	Test Set
Baseline	20.1	23.1
+ Boundary	14.5	15.1
+ Tina LM	17.9	19.3
+ Both	13.7	13.9

¹Muxing is the Chinese name for the planet Jupiter.

Figure 10. Example semantic frame for the sentence, “becoming mostly sunny in late afternoon,” along with its paraphrase into pinyin.

```

{c weather_event
  :pred
  {p becoming
    :topic
    {q weather_act
      :conditional “mostly”
      :name “sunny”
      :pred
      {p in_time
        :topic
        {q time_of_day
          :modifier “late”
          :name “afternoon” } } } } }

```

Pinyin Paraphrase:
bang4 wan3 zhuan3 zhu3 yao4 wei2 qing2 tian1
around evening becoming mostly sunny

categories are identified as classes for the top-layer class *n*-gram. Once the grammar is trained, it is written out as a finite state transducer, which can then be composed with the lexicon to define the search space for recognition. Table 3 shows system performance as each of these improvements were added. The error rates are given for syllables.

One of the most challenging aspects of multilingual implementations is natural language generation, since it is essentially a translation of existing weather reports from English. In preparing generation rules for multilingual systems, we encountered a number of new phenomena, which ultimately inspired our group to develop an entirely new version of GENESIS, our generation server (c.f. page 43).

Different mappings for particular concepts in a semantic frame posed a problem for generation in Chinese. For

<i>late</i>	“ ” ; \$:late
<i>afternoon</i>	“xia4 wu3” \$:late “bang4 wan3”

Figure 11. Selected vocabulary entries to illustrate mechanisms to creatively combine two words into a single mapping.

example, the semantic frame in Figure 10 contains a modifier “late” for the word “afternoon,” translated into Chinese as “bang4 wan3.” However, it is not possible to choose a single meaning for “late” that is correct for “morning,” “afternoon,” and “evening,” and likewise, it is not possible to choose a single word for “afternoon” that is correct for “early,” “late,” or no qualifier. GENESIS-II provides the capability for a vocabulary item to specify a “selector,” that is used to determine the surface form of any other vocabulary item that falls within its scope. Figure 11 contains examples from a GENESIS-II vocabulary file showing the placement of the selector “\$:late” in the vocabulary entry for “late,” and the association of the surface form “bang4 wan3” with that selector for the vocabulary item “afternoon.”

Other phenomena such as temporal order of adjectives and adverbs, which varies greatly from Chinese to English, were accounted for in a similar way in GENESIS-II. GENESIS-II provides a mechanism to allow a vocabulary item to generate a [key value] pair that is inserted directly into the semantic frame. This mechanism interacts with the message templates to essentially post-hoc edit the frame to reassign inappropriately labelled keys, and, consequently, to reposition their string expansions in the surface form generation.

MUXING has been deployed on a publicly available phone number, and we are in the process of collecting and analyzing data obtain from real users.

Reference

[1] C. Wang, S. Cyphers, X. Mou, J. Polifroni, S. Seneff, J. Yi, and V. Zue, “MUXING: A Telephone-access Mandarin Conversational System,” to appear in *Proc. ICSLP 2000*, Beijing, China, Oct. 2000.

ORION

Stephanie Seneff, Chian Chuu and Scott Cyphers

Over the past year, we have begun to explore the idea of off-line delegation, as a natural extension of on-line interaction. By this, we mean that the user engages the system in a conversation in order to specify a task that the system should perform off-line, contacting the user at a later time to provide the requested information.

To this end, we have developed a new system, ORION, which represents a distinct departure from our standard system design in three significant ways: (1) Delayed task resolution: the user initially engages ORION in a conversation in order to request that a task be carried out at some designated time in the future. (2) Server acts as a user: ORION consults other conversational systems at intervals in order to seek information that it needs to execute the user's task, and (3) System initiates the call: at the appropriate time, ORION calls back the user and provides the requested information. ORION can currently handle a variety of different types of tasks, as suggested by the list of example queries in Figure 12. It consults four other systems to retrieve critical information: MERCURY for flight schedules, PEGASUS for flight status, VOYAGER for traffic conditions, and JUPITER for weather.

The ORION system is configured using the GALAXY Communicator architecture. All of the servers communicate via a common hub. The ORION server plays two distinct roles, each implemented as a separate stream. One is devoted to the enrollment of new tasks and the other is concerned with the execution of existing tasks. Users first enroll by providing critical information about their name, appropriate phone numbers, and e-mail address. To edit existing tasks or add new tasks, the user interacts with ORION at a web site, using

spoken or typed interaction. The tasks are displayed in the graphical interface, and ORION engages the user in a mixed-initiative conversation until the task is fully specified. The user can also enter new tasks in a telephone-only interaction. If a new task needs to be executed today, it is sent to the agent stream for an immediate update. Eventually, the system will also support speech-only task editing.

Generally there is a stack of pending tasks being monitored by the agent stream. A structure encoding the status of each task is examined and updated at strategic times. Updates usually involve enquiring for information from one of the other servers, using a module-to-module subdialogue. For example, a call-back request on a flight would involve first determining from MERCURY the scheduled departure time, then requesting information about the actual status of the flight from the PEGASUS server at strategic intervals while the flight should be in the air. Finally, a call to the user is launched at the predesignated trigger time.

ORION represents the early stages of an ambitious long-term project. In the future, we envision that people will be able to take advantage of available on-line information systems such that routine tasks can be delegated to the computer as much as possible, thus freeing humans to attend to tasks that truly need their attention. We will explore how we can build systems that can easily customize and adapt to the users' needs and desires. In the process, we will also examine how a system such as ORION could be made more intelligent by incorporating into its decision-making process all available information at its disposal, both about the domain and about the individual user. We believe that ORION has tremendous

Figure 12. Examples of user requests within the ORION domain.

“Call me every weekday morning at 6 a.m. to wake me up.”

“Can you call me at 4:30 in the afternoon to remind me to take my medicine?”

“Call me a half hour before flight 20 lands in Dallas”

“Send me e-mail any time between 6 and 8 a.m. if the traffic on Route 93 Southbound is at a standstill.”

“Call Jim at 4 p.m. at 333-3333 to remind him to go to the dentist.”

potential, although we are still working out the details of some of the more difficult aspects of its design, including user identification, customization, resource allocation, and task completion verification.

Reference

- [1] S. Seneff, C. Chuu, and D. S. Cyphers, “ORION: From On-line Interaction to Off-line Delegation”, to appear in *Proc. ICSLP 2000*, Beijing, China, Oct. 2000.

Facilitating Spoken Language System Creation with SPEECHBUILDER

James Glass, Jef Pearlman and Eugene Weinstein

Over the past decade researchers in the Spoken Language Systems group have developed several different conversational systems. These mixed-initiative systems are research prototypes which require a large amount of expertise in human language technology (HLT) by the developers. This requirement represents an impediment to the rate at which new systems can be deployed, since it limits the pool of developers. To address this issue, we have begun to create a utility called SPEECHBUILDER, which will lower the barrier to novice and intermediate developers, and allow them to rapidly create conversational interfaces using state-of-the-art human language technology components.

From a research perspective, there are several benefits to making the technology more accessible to developers. First, there is an issue of automation; can infrastructure be put in place to fully automate all aspects of conversational system development? This task could involve research in missing technologies, such as improved pronunciation generation of new words, or unsupervised learning of acoustic and linguistic models. Second, there is the issue of portability; how can the technology be made to work robustly with a limited amount of training data from a new domain? Since system development is often tied to data collection, any data collected from new application domains will benefit research in portability by allowing researchers to evaluate robustness of HLT components.

The SPEECHBUILDER utility is intended to allow people unfamiliar with speech and language processing to create their own speech-based application. The initial focus of SPEECHBUILDER has been to allow developers to specify the knowledge

representation and linguistic constraints necessary to automate the design of speech recognition and natural language understanding. To do this, SPEECHBUILDER uses a simple web-based interface which allows a developer to describe the important semantic concepts (e.g., objects, attributes) for their application, and to show, via example sentences, what kinds of actions are capable of being performed. Once the developer has provided this information, along with the URL to their CGI-based back-end application, they can use SPEECHBUILDER to automatically create their own spoken dialogue system which they, and others, can talk to in order to access information.

Architecture

A SPEECHBUILDER application has two basic parts: first, the HLTs which perform speech recognition, language understanding etc., and second, the application back-end which takes a semantic representation produced by the language understanding component and determines what information to return to the user. The HLTs are automatically configured by SPEECHBUILDER using information provided by the developer, and run on compute servers residing at MIT. The back-end consists of a program (e.g., Perl script) created by the developer using the Common Gateway Interface (CGI) protocol, running on a CGI-capable web server anywhere on the Internet. The semantic representation produced by the HLTs takes the form of conventional CGI parameters which get passed to the back-end via standard HTTP protocols.

There are four CGI parameters which are currently used by SPEECHBUILDER: text, action, frame, and history. As may be

surmised, the text parameter contains the words which were understood from the user, while the action parameter specifies the kind of action being requested by the user. The frame parameter lists the semantic concepts which were found in the utterance. In their simplest form, semantic concepts are essentially key/value pairs (e.g., color=blue, city=Boston, etc). More complex semantic concepts have hierarchical structure such as:

```
time=(hour=11,minute=30,xm=AM),
item=(object=box,beside=(object=table))
```

The following examples illustrate possible action and frame values for different user queries:

<i>turn on the lights in the kitchen</i>
action=set&frame=(object=lights, room=kitchen, value=on)
<i>will it be raining in Boston on Friday</i>
action=verify&frame=(city=Boston,day=Friday,property=rain)
<i>are there any chinese restaurants on Main Street</i>
action=identify&frame=(object=(type=restaurant, cuisine=chinese, on=(street=Main,ext=Street)))
<i>I want to fly from Boston to San Francisco arriving before ten a m</i>
action=list&frame=(src=BOS,dest=SFO, arrival_time=(relative=before,time=(hour=10,xm=AM)))

Since a CGI program does not retain any state information (e.g., dialogue), the history parameter enables a back-end to provide information back to the HLT servers that can be used to help interpret subsequent queries. For example, in the following exchange the history parameter is used to keep track of local discourse context:

<i>what is the phone number of John Smith</i>
action=identify&frame=(property=phone,name=John+Smith)
<i>what about his email address</i>
action=identify&frame=(property=email) &history=(property=phone,name=John+Smith)
<i>what about Jane Doe</i>
action=identify&frame=(name=Jane+Doe) &history=(property=email,name=John+Smith)

Knowledge Representation

Semantic concepts and linguistic constraints are currently specified in SPEECHBUILDER via keys and actions. Keys usually define classes of semantically equivalent words or word sequences, so that all the entries of a key class should play the same role in an utterance. All concepts which are expected to reside in a frame must be a member of a key class. The following table contains example keys.

Key	Examples
color	red, green, blue
day	Monday, Tuesday, Wednesday
room	living room, dining room, kitchen
appliance	living room, dining room, kitchen

Actions define classes of functionally equivalent sentences, so that all the entries of an action class perform the same operation in the back-end. All example sentences will generate the appropriate action CGI parameter if they are spoken by the user. SPEECHBUILDER will generalize all example sentences containing particular key entries to all the entries of the same key class. SPEECHBUILDER also tries to generalize the non-key words in the example sentences so that it can understand a wider range of user queries than were provided by the developer. However, if the user does say something that cannot be understood, the action CGI parameter will have a value of "unknown", while the frame parameter will contain all the keys which were decoded from speech signal. The following table contains example actions.

Action	Examples
identify	<i>what is the forecast for Boston what will the temperature be on Tuesday I would like to know today's weather in Denver</i>
set	<i>turn the radio on in the kitchen please can you please turn off the dining room lights turn on the tv in the living room</i>
good_bye	<i>good bye thank you very much good bye see you later</i>

SPEECHBUILDER allows the developer to build a structured grammar when this is desired. To do this, the developer needs to bracket parts of some of the example sentences in the action classes, in order that the system may learn where the structure lies. For example, a bracketed sentence might look like “Please put source=(the blue box) destination=(on the table in the location=(kitchen)).” Based on the bracketing, SPEECHBUILDER creates hierarchy in the meaning representation and the frame parameter. Hierarchy can be recursive, or conversely, can be flattened so that sub-structure is removed from the final frame parameter. Note that bracketing a sentence only involves pointing out the hierarchy as the keys are still automatically discovered by SPEECHBUILDER.

Web Interface

The first version of SPEECHBUILDER has a web interface which allows developers to edit their domains from anywhere on the Internet. The utility allows a developer to modify or delete any of the applications which they have previously created, or create new ones. To create a speech application, a developer needs to provide to SPEECHBUILDER 1) a comprehensive set of semantic concepts, and example queries for their particular domain (specified in terms of keys and actions), and 2) the URL of a

CGI script which will take the CGI parameters produced for a user query, and provide the appropriate information. Once this has been done, the developer 1) presses a button for SPEECHBUILDER to compile the information for their application into a form needed by the human language components, 2) presses another button to start the human language components for their application running (on an MIT web server), and 3) calls the SPEECHBUILDER developer phone number and starts talking to their system.

The concepts and actions specified by the developer are stored in an XML representation which is stored on our local file system. Since the SPEECHBUILDER utility is a CGI script, the file is modified every time changes are made to the domain. If a developer wishes to edit the XML file themselves, it is possible to download the XML file by selecting that option at the upper left of the SPEECHBUILDER utility. Similarly, it is possible to upload an XML file into the user’s SPEECHBUILDER directory.

In addition to specifying constraints and example sentences for their application domain, the developer needs to build the back-end which will provide the actual domain-specific interaction to the user. To do this, the developer needs to have access to a CGI-capable web server, and place the script to be used at a URL matching the one specified to SPEECHBUILDER. Because of the flexibility of CGI, it doesn’t matter whether the CGI back-end is actually a Perl script, a C program pretending to be a web server itself, an Apache module, or any other particular setup, as long as it adheres to the CGI specification. All of our testing to date has been done using Perl and CGI.pm. We provide each developer with a sample

application domain when they register, and provide a useful Perl module for parsing the semantic arguments for developers creating their CGI script.

Human Language Technology

The current focus of SPEECHBUILDER has been on robust understanding so that the discourse and dialogue components used for our mainstream systems (e.g., JUPITER, MERCURY, etc) are not yet incorporated. In addition the language generation component is only used to create CGI parameters, and has not been used for response generation. These activities are the subject of ongoing and future research.

In the area of language understanding, the current interface with the speech recognizer is via the conventional N -best interface. As in all our other systems, a robust parsing mechanism is used to analyze the utterance if no full parse is found. For speech recognition, a bigram language model is derived from the language understanding component using a hierarchical n -gram structure [1]. Acoustic models are trained from telephone-based speech collected from all our domains. A phone-based out-of-vocabulary model is incorporated to more robustly identify spoken words which were not specified by the developer [2].

Status and Plans

As of this date, the SPEECHBUILDER utility has been used internally to create domains in the control of home appliances, as well as access to music and personnel information. Once the necessary infrastructure to support multiple developers has been put in place, we plan to expand the pool of developers to include people outside of our group.

One of the next phases of research will be to re-design our discourse component so that it may be used by SPEECHBUILDER. Future work will also develop an interface to create mixed-initiative dialogues which can automatically interface with our dialogue module. Finally, we would like to develop an interface for our language generation component, so that we can begin to develop multilingual conversational interfaces with SPEECHBUILDER without having to modify the application back-end.

References

- [1] C. Wang, S. Cyphers, X. Mou, J. Polifroni, S. Seneff, J. Yi and V. Zue, "MUXING: A Telephone-Access Mandarin Conversational System " to appear *Proc. 6th International Conference on Spoken Language Processing*, Beijing, China October 2000.
- [2] I. Bazzi and J. Glass, "Modeling Out-of-Vocabulary Words for Robust Speech Recognition" to appear *Proc. 6th International Conference on Spoken Language Processing*, Beijing, China October 2000.

Finite-State Transducer-Based Unit Selection for Concatenative Speech Synthesis

Jon Yi, James Glass, and Lee Hetherington

Corpus-based concatenative methods and unit selection mechanisms have recently received increasing attention in the speech synthesis community (e.g., [3,5]). Our previous work with speech synthesis has focused on using unit selection and waveform concatenation techniques to synthesize natural-sounding speech for constrained spoken dialogue domains [6]. Following Hunt and Black [3], we organized the synthesis constraints of unit selection into concatenation and substitution costs, which essentially prioritize where the speech signal is spliced and which units are appropriate for concatenation. Our concatenative synthesizer, called `ENVOICE`, is phonologically-based (i.e., symbolic), uses phones as the fundamental synthesis unit, and selects variable-length segments for concatenation. When combined with a domain-dependent corpus, it has produced very natural sounding speech for several of our spoken dialogue domains.

Our recent work has revolved around developing a more general framework for this synthesizer that would be easier to maintain, extend, and deploy. Following our successful use of finite-state transducers (FSTs) for speech recognition [2], we have also converted our synthesizer to an FST-based representation. Synthesis is now modeled as a composition of five FST components: the word sequence, W , a lexicon, L , containing baseform pronunciations, a set of phonological rules, P , a set of transition phones, T , (used primarily for search efficiency), and a synthesis component, S , which maps all possible phone sequences to waveform segments for a given speech corpus. Unit selection is accomplished with a Viterbi or N -best search. Just as we, and others, have found for speech

recognition, the FST formulation provides clarity, consistency, and flexibility. In our case, it also allows leveraging off of our previous work with FST-based speech recognition (e.g., pronunciation modelling and search).

One of the main challenges of converting our phonological synthesis framework to an FST representation was determining an efficient structure for the synthesis FST, S . Our solution was to introduce a series of domain-independent intermediate layers, which efficiently encapsulate both the substitution and concatenation costs between every phonetic segment in a speech corpus. This structure has the property that the size of the intermediate layers is fixed, so that the size of the FST grows linearly with the size of the corpus. This allows us to avoid pruning mechanisms which would need to be implemented if we had directly connected every segment in the corpus [1]. Furthermore, this synthesis framework incorporates both concatenation and substitution costs as part of the FST, and not just concatenation costs.

FST Representation

One of the key reasons we adopted an FST representation is its ability to completely and concisely capture finite-state constraints. Given a permissible input sequence, an FST can generate a graph of output sequences [4]. The process of transforming an input language into an output language is guided by the states and arcs specified in the FST topology. Arcs have optional input and output labels (the absence of a label is represented by the ϵ symbol), and can also have weights associated with them. For example, a lexical dictionary can be implemented with an FST

that maps words to phonemes, with weights possibly being used to model alternative pronunciations. Since FSTs can be cascaded in succession to effect further mappings, a second phoneme-to-phone FST could be used to transduce the phonemic sequence into a phonetic sequence. The overall composition would then map words to phones.

With this representation, unit selection for concatenative speech synthesis can be modeled as a mapping from words to waveform segments. Five FST components, W , L , P , T , and S , perform the intervening steps. This factoring allows us to independently model, design, and refine the underlying processes. When combined with a search, these components work in succession from an input sequence of words to produce a sequence of waveform segments suitable for concatenation. For more details of the individual FSTs, see the student report by Jon Yi.

Search

The role of the search component is to find the least-cost sequence of speech utterance segments for a given text input. Specifically, the search finds the least-cost path through the composition of $W \circ (L \circ P) \circ T$ with S . In keeping with our parallelism between recognition and synthesis, the search we use is essentially the same Viterbi-style dynamic programming beam search that is used for recognition, except that different graphs are searched. For synthesis, we optimize $W \circ (L \circ P) \circ T$ and then walk through its states in topological order, exploring compatible arcs in S at the same time. Pruning consists of dynamic programming pruning plus score- and count-based beam pruning, which are tuned to achieve

real-time synthesis.

To reduce latency when synthesizing a long system response, we break the response into chunks separated by sentence boundaries and explicitly referenced pauses or waveform segments called “shortcuts” (described in the next section). Since the state of the system is known at these boundaries, the searches for each chunk can be performed separately, allowing us to perform waveform output for one chunk while performing the search for a subsequent chunk.

Implementation

Based on the structures and algorithms described in the previous section, we have developed a set of software tools and servers for working with this new FST framework. These tools encompass the steps that are performed in the assembling, testing, and running of an FST-based concatenative speech synthesizer. We use command-line utilities for lexicon creation, constraints compilation, and corpus instrumentation, as well as for synthesis testing. The lexicon creation process reuses tools from our speech recognizer. The constraint kernel of the synthesis FST is compiled from substitution and concatenation costs matrices. The synthesis FST is then populated with phones from a corpus of time-aligned waveforms. FST synthesis can then be tested with arbitrary sequences of in-vocabulary words.

We have integrated FST synthesis as networked servers into the GALAXY Communicator architecture which we use for all of our spoken dialogue systems. Two servers fulfill the text-to-speech conversion component, and handle the separate tasks of unit selection and waveform concatenation.

Based on a client-server architecture, they communicate with a central hub which coordinates all tasks in a conversational system. The two servers are pipelined and perform synthesis at speeds sufficient for interactive purposes.

Within our GALAXY Communicator implementation, the first step of synthesis actually begins in our natural language generation server, GENESIS (c.f. page 43). GENESIS recursively expands internal meaning representations into text strings which can be displayed directly on a display, or sent to a synthesis server. The server relies on a message or template file, a lexicon, and a set of rewrite rules to perform generation. For synthesis, the message file is identical to that used for text generation. The lexicon can optionally be modified to expand abbreviations, or explicitly represent waveform segments. These synthesis “shortcuts” allow the developer to bypass the search when desired, and provide backwards compatibility with our earlier word and phrase concatenation work. Another feature of GENESIS which we use for synthesis is the ability to specify features for entries in the lexicon. We have used this to help select words and syllables with the correct prosodic context.

The GENESIS rewrite rules can be used to perform text preparation for synthesis beyond what may be needed for text generation. For example, we have designed regular-expression rules that rewrite flight numbers and times originally in numerical form into written form (e.g., 6425 -> sixty four twenty five, 11:05 -> eleven oh five). This configuration performs the responsibilities typically assumed by a TTS text pre-processing stage. Because it is part of the generation component, it offers increased

accuracy (e.g., unambiguous abbreviation expansion) and flexibility when developing multiple domains and languages.

In the next step of the synthesis chain, the unit selection server receives pre-processed text to synthesize from the natural language generation component. The word sequence is converted into a phonetic sequence by the lexical FST and prepared for searching by the transition label FST. If the word sequence is interrupted with waveform segment “shortcuts”, phonetic context must be maintained before and after the waveform segment to ensure correctness of the search.

Based on the results of the unit selection search, the waveform concatenation component receives instructions to concatenate the appropriate waveform segments. Currently, concatenation is performed without signal processing. For performance considerations, the waveform concatenation server loads the entire corpus of utterances from disk into memory at startup time. As concatenation instructions are received, waveform samples are streamed to the output audio server. The waveform concatenation and output audio servers can be co-located for efficiency.

We have converted several of our domains to use the ENVOICE synthesizer we have developed [6]. The most recent system consists of the MERCURY air travel domain for flight information and pricing (c.f. page 9). Synthesizer development typically begins once the natural language generation component has been completed for a displayful system. Since the synthesizer currently relies on a domain-dependent corpus, the most time-consuming process is usually the design of a set of prompts to be read. As utterances are recorded, they can

be transcribed with a speech recognizer and inserted into the synthesis FST.

For domain specific synthesis, we have used both manual and semi-automatic means of designing recording prompts. For example, static responses are recorded as a whole. For covering responses with more dynamic content, we use an underlying generation template and fill it in with different vocabulary items, such as numbers, and names of cities and airlines. We have also experimented with semi-automatic means of selecting recording prompts. In the past, for the purpose of synthesizing proper nouns, we have used iterative, greedy methods to compactly cover an inventory of stress-marked, syllable-like units [6].

There are many other issues which we plan to address in future work. In order to reduce the abruptness of some concatenation artifacts, we have begun to explore the use of signal processing techniques to modify both fundamental frequency and segment duration. To date, we have taken advantage of the constrained nature of outputs in our conversational domains, and have avoided the use of any kind of prosodic generation module. Prosody has mainly been incorporated at the lexical level in our GENESIS language generation module, and with our ongoing design of a general corpus for the natural-sounding synthesis of arbitrary words (e.g., proper nouns). We would like to investigate corpus-based prosodic generation in future work, however.

Finally, we are interested in developing synthesis capabilities for languages other than English, and are actively working on a version for Mandarin Chinese in a weather information domain (c.f. page 27). This system is currently using syllable onsets and

rhymes as the fundamental synthesis units with tokenized phrases as the lexical representation. We also have plans to work on Spanish and Japanese synthesizers in the near future.

References

- [1] M. Beutnagel, M. Mohri, and M. Riley, "Rapid Unit Selection from a Large Speech Corpus for Concatenative Speech Synthesis," *Proc. Eurospeech*, 607-610, Budapest, Hungary, 1999.
- [2] J. Glass, T. J. Hazen, and L. Hetherington, "Real-time Telephone-based Speech Recognition in the JUPITER domain," *Proc. ICASSP*, 61-64, Phoenix, AZ, 1999.
- [3] A. J. Hunt and A. W. Black, "Unit Selection in a Concatenative Speech Synthesis System Using a Large Speech Database," *Proc. ICASSP*, Atlanta, GA, 373-376, 1996.
- [4] E. Roche and Y. Shabes (eds.), *Finite-State Language Processing*, MIT Press, 1997.
- [5] Y. Sagisaka, "Speech Synthesis by Rule Using an Optimal Selection of Non-uniform Synthesis Units," *Proc. ICASSP*, 679-682, New York, 1988.
- [6] J. Yi and J. Glass, "Natural-sounding Speech Synthesis Using Variable-length Units," *Proc. ICSLP*, 1167-1170, Sydney, 1998.

Barge-in

Nikko Ström

An important aspect of conversational system design is the degree of naturalness afforded the user in interacting with the system. However, it is difficult to design a system which can deal with the complex nature of turn taking in a way that mimics natural human-human interactions. While it is desirable that the user be permitted to interrupt the system at any time, in practice this poses some technical difficulties that may actually lead to intrusive disruptions of the dialogue. The main problem is that it is extremely difficult to design a capability to detect true interruptions without ever mistaking random environmental noises as speech. One approach is to terminate the verbal response only after carefully verifying that the detected speech “makes sense,” i.e., that the recognition and understanding components can interpret it as meaningful speech. However, this approach suffers from two problems: (1) the user perceives that the system is unresponsive to their interruption, due to the perceptibly long delay before it stops talking, and (2) the superposition of system speech intermixed with user speech causes a degradation in recognition performance, because the echo cancellation signal processing is not 100% effective.

Given the above considerations, we have implemented a barge-in capability within the telephony server of our GALAXY system which has a sophisticated model permitting the system to respond immediately to perceived interrupts, and then continue in a graceful and intuitive fashion if the interrupt turns out to be a false alarm. The system is designed as follows:

1. As soon as a candidate input speech signal is detected, stop talking.
2. Process the recorded signal through recognition and understanding.
3. If recognition confidence is sufficiently high and the understanding component produces a meaningful semantic frame, proceed on to the next turn.
4. Otherwise, the system speaks a filled-pause signal (“um”), and continues talking, beginning at the most recent phrase boundary before the stopping point.

This approach allows the barge-in detector to occasionally make mistakes, since the recovery process is natural and non-disruptive.

The detection is based on a threshold on the amount of energy detected at the fundamental frequency of voicing. A simple F0 detection algorithm provides a frequency estimate, and the energy in the autocorrelation coefficient at that frequency is the thresholded parameter.

There may be times when it is desirable that the system disallow barge-in, for example if a disclaimer is presented, or perhaps there is an advertisement that the listener is required to hear. In these cases, the system communicates to the user that barge-in is disallowed by “raising its voice.” This percept is implemented by a simple high-frequency boost combined with a gain in the total energy. This prosodic signal is very clear and effective in communicating to the user the desired message, that barging in is not possible at this point.

T H E S I S

R E S E A R C H

GENESIS-II

Lauren Baptist

Language generation is a fundamental component of dialogue systems. In the GALAXY conversational systems, the generation component, GENESIS, transforms a meaning representation into a string in a target “language” (e.g., English, Japanese, SQL, speech waveforms). GENESIS’ role has slowly expanded over time, and, consequently, the original system is poorly suited to some of the tasks for which it has been adapted. In many cases, GENESIS exerts too much control by “hardwiring” certain features, such as conjunction generation and wh-query movement, making it difficult, if not impossible, to generate appropriately in some situations. Furthermore, as we discovered through our experiences with multilingual system development, GENESIS’ method for ordering constituents was often too constraining, leading to awkward realizations in the target language. Finally, the original system has a somewhat idiosyncratic specification, using different generation mechanisms for clauses, predicates, topics, lists, and keywords.

Over the past year, we have implemented a new version of GENESIS to resolve many of the original system’s idiosyncrasies and shortcomings. The resulting system, GENESIS-II, has fulfilled our main goals, which were to provide very straightforward methods for simple generation tasks, while also supporting the capability of handling more challenging generation requirements, such as movement phenomena, propagation of linguistic features, structural reorganization, generation from lists, and the context-dependent specification of word sense.

In GENESIS-II, generation is controlled by a set of generation rules in conjunction with a lexicon mapping terminal strings to their surface form realization. In defining the

syntax of the generation rules, we have focused on creating an expressive language with generalized mechanisms. This was achieved by carefully designing notations and commands in a generic way, so that they would enjoy wider utility. Thus, the same generation mechanism is used for clauses, predicates, topics, and keywords. Furthermore, users can explicitly (but recursively) specify the ordering of all constituents in the target string, allowing for the correct generation of strings that were impossible to generate in the original system. Similarly, notations and commands were carefully designed such that their utility would extend beyond the original requirement to other related aspects. For example, we added selectors to allow for context-sensitive word-sense disambiguation, which were also effective in prosodic selection for speech synthesis needs.

GENESIS-II also provides generic mechanisms for handling conjunctions and wh-query movement. Since the mechanisms for wh-movement involve reorganizing frame hierarchies, they also turned out to be useful in the generation of foreign languages with substantially different syntactic organizations.

GENESIS-II allows frames to be grouped into class hierarchies associated with common generation templates, resulting in a significant size reduction over the corresponding rules in GENESIS-I. For example, we were able to reduce the rule files for SQL in the JUPITER weather domain by 50%. This generality also leads to more efficient porting to new languages.

We have thus far used GENESIS-II in a number of specific domains and languages, both formal and natural. In our JUPITER domain, weather reports are being trans-

lated into three languages besides English: Spanish, Japanese, and Mandarin Chinese. The quality of the translation is greatly improved over what could have been achieved using the original GENESIS system. In the realm of formal languages, we use GENESIS-II both to convert a linguistic frame into an E-form and to generate database queries, often represented in SQL. In our MERCURY flight reservation domain, common generation rules are used for both speech and text generation, where selected entries in the speech lexicon can map directly to pre-recorded waveforms, selected for prosodic context when appropriate.

Reference

- [1] L. Baptist. *GENESIS-II: A Language Generation Module for Conversational Systems*. MIT Department of Electrical Engineering and Computer Science. S.M. thesis expected August 2000.

Modeling Out-of-Vocabulary Words for Robust Speech Recognition

Issam Bazzi

Current speech recognizers are capable of recognizing only a fixed set of words referred to as the recognizer vocabulary. No matter how large this vocabulary is, there are always out-of-vocabulary (OOV) words that cannot be correctly recognized. OOV words are a common occurrence in many speech recognition applications, and are a known source of recognition errors. For example, in our JUPITER weather information domain the OOV rate is approximately 2%, and over 13% of the utterances contain OOV words [1]. JUPITER utterances containing OOV words have a word error rate (WER) of 51%, while those containing only in-vocabulary words have a much lower WER of 10.4%. Although part of the increased WER on these OOV data is due

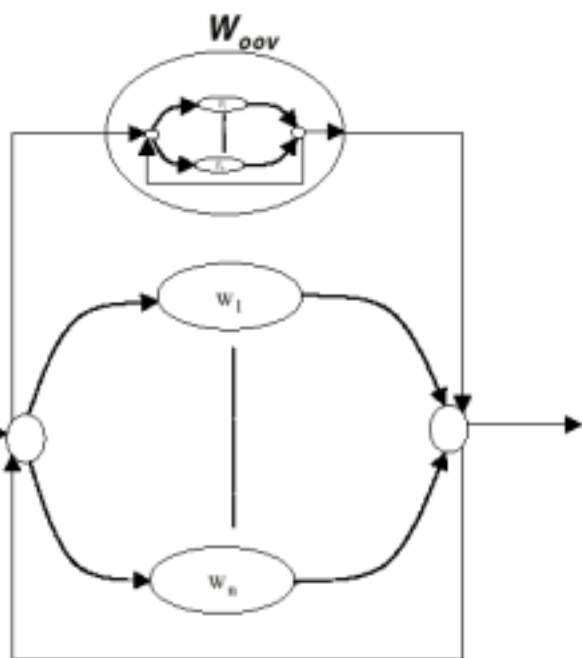
true that OOV words contribute to the increased WER. In this work, we introduce a tactic for incorporating an explicit OOV word model as part of the recognizer itself to allow for OOV detection and recognition.

To incorporate an explicit OOV model into the recognizer we start with a word-based recognizer. The search space for recognition can be viewed as finding the best path(s) in the composition

$$S = P \circ L \circ G,$$

where P represents the scored phonetic graph, L is the lexicon mapping pronunciations to lexical units, and G is the language model. Since an OOV word can consist of any sequence of phones (subject to language

e



sequences during recognition. One of the simplest word models is a phone recognizer. In FST terms, a phone recognizer can be represented as:

$$S_p = P \circ L_p \circ G_p$$

Where L_p and G_p are the phone lexicon and grammar, respectively.

To incorporate this OOV model into the word recognizer, we create a hybrid recognizer by adding an OOV word to the baseline word recognizer's vocabulary. The underlying model for this OOV word is the generic word model. Figure 13 shows how the word search space is augmented with the generic word model. We simply allow the search to have transitions into and out of the generic word model.

The transition into the generic word model is controlled via an OOV penalty (or cost) C_{ooV} . This penalty is related to the probability of observing an OOV word and is used to balance the contribution of the OOV phone grammar to the overall score of the utterance. The hybrid recognizer can be represented with FSTs as follows:

$$S_H = P \circ (L \cup (L_p \circ G_p \circ T_{ooV})) * \circ G'$$

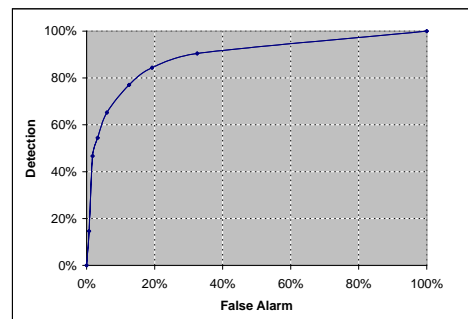
where S_H is the hybrid search space. T_{ooV} is the topology of the OOV word. G' is simply the same as G except for the extra

unknown word in the vocabulary.

The experiments we conducted for this work are within the JUPITER weather information domain [1]. The baseline system is a word-based system that uses a set of context-dependent diphone acoustic models, whose feature representation was based on the first 14 MFCCs averaged over 8 regions near hypothesized phonetic boundaries [2]. Diphones were modeled using diagonal Gaussians with a maximum of 50 mixtures per model. The word lexicon consisted of a total of 2,009 words, many of which have multiple pronunciations. Bigram language models were used both at the word-level, as well as at the phone-level for the OOV model. The main goal of these experiments was to demonstrate whether we can detect OOV words without significantly degrading the performance of the word recognizer on IV utterances. For this reason we measured word error rates (WERs) and OOV false detection (alarm) rates on In-Vocabulary. We also measured the OOV detection rate on the OOV test data to see how well we could detect OOV words.

For the series of experiments we present here, we varied the OOV penalty C_{ooV} . Figure 14 shows the Receiver Operating Characteristics (ROC) curve for several values of C_{ooV} . Figure 15 shows the WER for the IV test set as the false alarm rate increases on the IV data. As expected the WER increases with the increase in the false alarm rate. The results we obtained so far are quite encouraging. With a very simple generic word model, we were able to detect around half of the OOV words (47%) with a very small degradation in WER (0.3%) as well as a low false alarm (1.3%).

Figure 14. ROC curve for OOV detection



For our future work, we are working on incorporating a probabilistic duration model for OOV words. This duration model will require a minimum number of phones for an OOV word as well as probability scores for different word lengths. Another aspect of the approach we are working on is the use of larger units (syllables) to model the OOV word. Syllables should provide a more robust sub-word unit to model generic words. In addition, we are considering the use of classes of OOV words (instead of only one) such as an OOV model for city names, another for weather terms, and so on.

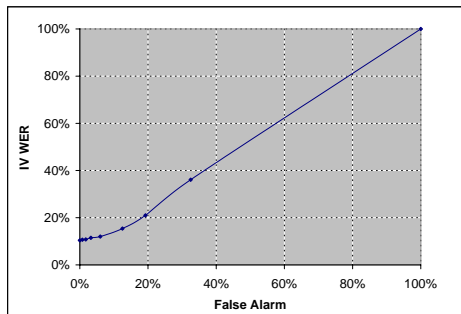


Figure 15. IV WER versus false alarm rate.

References

- [1] J. Glass, J. Chang, and M. McCandless, "A Probabilistic Framework for Feature-based Speech Recognition," *Proc. ICSLP*, pp. 2277–2280, Philadelphia, PA, 1996.
- [2] V. Zue, et al., "JUPITER: A Telephone-based Conversational Interface for Weather Information," *Proc. Speech and Audio Processing*, 88(1), 85–96, January, 2000.

Using Support Vector Machines for Spoken Digit Recognition

Issam Bazzi

Support Vector Machines (SVM) is a novel pattern classification technique that has been successfully applied to many pattern recognition problems in the areas of machine vision, text classification, marketing, and medicine [1]. The SVM framework is based on minimizing the expected risk of making a classification error. In contrast to traditional Bayesian classification methods, where probability density estimation is required for each class, an SVM does not require any density estimation. An SVM transfers the training data to a high dimensional space via a kernel function. Then, it finds an optimal separating hyper-plane that divides this high dimensional space into decision regions. The appeal of the SVM framework for speech recognition is twofold. First, SVMs have a great ability to generalize even with a small number of examples and a high-dimension feature space. Second, a trained SVM needs a small amount of computation to perform the recognition task. As a result, SVM constitutes an inexpensive classifier for systems with a small vocabulary.

In this work, we address the problem of recognizing isolated spoken digits in English. Although the problem is simple, it allows us to examine the potential use of SVMs for speech recognition, the preprocessing required to extract a fixed size feature vector from the temporal speech signal, and the size of the feature space that can be used in this recognition framework.

The input to the SVM classifier is a fixed-size feature vector that represents a whole word (digit). The SVM feature vector is generated from the Mel Frequency cepstral coefficients (MFCCs). However, concatenating all MFCC vectors associated with a word into one SVM feature vector

would result in a variable size feature set since different words have different duration. The approach we adopted is based on removing MFCC vectors at points in time where the MFCCs change the least until only a fixed number of MFCC vectors are left in the feature set.

An SVM is only a binary or a 2-way classifier. For our problem of digit recognition, we need to be able to classify among 10 classes. Performing N -way classification with a 2-way classifier is usually done using either a voting or a scoring mechanism. The first approach we explored was to train 10 binary classifiers. Each of the binary classifiers is trained to recognize a particular digit from all other digits. Having these 10 classifiers, we can test membership of a test point by running it through all of them. If all but one test negative, the one that tests positive will be our classification choice. This is not usually the case since more than one classifier could test positive (or none). Hence, we used a more general scheme: the class of choice is C_{opt} such that:

$$C_{opt} = \arg \max_i d_{SVM}[C_i]$$

Where $d_{SVM}[C_i]$ is the distance of the test point from the separating hyper-plane for classifier C_i . In the second approach, we build classifiers for each pair of digits, hence the term pair-wise classifiers. For 10 digits, the number of classifier is 45 (10-choose-2). Running the test point through each classifier gives us 45 hypotheses for the possible class the point could belong to. The class of choice is C_{opt} such that:

$$C_{opt} = \arg \max_i N_{SVM}[C_i]$$

Where N_{SVM} is the number of classifiers voting for class C_i . Hence deciding on the

class is done via a voting scheme among all classifiers.

For our work we used the 1992 and 1994 OGI speech database. For the experiments we conducted, we used the digit utterances from 133 speakers (two thirds for training and one third for testing). For the baseline, we used a Gaussian mixture classifier, where the best accuracy was 90.7% (9.3% classification error rate).

In one set of experiments, we investigate varying an important parameter of the SVM classifier, the variance of the Gaussian kernel used to map the feature space. Results are shown in Figure 16 for the 1-versus- N classifier with 60 PCA features.

In another set of experiments, we look at the impact of varying the number of PCA features on the accuracy of the 1-versus- N classifier with a Gaussian kernel. Figure 17 summarizes the results. The graphs indicate that we obtain the best performance when we use 45 PCA features.

Our best performing SVM classifier achieves a 94.9% accuracy on digit recognition and outperforms a Gaussian mixture classifier for the same training and testing data. However, current speech recognition systems can achieve 99% or more accuracy for the task of spoken digit recognition for phone numbers [2]. Given the scope of this work, the simplified approach we followed, and the fact that our method was purely based on the acoustic evidence, we believe that our results are quite encouraging as our first attempt to using the SVM framework for speech recognition. Current speech systems rely heavily on various sources of information that we did not utilize in our approach. There is much that can be done to improve the performance such as the use of more training data and incorporating

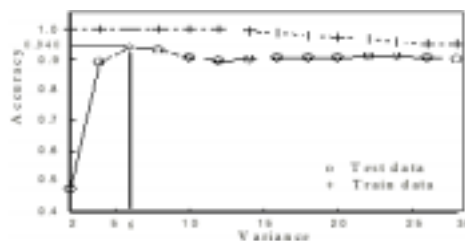


Figure 16. 1-vs- N classifier performance vs Gaussian variance.

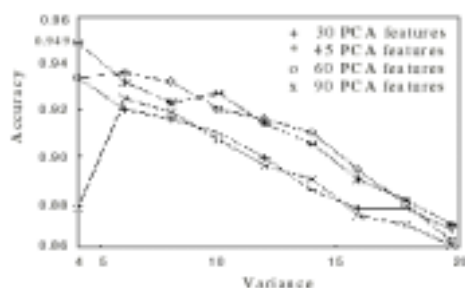


Figure 17. The effect of varying the number of PCA features.

phonological and pronunciation rules and higher-level language constraints into the recognition process.

References

- [1] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, 2(2), 1998.
- [2] J. Makhoul, et al, "State of the Art in Continuous Speech Recognition," *Proceeding Natl. Acad. Sci. USA*, Vol 92 pp.9956-9963, 1995.

Building a Speech Understanding System Using Word Spotting Techniques

Theresa Burianek and Timothy J. Hazen

This work discusses the development and evaluation of a simple word and phrase spotting understanding component within a spoken language system [1]. This use of simple word and phrase spotting techniques is explored as an alternative to the use of TINA, a natural language understanding system which attempts to perform full understanding of input utterances [2]. This topic is explored because the development of word and phrase spotting grammars is simpler and faster than the development of typical full TINA grammars. This approach may be easier for system developers who are not knowledgeable in linguistics or do not have the time to spend to develop a full-understanding grammar.

A word spotting understanding server was implemented within the GALAXY Communicator architecture. The grammar for this server is specified using the Java Speech Grammar Format (JSGF), which can represent basic context free grammars. The server reads the JSGF grammar and converts it into a finite state transducer (FST) representation whose input is words and whose output is concept [key:value] pairs. This approach is quite flexible and can even be incorporated directly into the recognizer without requiring an additional understanding server. The use of an external server allows additional pre-

processing of the recognizer's hypothesized word strings, such as confidence scoring rejection, to occur before the understanding FST is applied.

An evaluation of the capabilities of this server was conducted in the context of the JUPITER weather information domain [3]. The initial word spotting grammar was initially developed in a matter of hours and was refined over the course of a few days based on performance evaluations on a development test set. The word spotting technique was evaluated on an independent test set of JUPITER utterances by comparing its concept [key:value] pair understanding error rate against that of the TINA natural language understanding component [4]. The understanding evaluation also incorporated the use of word rejection based on confidence scoring. The results, presented in the table below, were encouraging and sustained the hypothesis that simple word and phrase spotting grammars can perform adequately for basic understanding tasks. Regardless of the use of word rejection, the full TINA grammar had approximately 25% fewer errors than the word spotting system. However, considerably more effort was expended on developing the TINA grammar and additional effort applied to the word and phrase spotting grammar should result in further reductions in its error rate.

Understanding Approach	Word Rejection Used?	Concept Error Rates (%)			
		Sub	Ins	Del	Total
Word Spotting	No	3.9	18.5	12.9	35.2
TINA	No	2.6	13.8	9	27.1
Word Spotting	Yes	2.6	6.2	17.8	26.6
TINA	Yes	1.8	4.3	13.6	19.7

Table 4. Concept error rates of the word and phrase spotting understanding server versus the TINA natural language server when using and not using rejection of words with poor recognition confidence scores.

References

- [1] T. Burianek, *Building a Speech Understanding System Using Word Spotting Techniques*, Master's thesis, MIT, Expected July 2000.
- [2] S. Seneff, "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics*, vol. 18, no. 1, March, 1992.
- [3] V. Zue, et al, "JUPITER: A Telephone-based Conversational Interface for Weather Information," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, January 2000.
- [4] J. Polifroni, et al, "Evaluation Methodology for a Telephone-based Conversational System," *Proceedings of the International Conference on Language Resources and Evaluation*, Granada, Spain, 1998.

A Three-Stage Solution to Flexible Vocabulary Speech Understanding

Grace Chung

For most conversational systems today, the gap in performance between sentences with out-of-vocabulary words and in-vocabulary sentences remains wide. It is important for systems to detect the incidence of unknown words and handle them intelligently. In the JUPITER weather domain, many user queries pertain to weather information for unknown cities, and currently the system either misunderstands the utterance or rejects it. We envision a future system that can inform the user whenever weather for the city in question is unavailable, and additionally incorporate this previously unseen city name into the lexicon. This system would be able to detect out-of-vocabulary items, dynamically incorporating them during recognition time. In the past, we have proposed a two-stage architecture where a domain-independent first stage, utilizing sublexical models, is interfaced via a phonetic network to a second-stage recognizer that utilizes probabilistic context free grammars for both subword and higher level linguistic knowledge in an integrated search. Recently, we have developed a three-stage implementation that enables our speech understanding system to detect unknown words, hypothesize their phonetic and orthographic transcriptions, and possibly incorporate them without additional training.

The current architecture consists of the following stages. The first stage utilizes sublexical information derived from the ANGIE framework. ANGIE, first introduced in [3], models sublexical phenomena via a hierarchical structure which is trainable and probabilistic. Our work focuses on novel ways to incorporate low-level linguistic models, including those derived from ANGIE, efficiently in the first stage. The output

from the first stage is a phonetic lattice. This guides the second-stage search which yields a word graph, indicating the locations of any detected unknown words. Finally, the third stage involves the application of a probabilistic natural language model by using the TINA natural language system [2].

Our first stage is designed to be flexible so that novel phonetic sequences associated with new words are supported by the language models, while recognition accuracy on in-vocabulary data is maintained. This is accomplished by maximizing linguistic constraints that utilize only low-level domain-independent information. Our methods are facilitated by a system which has adopted finite-state transducers (FSTs) to represent its language models. Previously in [1], we developed a method for encapsulating the hierarchical models of ANGIE within an FST. We have since introduced the column-bigram method, which translates more efficiently the context-free formalism of ANGIE into the more tractable FST representation for integration with a recognizer. After training on in-vocabulary data, this FST is capable of accepting, with non-zero probabilities, novel phonetic sequences pertaining to unknown words. While our previous representation simply memorizes training data and their assigned ANGIE probabilities, the column-bigram FST generalizes across unseen data with similar substructures. Yet it remains a compact representation of ANGIE's rich hierarchical space.

A second feature is the ability to augment the column-bigram FST with grapheme information. We are motivated by the desire to (1) derive letter spellings directly during recognition upon encountering an unknown word, and (2) utilize

grapheme information as an additional source of low-level constraint in the first stage. We have designed an ANGIE grammar that simultaneously captures letter-to-sound information and phonological phenomena in the probability models. A set of hybrid “letter-phonemes” enriches the probability space by encoding both spelling and phonemic information. These units were designed by augmenting the original set of phonemes with carefully chosen characteristics such as spelling, pronunciation, syllable position and stress, (e.g. *ea_l+* is a stressed long vowel with spelling “*ea*.”) In the JUPITER domain, the new enhanced ANGIE grammar leads to perplexity reduction, an indication of the tighter linguistic constraint it offers. This “letter-phoneme” grammar is subsequently used in a column-bigram FST that represents the corresponding ANGIE models.

The third feature of the first stage is the use of an optimized lexical space that has been automatically generated. This is driven by the need for greater efficiency in stage one but at the same time using linguistic models that combine syllable *n*-gram knowledge and sublexical probabilities together with spelling information. Our solution is to derive a novel set of syllable-level units via an automatic procedure which aims to improve the probability likelihood at every iteration. This procedure begins with the column-bigram FST and iteratively builds novel syllable-sized units by concatenating grapheme-based sequences. At each iteration, lexical units are chosen to improve probability modeling and reduce perplexity. Upon achieving convergence, a more compact syllable lexicon is produced.

The phonetic lattice from the first stage constrains a best-first search in the second stage. This recognizer allows the incidence

of unknown words and employs ANGIE’s dynamic parse mechanism to guide phonetic hypotheses. In the case where a novel phonetic sequence is proposed, and an ANGIE parse has succeeded, the associated probability score is returned with an out-of-vocabulary flag. A hypothesized spelling may also be accessed from ANGIE.

In the third stage, TINA parses word graphs constructed during the second stage, which may possibly contain unknown words. The highest scoring sentence hypothesis is selected according to TINA models. This method also computes a meaning representation for further processing in the dialogue system.

This flexible vocabulary system has been implemented in the JUPITER weather information domain and we conducted recognition experiments on utterances containing unknown city names. On an independent test set, we compare word (WER) and understanding (UER) error rate with a baseline recognizer that does not have the capability to handle unknown words. We also experiment with a two-stage variant that employs TINA integrated with ANGIE in the second-stage search [1]. In both cases, TINA has been trained such that unknown words are permitted to occur exclusively when an unknown city category can be admitted in the grammar. Compared with the baseline, we achieve up to 29.3% reduction in WER (from 24.6% to 17.4%) and 67.5% reduction in UER (from 67.0% to 21.8%) in the three-stage system. Such significant improvements are derived from increased in-vocabulary recognition performance combined with success in detecting unknown word occurrences. In a separate pilot experiment, we extract spelling hypotheses directly from our ANGIE

models during recognition. For the subset of utterances where an unknown city is correctly detected, we computed the letter accuracy to be 42.2%. Thus, we demonstrate the possibility of instantaneously incorporating the unknown city.

References

- [1] G. Chung and S. Seneff, "Towards Multi-Domain Speech Understanding Using a Two-Stage Recognizer," in *Proc. Eurospeech '99*, pp. 2655–2658, Budapest, Hungary, September 1999.
- [2] S. Seneff, "TINA: A Natural Language System for Spoken Language Applications," in *Computational Linguistics*, 18(1):61–86, March 1992.
- [3] S. Seneff, R. Lau, and H. Meng, "ANGIE: A New Framework for Speech Analysis Based on Morpho-phonological Modelling," in *Proc. ICSLP '96*, Vol. 1, pp. 110–113, Philadelphia, PA, October 1996.

Lexical Modeling of Non-Native Speech for Automatic Speech Recognition

Karen Livescu

Automatic speech recognizers have been observed to perform drastically worse on non-native speakers than on native speakers. We have investigated the effect of non-native speech on several components of a speech recognizer. Here we describe one aspect of this work, namely a technique for modeling the ways in which non-native speakers pronounce words differently from native speakers [2, 3].

In order to better model non-native pronunciation, or lexical, patterns, we have explored modifications to the lexicon to account for pronunciation variants that non-native speakers are likely to use. Ideally, we would like to collect entire word pronunciations and train their probabilities from real non-native data. However, since there are not enough instances of each word in the non-native training data available to us, we have instead chosen to derive simpler rules from the data, which we then apply to the baseline lexicon. Specifically, we have constrained ourselves in this initial investigation to context-independent phonetic confusion rules, i.e., substitutions, deletions, and insertions of phones irrespective of the neighboring phones. Although context-dependent rules would contain more information, the larger required number of parameters would be difficult to train from the limited amount of available training data.

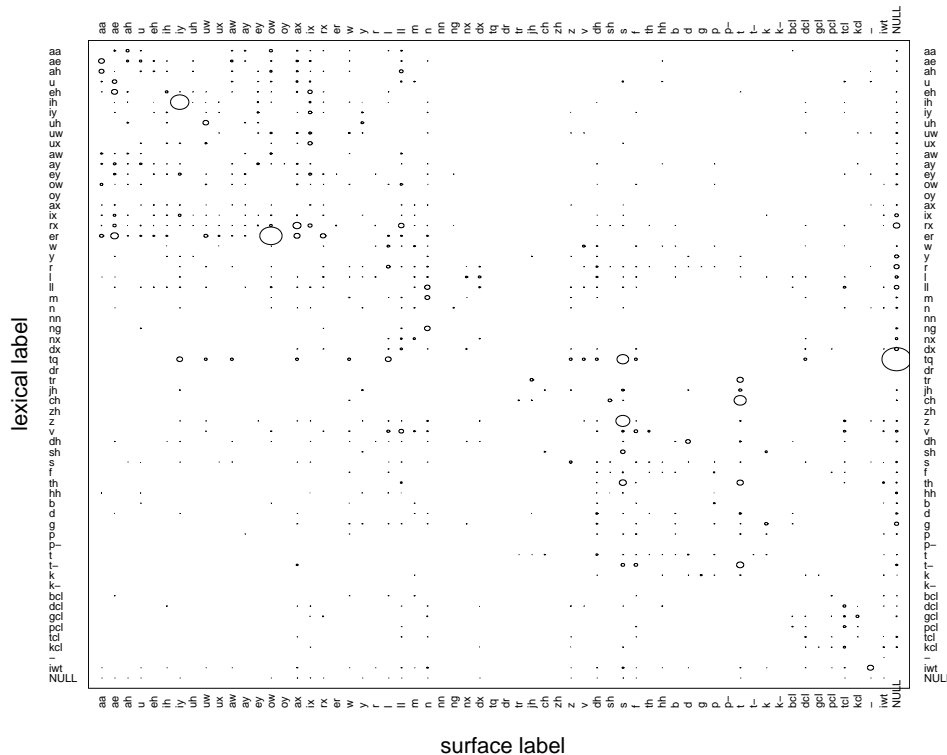
The speech recognizer for these experiments can be represented as a composition of finite-state transducers (FSTs), each of which models one of the components of the recognizer (acoustic-phonetic graph, lexicon, language model). In this framework, phonetic confusion rules can be easily and naturally incorporated into the recognizer by adding a simple

transducer between the acoustic-phonetic and lexicon FSTs. Each of the confusions is associated with a probability of that confusion occurring. This is equivalent to adding a large number of pronunciations to the lexicon, corresponding to applying each of the confusions to each of the baseline pronunciations, with two important distinctions: (1) we need not actually rebuild the lexicon for each experiment, thereby saving the space that would be needed to store all of the lexicons, and (2) we can perform the composition of the confusion FST with the lexicon during recognition, dynamically pruning out unlikely hypotheses and again reducing the space requirements of the recognizer. Using this architecture, then, we can experiment with many different versions of the confusion rules simply by changing the corresponding transducer.

In order to estimate the probabilities of the confusions, we need a phonetic transcription for each training utterance, aligned with the corresponding pronunciation according to the baseline lexicon. In our approach, these transcriptions are generated automatically (as described below) and aligned with the lexicon using an automatic string alignment procedure. Once the alignments are obtained, we estimate the probability of each phonetic confusion from its frequency in the alignments.

We have performed recognition experiments on utterances recorded from users of JUPITER [1], a conversational system providing weather information. For the initial set of experiments, we obtained the phonetic transcription of each non-native utterance by recognizing it with a phonetic recognizer. This produces an errorful transcription, since it does not use any

Figure 18. Bubble plot showing the estimated probability of each phonetic confusion. The y-axis shows the intended (lexical) phone and the x-axis shows the actual (surface) phone used by non-native speakers. The size of a bubble represents how likely the corresponding confusion is. Deletions are represented as confusions from some lexical phone to the “NULL” surface phone, and insertions are confusions from “NULL” to some non-null surface phone. For example, this plot indicates that an [ih], if mispronounced, is most likely to become an [iy]; and a glottalized t, [tq], is most likely to be deleted if mispronounced. The bubbles along the diagonal, corresponding to the probabilities of correct pronunciations, have been removed for clarity.



knowledge of the words in the utterance, but it serves as a starting point. After transcribing the non-native training set in this way, we obtain our first estimates of the confusion probabilities. Recognition experiments with this first iteration of confusion estimates indicated that they were not sufficiently well-trained to improve recognition performance.

However, once initial estimates for the confusion probabilities are obtained, they can be used to produce more accurate transcriptions. Using our knowledge of the word sequence in each training utterance, we combined the baseline lexicon with the initial confusion estimates to obtain the most likely pronunciation for each word. This produced a set of improved transcriptions, from which we computed new estimates of the confusion probabilities.

The resulting probabilities corresponded more closely to our expectations about likely phonetic confusions in non-native speech. Figure 18 shows the final probability estimates in the form of a bubble plot.

Using this new set of confusions, we performed experiments in which we varied the size of the confusion FST by including only the most likely confusions up to a certain pruning threshold. Varying the threshold allowed us to explore the tradeoff between a faster recognizer with fewer allowed confusions and a slower but more accurate recognizer. Figure 19 shows the word error rate of the recognizer for various settings of the pruning threshold, with a threshold of 0 corresponding to including no confusions (i.e. using the baseline lexicon) and 12 corresponding to allowing all of the confusions that occurred in the

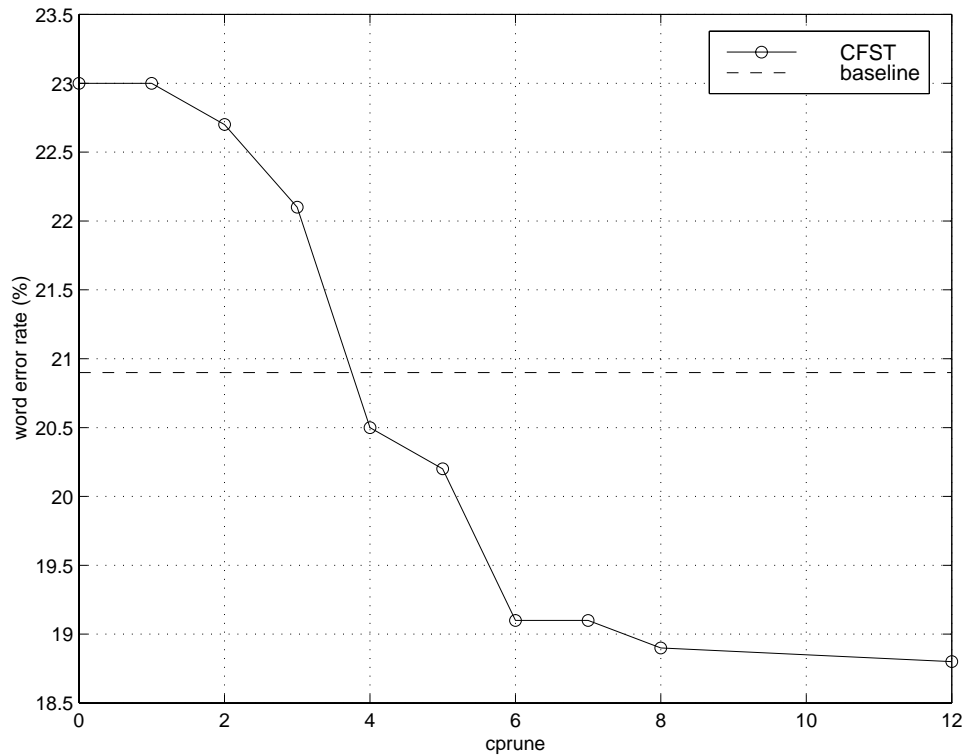


Figure 19. Word error rate on a non-native test set as a function of pruning threshold on the confusion probabilities, compared to a baseline recognizer with no confusions. The threshold c_{prune} corresponds to a negative log probability rather than a probability; therefore, the higher the threshold, the more confusions are allowed. The word error rate is higher than baseline at the lower pruning thresholds because of measures that were taken to save computation time.

training set. The word error rate monotonically decreases as we add more confusions. When all of the confusions are included, the error rate is reduced to 18.8%, a highly significant improvement from the baseline error rate of 20.9%.

These results demonstrate that, if the recognizer knows that an input utterance is non-native, then it can do significantly better by including simple context-independent phonetic confusion rules. These rules can be automatically extracted from a non-native training set and easily incorporated into a finite-state transducer-based recognition architecture. It is noteworthy that using the recognizer that achieved the best performance on non-native speakers did not degrade the performance on native speakers (although it did increase the running time), which is an encouraging sign for future work combining native and non-native

recognition. Other possibilities for future work in this area include performing additional iterations of confusion probability estimation, reducing the running time and memory requirements of the modified recognizer, and modeling context-dependent confusions as more data become available.

References:

- [1] J. R. Glass and T. J. Hazen, "Telephone-based Conversational Speech Recognition in the JUPITER Domain," in *Proc. ICSLP '98*, Sydney, Australia, December 1998.
- [2] K. Livescu, *Analysis and Modeling of Non-Native Speech for Automatic Speech Recognition*. S.M. thesis, MIT Department of Electrical Engineering and Computer Science, September 1999.
- [3] K. Livescu and J. Glass, "Lexical Modeling of Non-native Speech for Automatic Speech Recognition," in *Proc. ICASSP 2000*, Istanbul, Turkey, June 2000.

The Use of Dynamic Reliability Scoring in Speech Recognition

Xiaolong Mou

Speech recognition is usually formulated as a problem of searching for the best string of symbols, subject to the constraints imposed by the acoustic and language models. In implementing such a formulation, systems typically apply the constraints uniformly across the entire utterance. This does not take into account the fact that some units along the search path may be modeled and recognized more reliably than others. One possible way to incorporate reliability information is through word- and utterance-level rejection [1]. However, this approach only provides confidence information after the recognition is done. In contrast, we attempt to dynamically incorporate reliability information into the search phase in order to help the recognizer find the correct path.

In this work, we introduce the notion of dynamic reliability scoring that adjusts the partial path score while the recognizer searches through the composed lexical and acoustic-phonetic network. In our scheme, the recognizer evaluates the reliability of choosing a hypothesized arc to extend the current path by adding a weighted phonetic reliability score to the current path score. The reliability score is obtained from a reliability model that gives the likelihood of extending the current path using one specific arc as opposed to using its immediate competing alternatives. The reliability models are trained from transcribed speech data. First a forced alignment search is conducted to obtain the references to correct paths. Then, for each partial path along the forced path, the score of the arc that extends the forced path, denoted s , and the scores of the arcs that are not in the forced path, denoted M_i , are collected. After that, Gaussian models for the correct

scoring (i.e., scores of arcs that are in the forced path), M_s , and incorrect scoring (i.e., scores of those not in the forced path), M_i , are trained. Figure 20 shows typical trained reliability models M_s and M_i for a given arc in the lexical network. The further apart the two models and the smaller their variances are, the easier it would be to distinguish the correct arc from its immediate competing alternatives while searching.

After all the models are trained, we can obtain reliability measurement on the fly while searching through the network. The reliability measurement is essentially the likelihood that a particular arc in the network with score s is in the right path while its immediate competitors with acoustic scores M_i are not in the right path. It is given by the following formula (assuming the correct and competing paths are independent):

$$\begin{aligned} p(s | M_s) p(t_1, \dots, t_n | M_i) \\ = \frac{p(s | M_s)}{p(s | M_i)} p(s, t_1, t_2, \dots, t_n | M_i) \end{aligned}$$

Because we try to use the reliability score to help the recognizer choose a hypothesis arc from its immediate competing arcs, $p(s, t_1, t_2, \dots, t_n | M_i)$ is a constant factor here, and we can then just use the log domain score $\log(p(s | M_s)/p(s | M_i))$ as the reliability measurement, which saves a lot of computation effort during the search.

The recognizer's acoustic models and reliability models are trained and evaluated in an English weather information domain called JUPITER and a development Mandarin Chinese weather information domain called PANDA. For the JUPITER domain, the training set consists of 24,182 live utterances recorded over the phone and the test set consists of 1,806 utterances randomly

selected from the data collection independent of the training set. Both di-phone and segment models are used, and the reliability models are built on the normalized and combined boundary and segment scores. For the PANDA domain, the training set consists of 1,455 utterances and the test set consists of 244 utterances. Only segment models are used because of insufficient training data, and the reliability models are built on the normalized segment scores.

We have incorporated the reliability scheme into the segment-based, SUMMIT [2] speech recognition system. On the JUPITER test set of 1,806 utterances without out-of-vocabulary (OOV) words, we achieve 9.8% reduction in word error rate (WER) after using the reliability modeling scheme described above. On the PANDA test set of 244 utterances without OOV words, we achieve 12.4% reduction in WER. This demonstrates that reliability models can be used to address the fact that units along the search path are modeled and recognized with different reliability, and to help early recovery of search errors.

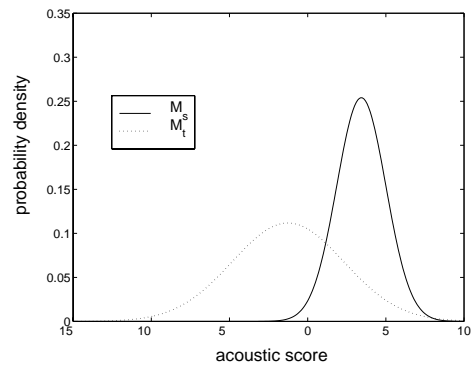


Figure 20. The reliability models M_s (correct scoring) and M_t (incorrect scoring) trained for an arc labeled [t] in the lexical network of the word “want”.

References

- [1] C. Pao, P. Schmid, and J. Glass, “Confidence Scoring for Speech Understanding Systems,” in *Proc. ICSLP '98*, Sydney, 1998.
- [2] J. Glass, J. Chang, and M. McCandless, “A Probabilistic Framework for Feature-based Speech Recognition,” in *Proc. ICSLP '96*, Philadelphia, 1996.

Subword-based Approaches for Spoken Document Retrieval

Kenney Ng

This thesis explored approaches to the problem of spoken document retrieval (SDR), which is the task of automatically indexing and then retrieving relevant items from a large collection of recorded speech messages in response to a user specified natural language text query. We investigated the use of subword unit representations for SDR as an alternative to words generated by either keyword spotting or continuous speech recognition. Our investigation is motivated by the observation that word-based retrieval approaches face the problem of either having to know the keywords to search for a priori, or requiring a very large recognition vocabulary in order to cover the contents of growing and diverse message collections. The use of subword units in the recognizer constrains the size of the vocabulary needed to cover the language; and the use of subword units as indexing terms allows for the detection of new user-specified query terms during retrieval. Four research issues were addressed:

- 1) What are suitable subword units and how well can they perform?
- 2) How can these units be reliably extracted from the speech signal?
- 3) What is the behavior of the subword units when there are speech recognition errors and how well do they perform?
- 4) How can the indexing and retrieval methods be modified to take into account the fact that the speech recognition output will be errorful?

In this thesis, we made the following contributions to research in the area of spoken document retrieval:

- An empirical study of the ability of different subword units to perform spoken document retrieval and their behavior in the presence of speech recognition errors.
- The development of a number of robust indexing and retrieval methods that can improve retrieval performance when there are speech recognition errors.
- The development of a novel spoken document retrieval approach with a tighter coupling between the recognition and retrieval components that results in improved retrieval performance when there are speech recognition errors.
- The development of a novel probabilistic information retrieval model that achieves state-of-the-art performance on standardized text retrieval tasks.

In the following sections, we give a brief summary of the main chapters in the thesis and finally close by mentioning some possible directions for future work.

Feasibility of Subword Units for Information Retrieval

We first explored a range of subword units of varying complexity derived from error-free phonetic transcriptions and measured their ability to effectively index and retrieve speech messages [5,6]. These experiments provide an upper bound on the performance of the different subword units since they assume that the underlying phonetic recognition is error-free. In particular, we examined overlapping, fixed-length phone sequences and broad phonetic class sequences, and non-overlapping, variable-length, phone sequences derived automati-

cally (multigrams) and by rule (syllables). We found that many different subword units are able to capture enough information to perform effective retrieval. We saw that overlapping subword units perform better than non-overlapping units. There is also a tradeoff between the number of phonetic class labels and the sequence length required to achieve good performance. With the appropriate choice of subword units it is possible to achieve retrieval performance approaching that of text-based word units if the underlying phonetic units are recognized correctly. Although we were able to automatically derive a meaningful set of subword “stop” terms, experiments using the stop-list did not result in significant improvements in retrieval performance.

Extracting Subword Units from Spoken Documents

Next, we trained and tuned a phonetic recognizer to operate on the radio broadcast news domain and used it to process the spoken document collection to generate phonetic transcriptions [7]. We then explored a range of subword unit indexing terms of varying complexity derived from these errorful phonetic transcriptions and measured their ability to perform spoken document retrieval. We found that in the presence of phonetic recognition errors, retrieval performance degrades, as expected, compared to using error-free phonetic transcriptions or word-level text units: performance falls to 60% of the clean reference performance. However, many subword unit indexing terms still give reasonable performance even without the use of any error compensation techniques. We also observed that there is a strong

correlation between recognition and retrieval performance: better phonetic recognition performance leads to improved retrieval performance. These experiments establish a lower bound on the retrieval performance of the different subword units since no error compensation techniques are used. We know that there are speech recognition errors, but we are not doing anything about them. Hopefully improving the performance of the recognizer and developing robust indexing and retrieval methods to deal with the recognition errors will help improve retrieval performance.

Robust Indexing and Retrieval Methods

We investigated a number of robust methods in an effort to improve spoken document retrieval performance when there are speech recognition errors [2]. In the first approach, the original query is modified to include near-miss terms that could match erroneously recognized speech. The second approach involves developing a new document-query retrieval measure using approximate term matching designed to be less sensitive to speech recognition errors. In the third method, the document is expanded to include multiple recognition candidates to increase the chance of capturing the correct hypothesis. The fourth method modifies the original query using automatic relevance feedback to include new terms as well as approximate match terms. The last method involves combining information from multiple subword unit representations. We studied the different methods individually and then explored the effects of combining them. We found that using a new approximate match retrieval metric, modifying the queries via automatic

relevance feedback, and expanding the documents with N -best recognition hypotheses improved performance; subword unit fusion, however, resulted in only marginal gains. Combining the approaches resulted in additive performance improvements. Using these robust methods improved retrieval performance using subword units generated from errorful phonetic recognition transcriptions by 23%.

Probabilistic Information Retrieval Model

We presented a novel probabilistic information retrieval model and demonstrated its capability to achieve state-of-the-art performance on large standardized text collections [3]. The retrieval model scores documents based on the relative change in the document likelihoods, expressed as the ratio of the conditional probability of the document given the query and the prior probability of the document before the query is specified. Statistical language modeling techniques are used to compute the document likelihoods and the model parameters are estimated automatically and dynamically for each query to optimize well-specified maximum likelihood objective functions. An automatic relevance feedback strategy that is specific to the probabilistic model was also developed. The procedure automatically creates a new query (based on the original query and a set of top-ranked documents from a preliminary retrieval pass) by selecting and weighting query terms so as to maximize the likelihood ratio scores of the set of documents presumed to be relevant to the query. To benchmark the performance of the new retrieval model, we used the standard ad hoc text retrieval tasks from the TREC-6 and TREC-7 text retrieval confer-

ences. Official evaluation results on the 1999 TREC-8 ad hoc text retrieval task were also reported [1]. Experimental results indicated that the model is able to achieve performance that is competitive with current state-of-the-art retrieval approaches.

Integrated Recognition and Retrieval

We presented a novel approach to spoken document retrieval where the speech recognition and information retrieval components are more tightly integrated. This was accomplished by developing new recognizer and retrieval models where the interface between the two components is better matched and the goals of the two components are consistent with each other and with the overall goal of the combined system. We presented a new probabilistic retrieval model which makes direct use of term occurrence probabilities that can be computed by the recognizer. We then described several ways to compute the desired term probabilities including using the top one recognition hypothesis, using N -best recognition hypotheses, expanding the term set to include approximate match terms, and modifying the speech recognizer to enable it to output the term occurrence probabilities directly. We evaluated the performance of the integrated approach and found that the probabilistic model performs slightly better than the baseline vector space retrieval model, and the addition of automatic relevance feedback resulted in a significant performance improvement. We then measured the retrieval performance of the integrated approach as different methods for estimating the term occurrence probabilities are used. We found that retrieval performance improves as more

sophisticated estimates are used. The best performance was obtained using term occurrence probabilities computed directly from the speech recognizer. The integrated approach improved retrieval performance by over 28% from the baseline. This is compared to an improvement of 23% using the robust methods.

Future Directions

The experimental results presented in this thesis demonstrate that subword-based approaches to spoken document retrieval are feasible and merit further research. There are a large number of areas for extension of this work. One area is to improve the performance of the extraction of the subword units from the speech signal. For example, more training data can be used to improve model robustness and more detailed and complex models can be used to try to capture more information from the speech signal. Another approach would be to modify the speech recognizer to recognize the subword units directly from the speech, rather than constructing them from a phonetic string. Another area of work is to improve the probabilistic information retrieval model. More sophisticated models such as higher order statistical n-gram language models and alternative probability smoothing techniques should be explored. Another interesting and potentially profitable area of research is on information fusion methods for spoken document retrieval [4]. We briefly looked at some simple methods for combining multiple subword unit representations. Although the performance improvements we obtained were small, the method of information combination still holds promise. Finally, the approaches presented in the thesis

should be evaluated on larger sets of data. This includes both the spoken document collection and the training set for the speech recognizer. More data will allow us to build more robust models and to further test the scalability and behavior of our systems.

References

- [1] Harman, D. K. (Ed.) (1999). Eighth Text Retrieval Conference (TREC-8), Gaithersburg, MD, USA. National Institute for Standards and Technology.
- [2] Ng, K. (1998). "Towards Robust Methods for Spoken Document Retrieval." In *Proc. ICSLP '98*, Sydney, Australia.
- [3] Ng, K. (1999). "A Maximum Likelihood Ratio Information Retrieval Model." In *Eighth Text Retrieval Conference (TREC-8)*, Gaithersburg, MD, USA.
- [4] Ng, K. (2000). "Information Fusion for Spoken Document Retrieval." In *Proc. ICASSP '00*, Istanbul, Turkey.
- [5] Ng, K. and V. Zue (1997). "An Investigation of Subword Unit representations for Spoken Document Retrieval." In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Posters: Abstracts, pp. 339.
- [6] Ng, K. and V. Zue (1997). "Subword Unit Representations for Spoken Document Retrieval." In *Proc. Eurospeech '97*, Rhodes, Greece, pp. 1607-1610.
- [7] Ng, K. and V. Zue (1998). "Phonetic Recognition for Spoken Document Retrieval." In *Proc. ICASSP '98*, Seattle, WA, USA, pp. 325-328.

Enabling Spoken Language Systems Design for Non-Experts

Jef Pearlman

For my thesis research I designed and implemented a utility for allowing non-experts to build and run spoken language systems. This involved the creation of both a web interface for the developer and a set of programs to support the construction and execution of the required internal systems. As shown in Figure 21, the utility made use of the underlying GALAXY Communicator architecture. The architecture was augmented by a server which allowed developers to communicate with their CGI-based application remotely over the web.

For this research, we concentrated on configuring the language understanding components of a spoken language system. By learning the required grammar from a set of simple concepts and sentence examples provided by the developer, we

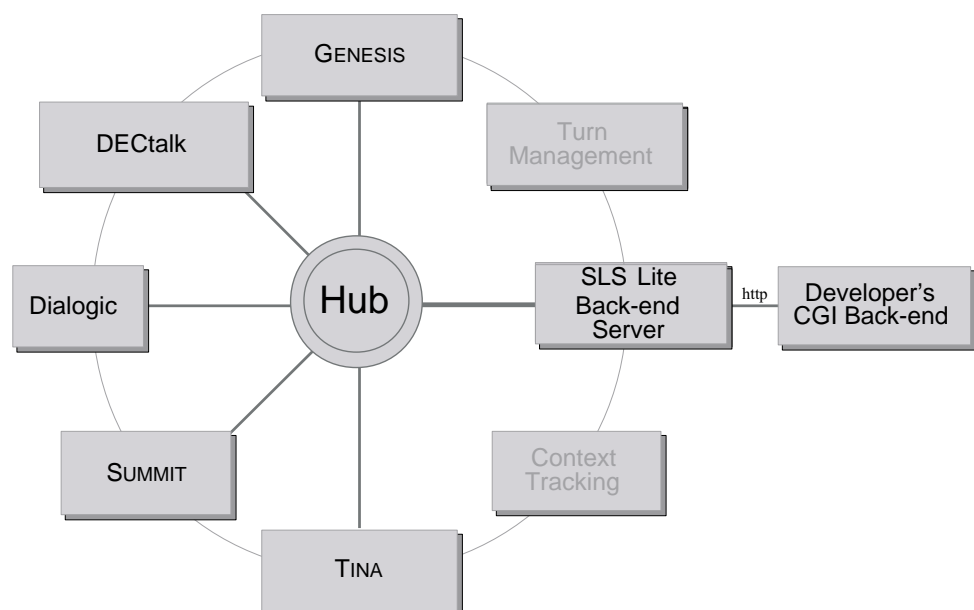
were able to build a system where non-experts could build grammars and speech systems. Developers could also easily specify hierarchy in domains where a more complex grammar was appropriate.

We demonstrated the utility, originally known as SLS-Lite, now called SPEECHBUILDER, by building several prototype domains ourselves, and allowing others to build their own. These included domains for controlling the appliances in a house, querying a directory of people in the MIT Laboratory for Computer Science, manipulating fictional objects, and asking questions about music.

Reference

- [1] J. Pearlman. *SLS-Lite: Enabling Spoken Language Systems Design for Non-Experts*. MIT Department of Electrical Engineering and Computer Science, M.Eng. thesis, expected August 2000.

Figure 21. GALAXY architecture used by SPEECHBUILDER utility.



Discriminative Training of Acoustic Models in a Segment-Based Speech Recognizer

Eric Sandness

In this thesis work we investigated discriminative training of acoustic models within the SUMMIT system [1,2]. In particular, we developed a new technique we called keyword-based discriminative training which focuses on optimizing a keyword error rate, rather than the error rate on all words. We hypothesize that improvements in keyword error rate correlate with improvements in understanding error rates. Keyword-based discriminative training is accomplished by modifying a standard minimum classification error (MCE) training algorithm so that only segments of speech relevant to keyword errors are used in the acoustic model discriminative training. When both the standard and keyword-based techniques are used to adjust Gaussian mixture weights, we find that keyword error rate reduction compared to baseline maximum likelihood (ML) trained models is nearly twice as large for the keyword-based approach. The overall word accuracy is also found to be improved for keyword-based training, and we ran experiments to investigate this phenomenon.

In this work we used an utterance-level MCE criterion [3,4] to train the Gaussian mixture weights. We also attempted to train the mixture means and variances, but training convergence was unreliable and considerably slower. All experiments were carried out in the JUPITER weather information domain, with boundary diphone acoustic models with up to 50 mixtures per model, and a bigram word-class language model.

For each training utterance, complete recognizer scores are computed for the correct word sequence and an N -best list of competing hypotheses. These scores are a sum of acoustic and non-acoustic (i.e.,

lexical and language model) scores. The acoustic scores are updated at each iteration while the non-acoustic scores remain constant.

We make use of what we call “hot boundaries” to perform keyword-based discriminative training. A hot boundary is a boundary (potential phone boundary) where the correct word sequence and competing N -best hypotheses contain a mismatched keyword. (See [1,2] for details.) In regions of the utterance where there are no keywords (correct or hypothesized) or there are no keyword mismatches, there will be no hot boundaries. In this way, hot boundaries represent the regions of the utterance relevant to optimizing the keyword error rate.

The keywords we used were selected by hand. Of JUPITER’s nearly 2,000-word vocabulary, we identified 1,066 keywords, primarily consisting of place names (e.g., Boston, Europe, India), weather terms (e.g., snow, humidity, weather, advisories), and dates/times (e.g., tomorrow, tonight, Thursday, weekend).

On a test set of 500 in-vocabulary utterances, the baseline ML-trained system achieved a keyword error rate of 6.0%. The baseline MCE-trained improved to 5.7%. The keyword-based MCE-trained system achieved more than twice the improvement at 5.2%.

Evaluating the overall word-error rates yielded surprising results. The baseline ML word error rate was 10.4%, and the baseline MCE improved to 9.7%. Surprisingly, the keyword-based MCE-trained system improved even more to 9.3%.

We did not expect the keyword-trained system to do as well over all words, and performed some follow up experiments to

deduce the reason. We hypothesized that many of the non-keywords (e.g., function words) are poorly articulated and thus may contribute to inferior acoustic models. In one experiment, we set the keywords to be all words except for 148 manually chosen function words. Training with this set of keywords produced an overall word error rate of 9.4%, very similar to the 9.3% we achieved with the manually selected keywords.

Other work in this thesis included the implementation of Gaussian selection [5,6]. The goal of Gaussian selection is to speed up computation of Gaussian mixture models by preselecting a reduced subset of mixtures to evaluate for each region of the feature space. We used binary vector quantization with 512 code words to divide the feature space. For each code word, mixtures whose mean is within a distance threshold of the code word mean are noted, with at least one mixture for each model associated with each code word. At run time, a feature vector is rapidly quantized and the relevant mixture components looked up and evaluated. Overall, the use of Gaussian selection speeds up SUMMIT in the JUPITER domain by a factor of 1.8 with the pruning thresholds used.

References

- [1] E. D. Sandness, *Discriminative Training of Acoustic Models in a Segment-Based Speech Recognizer*, M.Eng. thesis, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 2000.
- [2] E. D. Sandness and I. L. Hetherington, "Keyword-Based Discriminative Training of Acoustic Models," to appear in *Proc. ICSLP '00*, Beijing, vol. 3, pp. 135-138, October 2000.
- [3] W. Chou, B.-H. Juang, and C.-H. Lee, "Segmental GPD Training of HMM based Speech Recognizer," in *Proc. ICASSP '92*, San Francisco, pp. 473-476, March 1992.
- [4] F. Beaufays, M. Weintraub, and Y. Konig, "Discriminative Mixture Weight Estimation for Large Gaussian Mixture Models," in *Proc. ICASSP '99* Phoenix, pp. 337-340, March 1999.
- [5] E. Bocchieri, "Vector Quantization for the Efficient Computation of Continuous Density Likelihoods," in *Proc. ICASSP '93*, Minneapolis, pp. 692-695, April 1993.
- [6] K. Knill, M. Gales, and S. Young, "Use of Gaussian Selection in Large Vocabulary Continuous Speech Recognition Using HMMs," in *Proc. ICSLP '96*, Philadelphia, pp. 470-473, October 1996.

Analysis and Transcription of General Audio Data

Michelle S. Spina

In addition to the vast amount of text-based information available on the World Wide Web, an increasing amount of video and audio based information is becoming available to users as a result of emerging multimedia computing technologies. The addition of these multimedia sources of information have presented us with new research challenges. Mature information retrieval (IR) methods have been developed for the problem of finding relevant items from a large collection of text-based materials given a query from a user. Only recently has there been any work on similarly indexing the content of multimedia sources of information.

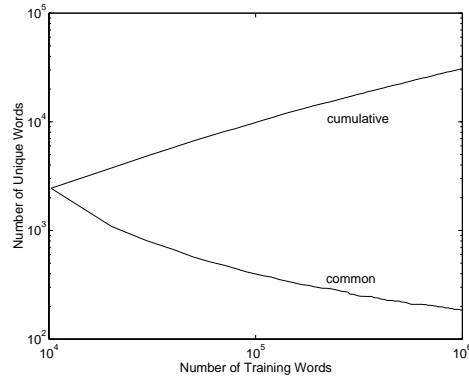
In this work, we focus on general audio data (GAD) as a new source of data for information retrieval systems. The main goal of this research is to understand the issues posed in describing the content of GAD. We are interested in understanding the general nature of GAD, both lexically and acoustically, and in discovering how our findings may impact an automatic indexing system. Specifically, three research issues are addressed. First, what are the lexical characteristics of GAD, and how do they impact an automatic recognition system? Second, what general sound classes exist in GAD, and how well can they be distinguished automatically? And third, how can we best utilize the training data to develop a GAD transcription system?

In our attempt to answer these questions, we first developed an extensive GAD corpus for study in this work. We chose to focus on the National Public Radio (NPR) broadcast of the Morning Edition (ME) news program. NPR-ME is broadcast on weekdays from 6 to 9 a.m. in the US, and it consists of news reports from national and

local studio anchors as well as reporters from the field, special interest editorials and musical segments. We chose NPR-ME after listening to a selection of radio shows, noting that NPR-ME had the most diverse collection of speakers and acoustic conditions and would therefore be the most interesting for study. We collected and orthographically transcribed 102 hours of data for lexical analysis. Ten hours were further processed for acoustic analysis and recognition experiments.

Next, we studied the lexical properties of GAD to gain a better understanding of the data, and to see how these data compare with those typically used in the ASR community. We then studied the properties of the GAD vocabulary [1]. We are interested in determining the size of the NPR-ME vocabulary and in observing how the vocabulary grows with time. This analysis discovered some potential problems for a general large vocabulary continuous speech recognition approach to the transcription of GAD. Figure 22 plots the relationship between the number of distinct words encountered (i.e., the recognizer's vocabulary) and the size of the training set as the training set size is increased. The upper curve of Figure 22 shows the cumulative sum of all the distinct words, and therefore represents the potential vocabulary of the recognizer. While the size of the vocabulary after 102 shows (over 30,000 words) is within the capabilities of current-day ASR systems, it is quite alarming that the growth of the vocabulary shows no sign of abating. Next, we found that even for large training set sizes and vocabularies, new words were still regularly encountered. With a training set of nearly one million words (resulting in over 30,000 unique vocabulary words), the

Figure 22. The number of distinct words as a function of the number of words encountered in the NPR-ME corpus.



out of vocabulary rate was just over 2%. A part-of-speech analysis determined that the new words were predominately proper nouns and nouns, which would be very important to recognize if we were describing the content of this data. This problem was magnified when we investigated the more realistic scenario of constructing a training set from an out-of-domain source. In this case, we found that the out of vocabulary rate nearly doubled to 4% [2].

We then examined the acoustic characteristics of GAD and developed a sound recognition system to segment the audio into its salient sound classes. For the NPR-ME corpus, we subjectively identified seven acoustically distinct classes based on visual and aural examination of the data. We found that these classes differed in their spectral characteristics, statistical profile, and segment duration. Specifically, we found that high quality, prepared speech constitutes only half of the entire corpus, another 25% of the data contains speech superimposed on other sounds, nearly 15% of the data was of telephone bandwidth, and the remaining 10% of the data was non-speech. We also found that while pure music segments are similar in length to speech segments, other non-speech segments are substantially shorter in length.

We achieved a 79.4% recognition accuracy for these seven classes on unseen data, using relatively straightforward acoustic measurements and pattern recognition and smoothing techniques. A speech/non-speech recognizer achieved an accuracy of 94.2%.

Finally, based on the results of our lexical analysis, we proposed a subword approach to the lexical transcription of GAD. Since we discovered that GAD contains a number of different acoustic speaking environments and the performance of ASR systems can vary a great deal depending on speaker, microphone, recording conditions and transmission channel, we proposed that the transcription of GAD would benefit from a preprocessing step that first segments the signal into acoustically homogeneous blocks so that appropriate models could be used during test. Specifically, we developed a phonetic recognizer for GAD and investigated a number of different training and testing techniques. We found that for all of the training techniques that we investigated the phonetic error rates varied widely across the NPR-ME speaking environments. By systematically exploring different system designs (one recognizer vs. multiple recognizers) and different training techniques, we were able to discover how each technique affected each environment.

The results of our experiments are summarized in Figure 23. If a single recognizer system is to be used, we found that training on all of the available data which contains a variety of speaking environments was more effective than using a smaller amount of homogeneous, clean data. This result held true even when we accounted for the discrepancy in the amount of training data available for the

two approaches. While we originally felt that the transcription of GAD would benefit from a preprocessing step that first segments the data into acoustically homogeneous blocks so that appropriate models could be used during test, overall, we found that such a multiple recognizer system achieved performance similar to a single multi-style recognizer. However, upon closer inspection of the results we found that the multi-style system primarily benefitted from the increased amount of data available for training. We may be able to utilize the strengths of both the multi-style and environment-specific approaches by developing interpolated models. By interpolating between the well-trained multi-style models and the more detailed environment-specific models, we may be able to improve our phonetic recognition results.

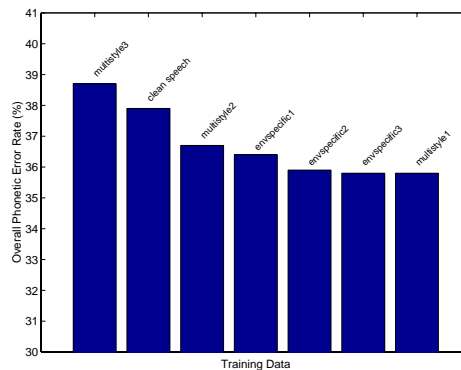


Figure 23. Summary of phonetic error rate results for different training methods. The multi-style1 system uses all of the available training data, the clean speech system uses only studio quality, wideband training data, the multi-style2 system uses an amount of training data comparable to the clean speech system and the multi-style3 uses an amount of training data comparable to each of the test speaking environment systems. Each of the environment-specific systems used the sound recognition system as a preprocessor to select the appropriate models for testing. The env-specific1 system uses the four original speaking classes, the env-specific2 system collapses music and noisy speech into a single class, and the env-specific3 system adds bandlimited models for the field speech data.

References:

- [1] M.S. Spina and V.W. Zue, "Automatic Transcription of General Audio Data: Preliminary Analysis," *Proc. ICSLP '96*, Philadelphia, PA, October 1996.
- [2] M.S. Spina. *Analysis and Transcription of General Audio Data*. Ph.D. thesis, MIT Department of Electrical Engineering and Computer Science, May 2000.
- [3] M.S. Spina and V.W. Zue, "Automatic Transcription of General Audio Data: Effect of Environment Segmentation on Phonetic Recognition," *Proc. Eurospeech '97*, Rhodes, Greece, September 1997.

Framework for Joint Recognition of Pronounced and Spelled Proper Names

Atiwong Suchato

The goal of this thesis was to demonstrate that a spoken name along with a spoken version of the spelling of the name could be combined to produce a proposed pronunciation and spelling to augment the vocabulary of an existing recognizer.

The system involves two recognizers, both implemented within the SUMMIT framework. The first one is based on a vocabulary consisting of a set of “morph” units that cover the majority of the syllables appearing in peoples’ names. The other recognizer has the 26 letters of the English alphabet as its vocabulary, and is used to recognize spoken spellings of the words.

The procedure is to first recognize the letters, supported by a letter trigram language model, into a set of candidate N -best spellings. The TINA system is used to parse the letter hypotheses into a set of candidate morph sequences, with associated probabilities, that are then written out as a finite state transducer, to serve as a supporting language model for the pronunciation recognition task.

Subsequently, the pronunciation of the paired hypothesis is processed through the morph recognizer, constrained by the morph graph obtained from the letter recognition task. The best scoring hypothesis is then consistent with both the spelling and the pronunciation, and its quality is then conditioned on the joint performance of the two recognizers.

We experimented with two versions of the morph lexicon, one of which (type-i) attempted to distinguish homomorphs (same spelling, different pronunciation) as distinct vocabulary entries, whereas the other one (type-ii) simply allowed multiple

pronunciations for the homomorphs, represented as a shared lexical unit. The latter made it much easier to create a lexicon of names decomposed into morphs, because the distinct homomorphs did not have to be specified lexically. We found that the collapsed homomorphs were in general better behaved, both in terms of overall recognition accuracy and in terms of the ability of the spellings to provide support to the pronunciation recognition task.

Our speech corpus consisted of material obtained from the Oregon Graduate Institute through the Linguistic Data Consortium, augmented with data obtained by asking users of the Jupiter weather domain to speak a name when they are first greeted. We separated the corpus into a training and a test set, and evaluated the system on both sets in order to assess the degree of generality of the learned statistics.

For language modelling, we obtained a text corpus of over 100,000 names, which were converted into morph sequences through a semi-automatic procedure. This corpus was used to train both the letter trigram for the letter recognizer and the recursive grammar that converts letters to morphs.

We performed a number of experiments, varying the conditions of the n -gram training and testing corpus and the size of the morph lexicon. We also tested the ideal condition where the letter recognition was “perfect,” which of course resulted in a huge gain in accuracy of the pronunciation task. In all cases, we measured phone recognition, letter recognition, and morph recognition accuracies. Details of the results can be found in the thesis, which showed that

slight gains were obtained in phone accuracy with the support of the errorful letter sequences from the letter-recognition task, and, likewise, the phone information led to slight gains in the letter-recognition task, but only for the type-ii morph lexicon, as discussed above.

Reference

- [1] A. Suchato. *Framework for Joint Recognition of Pronounced and Spelled Proper Names*. MIT Department of Electrical Engineering and Computer Science, S.M thesis expected September 2000.

Robust Pitch Tracking for Prosodic Modeling in Telephone Speech

Chao Wang

Reliable pitch detection is very crucial to the analysis and modeling of speech prosody. The fundamental frequency (F0) is found to be highly correlated with prosodic features such as lexical stress and tone, which provide important perceptual cues to human speech communication. However, most current automatic speech recognition and understanding (ASRU) systems under-utilize prosodic features, especially those related to F0. This is partially due to the lack of a robust parameter space for statistical modeling. More specifically, errors in F0 contours, both in terms of pitch accuracy and voicing decision, can affect feature measurements dramatically.

Various pitch detection algorithms (PDAs) have been developed in the past [1]. While some have very high accuracy for *voiced* pitch hypotheses, the error rate considering voicing decision is still quite high; and the performance degrades significantly as the signal condition deteriorates. We are interested in developing a PDA that is particularly robust for telephone quality speech and prosodic modeling applications. In particular, we address two problems: 1) missing fundamental and high noise level in telephone speech, and 2) discontinuity in the F0 contour due to the *voiced/unvoiced* dichotomy. To address the first problem, we adopt a frequency-domain signal representation and rely on the overall harmonic structure to estimate F0. To deal with discontinuity of the F0 space for prosodic modeling, we believe that it is more advantageous to emit an F0 value for each frame, even in unvoiced regions, and to provide separately a parameter to reflect probability of voicing. This is based on the considerations that, first, voicing decision errors will not be manifested as absent pitch

values; second, features such as those describing the shape of the pitch contour are more robust to segmental mis-alignments; and third, a voicing probability is more appropriate than a “hard” decision of 0 and 1, when used in statistical models.

On a logarithmic frequency scale, harmonic peaks appear at $\log F_0$, $\log F_0 + \log 2$, $\log F_0 + \log 3$, ..., etc. To find the F0 value, one can sum the spectral energy spaced by $\log 2$, $\log 3$, ..., etc., from the pitch candidate and choose the maximum [2]. This is equivalent to correlating the spectrum with an n -pulse template, where n is the number of included harmonics. We adopt a similar method to find $\log F_0$ for each frame, using a carefully constructed template in place of the pulse sequence. More importantly, we also obtain a reliable estimate of $\Delta \log F_0$ across two adjacent *voiced* frames of the speech by simple correlation. These two constraints are combined in a Dynamic Programming (DP) search to find an overall optimum solution. In the following, we introduce each module in more detail.

The logarithmically spaced spectrum was obtained by sampling a narrow band spectrum in the low-frequency region at linear intervals in the logarithmic frequency dimension. We define this representation as a *discrete logarithmic Fourier transform* (DLFT). Figure 24 shows the waveform, Fourier transform and DLFT for a 200 Hz pulse train and a *voiced* speech signal. The DLFT of the speech signal, sampled between 150 and 1500 Hz, has been normalized by mu-law conversion to flatten out the formant peaks. The weighted DLFT of the pulse train will be used as a template for F0 estimation.

The constraints for log F0 and delta log F0 estimations are captured by two correlation functions. The “template-frame” correlation provides log F0 estimation by aligning the speech DLFT with the template. The position of the correlation maximum should correspond to the difference of log F0 between the voiced signal and the impulse train. The “cross-frame” correlation provides delta log F0 constraints by aligning two adjacent frames of the signal DLFT spectra. The maximum of the correlation gives a robust estimation of the log F0 difference across two voiced frames. For unvoiced regions, it is observed that the “template-frame” correlation is more or less random, and the “cross-frame” correlation stays fairly flat both within an unvoiced region and upon transition of voicing status. This has important implications for our DP based *continuous* pitch tracking as described next.

We can easily formulate the problem of pitch tracking as a DP search given the two correlation constraints. We define the score function as the sum of the best past score as weighted by the cross-frame correlation and the template-frame correlation. The search is forced to find a pitch value for every frame, even in unvoiced regions. This is feasible because the cross-frame correlation stays relatively flat when at least one frame is unvoiced. Thus, upon transition into unvoiced regions, the best past score will be inherited by all candidate nodes; and the scores become somewhat random. However, once in voiced regions, the sequence of nodes corresponding to the true pitch values will emerge because of high internal scores enhanced by high cross-frame correlation coefficients.

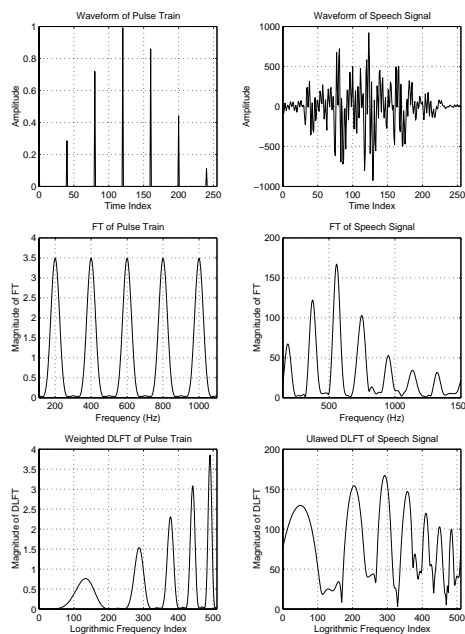


Figure 24. Windowed waveform, FT, and adjusted DLFT (refer to the text for details) for a pulse train and a speech signal.

We evaluated our PDA for both voiced pitch accuracy and tone classification accuracy, and compared the performances with those of an optimized PDA provided by Xwaves [3].

The Keele pitch extraction reference database [4] was used for voiced pitch accuracy evaluation, because it provides reference pitch as “ground truth.” While Xwaves performs well on studio speech, the performance degraded severely for telephone speech, particularly with regard to voicing decisions. Our PDA performs substantially better than Xwaves for telephone speech, and the overall gross error rate for studio and telephone conditions is nearly the same (4.25% vs. 4.34%).

We compared the tone classification performance using F0 contours derived by our system and Xwaves on a telephone-quality, Mandarin digit corpus [5]. We found that the classification error rate using our system was much lower than that using

Xwaves (19.2% vs. 25.4%); and the inclusion of an average voicing probability feature reduced error further down to 18.2% for our system. We tried two approaches to deal with the unvoiced frames when using Xwaves: 1) interpolate F0 from the surrounding voiced frames, and 2) bias the V/UV decision threshold to greatly favor “voiced”, followed by interpolation. However, neither method was particularly successful.

References

- [1] W. Hess, *Pitch Determination of Speech Signals*. Springer-Verlag, 1983.
- [2] D. Hermes, “Measurement of Pitch by Subharmonic Summation,” in *JASA*, vol. 83, no. 1, pp. 257-273, 1988.
- [3] D. Talkin, “A Robust Algorithm for Pitch Tracking (RAPT),” in *Speech Coding and Synthesis* (Elsevier, ed.), pp. 495-518, 1995.
- [4] F. Plante, G. Meyer, and W. Ainsworth, “A Pitch Extration Reference Database,” in *Proc. Eurospeech '95*, pp. 837-840, Madrid, Spain, 1995.
- [5] C. Wang and S. Seneff, “A Study of Tones and Tempo in Continuous Mandarin Digit Strings and Their Application in Telephone Quality Speech Recognition,” in *Proc. ICSLP '98*, pp. 635-638, Sydney, Australia, 1998.

Methods and Tools for Speech Synthesizer Design and Deployment

Jon Yi

In the recent year, we have re-engineered our concatenative speech synthesis tools in an effort to make them easier to use. We adopted finite-state methods to concisely capture principles of concatenative speech synthesis laid out in earlier works [1,2]. As a result we now have a set of tools to rapidly design concatenative speech synthesizers for constrained understanding domains. The deployment is made possible by a suite of networked servers operating within the GALAXY Communicator framework. Speech synthesizers which convert meaning representations to speech waveforms have been prototyped for the MERCURY, JUPITER and MUXING domains.

Because many aspects of automatic speech recognition can be modelled by finite-state automata (e.g., dictionary lookup and pronunciation variation), finite-state methods which seek to unify representation have recently gained popularity within the speech community. More specifically, weighted finite-state transducers (FST), which map an input regular language to an output regular language, have become the abstraction of choice. This transformation of input symbols to output symbols can be prioritized by weights. Once the algebraic algorithms have been implemented to manipulate and optimize transducers, the researcher is free to focus on the actual graphical structure of the transducers.

Lexical access in automatic speech recognition and unit selection in automatic speech synthesis ultimately make use of a search algorithm to seek the least-cost path through a search space. Just as researchers in our group (and elsewhere) have converted the speech recognition framework to make use of FSTs, we sought to do the same for speech synthesis, the inverse problem.

While pronunciation generation and variation components could be re-used from recognition, it was imperative to design a transducer structure that maps phonetic symbols to waveform segments. This structure, which is the key contribution of our recent work [3], participates in a Viterbi-style dynamic programming search during the final stage of unit selection.

In corpus-based concatenative speech synthesis approaches one finds the classical trade-off between knowledge and data. While synthesis constraints govern which units and where to concatenate, a corpus must provide enough examples of units in all potential contexts. Our new tools allow the complementary components to be separately refined. A language expert can craft the language-dependent synthesis constraints independently of the domain, whereas a domain expert can design and acquire a synthesis corpus according to the distributional properties of in-domain responses. As delineated in a companion research summary piece, a flexible FST architecture was designed to encapsulate both the synthesis constraints and corpus in a scalable fashion.

In concatenative speech synthesis there are essentially two tasks: unit selection and waveform generation. In our implementation, unit selection is achieved through FST methods as described above. Currently, waveform generation simply consists of sample re-sequencing. The actual deployment of our FST-based speech synthesis framework is facilitated through two Communicator-compliant components. A fat server performs unit selection and transmits the lightweight search result to a thin client which performs waveform generation using locally stored audio

waveforms. By moving waveform splicing closer to the end user, such a framework could be deployed in low-bandwidth environments (i.e., not sufficient for real-time audio transmission), where a monolithic approach might fail.

In summary, the concatenative speech synthesis system, ENVOICE, has undergone an extensive rewrite that should make for easier use and quicker development cycles. Knowledge and data are coupled in a finite-state transducer structure used for corpus-based unit selection. Finally, complementary search and waveform components work in tandem to produce interactive responses in spoken dialogue systems.

References

- [1] Jon R. W. Yi, *Natural-Sounding Speech Synthesis Using Variable-Length Units*, M.Eng. thesis, MIT Department of Electrical Engineering and Computer Science, May 1998.
- [2] Jon R. W. Yi and James R. Glass. Natural-Sounding Speech Synthesis Using Variable-Length Units. In *Proc. ICSLP '98*, Sydney, Australia, December 1998.
- [3] Jon R. W. Yi, James R. Glass, and I. Lee Hetherington. A Flexible, Scalable Finite-State Transducer Architecture for Corpus-Based Concatenative Speech Synthesis. To appear *Proc. ICSLP '00*, Beijing, China, October 2000.

T H E S E S

P U B L I C A T I O N S

P R E S E N T A T I O N S

S E M I N A R S

Doctoral Theses

Completed

K. Ng, *Subword-based Approaches for Spoken Document Retrieval*. February 2000.

M. Spina, *Analysis and Transcription of General Audio Data*. June 2000.

In Progress

G. Chung, *Towards Multi-Domain Speech Understanding with Flexible and Dynamic Vocabulary*. Expected December 2000.

C. Wang, *Incorporation of Prosody into Speech Recognition and Understanding*. Expected 2001.

Masters Theses

S.M. Thesis Completed

K. Livescu, *Analysis and Modeling of Non-Native Speech for Automatic Speech Recognition*. Supervisor: J. Glass. August 1999.

M.Eng. Theses Completed

S. Kamppari, *Word and Phone Level Acoustic Confidence Scoring for Speech Understanding Systems*. Supervisor: T.J. Hazen. September 1999.

E. Sandness, *Discriminative Training of Acoustic Models in a Segment-based Speech Recognizer*. Supervised by I.L. Hetherington. May 2000.

In Progress

T. Burianek, *Building a Speech Understanding System Using Word Spotting Techniques*. M.Eng. thesis. Supervisor: T.J. Hazen. Expected July 2000.

L. Baptist, *GENESIS-II: A Language Generation Module for Conversational Systems*. S.M. thesis. Supervisor: S. Seneff. Expected August 2000.

J. Pearlman, *SLS-Lite: Enabling Spoken Language Systems Design for Non-Experts*, M.Eng. thesis. Supervisor: J. Glass. Expected August 2000.

A. Suchato, *Framework for Joint Recognition of Pronounced and Spelled Proper Names*. S.M. thesis. Supervisor: S. Seneff. Expected: September 2000.

Publications

- G. Chung, S. Seneff and I.L. Hetherington, "Towards Multi-Domain Speech Understanding Using a Two-Stage Recognizer," *Proc. Eurospeech 99*, Budapest, Hungary, September 1999.
- S. Seneff, R. Lau and J. Polifroni, "Organization, Communication, and Control in the GALAXY-II Conversational System," *Proc. Eurospeech 99*, Budapest, Hungary, September 1999.
- J. Glass, "Challenges for Spoken Dialogue Systems," *Proc. 1999 IEEE ASRU Workshop*, Keystone, CO, December 1999.
- N. Strom, L. Hetherington, T.J. Hazen, E. Sandness and J. Glass, "Acoustic Modeling Improvements in a Segment-Based Speech Recognizer," *Proc. 1999 IEEE ASRU Workshop*, Keystone, CO, December 1999.
- V. Zue, et al., "JUPITER: A Telephone-Based Conversational Interface for Weather Information" *IEEE Transactions on Speech and Audio Processing*, Vol. 8 , No. 1, January 2000.
- J. Polifroni and S. Seneff, "GALAXY-II as an Architecture for Spoken Dialogue Evaluation" *Proc. Second International Conference on Language Resources and Evaluation (LREC)*, Athens, Greece, May 31-June 2, 2000.
- I. Bazzi and J. Glass, "Heterogeneous Lexical Units for Automatic Speech Recognition: Preliminary Investigations" *Proc. ICASSP2000*, Istanbul, Turkey, June 2000.
- S. Kamppari and T.J. Hazen, "Word and Phone Level Acoustic Confidence Scoring" *Proc. ICASSP2000*, Istanbul, Turkey, June 2000.
- K. Livescu and J. Glass, "Lexical Modeling of Non-native Speech for Automatic Speech Recognition" *Proc. ICASSP2000*, Istanbul, Turkey, June 2000.
- K. Ng, "Information Fusion for Spoken Document Retrieval" *Proc. ICASSP2000*, Istanbul, Turkey, June 2000.
- C. Wang and S. Seneff, "Robust Pitch Tracking for Prosodic Modeling in Telephone Speech" *Proc. ICASSP2000*, Istanbul, Turkey, June 2000.

Presentations

V. Zue. "Next Generation Speech-based Interfaces." Invited talk at the Bell Atlantic Speech Symposium. September 23, 1999.

V. Zue. "2001: Why the Hal Not?" Invited talk at SpeechWorks International, Boston, MA, September 28, 1999.

V. Zue. "Speech is It". Presentation at DARPA. September 29, 1999.

J. Glass. "Challenges for Spoken Dialogue Systems." IEEE Automatic Speech Recognition and Understanding Workshop, Keystone, Colorado, December 1999.

V. Zue. "Introduction to the MIT Laboratory for Computer Science". Presentation at NTT-MIT Collaboration Meeting, Tokyo, Japan. January 2000.

S. Seneff. "Multilingual Conversational Interfaces: An NTT-MIT Collaboration". NTT-MIT Collaboration Meeting, Tokyo, Japan. January 13, 2000

V. Zue. "The DARPA Communicator: An MIT Progress Report". Presentation at the DARPA Communicator PI Meeting, Charleston, SC, January 20, 2000

V. Zue. "Conversational Interfaces". Presentation to NOKIA, Helsinki, Finland, January 25, 2000.

V. Zue. "2001: Why the Hal Not?" Invited talk at the Telephony Voice User Interface Meeting, Phoenix, AZ, February 3, 2000.

V. Zue and J. Glass. "Conversational Interfaces: Advances and Challenges". February 8, 2000. IEEE

V. Zue "Introduction to the MIT Laboratory for Computer Science". Invited talk at MITRE, March 1, 2000.

V. Zue. "@ the Human Interface". Panel moderator at the Intel CCC Meeting, San Francisco, CA, March 16, 2000.

J. Glass. "Overview of the MIT Language Generation and Speech Synthesis Systems". DARPA Workshop on Corpus-Based Text Generation and Synthesis, Seattle, WA, April 30, 2000.

S. Seneff. and J. Polifroni. "Dialogue Management in the MERCURY Flight Reservation System". Satellite Workshop on Conversational Systems following ANLP/NAACL 2000, Seattle, Washington. May 4, 2000,

S. Seneff. "Spoken Dialogue Systems: Dialogue Management and System Evaluation Issues". Panel at Satellite Workshop on Conversational Systems following ANLP/NAACL 2000, Seattle, Washington, May 4, 2000.

S. Seneff. "Spoken Language Technology Research at MIT: A Progress Report". ITRI, June 15, 2000.

V. Zue "Acoustic Properties of Speech". Johns Hopkins workshop, Baltimore, MD, July 2000.

SLS Seminar Series

October 18, 1999

“Jana Speech API: Developing Speech Apps in Java”

Andrew Hunt & Willie Walker
Sun Microsystems

October 25, 1999

“Toward Automatic Summarization of Broadcast News Speech”

Sadaoki Furui
Tokyo Institute of Technology

February 28, 2000

“Persuasive Multi-modal Conversational Systems”

Ganesh Ramaswamy
IBM TJ Watson Research center

March 27, 2000

“Stochastic Suprasegmentals: Relationships Between Redundancy, Prosodic Structure and Core Articulation in Spontaneous Speech”

Matthew Aylett
University of Edinburgh

March 6, 2000

“Building Telephone-Based Spoken Dialog Applications”

Mike Phillips
SpeechWorks International

May 1, 2000

“Natural Language in L&H Voice Xpress: An Overview”

Jeff Adams
Lernout & Hauspie Speech products

May 22, 2000

Issues in Mandarin TTS Systems

Min Chu
Microsoft Research China

Photography by [Tony Rinaldo](#)

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

2000

July 2000

MIT Laboratory for Computer Science - Spoken Language Systems Group

MIT/LCS