In ICSLP-94, 2139–2142

THE BERKELEY RESTAURANT PROJECT

Daniel Jurafsky, Chuck Wooters,* Gary Tajchman,
Jonathan Segal, Andreas Stolcke, Eric Fosler, and Nelson Morgan

International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704, USA
& University of California at Berkeley†

ABSTRACT

This paper describes the architecture and performance of the Berkeley Restaurant Project (BeRP), a medium-vocabulary, speaker-independent, spontaneous continuous speech understanding system currently under development at ICSI. BeRP serves as a testbed for a number of our speech-related research projects, including robust feature extraction, connectionist phonetic likelihood estimation, automatic induction of multiple-pronunciation lexicons, foreign accent detection and modeling, advanced language models, and lip-reading. In addition, it has proved quite usable in its function as a database frontend, even though many of our subjects are non-native speakers of English.

1 OVERVIEW

The BeRP system functions as a knowledge consultant whose domain is restaurants in the city of Berkeley, California. As a knowledge consultant, it draws inspiration from earlier consultants like VOYAGER [15]. Users ask spoken language questions of BeRP, which directs questions to the user and then queries a database of restaurants and gives advice to the user, based on such use criteria as cost, type of food, and location.

The BeRP recognizer consists of six components: the RASTA-PLP *feature extractor*, a multilayer perceptron (MLP) *phonetic likelihood estimator*, a *Viterbi decoder* called Y_0, an HMM pronunciation *lexicon*, a bigram or SCFG *Language Model (LM)* and the *natural language backend*, including a database of restaurants. The whole system runs on a SPARCstation, although for speed we usually offload the phonetic likelihood estimation (the MLP forward pass) to special purpose hardware. Figure 1 gives an overview of the architecture.

Figure 2 gives some statistics that give the reader a feeling for the BeRP system as implemented.

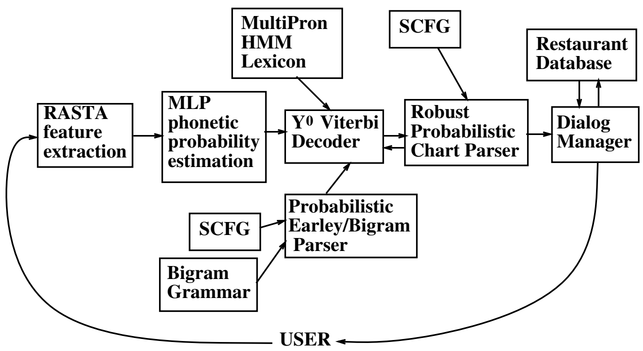

Figure 1: The BeRP Architecture

Training Corpus	4786 sentences + TIMIT	
Test Corpus	563 sentences	
Vocabulary	1274 words	
Data Base	1 database table, 150 restaurants	
Bigram	Perplexity 10.7 with 77% coverage	
Grammar	1389 handwritten SCFG rules	
Implementation	18,000 lines of C++	
Performance	Recognition	32.1% error
	Parsing	63% training 61% test
	Understanding	34% error

Figure 2: BeRP Status in June 1994

2 FEATURE EXTRACTION

BeRP uses RASTA-PLP [7] to extract features from digitized acoustic data. RASTA-PLP is a speech analysis technique that is robust to steady-state spectral factors in speech such as those imposed by different communication channels (i.e., different microphones). The fundamental idea of RASTA is to bandpass or highpass-filter each spectral trajectory to reduce the affect of stationary convolutional errors. In the case of RASTA-PLP this filtering is done for critical bands output after processing by a log or log-like nonlinearity. RASTA-PLP's robustness is an important feature for our frontend, since we bootstrap our system on databases collected under different recording conditions than those at ICSI; in general, we can change microphones fairly easily without severely impacting performance.

3 PROBABILITY ESTIMATION

The BeRP system uses a discriminatively-trained Multi-Layer Perceptron (MLP) in an iterative Viterbi procedure to estimate emission probabilities. [2] and [10] show that with a few assumptions, an MLP may be viewed as estimating the probability $P(q|x)$ where q is a subword model (or a state of a subword model) and x is the input acoustic speech data. We can then compute the likelihood $P(x|q)$ needed by the Viterbi algorithm by dividing by the prior $P(q)$, according to Bayes' rule; we ignore $P(x)$ since it is constant in time-synchronous Viterbi:

$$P(x \mid q) = \frac{P(q \mid x)P(x)}{P(q)} \tag{1}$$

Results with our architecture on the speaker independent DARPA Resource Management database for MLP monophone estimators [4] yield competitive performance (4–6% word error with the standard perplexity 60 wordpair grammar).

The estimator consists of a simple three-layer feed forward MLP trained with the back-propagation algorithm (see Figure 3).

*Currently at Dept. of Defense
† <jurafsky|wooters|tajchman|jsegal|stolcke|fosler|morgan>@icsi.berkeley.edu

The input layer consists of 9 frames of input speech data. Each frame, representing 10 msec of speech, is typically encoded by 9 RASTA PLP coefficients, 9 delta-RASTA PLP coefficients, and an energy term and a delta-energy term. Typically, we use 500-1000 hidden units. The output layer has 61 units, one for each of the context-independent phonetic classes used in our lexicon. The MLP is trained on phonetically hand-labeled speech (TIMIT) to produce the initial state alignments, and then trained in the iterative Viterbi procedure (forced-Viterbi providing the labels) with the BeRP corpus.

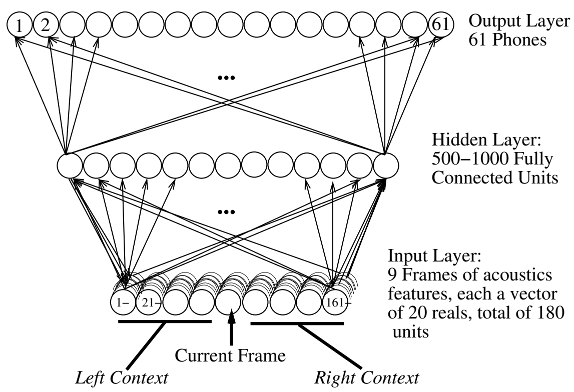

Figure 3: Phonetic Likelihood Estimator

In order to perform the phonetic likelihood estimation quickly, we usually offload the computation to special-purpose hardware, the Ring Array Processor (RAP) [9], although we have recently completed a vector-quantized single-layer perceptron version which runs in real-time on the SPARC. Recognition performance of the VQ system is lower than our RAP-based system (46% relative error) but as both systems improve, the VQ system may become good enough for on-line use.

4 VITERBI DECODER

Our decoder, called Y_0, implements the standard synchronous Viterbi algorithm. It has two non-standard features. One is the ability to efficiently perform backtraces on-line, so we can pass backtraces to the SCFG LM on each frame (see §6). Another is the use of a technique borrowed from the word-spotting literature to make the recognition more robust to the unknown words, fragments and repairs that occur in spontaneous speech.

The idea is to add a new word model, the *garbage word*, which is designed to match unknown words and word fragments. However, instead of generating garbage likelihoods through some trained model, we use the method of [1] to generate virtual likelihoods. At each frame, the phone probabilities for the garbage word are computed by averaging the top N phone probabilities from the MLP. Thus at each frame, the local distance of the garbage word will be worse than any word which matches the best few phones. But if no word can transition into the top few phones, the garbage word will have a greater local distance than other words, and will be selected. This should happen just in the case where no word adequately models the current frame, such as unknown word or fragments. The transition probabilities for the garbage word are computed by averaging the bigram transition probabilities for the top N words, with the same rationale. We generally use a value of N=10 for both local distance and word transition computation.

5 LEXICON

The BeRP system uses a multiple-pronunciation HMM lexicon, which we build by combining two lexicon-production methods. In the first method we use the *model merging* algorithm [11] to induce a word model directly from a set of pronunciations for the word culled from TIMIT, various dictionaries, and other sources. The algorithm begin with a very specific HMM producing exactly the set of input pronunciations, constructed by stringing together each observed phone sequence between a start and end state. Next, this initial model is simplified and generalized by repeatedly *merging* states until we reach a model with (locally) maximum posterior probability. See [14] for more details.

Our second method augments this bottom-up approach with top-down information. Following [5], we manually develop phonological rules and automatically apply them to a set of base-form or "dictionary" pronunciations, to automatically generate multiple pronunciations for any word in the dictionary. We then add these pronunciations to the set of pronunciations from TIMIT and other sources, and apply the merging algorithm.

Figure 4 shows the initial multiple-pronunciation model for the word *waiting*, computed from the samples in the TIMIT and other corpora and dictionaries, while Figure 5 shows the automatically merged model. Table 1 shows the significant improvement which our multiple pronunciation models provide over the single pronunciation models. (Note that our most recent results reported in §10 are somewhat improved from these.)

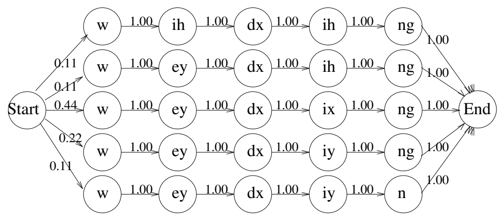

Figure 4: Unmerged HMM for the word "waiting"

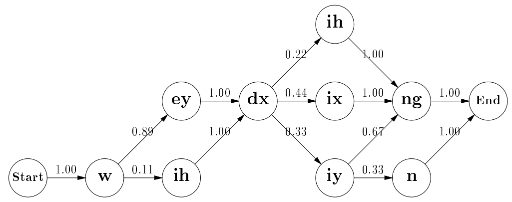

Figure 5: Merged HMM for the word "waiting"

System	Word Error	Ins	Dels	Subs
Single Pron	40.6	5.2	10.3	25.1
Multi-Pron	32.1	7.3	5.9	18.9

Table 1: Performance of Pronunciation Models.

6 LANGUAGE MODEL

The Y_0 recognizer uses a number of different language models, including a simple bigram as well as two advanced suites of methods. The first of these advanced methods uses an SCFG and an SCFG parser/generator to provide more sophisticated language information. The 1389 rules in the grammar are written by hand, but the rule probabilities are learned from the 4786-sentence BeRP corpus with the EM algorithm. We have experimented with a number of ways to use the SCFG information:

1. Smooth the bigram grammar by augmenting the corpus with a pseudo-corpus of sentences generated from the SCFG (extending the method of [15] for training word-pair grammars).

2. Use the *characteristic bigram* of the SCFG, which can be generated in closed form [12].
3. Use the SCFG directly as the LM for the recognizer, by using the probabilistic parser to compute word transition probabilities directly from the SCFG [8].
4. Mix the SCFG and smoothed bigram probabilities directly to provide word transition probabilities on each frame.

Table 2 presents our word error results, showing the results of applying these four methods. Note that the SCFG gave a significant 4.1% improvement in word error over the bigram. The SCFG and the SCFG-smoothed bigram performed equally, and the mixture model was slightly but not significantly better than either SCFG model; compiling the SCFG into a bigram seems to preserve most of the useful information.[1]

	Word Error
Bigram	33.7
SCFG-Smoothed Bigram	29.6
SCFG	29.6
SCFG/Bigram Equal Mixture	28.8

Table 2: SCFG Performance

In our most recent language model experiment, we train pragmatic-context-specific bigrams. Because BeRP is a mixed-initiative system, users often respond to questions asked by the system. For each question the system can ask, we build a subcorpus of responses from our training set, and train a bigram (smoothed with responses to other questions). Then during recognition, we switch among these bigrams depending on the system's latest question.

7 NATURAL LANGUAGE BACKEND

The BeRP natural language backend transforms word strings from the Y_0 decoder into database queries. The architecture is controlled by a template-filling *dialog manager*, which asks questions of the user in order to determine certain restaurant features, (such as cost, distance, and type of food) which are used to fill a query-template, and thus to query the database. For each template slot the system prompts the user with a question, although the user may choose not to answer the question. Consider the following user response:

```
i'd like to have indian food today
```

The *bottom-up stochastic chart parser* first computes all possible parses and semantic interpretations of the input, along with their grammatical probabilities. It uses a stochastic context-free attribute grammar in which simple semantic rules are encoded in the attributes, written in a generalization of the PostQuel database query language used by the BeRP database backend.[2] The parse tree for the input above is shown in Figure 6.

Each node of the parse tree is automatically annotated with a partial semantics. The semantics for the top node in Figure 6 is passed directly to the context module, which fills out all context-dependent and scope-dependent operators, such as temporal deictics ("now", "today") and negation ("not far"). The resulting semantics shown below indicates that the user is interested in an Indian restaurant which is open Monday ("REST.mon") for any meal (breakfast ("B"), lunch ("L") or dinner ("D")):

```
(REST.rest_type = "INDIAN") and (REST.mon ~ "[BLD]")
```

[1]These results are on a slightly different system than that used for the multiple-pronunciation and semantic performance tests.

[2]Our 'semantic' grammar encodes more information in the context-free portion than is common in purely 'syntactic' grammars by allowing very specific non-terminal symbols, which lowers the perplexity of the grammar. We are currently working on a unification-based augmentation.

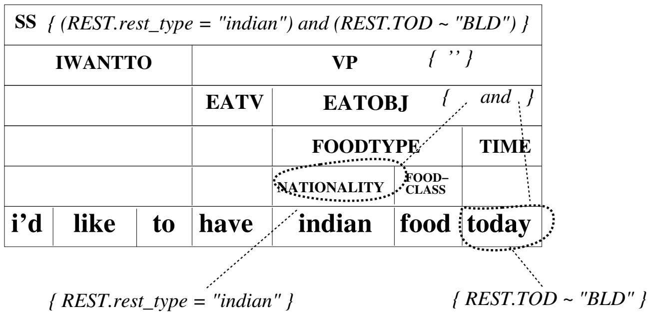

Figure 6: The parse-tree produced by the interpreter

An important consideration in the parser is robustness to grammatical and lexical gaps as well as recognizer errors. The BeRP parser bases its robustness on a bottom-up algorithm for parsing combined with a heuristic for combining parse fragments. The simple bottom-up algorithm (the CYK or bottom-up chart algorithm) will find every constituent which occurs in the input sentence, even those which are in ungrammatical contexts. Then if a sentence is not completely parsable, our *greedy fragment combination* algorithm iteratively builds a maximum partial covering by repeatedly selecting the constituent which covers the greatest number of input words without overlapping with the current covering set. If two non-overlapping constituents both cover the same number of words, the algorithm chooses the most informative one, (greatest number of semantic predicates) or on further ambiguity the one with the higher probability.

8 DATA COLLECTION

The BeRP system was bootstrapped with a Wizard of Oz[3] system [6], allowing us to collect naturalistic data before the first version of the system was built. The test subjects interacted with a windowing system monitored by the hidden operator (wizard) in another room. The wizard used an interface tool which helped formed database queries and generate canned responses to the user. With the 723 sentences collected from this system, we trained the MLP and the bigram LM for our recognizer and built a natural-language backend to serve as our prototype system, using this system in a second data-collection phase. Finally, we used the data from these first two efforts to build a third system, and collected further sentences.

Data was collected in a semi-quiet office, with no attempt at suppressing the environmental noise in the room. Speech was recorded digitally at 16 kHz sampling rate using a Sennheiser close-talking microphone. In all of our data-collection efforts, participants are first given an instruction sheet which gives them the task of finding information about one or more restaurants in Berkeley. The average age of our subjects was 32, and their native languages are summarized below:[4]

Language	Spkrs	Language	Spkrs	Language	Spkrs
American	71	British	9	French	3
German	29	Hebrew	4	Spanish	2
Italian	11	Mandarin	4	Japanese	2

Table 3: Speaker's Native Languages in the BeRP corpus

9 ACCENT

As Table 3 shows, the users of the BeRP system, like the citizens of the United States, speak English with a broad variety of accents.

[3]or PNAMBC (Pay No Attention to the Man Behind the Curtain).

[4]Plus one speaker each of Cantonese, East Indian English, Polish, Konkani, Turkish, Greek, Bulgarian, Kannada, and Australian.

In order to improve our robustness to accents, we have built baseline systems for automatically *detecting* and *modeling* foreign accents of English, while retaining high performance with native American English speech. We are focusing for the present on German-accented English, by building HMM word models which include German pronunciations, and using accent-detection algorithms to determine what weight to assign the foreign versus native pronunciations.

To build HMMs which include foreign pronunciations, we augmented the lexicon induction procedure (§5) with hand-written multi-language phonological rules. Given a set of canonical pronunciations as input, the rules generate a broad variety of multiple pronunciations which are then trained separately on German and American portions of the BeRP corpus. The procedure successfully found that final devoicing of stop consonants and fricatives was used only by Germans, while nasal and dental flapping were used by American but not Germans. Our database of German speech is still too small for conclusive quantitative results, however.

To detect foreign accent, in order to know when to use the foreign pronunciations, we have focused on *acoustic-phonetic* and *syntactic* information. Our initial acoustic-phonetic detector is an MLP with one output unit for each accent (the same architecture as in Figure 3 but with only two output units). The accent of the speaker who produced the input frame is used as the desired output. During test, we accumulate the MLP outputs over the course of a sentence to estimate the probability of each accent. Our initial system achieves 67.9% correct speaker-independent accent identification at the sentence level on an accent balanced test set. Our second detector uses syntactic information about the slightly different dialects used by native versus non-native speakers to help distinguish them. We trained different SCFGs for the German and American speakers, and used our probabilistic chart parser to compute $P(German|s)$ and $P(American|s)$ for each sentence s. The syntactic detector functions at 60% on the same test set as the phonetic detector.

We have just begun experimenting with combining accent detection and modeling. Our first experiment (see Table 4) used accent information to choose between an American-tuned system and a German-tuned system. Our preliminary conclusions are that while using all the data for training is a potent method, explicit accent information could be useful, as the true accent system (nonsignificantly) outperformed all others.

Information Used To Choose System	Lexicon	
	Single Pron	Multiple Pron
Accent ID	32.1	30.0
Sentence Probability	31.0	28.5
None (combined training only)	30.6	28.8
True Accent	29.3	28.3

Table 4: % word error on 554-sentence accent balanced test set.

10 RESULTS AND CONCLUSIONS

The BeRP system has proved quite successful as a framework for our speech-related experiments. In addition, it has proved quite usable in its function as a database frontend, with a word error rate of 32.1% and a semantic sentence error rate of 34.1%.[5]

The table below shows the system's overall sentence semantic error rate, measured by comparing with a hand-designed correct query component for each sentence. This correct query is compared to the query produced by the entire system and also to the query produced by just the backend operating without the recognizer on hand-transcribed sentences. Finally we show the results of comparing the recognizer output with the output produced by the natural language on perfect strings, to get an idea for the semantic performance of the recognizer.

	Semantic Error Rate
BeRP system	34.1
Backend alone	18.1
Recognizer alone	27.7

Table 5: BeRP semantic performance

We are currently extending the BeRP system to use visual data from the speakers' lips, either to aid in recognition, or to help determine when a speaker has started speaking [3]. Further details of the BeRP system are presented in [13] and [8].

Acknowledgments

Thanks to Emily Bender, Hervé Bourlard, Jerry Feldman, Hana Filip, Hynek Hermansky, Ron Kay, Yochai Konig, Robert Moore, Steve Omohundro, Patti Price, and Liz Shriberg. This work was partially funded by ICSI, an SRI subcontract from ARPA contract MDA904-90-C-5253, and ESPRIT project 6487 (The Wernicke project).

References

[1] J. Boite, H. Bourlard, B. D'hoore, and M. Haesen. A new approach towards keyword spotting. In *Proceedings of Eurospeech 93*, 1993.

[2] H. Bourlard and N. Morgan. Merging multilayer perceptrons & Hidden Markov Models: Some experiments in continuous speech recognition. In E. Gelenbe, editor, *Artificial Neural Networks: Advances and Applications*. North Holland Press, 1991.

[3] Chris Bregler, Stephen Omohundro, Yochai Konig, and Nelson Morgan. Using surface-learning to improve speech recognition with lipreading". In *Proc. 28th Annual Asilomar Conf. on Signals, Systems, and Computers*, Pacific Grove, CA, 1994. To appear.

[4] M. Cohen, H. Franco, N. Morgan, D. Rumelhart, and V. Abrash. Hybrid neural network/Hidden Markov Model continuous speech recognition. In *ICSLP-92*, 915—918, Banff, Canada, 1992.

[5] M. H. Cohen. *Phonological Structures for Speech Recognition*. PhD thesis, University of California, Berkeley, 1989.

[6] N. M. Frazer and G. N. Gilbert. Simulating speech systems. *Computer Speech and Language*, 5:81–99, 1991.

[7] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn. RASTA-PLP speech analysis technique. In *IEEE ICASSP-92*, I.121–124, San Francisco, CA, 1992.

[8] Daniel Jurafsky, Chuck Wooters, Gary Tajchman, Jonathan Segal, Andreas Stolcke, and Nelson Morgan. Integrating advanced models of syntax, phonology, and accent/dialect with a speech recognizer. In *AAAI Workshop on Integrating Speech and Natural Language Processing*, Seattle, 1994. to appear.

[9] N. Morgan. The ring array processor (RAP): A multiprocessing peripheral for connectionist applications. *Journal of Parallel and Distributed Computing*, 14:248—259, 1992.

[10] S. Renals, N. Morgan, H. Bourlard, M. Cohen, H. Franco, C. Wooters, and P. Kohn. Connectionist speech recognition: Status and prospects. TR-91-070, ICSI, Berkeley, CA, 1991.

[11] Andreas Stolcke and Stephen Omohundro. Best-first model merging for hidden Markov model induction. TR-94-003, ICSI, Berkeley, CA, January 1994.

[12] Andreas Stolcke and Jonathan Segal. Precise n-gram probabilities from stochastic context-free grammars. In *Proceedings of the 32nd ACL*, Las Cruces, NM, 1994. To appear.

[13] Charles C. Wooters. *Lexical Modeling in a Speaker Independent Speech Understanding System*. PhD thesis, University of California, Berkeley, CA, 1993. available as ICSI TR-92-062.

[14] Chuck Wooters and Andreas Stolcke. Multiple-pronunciation lexical modeling in a speaker-independent speech understanding system. In *ICSLP-94*, 1994. To appear.

[15] Victor Zue, James Glass, David Goodine, Hong Leung, Michael Phillips, Joseph Polifroni, and Stephanie Seneff. Integration of speech recognition and natural language processing in the MIT VOYAGER system. In *IEEE ICASSP-91*, I.713–716, 1991.

[5] These numbers do not include our recent tight-coupling augmentations.