THE AT&T WATSON SPEECH RECOGNIZER

V. Goffin, C. Allauzen, E. Bocchieri, D. Hakkani-Tür, A. Ljolje, S. Parthasarathy,
M. Rahim, G. Riccardi and M. Saraclar
AT&T Labs – Research
180 Park Ave, Florham Park, NJ 07932 USA
{vjg,allauzen,enrico,dtur,alj,sps,mazin,dsp3,murat}@research.att.com

ABSTRACT

This paper describes the AT&T WATSON real-time speech recognizer, the product of several decades of research at AT&T. The recognizer handles a wide range of vocabulary sizes and is based on continuous-density hidden Markov models for acoustic modeling and finite state networks for language modeling. The recognition network is optimized for efficient search. We identify the algorithms used for high-accuracy, real-time and low-latency recognition. We present results for small and large vocabulary tasks taken from the AT&T VoiceTone® service, showing word accuracy improvement of about 5% absolute and real-time processing speed-up by a factor between 2 and 3.

1. INTRODUCTION

Automatic Speech Recognition (ASR) has become an integral part of the emerging electronic contact market, enabling human-machine communication through spoken dialog. The success of ASR is attributed to several factors including the availability of large amounts of human-transcribed data for training, powerfull computers to support multi-channel and real-time operation, and algorithmic advances in areas of speech and signal processing leading to more accurate, efficient and robust modeling.

Although word accuracy has always been the standard measure used for evaluating speech recognition systems, many other factors need to be considered to successfully deploy a spoken dialog system. Several measures, including accuracy and robustness, real-time, memory foot-print, latency, barge-in detection and rejection of out of vocabulary events, need to be jointly optimized when deploying large-scale ASR services.

In this paper we describe the WATSON real-time speech recognizer, the product of several decades of speech research at AT&T. The recognizer is speaker-independent and uses context-dependent continuous density hidden Markov models (HMM) for acoustic modeling and finite-state networks for network optimization and search. For improved acoustic modeling, WATSON uses task-specific phoneme sets that are trained with linear discriminative analysis, vocal tract length normalization, constrained model optimization, maximum mutual information and maximum likelihood linear regression.

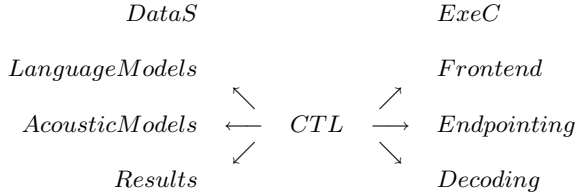
One innovation of the WATSON recognizer is its ability to support recognition for any vocabulary size. This is achieved by representing the search space as a weighted finite-state machine (FSM). The language model (G), rule-based or stochastic, determines valid word sequences and their probabilities, the lexicon (L) determines phone sequences for each word and the context FSM (C) selects the acoustic HMM for each phone. The final network (CLG) is composed and optimized using very general FSM algorithms such as composition, minimization and weight pushing. By substituting the G and recomposing the network, WATSON can support a range of tasks from small to large vocabulary, up to several millions of words. We will present results on two sets of data taken from the AT&T VoiceTone® service, a small vocabulary task (alpha-digits), and a large vocabulary task acquired from a natural language dialog application.

2. THE WATSON SPEECH RECOGNIZER

The WATSON environment includes a core recognizer, tools and libraries of reusable grammars (for dates, times, etc.). The core recognizer is a Viterbi decoder that can output lattices and confidence scores. The tools are for feature extraction, acoustic and language modeling, pronunciation modeling and network optimization. The environment enables users to create or customize real-time, multi-lingual, speech and speaker recognition applications.

2.1. Software Architecture

The core recognizer is organized around a Controller (CTL) that has access to a Data Store (DS) and an Execution Context (EC). The DS mediates access to all data: language models, acoustic models, word dictionaries, speaker profiles, and includes a results history. The data itself can reside in memory, in local files or in a separate database. The core EC represents all sections of the algorithm pipeline: feature extraction, feature normalization, speech silence detection, endpointing, bargein detection, decoding and scoring.



The CTL guides the acoustic signal and control parameters through the EC pipeline in real time. It divides the task into small processing steps during which the algorithms consume and produce chunks of data, all of which are attached to events in an *event queue*. The challenge lies in defining a complete set of well defined events. The CTL's role is to manage the event queue efficiently. It does this by calling on those algorithms that have registered interest in particular events and by monitoring their response times.

The CTL, DS and EC modules, and the event based processing, make it easier to extend the core configuration to handle tasks beyond simple recognition. One important result is that the CTL

does not grow in complexity. Adding a major extension may require refining the event model and enhancing existing modules but most of the new complexity is confined to the new module.

For example we handle Speaker Verification (SV), by adding an SV module to the EC. Like any other EC module it registers itself with the CTL and expresses interest in certain events, in this case this includes *phrase-result*. When *phrase-result* occurs, the CTL passes control to the SV module, along with access to the event queue and the DS. Recent utterances are accessible through the DS and so are individual speaker models, so SV can proceed. The SV disposition is added to the current result and control returns to the CTL. The result is eventually serialized and forwarded to the client, as before. The same paradigm is applied for supporting the real-time implementation of the Vocal Tract Length Normalization (VTLN) algorithm. Besides changing the front-end to generate wrapped features and adding a new VTLN module, the CTL itself remains independent of VTLN.

2.2. Acoustic Feature Extraction

WATSON acoustic features are derived from the mel-frequency cepstrum of audio frames. The frames are generated every 10 ms and cover a 20 ms time window. A vector of 21 cepstral coefficients and energy is computed for every frame, and mean-normalization is applied in real-time (with a 300 ms lookahead). To capture the speech signal dynamics, 11 consecutive cepstrum vectors are concatenated into a super-vector and then projected onto a 60 dimensional feature space. The projection combines a discriminative feature extraction technique known as Heteroscedastic Discriminant Analysis (HDA) [1] and a decorrelating linear transformation meant to ensure minimum loss of likelihood due to the diagonal covariance Gaussian mixture distributions of our HMM states.

2.3. Acoustic Modeling

The WATSON recognizer incorporates many recent advances in acoustic modeling. Parameter tying is available for mixture components, mixtures (states), and by careful design of the dictionaries even for HMMs and words. Any configuration of those levels that can be represented by a finite state network is supported [2].

In most of our applications we use Gaussian mixture tied-state three/four-state left-to-right HMM-based acoustic models. When models are needed to simultaneously support several tasks, an task-specific phoneme sets are used which provide task dependent performance, even if different tasks appear in the same utterance. An example would be a speaker who identifies him/her-self (name recognition), provides an account number (alphadigit recognition) embedded within conversational language (general English) [3]. Each part of the utterance is handled by a different part of the model, with its own phoneme set, with full context dependency across the task boundaries. Additionally, we use tree-based context clustering for general English while at the same time, and in the same model, we use head-body-tail models for digits.

All the acoustic models are initially trained using the Maximum Likelihood Estimation (MLE) criterion, followed by Maximum Mutual Information Estimation (MMIE) criterion.

In addition, WATSON provides low latency real-time Vocal Tract Length Normalization (VTLN) [4]. This algorithm is effective even on short utterances from a given speaker.

2.4. Acoustic Likelihood Calculation

For likelihood calculations we use parallel evaluation when possible. With Intel processors this means using SSE instructions to schedule up to 4 parallel floating point operations. We observe the expected reduction in likelihood calculation cycles.

Precalculating acoustic likelihoods for several frames at each active HMM state reduces the likelihood calculation load by another 10% to 40% [5].

Even after applying this look-ahead, the fraction of decoding time devoted to the state likelihoods is significant, ranging from 45% to 65%, depending on the task. Gaussian Selection (GS) [6] aims to limit the number of computed Gaussian exponents, by ignoring the Gaussians yielding negligible contributions to the state likelihoods. Our recent GS implementation uses non-overlapping shortlists of Gaussian neighbors, similar to [7]. During decoding, likelihood computation of an input vector is enabled only for a small fraction (e.g. 32 out of 256) of Gaussian shortlists nearest to the vector. States with no Gaussians in the selected shortlists, and states of "silence" models, are specially handled. The total computation speed-up provided by GS, after state lookahead, varies from 24% to 30%, measured on 6 tasks from connected alpha-digits to very large vocabulary continuous speech recognition.

2.5. Language Modeling

WATSON supports any language model that can be represented as a weighted finite state machine (FSM), and has tools to compile rule-based and stochastic grammars into FSMs.

Rule based grammars are used for lower complexity tasks and tasks for which no training data is available. Examples of these are dates, times, confirmations and account numbers.

Corpus based stochastic language models are used for higher complexity tasks, such as spoken dialog applications where users are prompted to speak naturally. These models can be implemented as weighted FSMs and in particular as Variable Nondeterministic Stochastic Automata (VNSA). A VNSA is a weighted FSM that can parse an arbitrary sequence of words from a given vocabulary. In its simplest implementation a state of a VNSA represents the history of a word sequence and the VNSA itself is a compact representation of the probability distribution over all possible word sequences. By appropriately defining the state space to incorporate lexical (word) and extra lexical information (e.g. part of speech tags), the VNSA formalism can generate a wide class of probability distributions (e.g. standard word n-gram, class-based, phrase-based). The transition probabilities and state space are learned via self-organizing algorithms [8, 9].

2.6. Search Algorithms

The different constraints used in speech recognition can naturally be represented by weighted automata and can then be combined to obtain a recognition network. Weighted *composition* is used to combine the component automata, while *determinization*, *minimization* and *weight-pushing* optimize the result in time and space. See [10] for a description of these general algorithms.

The construction and optimization of the recognition network (*transducer*) that we use is presented in details in [11]. Here, we will only give a brief outline of the construction. The recognition transducer N, mapping directly sequences of context-depend phones to sequences of words, is built by combining the context-dependency transducer C, the pronunciation dictionary L and the

language model G in the following way:

$$N = \pi_\epsilon(C \circ \det(\hat{L} \circ G))$$

where $\circ$ denotes the composition of weighted transducers, $\hat{L}$ indicates that disambiguation symbols have been inserted in L to ensure the determizability of $L \circ G$, det stands for the weighted determinization algorithm applied in the log semiring and π_ϵ replaces the disambiguation symbols with ϵ transitions. Additionally, weighted (encoded) minimization can be used to reduce the size of the intermediate and final transducers.

Ideally, the weight pushing algorithm [10] in the log semiring should be applied to N to ensure a distribution of the weights along the paths beneficial to the pruning efficacy of a standard Viterbi beam search. But, since we are applying determinization in the log semiring, it is actually sufficient to ensure the stochasticity of each components, *i.e.* L and G.

For L, this can be done by ensuring that the different pronunciations of a word are weighted according to a probability distribution – the uniform distribution for instance. For G, a source of non-stochasticity is the way the silence tokens are handled. Silence tokens are typically not modeled within language models. They are added after the fact, most commonly as free cost loops at various states in the automata representation of the model, such as the initial, final, and unigram states. This makes the model non-stochastic. As described in [11], we use a stochastic silence model where after the emission of each word, there is probability p of emitting a silence token and probability $1 - p$ of stopping emitting silence tokens and being able to emit the next word.

A synchronous Viterbi beam search is then used to find given an acoustic observation, the path in N that minimize the combination of the grammar weight, the weight of that path in N, and the acoustic weight.

The FSM library [12] is used for the representation and manipulation of weighted transducers and automata, the GRM library [13] for the construction of the language model G, and the `dmake` utility from the DCD library [14] for the construction and optimization of the recognition transducer.

2.7. Confidence Scores

Confidence scores are important in ASR for detecting out of vocabulary words and misrecognized phrases.

Scores can be of acoustic origin, computed through a likelihood ratio test where the numerator is the likelihood of speech feature frames along the best path and the denominator is the likelihood of the same frames either evaluated on a generic-speech HMM (*garbage score*) or taken from the best active network node for each frame (*unconstrained score*). The ratios are normalized using a task dependent curve to fall between 0 and 1 and then combined. Acoustic scoring works best for small rule based grammars.

We also use the lattice output of the ASR to generate word confidence scores. The algorithm is based on the *pivot* alignment for strings in the word lattice. To obtain word confidence scores, we combine the posterior probabilities of the transitions in the lattice corresponding to the same word, and which occur at around the same time interval. A detailed explanation of this algorithm and the comparison of its performance with other approaches is presented in [15].

To evaluate the quality of confidence scores we introduce a confidence threshold and look at the Equal Error Rate (EER) in a binary classification scheme where each word is classified as correctly recognized if its confidence score exceeds the threshold and as misrecognized otherwise. The EER is the point where the false rejection and false acceptance rates are equal. For example, on a test set of 2,174 utterances (31,018 words) from the AT&T Spoken Dialog System Database where the word accuracy is 70.5%, the EER is 22.5%. The quality of the scores increases, witnessed by a lower EER, as the recognition beam increases. The computation of word scores adds a 2-4% overhead to the total recognition time. Once computed, the scores can be combined (via arithmetic or geometric mean, etc.) to generate whole sentence scores.

2.8. Endpointing and Bargein

The role of endpointing is to keep excessive silence out of the decoder and to promptly detect the end of user speech. We use an energy based endpointer for speech detection and improve detection of end of sentence using feedback from the decoder.

Bargein allows the user to start talking before the prompt is over. For a smooth bargein the prompt has to be cut off as soon as the bargein is determined to be real and the determination has to happen quickly. This is obviously a balancing act. We have been using a language model based bargein where bargein is triggered when recognition reaches specially designated nodes.

3. EXPERIMENTAL RESULTS

We now present results from two sets of data acquired from the AT&T VoiceTone® service. In both cases the acoustic model used has been trained on over 100 hours of speech. This includes field data representing digits, names, addresses, as well as natural-language customer care utterances. For each of the tasks, we show the trade-off between word accuracy and CPU time/audio time by simply varying the width of the Viterbi search beam. Each result displays three curves, the first one showing the baseline ML performance, the second one showing improvements after MMIE training and network optimization, and the third one showing additional benefits after Gaussian selection and real-time VTLN. The baseline ML performance already includes SSE processing and likelihood precalculation.

The alpha-digit task (Figure 1) represents 1,000 7-character strings. The natural-language customer care task (Figure 2) represents 5,000 utterances. For a given operating point just after the knee in the curves (a) MMIE training and network optimization provide 3-4% absolute improvement in word accuracy and a factor of nearly two speed-up in processing time, and (b) by further adding Gaussian selection and real-time VTLN, we obtain over 5% absolute improvement in word accuracy and a factor of 2-3 speed-up in processing time over the baseline system. The results show the impact of WATSON's more advanced algorithms on different types of language models and task complexities.

4. SUMMARY

We have outlined the current configuration of the AT&T WATSON speech recognizer. The algorithms identified were selected for their high effectiveness during field performance. We showed a 5% absolute improvement in word accuracy and a factor of 2 to 3 speed-up in processing using these algorithms.

The WATSON recognizer is currently being used to support several large-scale speech applications as part of the AT&T VoiceTone®

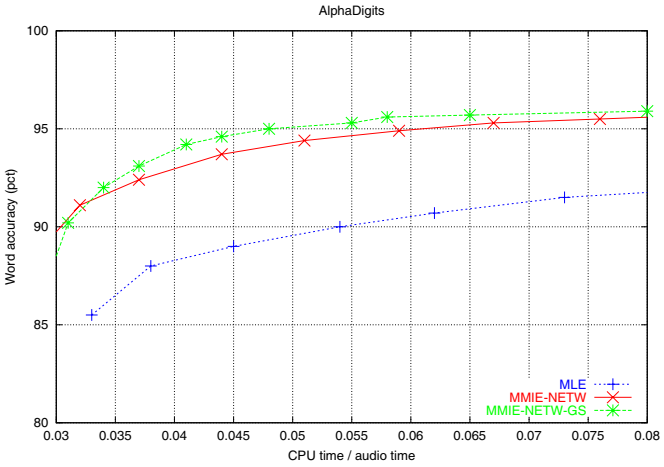


Fig. 1. Rule based grammar results (Alphadigit strings).

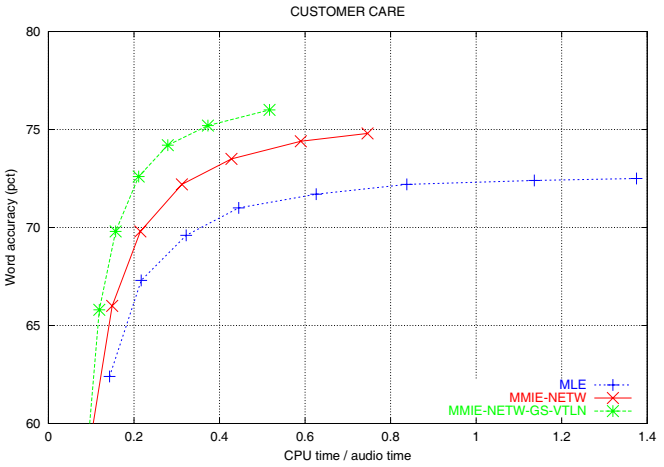


Fig. 2. Stochastic grammar results (Customer care dialog).

service. We continue to research WATSON configurations and algorithms that can improve the real-time performance of these applications.

5. ACKNOWLEDGMENTS

We would like to thank R. Knag and I. Arizmendi for their contribution in the development of the WATSON recognizer.

6. REFERENCES

[1] N. Kumar and G. Andreou, "Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition," *Speech Communication*, vol. 26, pp. 283–297, 1998.

[2] A. Ljolje, M. Saraclar, M. Bacchiani, M. Collins, and B. Roark, "The at&t rt-02-stt gt10xrt system," in *RT02 Workshop*, Vienna, Virginia, 2002.

[3] A. Ljolje, "Multiple task-domain acoustic models," in *Proc. ICASSP*, 2003.

[4] A. Ljolje, V. Goffin, and M. Saraclar, "Low latency real-time vocal tract length normalization," in *Text, Speech and Dialog*, Brno, Czech Republic, September 2004.

[5] M. Saraclar, M. Riley, E. Bocchieri, and V. Goffin, "Towards automatic real time broadcast news transcription," in *Proc. ICSLP*, 2002.

[6] E. Bocchieri, "Vector quantization for the efficient computation of continuous density likelihoods," in *Proc. ICSLP-96*, Mineapolis, April 1993, pp. 692–695.

[7] G. Saon, G. Zweig, B. Kingsbury, L. Mangu, and U. Chaudhari, "An architecture for rapid decoding of large vocabulary conversational speech," in *Proc. Eurospeech*, Geneva, 2003, pp. 1977–1980.

[8] G. Riccardi, R. Pieraccini, and E. Bocchieri, "Stochastic automata for language modeling," in *Computer Speech and Language*, 1996, vol. 10(4), pp. 265–293.

[9] G. Riccardi, A. L. Gorin, A. Ljolje, and M. Riley, "A spoken language system for automated call routing," in *Proc. ICASSP*, Munich, 1997, pp. 1143–1146.

[10] M. Mohri, "Finite-state transducers in language and speech processing," *Computational Linguistics*, vol. 23, no. 2, 1997.

[11] M. Riley C. Allauzen, M. Mohri and B. Roark, "A generalized construction of integrated speech recognition transducers," in *Proc. ICASSP*, 2004, vol. I, pp. 761–764.

[12] F. Pereira M. Mohri and M. Riley, "The design principles of a weighted finite-state transducer library," *Theoretical Computer Science*, vol. 231, no. 1, pp. 17–32, 2000, http://www.research.att.com/sw/tools/fsm.

[13] C. Allauzen M. Mohri and B. Roark, "A general weighted grammar library," in *Proceedings of the Ninth International Conference on Implementation and Application of Automata (CIAA 2004)*. 2004, vol. to appear of *Lecture Notes in Computer Science*, Springer, http://www.research.att.com/sw/tools/grm.

[14] M. Mohri and M. Riley, *DCD Library - Decoder Library, software collection for decoding and related functions*, AT&T Labs - Research, 2003, http://www.research.att.com/sw/tools/dcd.

[15] D. Hakkani-Tür and G. Riccardi, "A general algorithm for word graph matrix decomposition," in *Proc. ICASSP*, Hong Kong, 2003.