RASR – The RWTH Aachen University Open Source Speech Recognition Toolkit

D. Rybach, M. Bisani[1], P. Dreuw[1], C. Gollan[1], S. Hahn, G. Heigold[1], B. Hoffmeister[1], S. Kanthak[1], P. Lehnen, J. Lööf[1], D. Nolden, M. Pitz[1], M. Sundermeyer, Z. Tüske, S. Wiesler, A. Zolnay[1], R. Schlüter, H. Ney

Human Language Technology and Pattern Recognition,
Computer Science Department, RWTH Aachen University, 52056 Aachen, Germany
rasr@i6.informatik.rwth-aachen.de
www.hltpr.rwth-aachen.de/rasr

***Abstract*—RASR is the open source version of the well-proven speech recognition toolkit developed and used at RWTH Aachen University. The current version of the package includes state of the art speech recognition technology for acoustic model training and decoding. Speaker adaptation, speaker adaptive training, unsupervised training, discriminative training, lattice processing tools, flexible signal analysis, a finite state automata library, and an efficient dynamic network decoder are notable components. Comprehensive documentation, example setups for training and recognition, and tutorials are provided to support newcomers.**

I. Introduction

The interest in speech recognition technology has grown over the last years. For researchers it requires a lot of effort to develop a speech recognition system from scratch, which impedes innovations. Publicly available toolkits, often published under an open source license, facilitate the introduction to research in this area. A couple of open source system are available, for example CMU Sphinx [1], the HTK Toolkit [2], Julius [3], and Kaldi [4].

RASR (short for RWTH ASR) has been designed for the special requirements of research applications. On the one hand it should be very flexible, to allow for rapid integration of new methods, and on the other hand it has to be efficient, so that new methods can be studied on real-life tasks in reasonable time and system tuning is feasible. The flexibility is achieved by a modular design, where most components are decoupled from each other and can be replaced at runtime. The API is subdivided into several modules and allows for an integration of (high and low level) methods in external applications.

The applicability of our toolkit to real-life tasks has been proven by building several competitive large vocabulary systems in recent international research projects, for example TC-STAR (European English and Spanish) [5], GALE (Arabic, Mandarin) [6], [7], and Quaero (English, French, German, Polish, and Spanish) [8]. For some of these systems, we have to deal with huge vocabularies and need to process thousands of hours of speech data.

The flexibility of the toolkit allows for the rapid development of applications also in other domains, for example continuous sign language recognition using video input [9] and optical character recognition (OCR), in particular handwriting recognition [10]. The OCR system is publicly available, too (cf. Section VIII).

An important aspect for developing a system for a large vocabulary task is the support for grid-computing. Nearly all processing steps for acoustic model training and decoding can be distributed in a cluster computer environment. The parallelization scales very well, because we divide the computations on the segment level, which requires synchronization only at the end of the computation.

The toolkit is available for download on our website [2]. We publish our toolkit under an open source license, called "RWTH ASR License", which is derived from the Q Public License v1.0 This license grants free usage including redistribution and modification for non-commercial use. Publications of results obtained through the use of original or modified versions of the software have to cite our paper [11]. RASR runs on Linux and Mac OS X.

The RASR website also offers comprehensive documentation, tutorials, and recipes for system development. Support is offered in form of a forum as part of the website. Furthermore, we offer a ready-to-use recognizer for English.

In the remainder of this paper we describe the individual parts of the framework. First we depict the acoustic front-end and the used models. Then we present the decoder, lattice processing tools, the finite-state automata library, extensions, and finally the documentation and supplementary materials.

II. Signal Analysis

Methods for signal analysis are implemented in a generic framework, called Flow, which is described in the next section. The predefined acoustic features computed using this framework are defined in the following section.

A. Flow Networks

The Flow module offers a generic framework for data processing. The data flow is modeled by links connecting several data processing nodes to a network. The networks are created at runtime based on a network definition in XML

[1]Former staff members

[2]http://www.hltpr.rwth-aachen.de/rasr

documents, which makes it possible to implement or modify data processing tasks without modifying the software.

Flow networks are used to compute acoustic features as well as to generate and process dynamic time alignments, i.e. mappings from acoustic features to HMM states. Using a caching mechanism, which is also implemented as a node, acoustic features and alignments can be re-used in processing steps requiring multiple iterations.

B. Acoustic Features

The basic nodes in a Flow network implement the reading of waveforms from audio files, computing an FFT, miscellaneous vector operations, and different types of signal normalization. The networks included in the toolkit compute MFCC features and a voicing feature [12]. Temporal context can be incorporated by using derivatives of the acoustic features or an LDA transformation [13].

The flexibility of the Flow module allows for an easy implementation of other acoustic features as well as for the integration of externally computed features.

III. Acoustic Modeling

The acoustic model consists of the transition, the emission, and the pronunciation model. The pronunciation model gives for each word in the vocabulary a list of pronunciations together with a probability of the occurrence. A pronunciation is modelled by a sequence of context dependent phonemes. In the current version, the context is limited to triphones, including context across words.

Strict left-to-right HMM topologies are supported, each representing a context dependent phoneme. Except for silence, which is modeled by a single state, all HMMs consist of the same number of states. The transition model implements loop, forward, and skip transitions. The existing toolkit supports a global transition model which distinguishes only the silence state. Transitions leaving a word are penalized with an extra cost, the word penalty.

The emission probability of an HMM state is represented by a Gaussian mixture model. By default, globally pooled variances are used. However, several other tying schemes, including density-specific diagonal covariance matrices, are supported. The acoustic model score computations are optimized for globally pooled variances though.

We provide tools to convert HTK acoustic models to RASR models. However, not all parameters of the HTK models can be used, especially the state dependent transition probabilities.

A. State Tying: Phonetic Decision Trees

RASR includes tools to train classification and regression trees (CART) for phonetic decision trees [14]. The configuration of the CART training is flexible and supports a variety of phonetic decision tree based tyings. For example, English systems usually perform best when estimating a separate tree for each combination of central phoneme and HMM state. On the other hand, languages like Mandarin benefit from applying a less strict separation. In addition, the CART software supports randomization to generate several acoustic models for subsequent system combination.

State tying definitions from external tools can be imported by using look-up tables stored in simple text files.

B. Confidence Scores

The relation between the competing hypotheses in a word graph can be computed by estimating the lattice link posterior probabilities [15]. Depending on the lattice link labels and the structure of the lattice it is possible to compute confidence scores for different units, e.g. word, pronunciation, or HMM state confidence scores.

For the unsupervised refinement or re-estimation of the acoustic model parameters (unsupervised training) the toolkit supports the generation and processing of confidence weighted state alignments. Confidence thresholding on state level is supported for unsupervised training as well as for unsupervised adaptation methods. The toolkit supports different types of state confidence scores, all described in [16]. The emission model can be re-estimated based on the automatically annotated observations and their assigned confidence weights, as presented in [17].

C. Speaker Normalization and Adaptation

RASR supports several methods for speaker normalization and adaptation: Vocal tract length normalization (VTLN) [18], maximum likelihood linear regression (MLLR) [19], feature space MLLR (fMLLR, also known as constrained MLLR, CMLLR) [20], and dimension reducing affine transforms [21].

VTLN is implemented as a parametric linear warping of the MFCC filter bank, as described in [22]. The parameter is estimated using maximum likelihood. Support for one pass, or so called fast VTLN [18], is also included, by using Gaussian mixture model classifiers for choosing the warping factors.

Both, VTLN and fMLLR are implemented in the feature extraction front-end, allowing for use in both recognition and in training, thus supporting speaker adaptive training.

For MLLR, a regression class tree approach [23] is used to adjust the number of regression classes to the amount of adaptation data available. As a variation, it is possible to do adaptation using only the offset part (and not the matrix part) of the affine transform.

All the adaptation methods can be utilized for both unsupervised and supervised adaptation. fMLLR as well as MLLR estimation can make use of weighted observations, as produced by the confidence measures described in the previous section, allowing for confidence based unsupervised adaptation.

D. Acoustic Model Training

RASR includes tools for the estimation of Gaussian mixture models by both standard maximum likelihood training and discriminative training using the minimum phone error (MPE) criterion [24]. All training steps can be parallelized in a cluster computer environment, which is indispensable for state-of-the-art amounts of training data.

The offered documentation (cf. Section IX) includes training recipes (configuration files and shell scripts), which can be easily adapted for other tasks.

IV. Language Modeling

The toolkit does not include tools for the estimation of language models. However, the decoder supports N-gram language models in the ARPA format, produced e.g. by the SRI Language Modeling Toolkit [25]. The order of the language model is not limited by the decoder. Class language models, defined on word classes instead of words, are supported as well. Alternatively, a weighted finite state automaton representing a (weighted) grammar can be used.

V. Decoder

The decoder included in our toolkit is based on the history conditioned lexical tree (HCLT) search [26]. HCLT search is a one-pass dynamic programming algorithm which uses a pre-compiled lexical prefix tree as representation of the pronunciation dictionary. The search space is constructed dynamically by integrating parts of the LM as needed during search. Thereby the decoder can deal with huge vocabularies and complex language models in a memory efficient way [27].

The beam search strategy retains for every time step only the most promising hypotheses. Hypotheses with a too low score compared to the best state hypothesis are eliminated by *acoustic pruning*. The beam width, i.e. the number of surviving hypotheses, is defined by a threshold. *Language model pruning* is applied to the word start hypotheses after applying the language model, which further decreases the active search space. In addition, histogram pruning restricts the absolute number of active hypotheses.

The acoustic pruning can be refined by incorporating the language model probabilities as early as possible using a language model look-ahead [28]. The anticipated language model probability for a certain state in the tree is approximated by the best word end reachable.

The tree lexicon is constructed from the tied HMM-state sequences of the pronunciations of the words in the vocabulary. Across-word context dependent models are supported by the decoder as well [29].

The decoder can generate word graphs (lattices) which is a compact representation of the set of alternative word sequences with corresponding word boundaries [30]. This word graph can be used in later processing steps. Our system produces word graphs as finite-state automata with attached word boundaries or alternatively in the HTK standard lattice format.

The computation of acoustic likelihoods can be optionally accelerated by the use of SIMD instructions [31], batched computations, and density pre-selection. Scalar quantization can be applied to both acoustic feature vectors and means of the mixture models, thus reducing the score computation to integer operations.

VI. Lattice Processing

Lattice processing tools can be used for the post-processing of the word graphs generated by the decoder. RASR includes a feature-rich framework for lattice processing. Major methods implemented in this framework are: several techniques for confusion network (CN) construction, CN decoding, lattice- and CN-based system combination [32], n-best list generation, and word confidence score estimation.

The individual methods can be combined with basic operations (e.g. lattice pruning, file operations, format conversion) to form a data processing network similar to Flow (cf. Section II-A), yielding an implementation of a complete post-processing pipeline.

VII. Finite-state Automata

RASR uses finite-state automata for several tasks. The computation of dynamic time alignments, required for acoustic model training and speaker adaptation, uses automata for the construction and representation of the search space. Furthermore, the word lattices generated by the speech recognizer are represented by finite-state automata. Therefore, the lattices generated can easily be post-processed by algorithms defined on weighted finite state transducers.

Finite-state automata are handled by the included RWTH FSA Toolkit [33], which is also available separately under an open source license.

VIII. Extensions

RWTH OCR [3] is an add-on for RASR which adds support for image (sequence) processing and can be used to develop competitive handwriting recognition systems [10].

For educational purposes, we offer a small add-on containing two basic and simple decoders, which can be used in lab courses for example.

IX. Documentation

The documentation is organized in a wiki [4] and covers all steps of the acoustic model training, multi-pass recognition, and describes the common concepts of the software and the used file formats. Emerging questions are answered by our developers in a support forum.

For a quick introduction, we created a step-by-step recipe for the development of a small (100 words) recognizer based on the freely available CMU Census Database. A more verbose tutorial describes the development of an open vocabulary ASR system from scratch, including acoustic model training, language model training, grapheme to phoneme conversion, and system evaluation based on RASR and other open source software tools [34]. Both tutorials can be found in the wiki.

In addition, we offer the acoustic model (triphones, 900K densities), the 4-gram language model (7.5M multi-grams), and the pronunciation dictionary (60K words) developed for our EPPS English system together with a ready-to-use one-pass recognition setup.

[3] http://www.hltpr.rwth-aachen.de/rwth-ocr
[4] http://www.hltpr.rwth-aachen.de/rasr/manual

REFERENCES

[1] W. Walker, P. Lamere, P. Kwok, R. S. Bhiksha Raj, E. Gouvea, P. Wolf, and J. Woelfel, "Sphinx-4: A flexible open source framework for speech recognition," Sun Microsystems, Inc, Tech. Rep. SMLI TR-2004-139, Nov. 2004.

[2] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*. Cambridge University Engineering Department, 2006.

[3] A. Lee, T. Kawahara, and K. Shikano, "Julius – an open source real-time large vocabulary recognition engine," in *EUROSPEECH*, Aalborg, Denmark, Sep. 2001, pp. 1691–1694.

[4] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *ASRU*, Big Island, Hawaii, USA, Dec. 2011.

[5] J. Lööf, C. Gollan, S. Hahn, G. Heigold, B. Hoffmeister, C. Plahl, D. Rybach, R. Schlüter, and H. Ney, "The RWTH 2007 TC-STAR evaluation system for European English and Spanish," in *INTERSPEECH*, Antwerp, Belgium, Aug. 2007, pp. 2145–2148.

[6] D. Rybach, S. Hahn, C. Gollan, R. Schlüter, and H. Ney, "Advances in Arabic broadcast news transcription at RWTH," in *ASRU*, Kyoto, Japan, Dec. 2007, pp. 449–454.

[7] C. Plahl, B. Hoffmeister, M.-Y. Hwang, D. Lu, G. Heigold, J. Lööf, R. Schlüter, and H. Ney, "Recent improvements of the RWTH GALE Mandarin LVCSR system," in *INTERSPEECH*, Brisbane, Australia, Sep. 2008, pp. 2426–2429.

[8] M. Sundermeyer, M. Nußbaum-Thom, S. Wiesler, C. Plahl, A. El-Desoky Mousa, S. Hahn, D. Nolden, R. Schlüter, and H. Ney, "The RWTH 2010 Quaero ASR evaluation system for English, French, and German," in *ICASSP*, Prague, Czech Republic, May 2011, pp. 2212–2215.

[9] P. Dreuw, D. Rybach, T. Deselaers, M. Zahedi, and H. Ney, "Speech recognition techniques for a sign language recognition system," in *INTERSPEECH*, Antwerp, Belgium, Aug. 2007, pp. 2513–2516.

[10] P. Dreuw, D. Rybach, G. Heigold, and H. Ney, *RWTH OCR: A Large Vocabulary Optical Character Recognition System for Arabic Scripts*. London, UK: Springer, Apr. 2011, ch. Part I: Development.

[11] D. Rybach, C. Gollan, G. Heigold, B. Hoffmeister, J. Lööf, R. Schlüter, and H. Ney, "The RWTH Aachen University open source speech recognition system," in *INTERSPEECH*, Brighton, U.K., Sep. 2009, pp. 2111–2114.

[12] A. Zolnay, R. Schlüter, and H. Ney, "Extraction methods of voicing feature for robust speech recognition," in *European Conference on Speech Communication and Technology*, vol. 1, Geneva, Switzerland, Sep. 2003, pp. 497–500.

[13] R. Haeb-Umbach and H. Ney, "Linear discriminant analysis for improved large vocabulary continuous speech recognition," in *ICASSP*, vol. 1, San Francisco, CA, USA, Mar. 1992, p. 1316.

[14] K. Beulen, E. Bransch, and H. Ney, "State-tying for context dependent phoneme models," in *EUROSPEECH*, vol. 3, Rhodes, Greece, Sep. 1997, pp. 1179–1182.

[15] G. Evermann and P. Woodland, "Large vocabulary decoding and confidence estimation using word posterior probabilities," in *ICASSP*, Istanbul, Turkey, Jun. 2000, pp. 1655 – 1658.

[16] C. Gollan and M. Bacchiani, "Confidence scores for acoustic model adaptation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Las Vegas, NV, USA, Apr. 2008, pp. 4289–4292.

[17] C. Gollan and H. Ney, "Towards automatic learning in LVCSR: Rapid development of a Persian broadcast transcription system," in *Interspeech*, Brisbane, Australia, Sep. 2008, pp. 1441–1444.

[18] L. Welling, S. Kanthak, and H. Ney, "Improved methods for vocal tract normalization," in *ICASSP*, vol. 2, Phoenix, AZ, USA, Mar. 1999, pp. 761 – 764.

[19] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer Speech and Language*, vol. 9, no. 2, pp. 171 – 185, Apr. 1995.

[20] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, no. 2, pp. 75 – 98, Apr. 1998.

[21] J. Lööf, R. Schlüter, and H. Ney, "Efficient estimation of speaker-specific projecting feature transforms," in *ICSLP*, Antwerp, Belgium, Aug. 2007, pp. 1557 – 1560.

[22] E. Eide and H. Gish, "A parametric approach to vocal tract length normalization," in *ICASSP*, vol. 1, Atlanta, GA, USA, May 1996, pp. 346 – 349.

[23] C. Leggetter and P. Woodland, "Flexible speaker adaptation using maximum likelihood linear regression," in *ARPA Spoken Language Technology Workshop*, Austin, TX, USA, Jan. 1995, pp. 104 – 109.

[24] D. Povey and P. Woodland, "Minimum phone error and I-smoothing for improved discriminative training," in *ICASSP*, Orlando, FL, USA, May 2002, pp. 105–108.

[25] A. Stolcke, "SRILM - an extensible language modeling toolkit," in *ICSLP*, Denver, CA, USA, Sep. 2002.

[26] H. Ney and S. Ortmanns, "Progress in dynamic programming search for LVCSR," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1224–1240, Aug. 2000.

[27] D. Rybach, R. Schüter, and H. Ney, "A comparative analysis of dynamic network decoding," in *ICASSP*, Prague, Czech Republic, May 2011, pp. 5184–5187.

[28] S. Ortmanns and H. Ney, "Look-ahead techniques for fast beam search," *Computer Speech and Language*, vol. 14, no. 1, pp. 15–32, Jan. 2000.

[29] A. Sixtus and H. Ney, "From within-word model search to across-word model search in large vocabulary continuous speech recognition," *Computer Speech and Language*, vol. 16, no. 2, pp. 245–271, May 2002.

[30] S. Ortmanns, H. Ney, and X. Aubert, "A word graph algorithm for large vocabulary continuous speech recognition," *Computer Speech and Language*, vol. 11, no. 1, pp. 43–72, Jan. 1997.

[31] S. Kanthak, K. Schütz, and H. Ney, "Using SIMD instructions for fast likelihood calculation in LVCSR," in *ICASSP*, Istanbul, Turkey, Jun. 2000, pp. 1531–1534.

[32] B. Hoffmeister, "Bayes risk decoding and its application to system combination," Ph.D. dissertation, RWTH Aachen University, Aachen, Germany, Jul. 2011.

[33] S. Kanthak and H. Ney, "FSA: An efficient and flexible C++ toolkit for finite state automata using on-demand computation," in *ACL*, Barcelona, Spain, Jul. 2004, pp. 510–517.

[34] S. Hahn and D. Rybach, "Building an open vocabulary ASR system using open source software," in *INTERSPEECH*, Florence, Italy, Aug. 2011, Tutorial M3.