Quaero 2010 Speech-to-Text Evaluation Systems

Sebastian Stüker, Kevin Kilgour, and Florian Kraft

1 Introduction

Our laboratory has used the HP XC4000, the high performance computer of the federal state Baden-Württemberg, in order to participate in the third Quaero evaluation (2010) for automatic speech recognition (ASR).

State-of-the-art ASR research systems usually employ techniques which require the parallel execution of several recognition systems for the purpose of system combination. The use of unsupervised adaptation techniques further requires the execution of several stages or passes of ASR systems. This leads to the fact that modern research systems process speech only with a run-time of many times realtime, under certain circumstances up to 100 times real-time. The process of speech recognition in this form can be easily parallelized at speaker level in independent processes without the need for inter-process communication. Therefore, the scheduling system of the XC4000 in combination with its global, high performing file space, is an ideal environment for executing such an evaluation.

In this paper we report on our 2010 ASR evaluation systems for English and German that we, at least in part, trained and executed on the XC4000, and that are an extension of our 2009 systems [11].

2 Quaero

Quaero (http://www.quaero.org) is a French research and development program with German participation. It targets to develop multimedia and multilingual indexing and management tools for professional and general public applications such

Sebastian Stüker · Kevin Kilgour · Florian Kraft
Research Group 3-01 ‘Multilingual Speech Recognition’, Karlsruhe Institute of Technology, Karlsruhe, Germany, e-mail: sebastian.stueker@kit.edu, kevin.kilgour@kit.edu, florian.kraft@kit.edu

as the automatic analysis, classification, extraction, and exploitation of information. The projects within Quaero address five main application areas:

- Multimedia Internet search
- Enhanced access services to audiovisual content on portals
- Personalized video selection and distribution
- Professional audiovisual asset management
- Digitalization and enrichment of library content, audiovisual cultural heritage, and scientific information.

Also included in Quaero is basic research in the technologies underlying these application areas, including automatic speech recognition, machine translation, and speech-to-speech translation. The vision of Quaero is to give the general public as well as professional user the technical means to access various information types and sources in digital form, that are available to everyone via personal computers, television, and handheld terminals, across languages.

Quaero is organized as a program consisting of seven projects. Five projects are concerned with applications. In addition, one project, the *Core Technology Cluster* (CTC), conducts basic research in the technologies underlying the application projects, and one project is concerned with providing the data resources necessary for the research within CTC.

Our laboratory is mainly involved in the CTC project. Two of the technologies under investigation are *Automatic Speech Recognition* (ASR), i.e. the automatic transcription of human speech into written records and *Machine Translation* (MT). Within Quaero research is driven by competitive evaluation and sharing of results and technologies employed. This process is called *coopetition*. Evaluations are conducted once a year on a predefined domain and a set of languages. As the project continues the number of languages to address will grow. Also the performance of the recognition systems developed within the project is expected to improve.

The third evaluation conducted in August 2010 was the second, real evaluation after the baseline evaluation in 2008 and the first evaluation in 2009. Seven languages were addressed: English, French, German, Greek, Polish, Russian, and Spanish. We participated in the languages English, German, Russian, and Spanish. The test data for the evaluation consisted of various audio files collected from the World Wide Web, including broadcast news, lectures, and video blogs.

3 English Evaluation Recognition Systems

For the 2010 English ASR evaluation within Quaero we participated with a recognition system that is a further development of our 2010 evaluation system [11]. The system has been trained and tested with the help of the Janus Recognition Toolkit that features the IBIS single pass decoder [13]. In general all recognition systems employ left-to-right Hidden Markov Models (HMMs), modeling phoneme sequences with 3 HMM states per phoneme. The general features of the system are the same as for 2009. Improvements over the 2009 system came from increased amounts of training data, and the use of a large language model that made it necessary to use memory mapping in order to share the language model among several processes running on the same node.

3.1 Front-End

We trained systems for two different kinds of acoustic front-ends. One is based on the traditional Mel-frequency Cepstral Coefficients (MFCC) obtained from a fast Fourier Transform and the other on the warped minimum variance distortionless response (MVDR). The second front-end replaces the Fourier transformation by a warped MVDR spectral envelope [16], which is a time domain technique to estimate an all-pole model using a warped short time frequency axis such as the Mel scale. The use of the MVDR eliminates the overemphasis of harmonic peaks typically seen in medium and high pitched voiced speech when spectral estimation is based on linear prediction.

For training, both front-ends have provided features every 10 ms. During decoding this was changed to 8 ms after the first stage. In training and decoding, the features were obtained either by the Fourier transformation followed by a Mel-filterbank or the warped MVDR spectral envelope.

For the MVDR front-end we used a model order of 22 without any filter-bank since the warped MVDR already provides the properties of the Mel-filterbank, namely warping to the Mel-frequency and smoothing. The advantage of this approach over the use of a higher model order and a linear-filterbank for dimensionality reduction is an increase in resolution in low frequency regions which cannot be attained with traditionally used Mel-filterbanks. Furthermore, with the MVDR we apply an unequal modeling of spectral peaks and valleys that improves noise robustness, due to the fact that noise is mainly present in low energy regions.

Both frond ends apply vocal tract length normalization (VTLN) [17]. For MFCC this is done in the linear domain, for MVDR in the warped frequency domain. The MFCC front-end uses 13 cepstral coefficients, the MVDR front-end uses 15. The mean and variance of the cepstral coefficients were normalized on a per-utterance basis. For both front-ends seven adjacent frames were combined into one single feature vector. The resulting feature vectors were then reduced to 42 dimensions using linear discriminant analysis (LDA).

3.2 Acoustic Model Training

We trained acoustic models for two different kinds of phoneme sets *P1* and *P2*. P1 is a version of the Pronlex phoneme set which consists of 44 phonemes and allophones while P2 is a version of the phoneme set used by the CMU dictionary that consists of 45 phonemes and allophones. We trained models for all four combinations of the two phoneme sets and the two acoustic front-ends described above.

Unlike last year, for this year's evaluation we only trained acoustic models on one set of acoustic model training data. The training contains approximately 80 h of English EPPS data provided by RWTH Aachen within the TC-STAR project [6], 9.8 h of TED data [8], and 167 h of unsupervised EPPS training material that had been collected within TC-STAR by RWTH Aachen but had not been manually transcribed. Transcriptions for the unsupervised training material were obtained by adapting an acoustic model of last year's system on automatic transcriptions provided by RWTH Aachen on that data. We then decoded the data, using the segmentation provided by RWTH Aachen. It further contained 140 h of BroadCast News data from the HUB-4 corpus, and approx. 50 h of in-domain training data provided by the Quaero consortium.

All models are semi-continuous quinphone systems that use 16000 distributions over 4000 codebooks. They were trained using incremental splitting of Gaussians training, followed by 2 iterations of Viterbi training. For all models we used one global semi-tied covariance (STC) matrix after LDA [5] as well as Vocal Tract Length Normalization. In addition to that feature space constraint MLLR (cMLLR) speaker adaptive training [3] was applied on top.

We improved the acoustic models further with the help of Maximum Mutual Information Estimation (MMIE) training [10]. We applied MMIE training firstly to the models after the 2 Viterbi iterations, and secondly to the models after the FSA-SAT training, taking the adaptation matrices from the last iteration of the maximum likelihood FSA-training and keeping them unchanged during the MMIE training.

This resulted in eight different acoustic models: for each combination of front-end, MVDR and MFCC, and phoneme set, P1 and P2, one set of models trained with VTLN plus MMIE, and one with FSA-SAT plus MMIE. From now on we refer to these models as *P1-MFCC-VTLN*, *P1-MVDR-VTLN*, *P2-MFCC-VTLN*, *P2-MVDR-VTLN*, *P1-MFCC-SAT*, *P2-MFCC-SAT*, *P1-MVDR-SAT*, *P2-MVDR-SAT*.

3.2.1 Segmentation and Clustering

Segmenting the input data into smaller, sentence-like chunks used for recognition was performed with the help of a fast decoding pass on the unsegmented input data in order to determine speech and non-speech regions. Segmentation was then done by consecutively splitting segments at the longest non-speech region that was at least 0.3 seconds long. The resulting segments had to contain at least eight speech words and had to have a minimum duration of six seconds. The maximum segment length was limited to 30 seconds.

In order to group the resulting segments into several clusters, with each cluster, in the ideal case, corresponding to one individual speaker we used the same hierarchical, agglomerative clustering technique as last year which is based on TGMM-GLR distance measurement and the Bayesian Information Criterion (BIC) stopping criterion [7]. The resulting speaker labels were used to perform acoustic model adaptation in the multipass decoding strategy described below.

3.2.2 Language Model and Test Dictionary

Using a 130k vocabulary, a 4gram case sensitive language model with modified Kneser-Ney smoothing was built for each of the text sources listed in Table 1. This was done using the SRI Language Modelling Toolkit [14]. The effects of the different text sources on the performance of the language model can be seen in Table 2[1]. The quick transcripts of the Quaro training data was cleaned and split into a 601k word training set and a 615k word tuning set. The aforementioned language models built from the text sources in Table 1 were interpolated using interpolation weights estimated on this tuning set resulting in a 50 GByte language model with 101 079k 2grams, 424 062k 3grams, and 978 979k 4grams. To select the vocabulary the de-

Table 1 English text sources

Corpus	Wordcount
UK parliamentary debate (Hansard)	49 681k
EPPS acoustic training data	750k
EPPS text data	33 044k
UN Parallel Text (English)	40 991k
Hub4 Broadcast News data	832k
Web dump (pre 2008)	643 236k
Quaero 2010 training transcripts	1 216k
Quaero 2010 training texts	1 764 227k
Web dump (February 2010)	4 764 752k
Web dump (December 2009)	1 488 881k
Gigaword 4th Edition including 2008 texts	1 800 434k
Google Ngrams	–
Total	10 588 044k

Table 2 WER (in %) on the Quaero 2010 development with varying language models. All other ASR components are the same as in our 2009 System

Language model		Case Dependant		Case Independant	
Name	Discription	pruned	not pruned	pruned	not pruned
LM1	same as Quaero2009 LM	34.31	–	31.99	–
LM3	LM1 + new tuning set	34.39	–	32.04	–
LM5	LM3 + Q2010 training transcripts	32.56	32.47	31.03	30.93
LM6	LM5 + Q2010 training texts	31.46	31.28	30.39	30.19
LM7	LM6 + Web dump (February 2010)	31.39	31.26	30.36	30.25
LM8	LM7 + Web dump (December 2010)	31.26	–	30.21	–
LM9	LM8 + Gigaword 4th Edition	31.35	31.06	30.34	30.12
LM10	LM9 + Google Ngrams	–	30.94	–	30.01

[1] A keen ovserver may notice that LM2 and LM4 do not appear in this table. Their obmission is not due to a dislike of the numbers 2 and 4 but rather the result of dead-ends in the development of the language model for our evaluation system.

velopment data text was split into a tuning set and a test set with each containing approximately half the text of every *show*. For each of our English text sources (see Table 1) we built a Witten-Bell smoothed unigram language model using the union of the text sources' vocabulary as the language models' vocabulary (global vocabulary). With the help of the maximum likelihood count estimation method described in [15] we found the best mixture weights for representing the tuning set's vocabulary as a weighted mixture of the sources' word counts thereby giving us a ranking of all the words in global vocabulary by their relevance to the tuning set. While the baseline 64k vocabulary had an OOV rate of 3.9% when measured on the validation set, the OOV rate of the vocabulary containing only the top ranked 64k words was 2.9%. This vocabulary was slowly increased until the OOV rate was under 1%. The final 130k vocabulary had a case sensitive OOV rate of 0.73%.

Pronunciations missing from the initial dictionary were created either manually or automatically with the help of Bill Fisher's tool [2] for P1 and Festival [1] for P2 respectively.

3.2.3 Decoding Strategy and Results

Decoding within our recognition system was performed in two stages. The acoustic models of the second stage were adapted on the output(s) from the previous stage using Maximum Likelihood Linear Regression (MLLR) [9], Vocal Tract Length Normalization (VTLN) [17], and feature-space constrained MLLR (cMLLR) [3]. For the second and third stage the frame shift during recognition was changed to 8 ms.

In the first stage we used the acoustic models P1-MFCC-VTLN, P1-MVDR-VTLN, P2-MFCC-VTLN, and P2-MVDR-VTLN. The resulting word lattices of P1-MFCC-VTLN and P1-MVDR-VTLN were then combined via confusion network combination to the output *o1*, those of P2-MFCC-VTLN and P2-MVDR-VTLN to *o2*. In this first stage we adapted the acoustic models using incremental VTLN and incremental fMLLR on a per speaker basis.

For the second stage P2-MFCC-SAT and P2-MVDR-SAT were adapted on o1, P1-MFCC-SAT and P1-MVDR-SAT were adapted on o2. The result of the different models were then combined via confusion network combination to the final output.

On the official 2010 development set the system achieved a word error rate of 24.0%.

4 German Evaluation Recognition System

All speech recognition experiments described in the following were performed with the help of the Janus Recognition Toolkit (JRTk) and the Ibis single pass decoder [12].

4.1 Front-End

We applied two different frontends: The WMVDR approach and the conventional MFCC approach. The front-end uses a 42-dimensional feature space with linear discriminant analysis and a global semi-tied covariance (STC) transform [4] with utterance-based cepstral mean and variance normalization. The 42-dimensional feature space is based on 20 cepstral coefficients for the MVDR system and on 13 cepstral coefficients for the MFCC system.

4.2 Acoustic Model Training

The training setup was based on last years evaluation system. We have used the following training material: Quaero development data set 2009 (13 hours), Quaero training data set 2009 (6 hours epps, 14 hours web data), Quaero training data set 2010 (51 hours), Verbmobil (67 hours), recordings of the Landtag Baden-Wuerttemberg (123 hours), Tagesschau (17 hours), isl-database (16 hours), Globalphone (19 hours), inhouse lecture and talk recordings (26 hours).

All the acoustic data is in 16 kHz, 16 bit quality. Acoustic model training was performed with fixed state alignments and VTLN factors, which were written by our last years evaluation system. The system trained uses left-right hidden Markov Models (HMM)s without state skipping with three HMM states per phoneme. Additional to last years setup with 2000 distributions and codebooks with up to 128 Gaussians per model using the MVDR frontend, we trained the same setup with the MFCC frontend and for both frontends also new systems with 4000 distributions. The adapted gender independent acoustic model training (given the vocal tract normalization values for each speaker by the previous system) can be outlined as follows:

- Training of the linear discriminant analysis matrix
- Extraction of samples
- Incremental growing of Gaussians
- Training of one global STC matrix
- Second extraction of samples
- Second incremental growing of Gaussians
- Three iterations of Viterbi training
- Three iterations of FSA-SAT speaker adaptive training.

For the 4000 distribution systems we skipped the second incremental growing of Gaussians, since we couldn't see gains from that in other systems.

4.3 Language Model Training and Evaluation

Using the same methods described in the Language Model section of the English evaluation system we selected a 300k vocabulary with which a 4gram case sensitive language model was built for each of our German text sources. Because interpolating several language models with interpolation weights estimated on only the aforementioned tuning text produced a language model that performed poorly, we added some more general text to the tuning text and reestimated the interpolation weights. This produced a language model which outerperformed the base system language model.

4.4 Decoding Strategy and Results

After a segmentation pass and speaker clustering we decoded for both frontends MVDR and MFCC both setups with 2000 and 4000 distributions using the speaker independent acoustic models. The result of a cnc combination applied on all four

Table 3 WERs on the German Quaero development set 2010

ID	pass	AM	LM	WER in % (ci/cs)
S	Segmentation	2000 MVDR	LM01	35.5 / 36.4
A	1st	4000 MVDR	LM01	30.0 / 31.2
B	1st	4000 MFCC	LM01	29.8 / 30.9
C	1st	2000 MVDR	LM01	30.8 / 31.9
D	1st	2000 MFCC	LM01	31.2 / 32.3
E	cnc A+B+C+D			28.3 / 29.4
F	2nd	4000 MVDR	LM02	26.8 / 28.0
G	2nd	4000 MFCC	LM02	27.0 / 28.0
H	2nd	2000 MVDR	LM02	27.7 / 28.8
I	2nd	2000 MFCC	LM02	27.9 / 29.0
J	cnc F+G+H+I			26.1 / 27.2

Table 4 WERs on the German Quaero evaluation set 2010

ID	pass	AM	LM	WER in % (ci/cs)
S	Segmentation	2000 MVDR	LM01	33.2 / 34.1
A	1st	4000 MVDR	LM01	28.1 / 29.3
B	1st	4000 MFCC	LM01	28.3 / 29.6
C	1st	2000 MVDR	LM01	29.2 / 30.4
D	1st	2000 MFCC	LM01	29.8 / 31.0
E	cnc A+B+C+D			26.8 / 28.0
F	2nd	4000 MVDR	LM02	25.3 / 26.5
G	2nd	4000 MFCC	LM02	25.7 / 26.8
H	2nd	2000 MVDR	LM02	26.3 / 27.5
I	2nd	2000 MFCC	LM02	26.5 / 27.7
J	cnc F+G+H+I			24.6 / 25.7
K	compound merging			24.1 / 25.2

systems was used to adapt the 2nd pass systems with incremental VTLN adaptation, constrained MLLR and MLLR. In the second pass speaker adapted FSA-SAT models and a bigger language model were used. Finally we combined the four 2nd pass systems again using cnc combination and applied compound merging.

On the official 2010 german development set the system achieved a word error rate of 26.1%. Tables 3 and 4 show the word error rates of the individual stages and combinations on the Quaero 2010 development and evaluation sets.

5 Parallelization Utilized

The XC4000 was utilized at different stages in the development and application of the ASR systems, training as well as decoding. For training the SLURM scheduler was used to start several processes in parallel that work on disjunct portions of the training data and synchronize and combine their results after every training iteration using an in-house synchronization mechanism that uses flag-files.

For decoding, the SLURM scheduler was thus used to simply start several processes in parallel that in theory can run independently of each other, as for decoding every speaker can be treated independently. The runtime of the different jobs per speaker can vary greatly, depending on how much speech is associated with one speaker. Due to the accounting system of the queue on the cluster—one process still running on one CPU on one node, while all the other processes belonging to one scheduled job have already finished, will lead to being charged the same amount of CPU time as if all processes were still running—as little processes per scheduled job as possible were sought. However, due to the fact that only 10 jobs in the production environment per user are allowed—no matter how many nodes or CPUs are actually employed—this would mean that only 40 parallel jobs could be started, if 4 CPUs per node in the cluster are assumed. In order to make use of more parallelization, therefore up to 8 or 16 processes per scheduled job were committed to the queue, even though that potentially means that one is being charged for runtime of nodes on which all processes have already finished.

5.1 Shared Memory Language Model

An important part of an ASR system is its language model which models the apriori probabilities of word sequences. Good language models generally require a large amount memory but remain unchanged after being loaded into RAM. Our 2009 English system for example required about 8 GBytes to run, almost 7 GBytes of which was used up by the language model. Because our language model was 11 GBytes big—even compressed in an easy to load binary format—a modification was made to our ASR system so that we could load the language model into a region of shared memory and allow multiple decoder instances running on different cores to access it. A standard 4 core 16 Gbyte XC4000 node was able to start 3–4 instances of our system with an 11 GByte language model compared to just 2 instances of our older system with a 7 Gybte language model or only a single instance of our new system without the shared memory language model.

5.2 One-Button System

The training and testing systems usually consisted of several sequential steps. Each step is parallelized in the xc4000 cluster using SLURM. The parallelization of a step is implemented using a master-slave system.

5.2.1 Parallelization of a Step

The master holds a list of speakers or phonemes or any data that can be divided and executed independently. In this One-Button system the master can be a single master or a multiple master (for a step which runs multiple iterations). The clients (slaves) process the actual tasks. The master communicates with the clients and give each available client what needs to be processed by that individual client (from the list that the master holds). The communication is done using socket communication, so the client would have to know the port and the hostname of the master. Everytime a client completed it's job it will notify the master and then get a new job or—if there is no more job left—wait for the others to complete their job. At the end of a step the master would wait for a minute (to make sure all jobs are really completed) and then notify the system to run the next step.

5.2.2 Detailed Description

The One-Button system manages the whole steps. It reads the defined configuration of each step and the order of the execution and then running each step in the cluster according to that configuration. It can submit in development mode or in production mode, defined in the configuration.

The One-Button system can do the following things:

- Run the whole steps from start to end – ./DO.train_system "training_id"
- Run a step only – ./DO.step "training_id" "step_name_or_order-index"
- Run from a step and continue until the last step – ./DO.from "training_id" "step_name_or_order-index"
- Cleaning the resulted outputs of a step (this is especially useful when a step is crashed and it need to be reset) – ./DO.step.clean "training_id" "step_name_or_order-index"
- Continue a step and ignore the crashed or half-finished items: Don't do any cleaning and just continue with DO.from or DO.step
- Inform the user of the current training progress (the completion progress of every step).

All outputs including the log file are located in the working directory of a setup. So to re-run the whole step from the beginning, the only thing to do is just removing the working directory and re-run the whole steps. The One-Button system also automatically backups logfiles and the parallelization list to make it possible to continue a step where it previously crashed or canceled.

5.2.3 Configuration Parameters

There are two part of configurations: the global configuration and the step configuration. The global one consists of: name (training id), working directory, jobname prefix, duration (maximum time that will be allowed for each step), and the submit mode (development or production, as explained before). Our toolkit binary used can be defined globally or locally for each step. In other words each step can use a different binary.

The configuration of each step is located in one file (the "steps" file). Each row defines the configuration of each step. The steps will be executed in the same order as it occurs in the steps-file, from the first row until the last one.

Format for each row in the steps-file: "steps_name", "slave", "master", "original_list", "memory", "janus(opt)", where:

- "steps_name": the name of the step, the script name is scripts/"steps_name".tcl
- "slave": the number of parallelized workers
- "master": the number of iteration, 1 means single iteration
- "original_list": the name of the parallelization list (speaker list, cbslist, etc)
- "memory": memory that will be provided for this step
- "janus(opt)": optional, janus binary to be used only in this step.

Other configuration files that are required by a particular step (featAccess, featDesc, traindesc, etc) are located in the desc-directory "training_id".desc/* and the parallelization list in "training_id".desc/Lists/*. Furthermore the cleaning would also require another config to list the outputs that each step produced: config. "training_id".clean.

5.2.4 Reusability

The basic idea of the One-Button system is to separate configuration files from the output files (including the log files). All configuration files needed are put in one directory, all output files are put in one other directory. This separation improves to analyze and to manage the output of different setups. Thus we can run different setups and test different parameters just by copying the configuration file and description folder, changing few parameters, and running the whole training or test system with multiple steps using just one button.

Acknowledgments. This work was realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation. The Research group '3-01 Multilingual Automatic Speech

Recognition' received financial support by the 'Concept of the Future' of Karlsruhe Institute of Technology within the framework of the German Excellence Initiative.

References

1. A.W. Black and P.A. Taylor. The festival speech synthesis system: System documentation. Technical report, Human Communication Research Centre, University of Edinburgh, Edinburgh, Scotland, United Kingdom, 1997.
2. W.M. Fisher. A statistical text-to-phone function using ngrams and rules. In *Proceedings the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Phoenix, AZ, USA, December 1999. IEEE.
3. M.J.F. Gales. Maximum likelihood linear transformations for hmm-based speech recognition. Technical report, Cambridge University, Engineering Department, May 1997.
4. M.J.F. Gales. Semi-tied covariance matrices. 1998.
5. M.J.F. Gales. Semi-tied covariance matrices for hidden Markov models. Technical report, Cambridge University, Engineering Department, February 1998.
6. C. Gollan, M. Bisani, S. Kanthak, R. Schlüter, and H. Ney. Cross domain automatic transcription on the tc-star epps corpus. In *Proceedings of the 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, Philadelphia, PA, USA, March 2005.
7. Q. Jin and T. Schultz. Speaker segmentation and clustering in meetings. In *Proceedings of the 8th International Conference on Spoken Language Processing (Interspeech 2004 – ICSLP)*, Jeju Island, Korea, October 2004. ISCA.
8. E. Leeuwis, M. Federico, and M. Cettolo. Language modeling and transcription of the TED corpus lectures. In *International Conference on Acoustics, Speech, and Signal Processing*, Hong Kong, China, March 2003.
9. C.J. Leggetter and P.C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9:171–185, 1995.
10. D. Povey and P.C. Woodland. Improved discriminative training techniques for large vocabulary continuous speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, Salt Lake City, UT, USA, May 2001.
11. S. Stüker, K. Kilgour, and J. Niehues. Quaero speech-to-text and text translation evaluation systems. In *High Performance Computing in Science and Engineering '10 – Transactions of the High Performance Computing Center, Stuttgart (HLRS) 2010*. Springer, Heidelberg, 2010.
12. H. Soltau, F. Metze, C. Fügen, and A. Waibel. A One Pass-Decoder Based on Polymorphic Linguistic Context Assignment. Trento, Italy, 2001.
13. H. Soltau, F. Metze, C. Fügen, and A. Waibel. A one pass-decoder based on polymorphic linguistic context assignment. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU '01)*, pages 214–217, Madonna di Campiglio Trento, Italy, December 2001.
14. A. Stolcke. SRILM – An extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904, Denver, CO, USA, 2002. ISCA.
15. A. Venkataraman and W. Wang. Techniques for effective vocabulary selection. Arxiv preprint cs/0306022, 2003.
16. M.C. Wölfel and J.W. McDonough. Minimum variance distortionless response spectralestimation, review and refinements. *IEEE Signal Processing Magazine*, 22(5):117–126, September 2005.
17. P. Zhan and M. Westphal. Speaker normalization based on frequency warping. In *Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, April 1997.