Search in speech, language identification and speaker recognition in Speech@FIT[1]

Jan ČERNOCKÝ, Lukáš BURGET, Petr SCHWARZ, Pavel MATĚJKA, Martin KARAFIÁT, Ondřej GLEMBEK, Jiří KOPECKÝ, Igor SZÖKE, Michal FAPŠO, František GRÉZL, Valiantsina HUBEIKA, Ilya OPARIN

[1] Speech@FIT, Department of Computer Graphics and Multimedia, Faculty of Information Technology, Brno University of Technology

speech@fit.vutbr.cz

Abstract. *This paper describes "search in speech" techniques developed in the Speech@FIT research group at FIT BUT in the last couple of years. It concentrates on spoken term detection (STD) and presents our system for NIST STD 2006 evaluations in detail. It also briefly mentions our systems for speaker and language recognition.*

Keywords

Speech recognition, speaker recognition, language identification, spoken term detection, Speech@FIT.

1. Introduction

Speech is the most important modality in human to human communication. Even in face-to-face communication, it might contain more than 80% of the information and in case of a telephone conversation, this proportion goes up to 100%. Speech is omni-present in many electronic media (land-line and mobile networks, IP telephony, dedicated channels, ...).

In many applications (eLearning, meeting recognition, audio archive search), the problem is usually not to **obtain** the speech but to **efficiently process** thousands of hours of speech records. Typically, speech is processed by human experts, but this processing capacity is always limited: by lack of qualified personnel, lack of budget, insufficient knowledge of foreign languages, insufficient security clearances (for security applications), or combination of the above.

Automatic speech search techniques can help to find the requested information – the "needle in a haystack" – in reasonable time and without extensive human labor. However, they should not be considered almighty – they *are not able* to replace qualified personnel that will always have to do the final analysis and decisions. They *are able* to automate some processing steps and "limit the search space" for humans.

Let us first define the categories of speech search:

- The task of **Large Vocabulary Continuous speech recognition (LVCSR)** is to determine "what was said" or to provide textual transcription of speech. The best performing LVCSR systems are based on complex acoustic models trained on thousands hours of transcribed speech and on language models trained on gigabytes of text. The biggest challenge in LVCSR is processing spontaneous data for which developers lack speech and language resources.
- In some situations, LVCSR is not the best suited to find the information in speech — as it is constrained by recognition vocabulary, it is hard to find rare information such as new proper names (companies, politicians, geographical). In this case, it is necessary to use **phonetic search** that operates not on the output of word recognizer, but rather on oriented graphs containing smaller units – phonemes.
- As important as the contents of speech is the information "who said it" – this is addressed by **speaker recognition**. We speak about **speaker identification** if the task is to choose one out of a set of N speakers, or about **speaker verification**, where it s necessary to confirm the claimed identity of a speaker.
- Last but not least, **language identification** is needed in speech search to be used as search criterion itself or for routing speech segments to appropriate language-dependent recognizer.

This paper deals mainly with the first two mentioned topics — LVCSR and phonetic search, jointly called **spoken term detection**, mainly in light of first edition of NIST Spoken term detection (STD) evaluations [3]. At the end, we will also briefly mention our work in speaker recognition and language identification, although we will mainly refer the reader to our respective summary publications.

The paper is organized as follows: section 2 defines the techniques of spoken term detection, section 3 presents the

NIST STD 2006 evaluations, including our system and results. Sections 4 and 5 deal respectively with speaker recognition and language identification and 6 concludes the paper.

2. Spoken term detection techniques

Unlike search in text, where the indexing and search is the only "science", spoken term detection is a more complex process that needs to address the following points:

- conversion of speech to discrete symbols that can be indexed and searched – LVCSR and phoneme recognizers are used. Using phoneme recognizer allows to deal with out-of-vocabulary words (OOVs) that can not be handled by LVCSR.
- accounting for inherent errors of LVCSR and phoneme recognizer – this is usually solved by storing and searching in word or phoneme lattices (Fig. 1) instead of 1-best output.
- determining the confidence of a query – in this work done by evaluating the likelihood ratio between the path with searched keyword(s) and the optimal path in the lattice.
- processing multi-word queries, both quoted (exact sequences of words) and unquoted.
- providing an efficient and fast mechanism to obtain the search results in reasonable time even for huge amounts of data.

2.1 LVCSR-based search

LVCSR lattices (upper panel in Fig. 1) contain nodes carrying word labels and arcs, determining the timing and acoustic (L_a^{lvcsr}) and language model (L_l^{lvcsr}) likelihoods generated by an LVCSR decoder. Usually, each speech record is first broken into segments (by speaker turn or voice activity detector) and each segment is represented by one lattice. The confidence of a keyword KW is given by

$$C^{lvcsr}(KW) = \frac{L_\alpha^{lvcsr}(KW) L^{lvcsr}(KW) L_\beta^{lvcsr}(KW)}{L_{best}^{lvcsr}}, \quad (1)$$

where the $L^{lvcsr}(KW) = L_a^{lvcsr}(KW) L_l^{lvcsr}(KW)$.

The forward likelihood $L_\alpha^{lvcsr}(KW)$ is the likelihood of the best path through lattice from the beginning of lattice to the keyword and the backward likelihood $L_\beta^{lvcsr}(KW)$ is the likelihood of the best path from the keyword to the end of lattice. For node N, these two likelihoods are computed by the standard Viterbi formulae:

$$L_\alpha^{lvcsr}(N) = L_a^{lvcsr}(N) L_l^{lvcsr}(N) \max_{N_P} L_\alpha^{lvcsr}(N_P) \quad (2)$$

$$L_\beta^{lvcsr}(N) = L_a^{lvcsr}(N) L_l^{lvcsr}(N) \max_{N_F} L_\beta^{lvcsr}(N_F) \quad (3)$$

where N_F is a set of nodes directly following node N (nodes N and N_F are connected by an arc) and N_P is a set of nodes directly preceding node N. The algorithm is initialized by setting $L_\alpha^{lvcsr}(first) = 1$ and $L_\beta^{lvcsr}(last) = 1$. The last likelihood we need in Eq. 1: $L_{best}^{lvcsr} = L_\alpha^{lvcsr} = L_\beta^{lvcsr}$ is the likelihood of the most probable path through the lattice.

2.2 Phonetic search

The main problem of LVCSR is the dependence on recognition vocabulary. The phonetic approach overcomes this problem by conversion of query to a string of phonemes and searching this string in a phoneme lattice (lower panel in Fig. 1). The lattice has similar structure as word lattice (section 2.1), but phonemes P populate nodes instead of words.

The confidence of keyword KW consisting of string of phonemes $P_b \ldots P_e$ is defined similarly as in Eq. 1 by:

$$C^{phn}(KW) = \frac{L_\alpha^{phn}(P_b) L_\beta^{phn}(P_e) \prod_{P \in P_b \ldots P_e} L_a(P)}{L_{best}^{phn}}, \quad (4)$$

where $L_\alpha^{phn}(P_b)$ is the forward Viterbi likelihood from the beginning of lattice to phoneme P_b, the product is the likelihood of the keyword, and $L_\beta^{phn}(P_e)$ is the likelihood from the last phoneme till the end of the lattice. L_{best} is the likelihood of the optimal path.

2.3 LVCSR lattice indexing

The **indexing** of LVCSR lattices is inspired by [4]. It begins with the creation of lexicon which provides a transformation from word to a unique number (ID) and vice versa. Then, a forward index is created storing each hypothesis (the word, its confidence, time and nodeID in the lattice file) in a hit list. From this index, an inverted index is created (like in text search) which has the same structure as the forward index, but is sorted by words and by confidence of hypotheses. Each speech record is represented by many lattices. The inverted index tells us, in which lattice and at which time the keyword appears.

In the **search** phase, the inverted index is used to find occurrences of words from query. An important feature of our system is the generation of the most probable **context** of the found keyword – a piece of the Viterbi path from the found keyword forward and backward. For all matching occurrences, the searcher therefore loads into the memory a small part of lattice within which the found word occurs. Then, the searcher traverses this part of lattice in forward and backward directions selecting only the best hypotheses; in this way it generates the most probable string which traverses the found word.

2.4 Multi-word queries

A usable system for STD should support queries of type

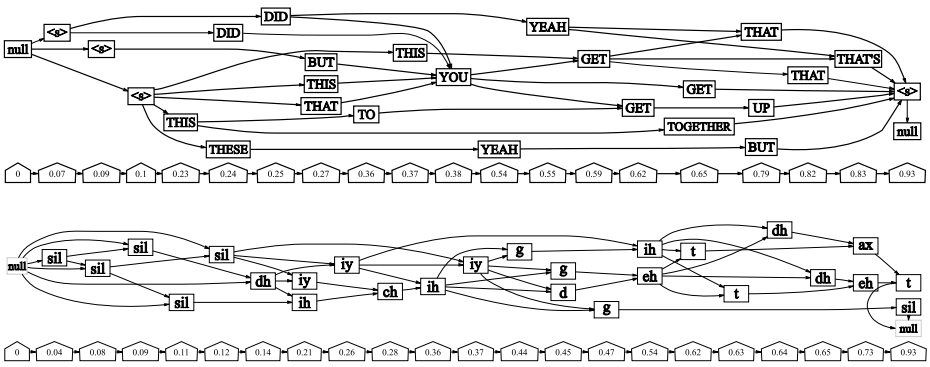

Fig. 1: Example of a word and phoneme lattices

`word1 word2 word3` and `"word1 word2 word3"` with the former one representing finding words in random order with optional spaces in between (in opposite to text-search where we work within a document, we specify a time-context) and the later one representing the exact match. Provided the query Q is found in the lattice, we again need to evaluate its confidence $C^{lvcsr}(Q)$. Similarly to Eq. 1, this is done by evaluating the likelihood of the path with all the words w_i belonging to the query and dividing it by the likelihood of the optimal path:

$$C^{lvcsr}(Q) = \frac{L^{lvcsr}_{rest} \prod_i L^{lvcsr}(w_i)}{L^{lvcsr}_{best}}, \quad (5)$$

where L^{lvcsr}_{rest} is the likelihood of the "Viterbi glue": optimal path from the beginning of the lattice to $w_{earliest}$, connections between words w_i (for unquoted query) and optimal path from w_{latest} to the end of the lattice. In other words, L^{lvcsr}_{rest} represents everything except the searched words. We should note, that each time we deviate the Viterbi path from the best one, we loose some likelihood, so that $C^{lvcsr}(Q)$ is upper-bounded by $\min_i C^{lvcsr}(w_i)$ — actually the confidence of the worst word in the query.

The same index as for single-word queries (keywords) is used. Processing of a query involves the following steps:

1. Based on frequencies of words, the least frequent one from the query, w_{lf}, is taken as first and all its occurrences are retrieved.
2. The search proceeds with other words and verifies if they are within the specified time interval from w_{lf} (for non-quoted queries) or joint to w_{lf} (for quoted ones). The internal memory representation resembles again a lattice. In such way, a candidate list is created.
3. The list is pre-sorted by the upper-bound of query confidence, as described above. The list is then limited to the pre-determined number of candidates (usually 10).
4. For these candidates, the evaluation of correct confidence is done according to Eq. 5. While looking for the "Viterbi glue", the Viterbi algorithm is extended before and after the part of lattice containing Q in order to obtain the left and right contexts.

2.5 Indexing phoneme lattices

While the indexing of word lattices is straightforward, indexing phoneme lattices is more tricky: in advance, we do not know what we will search for. Yu and Seide in [6] and Siohan and Bacchiani in [7] have chosen indexing sequences of phonemes with variable length, we have however investigated a simpler approach making use of overlapping tri-phonemes and indexing similar to multi-word queries. The use of tri-phonemes was also recommended in [5] as the best balance between number of units and number of units' occurrences in a corpus.

In the indexing phase, tri-phonemes T_i are selected in lattices. For each T_i, its confidence is evaluated by Eq. 4 as if T_i was a keyword. In case this confidence is higher than a pre-determined threshold, the tri-phoneme is inserted into the index. The search stage is similar to multi-word quoted queries:

1. The searched keyword generates a set of overlapping tri-phonemes. Based on their frequencies in the index, the least frequent one T_{lf}, is taken as first and all its occurrences are retrieved.
2. The search proceeds with other tri-phonemes and verifies that they form a chain in time (with a security margin between adjacent tri-phonemes). Similarly to multi-word queries, the internal memory representation has again the form of lattice. In such way, a candidate list is created.
3. The confidence of keyword is again upper-bounded by the confidence of the worst tri-phoneme. Based on these, the list is pre-sorted and limited to the pre-determined number of candidates (usually 10).
4. For these candidates, we go into the respective phoneme lattices and evaluate the correct confidence using Eq. 4.

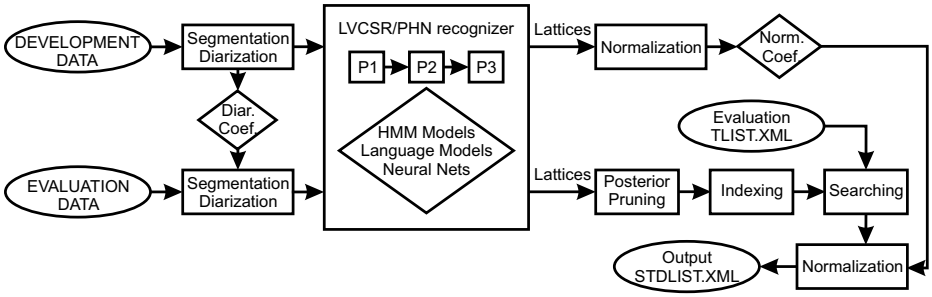

Fig. 2: BUT system for spoken term detection.

We have verified, that in case no thresholds are applied in the index, we obtain exactly the same accuracy of search that in case phoneme lattices are processed directly.

3. NIST STD evaluations 2006

The first edition of Spoken term detection evaluation was organized to facilitate research and development of technology for finding short word sequences rapidly and accurately in large heterogeneous audio archives [3]. In this paper, we will deal only with US English tasks.

3.1 Data and metrics

There were three kinds of data with the following amounts available for both the development and evaluation:

- broadcast news (BCN) – 2.2 hours,
- conversational telephone speech (CTS) – 3 hours
- meeting speech (MTG) recorded over multiple distant microphones (MDM) – 2 hours.

For all sets, NIST has defined 1100 search-terms[1] having 1,2,3 and 4 words:

- 42 of them do not appear in any of BCN, CTS and MTG data
- 898 of 1100 appear in BCN with ≈4900 occurrences
- 411 of 1100 appear in CTS with ≈5900 occurrences and
- 241 of 1100 appear in MTG with ≈3700 occurrences
- 160 of 1100 appear in all three BCN, CTS and MTG.

Examples of terms are: “dr. carol lippa”, “bush’s father george bush”, “thousand kurdish”, “senator charles”, “nato chief”, “every evening”, “kostunica”, “audio”, “okay”.

The main mean for comparison of different systems were detection error trade-off (DET) curves, displaying, for various thresholds θ, the false alarm probability $P_{FA}(\theta)$ on x-axis and miss probability $P_{MISS}(\theta)$ on the y-axis:

$$P_{MISS}(\theta) = \underset{term}{avg}\{1 - N_{correct}(term,\theta)/N_{true}(term)\}$$
$$P_{FA}(\theta) = \underset{term}{avg}\{N_{spurious}(term,\theta)/N_{NT}(term)\}$$

where $N_{correct}(term,\theta)$ is the number of correct detections of $term$ with a score greater or equal to θ, $N_{spurious}(term,\theta)$ is the number of spurious (incorrect) detections of $term$ with a score greater or equal to θ, $N_{true}(term)$ is the number of occurrences of $term$ in corpus and $N_{NT}(term)$ is the number of opportunities for incorrect detection of $term$ which is equal to length of the corpus in seconds minus $N_{true}(term)$.

NIST defined so called $TWV(\theta)$ (Term-Weighted Value) metric to “score” a system by one number. Term weighted value is computed by first computing the miss and false alarm probabilities for each term separately, then using these and a pre-determined prior probability to compute term-specific values, and finally averaging these term-specific values over all terms to produce an overall system value:

$$TWV(\theta) = 1 - \underset{term}{avg}\{P_{MISS}(term,\theta) + 999.9\, P_{FA}(term,\theta)\}$$

Threshold θ_M is found on development data by maximization of $TWV(\theta)$. $TWV(\theta_M)$ is then computed on evaluation data with θ_M threshold and denoted as $ATWV$ (Actual Term-Weighted Value, see evaluation plan [3] for further details).

3.2 BUT system

Our system was based on combination of LVCSR and phonetic-based search and its overall structure is shown in Figure 2. The detailed description of **recognizers** used in the system is beyond the scope of this paper and can be found in our system description for NIST [8] and in [9].

The **indexing and search** followed closely the theoretical description given above. In the *indexing phase*, word lattices were first converted to forward index (word unigrams). Forward index was than sorted to inverted index. Lattice were converted to binary format. The same processing was applied to phoneme lattices.

[1] “quoted” queries where “quoted” refers to Google and similar search engines and means that no other word(s) can appear inside the query.

task	EVAL ATWV Merged	EVAL MTWV Merged	EVAL MTWV LVCSR	DEVEL MTWV Merged
BCN	0.6541	0.6558	0.6305	0.7020
CTS	0.5235	0.5344	0.5301	0.5580
MTG	0.0549	0.0731	0.0695	0.2950

Tab. 1: Minimum (M) TWV and actual (A) ATWV values for individual and merged systems.

While *searching* a query, in-vocabulary (IV) tokens are searched in inverted index to estimate their position in lattices and than they are verified in the lattice.

Out-of-vocabulary (OOV) tokens are converted to phoneme strings. Automatic grapheme-to-phoneme (G2P) tool based on rules (derived from AMI recognition vocabulary and BEEP dictionary) is used for the conversion. Then the phoneme string is split into a train of overlapped triphonemes. Then they are also searched in inverted index (phoneme) and verified in lattice (phoneme). OOVs shorter than 3 phonemes are not searched and are dropped.

If all tokens were successfully verified, the time and score is produced. The score is computed as sum of IV (LVCSR) part and OOV (PHN) part. IV scores are computed by Viterbi approximation using likelihood ratio in word lattice and then normalized. OOV scores are computed by Viterbi approximation using likelihood ratio in phoneme lattice and then normalized.

3.3 Results

The results of LVCSR systems for different tasks in terms of word error rate (WER) evaluated on the development sets, are the following: BCN 21.03%, CTS 22.83% and MTG 46.65%. The oracle results obtained by scoring the path in lattice that matches the best the reference, are respectively: BCN 9.06%, CTS 8.32% and MTG 21.79%. It is obvious that while BCN and CTS results are good and comparable to the state-of-the-art, the recognition on meetings is worse. This is due to the MDM condition, for which all the systems in NIST RT'06 evaluation performed quite poorly.

The STD results on all three conditions in terms of DET curves on development data can be seen in Fig. 3 and the results in terms of TWV are are summarized in Table 1. First, we can see that the results on meetings are even worse than for the development data suggesting a problem with the data. Unfortunately, we are not able to analyze this in detail, as NIST does not intend to provide word transcriptions for the evaluation data. In the other tasks, the results were satisfactory and we have seen the actual TWV not differing substantially from minimum TWV – a sign of good estimation of the optimal threshold.

Except for BCN, we see minimum effect of merging phonetic search with LVCSR, this is however caused by the term-lists provided – in CTS data, we have counted only 6 OOVs out of all 1100 requested terms.

4. Speaker recognition

Speaker recognition and verification is another important activity in Speech@FIT. In 2006, we have formed the "STBU" consortium - BUT in cooperation with TNO Human factors (the Netherlands), Spescom Data Voice and Stellenbosch University (both South Africa). The system we produced included a combination of 3 acoustic classification techniques: (1) Gaussian mixture models (GMM) classifying directly speech features, (2) Support vector machines (SVM) processing super-vectors of GMM means and (3) SVM-classification of MLLR adaptation matrices from LVCSR system.

Great care was given to transmission channel compensation and score normalization: the system includes techniques such as feature mapping, eigen-channel adaptation and nuisance attribute projection (NAP). The scores are normalized by classical t-norm technique and fusing of systems was performed using logistic linear regression.

The STBU system as well as BUT's independent submission recorded important success in NIST SRE evaluations organized in spring 2006. We have scored among the best (NIST rules prohibit us disclose the exact position of our system) of almost 40 academic and industrial laboratories from all over the world. The details on our systems can be found in [2].

5. Language identification

The task of LID (also called language recognition - LRE) is to detect the language a particular speech segment was spoken. Speech@FIT works also in this direction, its system uses a combinations of two techniques:

Acoustic LID determines the language directly on the basis of features derived from the speech signal. This approach can for example well separate between French and English - in the former, the nasal cavity is more frequently open which is directly translated into speech features. Speech@FIT improved the existing technologies by adding discriminative training of acoustic models.

In **Phonotactic** LID, speech is first transcribed by phoneme recognizer into strings or graphs (lattices) of phonemes. On these, "language" models are trained to capture statistics of couples and triples of phonemes. In this way, German and English can be for example separated based on different statistics of "und" and "and". Speech@FIT group pioneered the use of so called "anti-models" for this task.

In NIST LRE-2005 evaluations, we have scored the

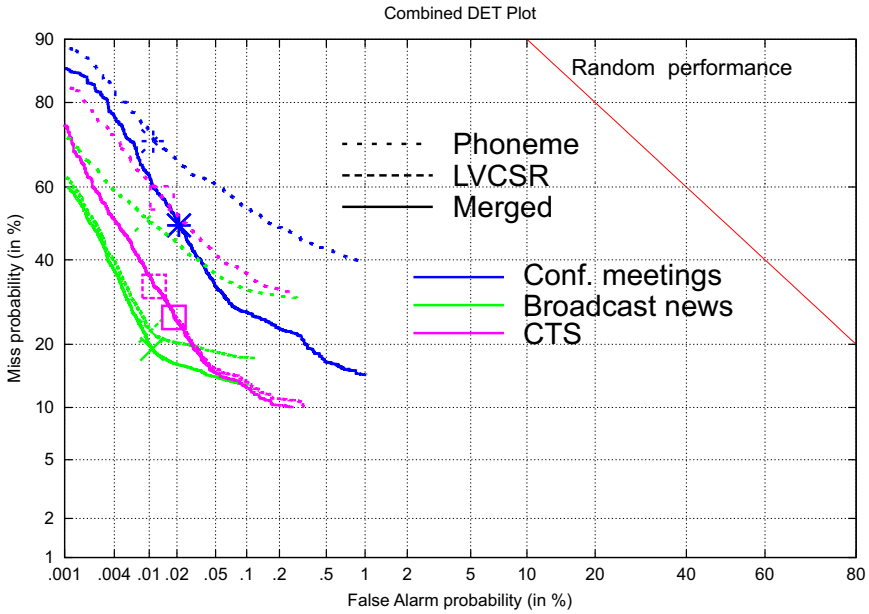

Fig. 3: DET curves for development US English data: LVCSR, phoneme-based and merged systems

second in the primary condition (30 second speech segments) and the best in two secondary conditions (10s and 3s) in competition of 12 academic and industrial laboratories from all over the world. The full description of our system can be found in [1].

6. Conclusions

Ten years after its founding, Speech@FIT group at FIT BUT has become one of the leading European labs dealing with speech processing. Experienced with keyword spotting and speech, speaker and language recognition, the group recently succeeded in several evaluations organized by U.S. National Institute of Standards and Technology. We are however far from exaggerated self-satisfaction. Good results usually bring more and more open questions and we feel that in some areas, we are still at the beginning of serious research. But at least, we can say we have good baselines for our future work.

7. Acknowledgments

This work was partly supported by European projects AMIDA (IST-033812) and Caretaker (FP6-027231), by Grant Agency of Czech Republic under project No. 102/05/0278 and by Czech Ministry of Education under projects No. MSM0021630528 and LC 06008. The hardware used in this work was partially provided by CESNET under projects No. 119/2004, No. 162/2005 and No. 201/2006. Lukáš Burget was supported by Grant Agency of Czech Republic under project No. GP102/06/383. We are grateful to David van Leeuwen (TNO, the Netherlands) for the diarization of NIST STD data, to Vinny Wan (University of Edinburgh) for language model training and to IDIAP speech guys for beam-forming. In speaker recognition, we are grateful to Niko Brummer (SDV), Albert Strasheim (Stellenbosch) and David van Leeuwen for all the help and excellent cooperation.

References

[1] P. Matějka, L. Burget, P. Schwarz, J. Černocký: Brno University of Technology System for NIST 2005 Language Recognition Evaluation, In: *Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop*, San Juan, Puerto-Rico, 2006, pp. 57-64.

[2] P. Matějka, L. Burget, P. Schwarz, O. Glembek, M. Karafiát, F. Grézl, J. Černocký, D. A. van Leeuwen, N. Brümmer and A. Strasheim: STBU System for the NIST 2006 Speaker recognition evaluation, accepted to ICASSP 2007, Hawaii, 2007.

[3] NIST Spoken Term Detection Evaluation, http://www.nist.gov/speech/tests/std/

[4] Sergey Brin, Lawrence Page: *The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Science Department*, Stanford University, 1998.

[5] K. Ng: *Subword-Based Approaches for Spoken Document Retrieval*, PhD thesis, MIT, USA, February 2000.

[6] P. Yu and F. Seide: Fast two-stage vocabulary independent search in spontaneous speech, in *Proc. ICASSP 2005*, Philadelphia, 2005.

[7] O. Siohan and M. Bacchiani: Fast vocabulary-independent audio search using path-based graph indexing, in *Proc. Eurospeech 2005*, Lisboa, Portugal, 2005.

[8] I. Szöke, M. Fapšo, M, Karafiát, L. Burget, F. Grézl, P. Schwarz, O. Glembek, P. Matějka, S. Kontár and J. Černocký Jan: BUT System for NIST STD 2006 - English, available from http://www.fit.vutbr.cz/speech/std/2006/, file but_06_std_eval06_eng_all_spch_p-BUT-STBU-MERGED_1.txt

[9] I. Szöke, M. Fapšo, M. Karafiát, L. Burget, F. Grézl, P. Schwarz, O. Glembek, P. Matějka, J. Kopecký and J. Černocký: Spoken term detection system based on combination of LVCSR and phonetic search, submitted to 4th Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms (MLMI), June 2007, Brno, Czech Republic.