EFFICIENT SUBWORD LATTICE RETRIEVAL FOR GERMAN SPOKEN TERM DETECTION

Timo Mertens[1,2] *and Daniel Schneider*[2]

1. Department of Electronics and Telecommunications, NTNU, Trondheim, Norway
timo.mertens@iet.ntnu.no
2. Fraunhofer IAIS, Schloss Birlinghoven, 53754 Sankt Augustin, Germany
daniel.schneider@iais.fraunhofer.de

ABSTRACT

We present a lattice-based STD method for German broadcast news data and compare it to a previously proposed fuzzy search. Due to the important out-of-vocabulary (OOV) problem in German, we evaluate suitable subword indexing units for lattice retrieval. Hybrid lattice retrieval of words and subwords is investigated because of the robust nature of words as an indexing unit. We show that by using efficient lattice graph and score pruning techniques, precision of subword retrieval is increased by 8% absolute with only a small loss in recall. Additionally, a speed-up of up to 6 times can be observed.

Index Terms— spoken term detection, spoken document retrieval, speech recognition, speech search

1. INTRODUCTION

Due to increasing computation power and storage capabilities it has become feasible to archive large amounts of digital media. For instance, the archive of the French Institut National de l'Audiovisuel (INA) [1] contains more than 3 million hours of digitized video and audio programs. An interesting task for end-user archivists is to find the exact occurrences of a spoken term or phrase in the collection, the so-called Spoken Term Detection (STD).

Systems designed for STD typically use large vocabulary continuous speech recognition (LVCSR), which requires a pre-defined lexicon containing all terms that can be recognized. If the user searches for a term that is not in the lexicon, the spoken occurrence of that term cannot be found. The so-called out-of-vocabulary (OOV) problem is particularly significant in languages with a rich morphology and active compounding such as German. In [1] it was observed that the OOV rate in German broadcast news data is at 6.1% with a recognition vocabulary size of 60k words. In comparison, they found that English has an OOV rate of about 1% at the same vocabulary size. This implies that the OOV problem has more impact on German STD as a standard 60k recognition vocabulary is not able to cover the infinite compounding and morphological variants which are not present in English. To alleviate this problem, speech recognition based on subwords instead of words has been applied previously [2, 3, 4]. Subword units are compositional and have a fixed inventory size which implies that the set of possible tokens is finite and known *a priori* (in our German test data we found 49 phonemes and 10k distinct syllables).

An important aspect of STD which has not received much attention is the efficiency of the vocabulary independent approach. As subword approaches typically produce higher error rates, complex retrieval algorithms are required to cope with incorrect recognition hypotheses. This means that relatively fast text retrieval techniques cannot be used on erroneous transcripts. In this paper we compare two methods for vocabulary independent STD on German speech: a fuzzy syllable search algorithm which has already been successfully applied on German data [5] and a new alternative based on a hybrid lattice approach. As no work has been done on subword German lattice STD before, we investigate different indexing units for lattice retrieval and their effect on accuracy and retrieval time. Furthermore, different techniques during indexing of and retrieval from lattices are evaluated.

This paper is organized as follows. Section 2 introduces subword retrieval techniques followed by Section 3 which presents our lattice-based retrieval system. We describe the evaluation of the system in Sections 4 and 5. Finally, we conclude this work in Section 6.

2. SUBWORD RETRIEVAL

In many state of the art systems for STD, words are used as baseline due to their recognition robustness. In order to alleviate the OOV problem imposed by the fixed word vocabulay, different subword units have been studied as an alternative. For English, phones are a common indexing unit [2, 4, 6]. German, unlike English, makes use of active compounding and has a rich inflectional morphology. In [3] it was shown for Turkish, which shares similarities in its morphological nature with German, that phones do not contribute significantly to the performance of their system. Syllables were proposed for German subword retrieval in [7]. They argue that syllables present a good tradeoff between robustness in terms of acoustic context and inventory size.

The fuzzy syllable search algorithm proposed in [8] is based on a similarity measure between the syllable representation of the query and the possibly erroneous ASR syllable transcript. To find the occurrence of a query in the collection, each position of each document transcription is hypothesized as a possible starting point of the syllable sequence given by the query. The actual distance between the syllable query and the current position is estimated by the edit distance between the syllable sequence of the query and the transcript at the given position. The penalty for substituting two syllables is in turn estimated with an edit distance between the phoneme sequences of the two syllables. A confusion matrix, estimated from held-out data, is used to weight phoneme substitutions. Positions in the transcript with a distance below a certain threshold are considered as a hit. The advantage of this approach is robustness against

[1] http://www.ina.fr

recognition errors by simulating a wider search space, i.e. allowing for likely token confusions. On the other side, using a confusion matrix is an *a priori* calculation of costs, which means that strong claims about possible confusions are made. By setting the similarity threshold relatively tight, high precision values can be achieved but at the cost of low recall. If the threshold is lowered, however, the results become rather imprecise due to the fuzzy and thus inaccurate nature of the algorithm. Furthermore, the complexity of the fuzzy search algorithm grows linearly with respect to collection size *and* number of syllables in the query. Another widely used approach for coping with recognition inaccuracy is keyword search on word and subword lattices [3, 6, 9]. Compared to fuzzy search with its inherent high complexity, lattice search is carried out on a compact representation of the search space. Furthermore, elaborate indexing techniques exist for efficient retrieval [10]. Effective graph pruning (GP) is essential for syllable and phone lattices, as subword graphs tend to contain a large number of nodes, resulting in high storage requirements and intolerable response times.

In this paper we focus on the indexing and retrieval aspects of a lattice retrieval system for German in order to (i) gain understanding on how phone, syllable and word lattice retrieval behave on German data with varying configurations and (ii) study the difference between fuzzy search and lattice retrieval.

3. SYSTEM DESCRIPTION

A research prototype was developed in order to evaluate word and subword lattice retrieval on German data. The open source LVCSR engine Julius [11] was used to create word and subword lattices. Each lattice is stored in a global inverted index, i.e. each node is stored together with all relevant information, including document and node ID as well as the confidence score of the node.

The retrieval component takes a given query and splits the string into subword tokens using grapheme-to-phoneme conversion. We found that syllables follow a Zipfian distribution in German. Hence, for efficiency reasons, we use the query syllable with the lowest prior probabilty to obtain a truncated set of lattices from the index in which the query could be present. The prior probability of each syllable was estimated on a newswire corpus of 80M words described in [5]. A depth-first search algorithm enters each returned lattice at the first query syllable and retrieves all paths containing the query.

For each found path, a query confidence score can be calculated. If the query score falls below a certain threshold, it is classified as a false positive and can thus be rejected. Using score pruning, the system can be optimized towards either precision or recall. Traditionally, query confidence scores can be computed as follows [2]:

$$\mathcal{S}_u(Q) \approx \frac{L_\alpha(q_f) L_\beta(q_l) \prod_{q \in q_f \ldots q_l} L_{AM}(q) L_{LM}(q)}{L_{best}} \qquad (1)$$

where $\mathcal{S}_u(Q)$ is the confidence score for a query Q and a given indexing unit u. L_α and L_β correspond to the forward / backward likelihoods leading in and out of the query path. L_{AM} and L_{LM} refer to the acoustic and language model likelihoods for a query token q. L_{best} is the likelihood of the Viterbi path through the lattice. q_f and q_l are the first and last nodes of the query path in the lattice. Instead of calculating the query score during runtime, we calculate token confidence scores for each node during indexing by restricting (1) to one token and not a sequence. We then use (2) on search time to create a query score.

$$\mathcal{S}_{unit}(Q) = \prod_{q \in Q} \mathcal{S}_{unit}(q) \qquad (2)$$

This, especially on large collections, should be more efficient than storing and accessing AM/LM and forward/backward scores during retrieval. If retrieval is performed on more than one indexing unit, results referring to the same utterance need to be merged. Different ways of combining word and subword results have been proposed [6]. We evaluate two hybrid search methods: combined search, which retrieves from the word and subword indexes simultaneously and OOV search which retrieves from the subword index only if the query contains an OOV term. The first search should obtain the best coverage but could result in low precision due to the less robust nature of subwords. On the other hand, OOV search should be more efficient than combined search as only one index is accessed per search request

4. EXPERIMENTAL METHODOLOGY

4.1. Data

The data used in this work consists of German broadcast radio programmes. The dataset is the same as in [5] and consists of 3.5 hours of manually segmented speech data. The material was extracted from broadcast shows with differing degrees of difficulty for the ASR. *Deutsche Welle Funkjournal* is a studio-quality planned speech broadcast contributing 45 minutes to the collection. *WDR Der Tag* consists of reports and opinions and adds 60 minutes. Finally, 105 minutes of *MonTalk*, an interview talk show, make up the rest of the collection. Hence, half of the corpus contains planned speech while the other half is governed by spontaneous conversational speech. This dataset is quite diverse in its recognition difficulty: broadcast news material is usually easy to recognize whereas broadcast conversational speech as in *MonTalk* presents a more difficult task.

The query set used for evaluation consists of 213 single or multiword phrases, with a total of 321 words, selected by professional archivists from German broadcasters. The longest query in the set is *Referenzkurse der europäischen Zentralbank Frankfurt* (reference exchange rates of the European central bank Frankfurt). The example shows the common phenomenon of active compounding in German. If one were to search for *Referenz*, a hit in *Referenzkurs* would be returned. This is commonly regarded as a false positive by the evaluation measure although most users of such a system would not classify the result as incorrect per se. In the whole 3.5 hours collection there are 549 query occurrences out of which 51 contained an OOV term.

4.2. Evaluation Setup

Julius was used to transcribe the data set and to produce the lattices for all experiments. The acoustic models were trained on 32 hours of training data and result in a model size of 12932 triphones. Trigram phoneme, syllable and word language models were trained from 80 million words. Using a 10k syllable and a 65k word dictionary, the 1-best recognition error rates of the evaluation data are at 41.6% for words, 31.2% for syllables and 21.8% for phonemes. The efficiency experiments were carried out on a Desktop Linux system using a 2.66GHz Intel CPU.

We use the standard metrics *Precision*, *Recall* and the *F-score* to evaluate our system.

5. RESULTS

We presented different aspects of fuzzy and lattice retrieval that are of interest when developing a STD approach for German. First, we compare the results of lattice and fuzzy retrieval on syllables. Next, we investigate whether phones present an alternative subword unit for German. The results of graph and score pruning are presented, followed by the best configurations for word, syllable and hybrid lattice retrieval which are compared to the best fuzzy syllable / exact word search results.

5.1. Subword Retrieval

[5] found that the distance threshold for fuzzy syllable search that maximizes the F-score is found at 0.85. Table 1 shows the accuracy results at this value. On 3.5 hours of data, the algorithm takes 46.51 seconds to process all 213 queries. The table shows also the performance of pure syllable lattice search with previously applied GP. In terms of true (TP) and false positives (FP), fuzzy search finds 383 TP and 70 FP out of 549 TP. Lattice search finds 15 TP less but at the same time accepts 22 fewer FP as well. We found that both search approaches cover a different part of the search space: 29 TP from fuzzy search could not be obtained with lattice search whereas 14 TP were only found in lattices. A higher recall for fuzzy search is due to the fact that fuzzy retrieval accepts tokens which are unlikely lattice hypotheses, as the fuzzy syllable distance is not influenced by acoustic or language model likelihoods. The drawback of reaching a high recall with fuzzy search is a low level of precision, which is not the case for lattice retrieval.

When using phones as subword indexing unit for lattices, precision and recall both drop considerably. Furthermore, the runtime rises to 5.41 hours for all 213 queries. The relatively low accuracy is mainly due to unintentionally dismissing correct hypotheses during lattice pruning. The main problem of this indexing unit is that the size of the indexed graphs explodes. Because phones have little discrimination power, too many competing paths are proposed per utterance, which leads to an unmanageable amount of nodes to store and to search. Even when retrieving only the least frequent query token, too many occurrences of this anchor are returned, resulting in too many applications of the search algorithm. Thus, phones do not present a feasible alternative to syllables as an indexing unit due to the severe efficiency drawback.

Table 1. Results of subword retrieval.

Approach	Precision	Recall	F-score	Runtime
Fuzzy syll	85%	69%	76%	46.51 *sec*
Latt syll	89%	67%	77%	8.25 *sec*
Latt phone	78%	52%	62%	5.41*hours*

5.2. Lattice Pruning

Two different pruning methods are evaluated in this subsection. Graph pruning of the search space is applied offline and determines the number of allowed hypotheses per graph. The document collection was converted into lattice format multiple times with varying graph pruning intensity. A low parameter allows only for few competing hypotheses to be indexed and vice versa. We investigate the effect of graph pruning on both precision and recall as well as the number of nodes that are indexed for the whole collection which indicates the runtime to process all queries. Figure 1 shows the results. It can be seen that when allowing more hypotheses to be indexed, recall increases rapidly. At the same time precision decreases but not at the same rate as recall increases. Recall levels off at $GP = 5$ whereas precision decreases up to $GP = 30$. Furthermore, the collection-wide number of nodes increases up to the same parameter value, after which it levels off. A rise in nodes implies that the retrieval time should increase as well since more indexed tokens lead to more anchor tokens returned from the inverted index. A statistically significant relationship between the number of nodes and the runtime is observed for words and syllables: $r(10) = .96$, $p \leq .01$ (words), $r(10) = .98$, $p \leq .01$ (syllables). This shows that indexing paths after $GP = 5$ only decreases precision (more FP are added) and increases runtime significantly. For all experiments, this pruning value was used for this dataset.

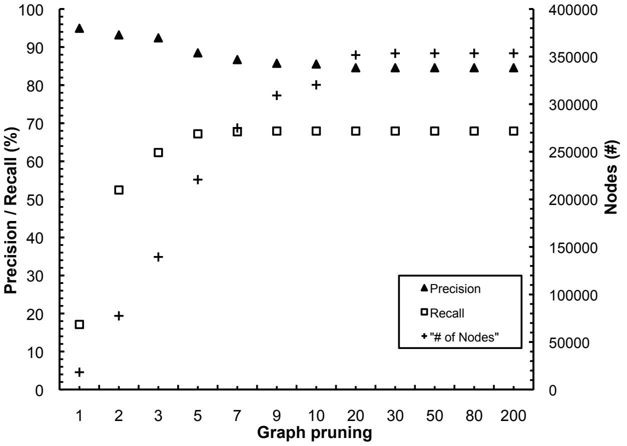

Fig. 1. Precision/Recall and node behavior of syllable lattice retrieval with varying amount of graph pruning (GP).

Post processing score pruning is done by setting a threshold between 0 and 1 such that the F-score is maximized (given that there is neither a preference for precision nor for recall). Figure 2 shows the precision vs. recall curves for syllable and word lattice retrieval. For lower thresholds, precision increases, i.e. FP are rejected, while recall does not decrease too much. At a threshold of around 0.6, however, precision starts to decrease as well. Token scores are multiplied to get a query score according to (2) . Thus, queries, especially longer ones, get a score below 1. With high thresholds, more TP than FP are pruned (since most FP were already pruned at lower levels) which leads to a decrease in precision and recall.

5.3. Hybrid Retrieval

Instead of only retrieving from a subword index, hybrid methods combine word and subword results. [5] used an exact search on the 1-best word transcription to complement subword retrieval. The results of the different hybrid retrieval techniques are presented in Table 2. We found that pure lattice word search is faster (1.52s runtime) than syllable search and performs more robustly in general, but suffers from the OOV problem. When merging both approaches, the OOV search presents the best tradeoff between accuracy and efficiency because the runtime is substantially lower than for combined search (combined: 9s vs. OOV: 2.1s) and only a small loss in recall

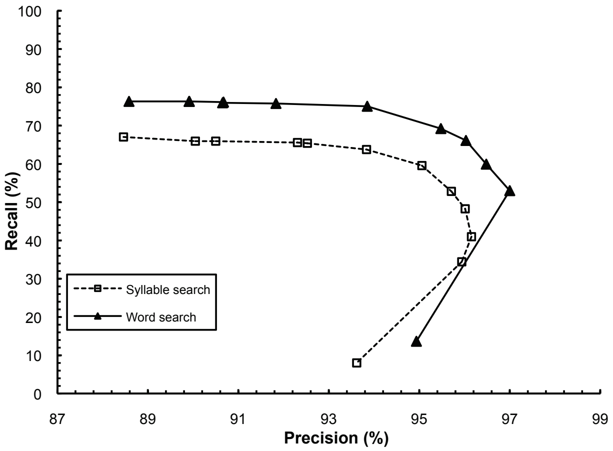

Fig. 2. Precision/Recall behavior of words and syllable lattice retrieval with varying score pruning.

is observed. Precision even increases due to the fact that the uncertain syllable lattices are accessed less frequently. These numbers were obtained without score pruning at runtime. Table 3 compares the best results achieved on lattices to the best fuzzy search results available from [5]. Because accuracy is considerably higher for word lattice search than for exact word search on 1-best transcripts and due to the presented tuning methods, the precision of lattice retrieval is increased by 8% absolute compared to the fuzzy syllable/exact word search.

Table 2. Results of hybrid retrieval.

Retrieval Approach	Precision	Recall	F-score
Fuzzy hybrid	86%	79%	82%
Latt Hybrid (Combined)	85%	83%	84%
Latt Hybrid (OOV)	89%	81%	84%

Table 3. Results of the best retrieval configurations.

Retrieval Approach	Precision	Recall	F-score
Best Fuzzy subword	85%	69%	76%
Best Latt subword	93%	65%	77%
Best Fuzzy hybrid	86%	79%	82%
Best Latt Hybrid	94%	79%	86%

6. CONCLUSION

We presented lattice indexing and retrieval as an alternative to fuzzy subword search for German STD. One finding was that lattices present a more reliable representation of the search space than fuzzy search to alleviate the important OOV problem in German. By using an adequate subword unit size and different pruning techniques, syllable lattice retrieval resulted in an absolute increase of 8% in precision and a 4% absolute loss in recall compared to fuzzy syllable search. Runtime was found to be six times faster. The best hybrid lattice system achieved 94% precision, 79% recall and an F-score of 86%, which means a 8% gain in precision at the same recall value compared to the best 1-Best hybrid. The results show that even at a WER of 40%, vocabulary independent STD for German data is feasible.

Future work should concentrate on using more condensed representations of lattices such as confusion networks or position specific posterior lattices. Furthermore, to bridge the gap in recall between fuzzy and lattice retrieval, a degree of fuzziness should be included when retrieving from lattice indexes by accepting lattice tokens within a given edit distance.

7. ACKNOWLEDGEMENTS

The work for this contribution was supported by the EU IST programme of the 6th FP for RTD within the VITALAS project (contract IST-045389). During a part of the project, the first author was affiliated with The University of Edinburgh.

8. REFERENCES

[1] K. McTait and M. Adda-Decker, "The 300k LIMSI German Broadcast News Transcription System," in *Proc. EUROSPEECH*, 2003.

[2] L. Burget, J. Cernocky, M. Fapso, M. Karafiat, P. Matejka, P. Schwarz, P. Smrz, and I. Szöke, "Indexing and search methods for spoken documents," in *Proc. TSD 2006*, 2006.

[3] S. Parlak and M. Saraclar, "Spoken term detection for turkish broadcast news," *Proc. ICASSP*, pp. 5244–5247, 2008.

[4] J. Mamou, Y. Mass, B. Ramabhadran, and B. Sznajder, "Combination of Multiple Speech Transcription Methods for Vocabulary Independent Search," in *Searching Spontaneous Conversational Speech Workshop, SIGIR*, 2008, pp. 20–27.

[5] D. Schneider, J. Schon, and S. Eickeler, "Towards Large Scale Vocabulary Independent Spoken Term Detection: Advances in the Fraunhofer IAIS Audiomining System," in *Searching Spontaneous Conversational Speech Workshop, SIGIR*, 2008, pp. 34–41.

[6] M. Saraclar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *Proc. HLT-NAACL*, 2004.

[7] M. Larson and S. Eickeler, "Using Syllable-Based Indexing Features and Language Models to Improve German Spoken Document Retrieval," in *Proc. EUROSPEECH*, 2003.

[8] M. Larson, S. Eickeler, and J. Köhler, "Supporting radio archive workflows with vocabulary independent spoken keyword search," in *Searching Spontaneous Conversational Speech Workshop, SIGIR*, 2007.

[9] I. Szöke, M. Fapso, L. Burget, and J. Cernocky, "Hybrid word-subword decoding for spoken term detection," in *Searching Spontaneous Conversational Speech Workshop, SIGIR*, 2008, pp. 42–49.

[10] R.P. Yu, K. Thambiratnam, and F. Seide, "Word-Lattice Based Spoken-Document Indexing with Standard Text Indexers," in *Searching Spontaneous Conversational Speech Workshop, SIGIR*, 2008, pp. 54–61.

[11] A. Lee, T. Kawahara, and K. Shikano, "Julius—an Open Source Real-Time Large Vocabulary Recognition Engine," in *Proc. EUROSPEECH*, 2001.