# FAST TWO-STAGE VOCABULARY-INDEPENDENT SEARCH IN SPONTANEOUS SPEECH

*Peng Yu and Frank Seide*

Microsoft Research Asia, 5F Beijing Sigma Center, No. 49 Zhichun Rd., 100080 Beijing, P.R.C.
{t-rogery,fseide}@microsoft.com

## ABSTRACT

For efficient organization of speech recordings - meetings, interviews, voice mails, lectures - the ability to search for spoken keywords is an essential capability. In [1, 2], we presented our work on vocabulary-independent search in spontaneous speech. That method involved linear scanning of phonetic lattices, and thus did not scale up to large collections. In this paper, we present a two-stage approach to fast search: first we retrieve *segments* from an index-like structure that are promising to contain the keyword, then we locate *individual* keyword occurrences by a detailed linear lattice scan. However, designing an efficient vocabulary-indepedent indexing structure is non-trivial. We use a "soft" index similar to [3] that provides *expected term frequencies* (ETFs) of query terms. We propose to approximate ETFs by $M$*-gram phoneme language models* estimated on the lattices (one per segment). Our index stores these language models in an inverted structure. Word spotting experiments on voicemails show that with this two-stage method, we lose under 4% FOM (Figure Of Merit) relative at a 25-times speed-up compared with a full linear search.

## 1. INTRODUCTION

Audio search – finding spoken words in speech recordings – is an important topic. Many approaches have been reported in literature [4, 5, 6, 1, 2]. In [1], we presented a phonetic approach to vocabulary-independent audio search. We search *lattices* instead of top-1 recognizer output to improve recall and use *phoneme* lattices instead of word lattices to handle unlimited vocabulary. In [2], we extended this work by a hybrid word-level/phonetic combination.

However, direct lattice search involves linear scanning of lattices, and thus does not scale up to large collections. For a more efficient search, some form of index is needed.

In [5], it is proposed to index individual arcs of lattice (lattice inversion). [6] proposes to store phone $M$-grams at all time locations as index. Both index individual keyword locations, and suffer from a large index size. When searching multi-label expressions such as phonetic strings, [5] needs to consider sequencing, so that only a speed-up for the first label in the query string is achieved, while for every following label, a search operation equivalent to a direct linear search is needed.

Instead, we decided to index segments that contain keywords with low time resolution of about 15 seconds, and propose a two-stage fast search approach – first quickly locate promising segments by index, then use linear search in the selected segments only. However, in our context, the concept of "indexing" is non-trivial. What units should be indexed in a *vocabulary-independent* index? What statistics should be stored to represent recognition *alternates*?

[3] addresses these questions by (1) indexing *expected term frequencies* (ETF, also known as "expected counts") (2) limiting the index to fixed-length sub-strings, and (3) approximating the ETFs of a query by the minimum of all of its sub-strings. The authors explore this approach for known-vocabulary queries, but rightly point out the fitfullness for handling unlimited vocabulary. In our view, the major shortcoming of this method is that the worst-matching sub-string dominates the estimate, and no sequence relationship amongst the individual sub-strings is exploited.

To overcome this, we propose to approximate ETFs of a query's phoneme sequence in a different way: by $M$*-gram phoneme language models* estimated on lattices (one per segment). Our "index" is the collection of these language models, stored in an inverted structure.

The main contributions of this paper are: first we propose a two-stage fast search algorithm for vocabulary-independent search. Secondly, we propose a novel method to estimate ETFs from sub-string statistics, which is the key part of the two-stage search.

The paper is organized as follows. In section 2 we will recapitulate lattice-based audio search. Section 3 will introduce the two-stage search framework, including the key part of this paper, the vocabulary-independent index. Section 4 reports experimental results, and section 5 will conclude the paper.

## 2. LATTICE-BASED WORD SPOTTING

We briefly recapitulate our phonetic approach to vocabulary-independent search in spontaneous speech [1]. In phonetic search, "indexing" consists of using speech recognition to generate a phonetic representation of each audio file – a "lattice" of scored phoneme hypotheses – and representing them in a structure efficient for searching. "Search" means rapidly locating all sub-paths in the lattice set that match the query string's phonetic representation, and it is "found" where a match's "confidence" is above a certain threshold.

We define the "confidence" of a match as its "word posterior probability"[1] $P(*, t_s, Q, t_e, *|O^i)$, i.e. the sum of the probabilities of all paths that contain the query string $Q$ from $t_s$ to $t_e$ in audio segment $i$:

$$P(*, t_s, Q, t_e, *|O^i) = \sum_{\substack{\forall WT: \exists k,l: t_k = t_s \wedge t_{k+l} = t_e \\ \wedge w_k, \ldots, w_{k+l-1} = Q}} P(WT|O^i) \quad (1)$$

where $O^i$ represents the $i$-th audio segment in the audio collection, $W = (w_1, ..., w_N)$ a hypothesized phonetic transcription, $T = (t_1, ..., t_{N+1})$ the associated time boundaries, and $P(WT|O^i)$ the posterior probability of the hypothesized $W$ and $T$. Eq. (1) can be efficiently approximated by forward-backward scoring of lattices.

[1] "Word posterior probability" is the common term to refer to the expression $P(*, t_s, Q, t_e, *|O)$. It is, in fact, not a probability: Summing it up over all combinations of $(t_s, Q, t_e)$ does not give 1 but the total expected term count.

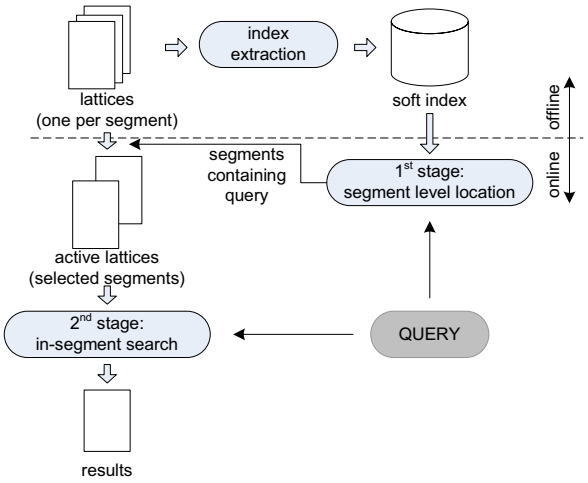


**Fig. 1**. Flow chart of the two-stage search.

The same formalism can also be applied in a LVCSR context to word lattices directly, if unknown-vocabulary queries are no issue, and both can be combined into a word/phonetic hybrid which can lead to substantial accuracy improvement [2].

## 3. TWO-STAGE FAST SEARCH

We propose here a two-stage fast-search: given a query keyword, first find promising segments that likely contain the keyword, then apply detailed linear search in these segments. Fig. 1 shows the flow chart of the search process. The idea is to realize the first stage by some pre-stored index such that the time cost is small compared to the second stage. The second stage is realized by symmetric dynamic programming as described in detail in [1]. The remainder of this section will only discuss the first stage.

### 3.1. Indexing Expected Term Frequencies

In [3], audio clips are ranked by *expected term frequency* (ETF) for the purpose of retrieval. We propose to apply the same ranking for the purpose of two-stage search. For a given query $Q$, the ETF for each segment $O^i$ can be computed as follows:

$$\begin{aligned} \mathrm{ETF}_i(Q) &= E_{WT|O^i}\{\mathrm{TF}_W(Q)\} \\ &= \sum_{\forall WT} \mathrm{TF}_W(Q) P(WT|O^i) \\ &= \sum_{\forall t_s, t_e} P(*, t_s, Q, t_e, *|O^i) \end{aligned} \quad (2)$$

with $\mathrm{TF}_W(Q)$ being the term frequency of query term $Q$ in $W$.

If the query vocabulary is known at the time of indexing, we can directly calculate and store the ETFs of all keywords for all segments. However, this is impossible for an open query vocabulary. Instead, we approximate ETFs from statistics of a limited set of sub-strings (the *index keys*) derived from each audio segment. Our "index" stores these sub-string statistics. The remainder of this section will detail this method.

In the following, query terms shall be represented as phoneme sequences[2] $Q = (q_1, ..., q_l)$. If pronunciation variants exist, the expected term frequencies of all pronunciations need to be summed up, possibly weighted with a pronunciaton prior.

[2]The problem converting from graphemic to phonetic form is not the subject of this paper.

### 3.2. Upper-bound Approximation

Before introducing our new method, we want to review an approach proposed by Allauzen *et al.* [3] that we want to compare ours with in the results section. We call it *upper-bound approximation*.

Allauzen proposes to exploit that the ETF of a string is bounded by all ETFs of its sub-strings. The minimum ETF of a set of sub-strings of the query $Q$ is an upper-bound of the ETF of $Q$. If the sub-strings set is representative enough, the upper-bound is approaching the ETF of $Q$ (if $Q$ itself belongs to the sub-string set, the upper bound exactly equals ETF of $Q$). [3] approximates ETFs as:

$$\mathrm{ETF}_i(Q) \approx \min_{\forall u: u \subset Q \wedge u \in \mathcal{K}} \mathrm{ETF}_i(u)$$

with $u \subset Q$ meaning that $u$ is a sub-string of $Q$, and $\mathcal{K}$ being the index-key set. The index is designed to store the ETFs of all sub-strings $u \in \mathcal{K}$.

In [3], the method is used for in-vocabulary keywords. Nevertheless, the method is directly applicable for unlimited vocabulary.

### 3.3. $M$-gram Approximation

The problem we see with the upper-bound approximation is that only the minimum ETF of sub-strings is used for the approximation. No sequential information is utilized. We propose to make use of this and borrow some idea from $M$-gram language modeling.

We define $P(Q|O^i)$ as the probability of observing query string $Q$ at any word boundary in the recording $O^i$. Then the ETF is

$$\mathrm{ETF}_i(Q) = \tilde{N}_i \cdot P(Q|O^i) \quad (3)$$

with $\tilde{N}_i$ being the expected number of words in document $i$.

Now we make the assumption that $P(Q|O^i)$ can be approximated by an $M$-gram model:

$$P(Q|O^i) \approx \tilde{P}(Q|O^i) = \prod_{i=1}^{l} \tilde{P}(q_i|q_{i-M+1}, \cdots, q_{i-1}, O^i) \quad (4)$$

We estimate the $M$-gram probabilities for each segment's lattice such that the model best matches (in the Maximum-Likelihood sense) the phoneme-string posterior distribution defined by the lattice. The resulting estimator has the following form and solution:

$$E_{WT|O^i}\{\log \tilde{P}_{\mathrm{MELL}}(W|\mathcal{M})\} \stackrel{!}{=} \max$$

$$\tilde{P}_{\mathrm{MELL}}(q_j|q_{j-M+1}, \cdots, q_{j-1}, O^i) = \frac{\mathrm{ETF}_i(q_{j-M+1}, \cdots, q_j)}{\mathrm{ETF}_i(q_{j-M+1}, \cdots, q_{j-1})}$$

where "MELL" stands for "Maximum Expected Log-Likelihood." This estimator has two important properties: If the lattice degenerates to a single path, MELL reduces to ML; and for $M \geq$ length of $Q$, $P(Q|O^i) = \tilde{P}_{\mathrm{MELL}}(Q|O^i)$.

### 3.4. Selection Of Index-Key Set

For larger $M$, the vast majority of phoneme $M$-grams occur very rarely or are even phonotactically impossible. To strike a balance between accuracy and index size, we want to use a variable-length index-key set that includes only probable long index keys, while keeping a complete set of short index keys to ensure coverage. We choose the index-key set by extracting from a large background dictionary all phoneme sub-strings up to a maximum length.

For sub-strings not contained in the background dictionary, we introduce a back-off/fallback strategy. Again, we borrow a technique from traditional $M$-gram language modeling. When a long

**Table 1**. Test-set token perplexities and top-1 error rates.

| setup | word LM | phonetic LM |
|---|---|---|
| token perplexity | 442 | 43 |
| token error rate | 52.7% | 44.2% |

history is available in the index-key set, it will be used directly, otherwise, we fall back to a shorter history:

$$\tilde{P}(q_j|q_{j-l+1},...,q_{j-1},O^i) = \begin{cases} \tilde{P}_{\text{MELL}}(q_j|q_{j-l+1},...,q_{j-1},O^i) \\ \quad \text{if } (q_{j-l+1},...,q_j) \in \mathcal{K} \\ B(q_{j-l+1},...,q_{j-1}|O^i)\cdot\tilde{P}(q_j|q_{j-l+2},...,q_{j-1},O^i) \\ \quad \text{if } (q_{j-l+1},...,q_j) \notin \mathcal{K} \\ \quad \text{but } (q_{j-l+1},...,q_{j-1}) \in \mathcal{K} \\ \tilde{P}(q_j|q_{j-l+2},...,q_{j-1},O^i) \\ \quad \text{otherwise} \end{cases} \quad (5)$$

Here $B$ is the *backoff weight*, calculated such that for each history $h$, $\sum_{\forall q_j} \tilde{P}(q_j|h,O^i) = 1$.

## 4. RESULTS

### 4.1. Setup

We evaluate our system on the LDC Voicemail corpus [7], which consists of two parts, VM-I and VM-II. All test data in VM-I and the entire VM-II (training and test portions), totally about 15h, are used as our test set (we do not use the VM-II designated training portions anywhere for training purposes). Of this test set, 1.5h are set aside as a dev set for algorithm development and parameter tuning. All results below are for the remaining evaluation set, about 13.5h. This set has been split into 3224 segments with average duration of about 15s. The acoustic model was trained on 309h of the Switchboard corpus.

The keyword set was selected by an automatic procedure as in [1]. The resulting keyword set has 6058 entries, 2295 (37.9%) of which are OOV[3] w.r.t. our word-based baseline. 3050 (50.3%) of these keywords are single words, the other 3008 (49.7%) are compound words. Example keywords are *adapter*, *semiconductor*, and *multiple-database-search*.

To be domain independent, no language model has been trained on any voicemail data. The phonetic language model is trained on transcriptions of 309 hours of Switchboard, of 50 hours of LDC Broadcast News 96 training, and from about 87000 background dictionary entries, a total of 11.8 million phoneme tokens. In addition, we have also built a word-level system for our hybrid experiments. The word language model is trained on Switchboard transcriptions, with a dictionary of 27463 words. Table 1 shows the test-set token (word or phoneme) perplexities and top-1 error rates.

We measure search accuracy by the common "Figure Of Merit" (FOM) defined by NIST (National Institute of Standards & Technology) as the average of detection/false-alarm curve over the range [0..10] false alarms per hour per keyword.

### 4.2. Upper-bound And $M$-gram Approximation

Fig. 2 compares these two basic sub-string approximations. We only use the single-word query set (3050 words) because we want to keep effects from cross-word indexing separate.

The horizontal axis represents the percentage of *active segments* - segments selected for linear search, while the vertical axis

[3] A word sequence is out of vocabulary if at least one of its words is.

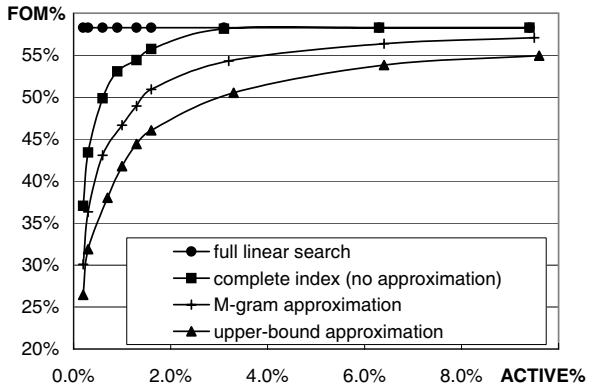


**Fig. 2**. Comparison of upper-bound and $M$-gram approximation ("full 4-gram" index-key set, single-word queries only).

shows the FOM. The line "full linear search" means detailed linear search on *all* segments. The line "complete index" represents an oracle experiment where the index-key set includes the full-length pronunciations of all keywords, thus ETFs of all keywords are directly available (no need to approximate from sub-strings). This setup reaches full accuracy with only 4% active segments (25 times speed-up), which confirms our choice of ranking by ETF. It also gives the theoretical upper-limit of any sub-string approximation methods.

The other two lines show the two-stage search results with upper-bound and our $M$-gram approximation respectively. The index-key set includes all sub-strings with the same length limitation of 4 ("full 4-gram" set) used in [3]. Compared to the upper-bound approximation, $M$-gram approximation halves the gap to the theoretical upper-limit ("complete index").

### 4.3. Index-Key Set Selection

Fig. 3 compares different index-key sets and evaluates our method of extracting index-keys from background dictionary.

When the length of index key ($M$) grows from 4 to 5 and 6, the "full $M$-gram" line is approaching the theoretical upper-limit ("complete index"). With 4% active segments, the "full 6-gram" is only 0.8 points below the "full linear search". However, the index set is untolerably large – about 2 billion entries per hour of speech.

Next we replaced the key set by one extracted from a background dictionary ("background $M$-gram") with 98877 words. In

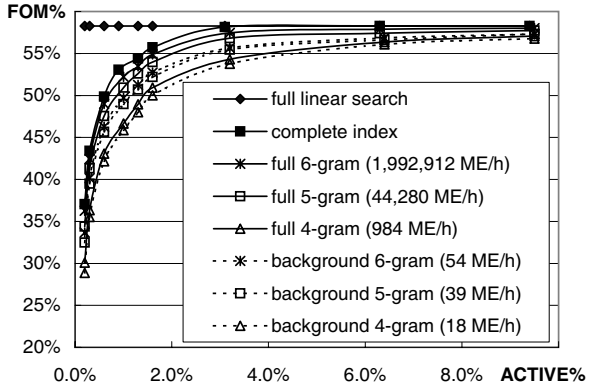


**Fig. 3**. Comparison of different index-key sets. Index sizes are shown in million entries per hour (ME/h). Single-word queries only.

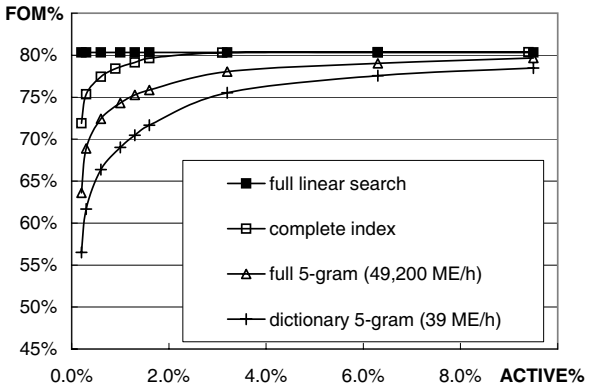


**Fig. 4**. Results for compound-word queries.

our 3050 keywords set, 670 (22%) are not in the background dictionary. The backoff/fall-back strategy in Eq. 5 is used here. From "full 6-gram" to "background 6-gram", we lose another 2-points FOM with 4% active segments, while the index size is dramatically reduced from 2 billion to 54 million entries per hour of speech.

### 4.4. Compound Words (Multi-Label Queries)

Fig. 4 shows our experiments with compound-word queries (3008 words). In linear lattice search, we found when matching compound words, allowing an optional silence (`sil`) at word boundaries yielded 5% FOM improvement. This can directly be applied to ETF estimation for compound-word queries by summing up ETF for all pronunciations with and without `sil`. The "full 5-gram" set now includes also 5-grams with `sil` (e.g. `n-uw-sil-y-ao`). The figure shows that with 4% active segments, the "full 5-gram" is 2-points below "complete index".

The "background 5-gram" setup uses the same 5-grams from background dictionary. Cross-word combinations should ideally be included but is infeasible due to their sheer number. We include a single unigram for `sil` in the index-key set. At 4% active segments, we get another 2-points FOM drop. The drop is larger than in the case of single-word queries (about 1.5 points). The reason is of course that our background index does not include the cross-word phoneme combinations.

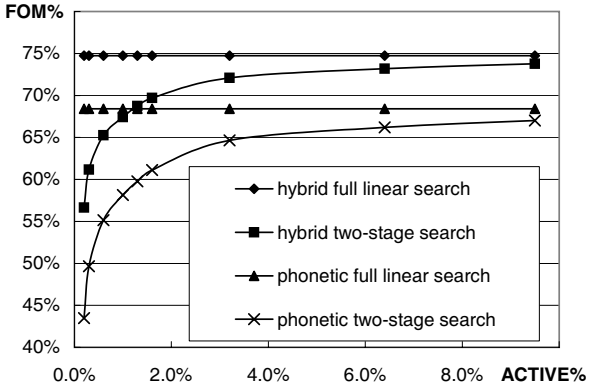


**Fig. 5**. Results for entire query set (single and compound word queries, "background 5-gram"), and comparison with hybrid word-level/phonetic two-stage search.

### 4.5. Entire Query Set and Hybrid Indexing/Search

Fig. 5 shows the result on the entire query set (both single-word and compound-word queries, 6058 words), using the "dictionary 5-gram" setup. With 4% active segments, two-stage search is about 3-points below the full linear search (to distinguish from the following hybrid results, the two setups are labeled "phonetic two-stage search" and "phonetic full linear search").

Fig. 5 also shows results for the hybrid word/phonetic search briefly introduced in section 2. Our indexing/two-stage search scheme can easily be applied. The line "hybrid full linear search" shows results for the hybrid system using full linear search. The pure word-based system by itself has very poor accuracy (44%, not shown in the Fig.), caused by both language model mismatch and out-of-vocabulary queries. But taken together, we still achieve a 6.5-points improvement over the pure phonetic system.

The line tagged "hybrid two-stage search" represents the hybrid two-stage search result. The index used in the first stage combines the phonetic "background 5-gram" setup and a word-level index with fixed vocabulary (realized with the method in section 3). Compared with "phonetic two-stage search," a consistent 6-7 point improvement is achieved over the entire curve.

## 5. CONCLUSION

We have presented a vocabulary-independent two-stage fast search method. In the first stage, an ETF based "soft" index is used to select segments that are likely to contain query keywords (active segments), while in the second stage, linear search is applied only to these active segments.

To be vocabulary-independent, we proposed a sub-string based $M$-gram approximation for estimating ETF for any query keywords. We have also looked into the problem of index key selection (balance between index size and accuracy); compound words queries (cross-word phoneme combinations, pause between words); and extension of the indexing/two-stage framework to hybrid word-level/phonetic search.

Experiments show with this two-stage method, we lose only about 4% FOM relative at 25 times speed up.

## 6. ACKNOWLEDGEMENTS

The authors wish to thank Dr. Asela Gunawardana for sharing his acoustic models for Switchboard, and Dr. Ciprian Chelba for frequent and fruitful discussions.

## 7. REFERENCES

[1] F. Seide, P. Yu, *et al.*, Vocabulary-Independent Search in Spontaneous Speech. *Proc. ICASSP'04*, Montreal, 2004.

[2] P. Yu, F. Seide, A Hybrid Word / Phoneme-Based Approach for Improved Vocabulary-Independent Search in Spontaneous Speech. *Proc. ICLSP'04*, Korean, 2004

[3] C. Allauzen, M. Mohri, M. Saraclar, General Indexation of Weighted Automata - Application to Spoken Utterance Retrieval. *Proc. HLT'04*

[4] J. Garofolo, TREC-9 Spoken Document Retrieval Track. National Institute of Standards and Technology, `http://trec.nist.gov/pubs/trec9/sdrt9_slides/sld001.htm`

[5] M. Saraclar, R. Sproat, Lattice-based search for spoken utterance retrieval. *Proc. HLTNAACL 2004*

[6] S. Dharanipragada, S. Roukos, A multistage algorithm for spotting new words in speech. IEEE Transactions on Speech and Audio Processing, vol. 10, issue 8, pp. 542-550, 2002.

[7] M. Padmanabhan et al, Voicemail Corpus Part I (LDC98S77) and Part II (LDC2002S35). Linguistic Data Consortium, `http://www.ldc.upenn.edu`