WORD-LATTICE BASED SPOKEN-DOCUMENT INDEXING WITH STANDARD TEXT INDEXERS

Frank Seide, Kit Thambiratnam, and Roger Peng Yu

Microsoft Research Asia, Beijing Sigma Center, 49 Zhichun Rd., 100080 Beijing, P.R.C.
{kit,fseide,rogeryu}@microsoft.com

ABSTRACT

Indexing the spoken content of audio recordings requires automatic speech recognition, which is as of today not reliable. Unlike indexing text, we cannot reliably know from a speech recognizer whether a word is present at a given point in the audio; we can only obtain a probability for it. Correct use of these probabilities significantly improves spoken-document search accuracy.

In this paper, we will first describe how to improve accuracy for "web-search style" (AND/phrase) queries into audio, by utilizing *speech recognition alternates* and *word posterior probabilities* based on *word lattices.*

Then, we will present an end-to-end approach to doing so *using standard text indexers*, which by design cannot handle probabilities and unaligned alternates. We present a sequence of approximations that transform the numeric lattice-matching problem into a symbolic text-based one that can be implemented by a commercial full-text indexer.

Experiments on a 170-hour lecture set show an accuracy improvement by 30-60% for phrase searches and by 130% for two-term AND queries, compared to indexing linear text.

Index Terms— Audio Indexing, Word Lattice, Posterior, Full-Text Indexing

1. INTRODUCTION

For efficient management of digital audio assets—audio or video recordings with intrinsic value—keyword search is an essential tool. However, audio and video is among the least accessible content for search engines. While for Internet video, audio/video search engines often have significant metadata at their disposal (anchor text, surrounding text, closed captions, editorial description texts), we frequently have no such luxury for typical "low-production value" enterprise audio like phone-conference recordings or video-taped presentations. We cannot escape "cracking open" the audio by speech recognition to *index the spoken content itself.* Such enterprise audio is still a challenge for today's speech-recognition technology, which achieves word accuracies of only 50-70% [1, 2, 3].

In this paper, we focus specifically on "web-search style" queries—the kind of search that most computer users are familiar with: few keywords only, and for a document to match, it *must contain all query terms*. However, unlike indexing text, we cannot reliably know from a speech recognizer whether a word is present at a given point in the audio; we can only obtain a probability for it.

Thus, in this paper, we cast the problem of web-style search into audio as *computing the posterior probability that an audio document contains all query terms and phrases* and comparing it to a threshold representing the minimum precision acceptable to the user[1]. We will first show how this probability can be *efficiently computed from word lattices.* Secondly, we will show how this can be approximately implemented *using standard full-text indexing engines*—despite the fact that, by their design, they cannot handle probabilities and unaligned alternates. Reusing existing text search engines is highly desirable because they are complex systems involving substantial investments (both development and deployment), and also to allow a seamless search experience across audio/video and text. With "reusing" we literally mean *without code modification*, because for complex commercial-grade systems, even small code changes are costly and bear significant risk. We present a sequence of approximations to map the word-lattice structure and to transform the numeric lattice-matching problem *into a symbolic text-based one implementable by a commercial full-text indexer*.

Related work includes the TREC (Text REtrieval Conference) "Spoken-Document Retrieval" (SDR) benchmarks [4]. Just by indexing speech-to-text transcripts—with error rates as high as 35%—search accuracies on par with indexing manual transcripts were achieved, and some researchers claimed SDR to be a "solved problem" [4]. However, this result is due to high redundancy in the relatively long topic descriptions and news segments, and the "soft" vector-space ranking. It does not translate to our task of selecting documents that contain *all* query terms. Rather, our problem at hand is one of "Spoken-Term Detection" (STD) or "keyword spotting."

Traditionally, *keyword spotters* are small-vocabulary systems whose vocabulary contains only the query terms (non-keyword speech is discerned with garbage- or anti-models [5]) or are based on phonetic lattices (avoiding the need to know the dictionary at indexing time, e.g [6]). Such systems do not scale well—they do not really allow indexing, and, lacking a language model, suffer from poor Precision—and are unsuitable for large-scale enterprise databases.

The alternative, *continuous-speech recognition* based word spotters like ours, achieve good Precision by replacing the garbage model with large vocabularies of 100,000+ words and m-gram language models. However, for our web-style queries (all terms must match), today's accuracy levels of 50–70% [1, 2, 3] lead to poor Recall when just "naïvely" indexing speech-to-text transcripts. Significant improvements are achieved by taking the probabilistic nature of speech recognition into account, based on word lattices [8, 9, 6, 7, 10]: *Word confidence scores* help to reduce false positives and allow ranking by reliability and thresholding, and—this is the core of this paper—searching *alternative recognition candidates* boosts Recall for multi-word queries. Fig. 1 shows an example word lattice.

[1]Relevance ranking should then be applied to the subset of documents that match the boolean condition specified by the query. For the purpose of this paper, however, we leave the challenge of relevance weighting—arguably a non-speech problem—to our colleagues in the IR community.

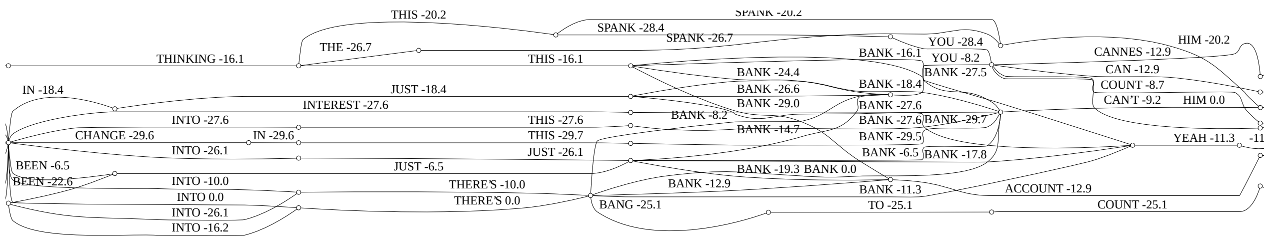

Fig. 1. Word-lattice example for the word sequence "...into this bank account."

When indexing enterprise-scale audio collections of thousands of hours, search efficiency is a concern. Word lattices cannot be represented by commercial full-text indexers out of the box. However, *conceptually* they can be indexed like text [8]. Related work includes the position-specific posterior-lattice (PSPL) approximation in which word hypotheses are merged w.r.t. word position, which is representable by text indexers and is treated as a hidden variable [9]. However, this method involves relatively high overgeneration and loses time information for individual hypotheses. *Confusion networks* [11] align a speech lattice with its top-1 transcription, yielding a parsimonious "sausage"-like approximation of lattices. However, "sausages" do not lend themselves for indexing due to the presence of a large number of null links. In [10] we have proposed "Time-based Merging for Indexing" (TMI) to significantly reduce lattice sizes by merging of similar hypotheses. In [12], based on the lessons of [9, 11, 10], we introduced the "Time-Anchored Lattice Expansion" (TALE) that maps the complex time structure onto the word-position concept of text indexers in a way that avoids the high-overgeneration and null-link problems. The paper at hand extends TALE in that we address the problem of storing posteriors and performing probabilistic searches using a symbolic representation.

This paper is organized as follows. Next, we establish the theoretical foundation of word-lattice-based speech indexing. Section 3 then introduces the proposed method to index word lattices with standard text indexers, and section 4 presents experimental results.

2. POSTERIORS AND LATTICES

In this section, we want to show how to compute the *document-hit posterior*—the posterior probability that an audio document D fulfills the *boolean condition* specified by the query Q—and how to do so efficiently from lattices, where Q is assumed to be of "web-search" type, i.e. (1) all query terms must be found in matching documents and (2) quoted terms must match in exact order (phrase queries)[2]. We denote this as $R(Q,D) = 1$ for a relevant document, 0 otherwise.

Because the true transcription of the audio is unknown, $R(Q,D)$ is not directly known. All we are provided with by the speech recognizer is

- hypotheses (W,T) on what might have been said, with $W = (w_1, w_2, ..., w_N)$ = hypothesized transcription of the audio database and $T = (t_1, t_2, ..., t_{N+1})$ = associated word time boundaries, and
- associated posterior probabilities $P(WT|O)$ (O for observation denoting the totality of all audio), denoting the probability that transcription hypothesis (W,T) was indeed the underlying event that caused the generation of the audio we observed.

[2]For the purpose of this paper, we shall not be concerned with relevance weighting, but simply *consider all audio documents that match the boolean condition equally relevant*.

With this, we can compute the *posterior probability that document D matches query Q* as:

$$\begin{aligned} P(R(Q,D)|O) &= E_{WT|O}\left\{R_{WT}(Q,D)\right\} \\ &= \sum_{WT} P(WT|O) \cdot R_{WT}(Q,D) \qquad (1) \\ &= \prod_{q\in Q}\left[1 - \prod_{\forall(t_s,t_e)}(1 - P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O))\right] \qquad (2) \end{aligned}$$

where $R_{WT}(\cdot)$ denotes relevance *w.r.t. the hypothesized transcription/alignment*, and $P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O)$ the *term-hit posterior* probability for the hypothesis that query term q occurred with time boundaries t_s and t_e. Eq. (2) simply expresses the probability that each query term is present at least once[3]. By comparing $P(R(Q,D)|O)$ against a pre-determined threshold $P_{\min}$, we can select the documents that match the query with a probability of $P_{\min}$ or higher, where the threshold would be chosen so that a target Precision will be reached, such as 75%.

Term-hit posteriors are computed by marginalizing out over all transcription hypotheses that contain query term q with time boundaries t_s and t_e (i.e. transcription hypotheses that match a "template" $*\text{-}t_s\text{-}q\text{-}t_e\text{-}*$):

$$P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O) = \sum_{\substack{WT:\exists k,l:t_k\approx t_s\wedge t_{k+l}\approx t_e \\ \wedge w_k,\ldots,w_{k+l-1}=q}} P(WT|O) \qquad (3)$$

We are not concerned with the context of the query keyword occurrence, so Eq. (3) marginalizes out all contexts. Also, multiple hypotheses for the same query keyword with slightly dissimilar time boundaries are considered "estimation noise" and marginalized out as well. Lastly, to deal with phrases (multiple words "in quotes" that must match in precise sequence), we treat multi-word phrases simply as single query terms, i.e. a query term q may actually be a *sequence of multiple words* that are required to match in precise sequence.

Term hit posteriors can computed efficiently and easily from word lattices. A lattice is a compact representation of speech recognition word hypotheses as a directed acyclic graph (DAG) like the one shown in Figure 1. For every word in the audio, a lattice contains multiple alternative hypotheses that were considered during the recognition process, including word matching probabilities and time boundaries.

Defining a lattice as $\mathcal{L} = (\mathcal{N}, \mathcal{A}, n_{\text{start}}, n_{\text{end}})$, each lattice arc $a = (S[a], E[a], W[a], P_{\text{arc}}[a]) \in \mathcal{A}$ represents a word hypothesis, while nodes $n \in \mathcal{N}$ represent context conditions and time $t[n]$.[4] $S[a]$, $E[a] \in \mathcal{N}$ denote the start and end node of the arc, and $W[a]$ its word identity. Any route through the graph from the lattice start

[3]Eq. (2) assumes that the query-term matches are independent (non-overlapping and sufficiently far enough from each other).

[4]In alternative definitions of lattices found in literature, nodes represent words and arcs word transitions.

node n_{start} to its end node n_{end} constitutes an utterance hypothesis whose probability can be derived from the weights along the route.

For the purpose of this paper, arc weights are *word posterior probabilities* $P_{\text{arc}}[a]$—the posterior probability of the hypothesis that word $W[a]$ occurred covering the time range $t[S[a]]...t[E[a]]$, in the context of arcs entering $S[a]$ and arcs leaving $E[a]$. $P_{\text{arc}}[a]$ is the aggregate posterior probability of all routes through the graph containing arc a. The $P_{\text{arc}}[a]$ are computed from acoustic likelihoods and language-model probabilities obtained from the speech recognizer, using the familiar forward-backward procedure [13].

With this, term-hit posteriors $P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O)$ (Eq. (3)) can be easily computed from arc posteriors $P_{\text{arc}}[a]$ as:

$$P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O) \quad = \sum_{\substack{WT:\exists a:t[S[a]]\approx t_s \wedge t[E[a]]\approx t_e \\ \wedge W[a]=q}} P_{\text{arc}}[a] \tag{4}$$

for *single-word queries* ($|q| = 1$). I.e., it is simply the arc-posterior sum over all arcs a that match q in a given time range. For multi-word queries $q = (q_1, ..., q_K)$, we need to take the sum over all connected routes $(a_1, ..., a_K)$ that begin at time t_s, end at t_e, and constitute the token sequence q:

$$P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O) \quad = \sum_{\substack{\forall(a_1,\cdots,a_K): \\ t[S[a_1]]\approx t_s \\ \wedge t[E[a_K]]\approx t_e \\ \wedge W[a_1,\cdots,a_K]=q}} P_{\text{arc}}[a_1] \prod_{k=2}^{K} \frac{P_{\text{arc}}[a_k]}{\sum_{a':S[a']=S[a_k]} P_{\text{arc}}[a']}$$

3. INDEXING WORD LATTICES WITH STANDARD FULL-TEXT INDEXERS

We have defined the task as retrieving all audio documents D that contain all query terms/phrases $q \in Q$, such that a certain Precision is met. We have further derived how this is achieved by thresholding against the posterior probability that an audio document contains all query terms using word posteriors obtained from word lattices.

We now want to show how this can be approximately achieved by *reusing existing text indexers*—despite the fact that word lattices cannot be mapped exactly onto a text index due to two issues:

- *Word-position concept:* Text indexers locate word occurrences by word-position counts in the running text, for phrase matching and proximity-based relevance scoring. However, the concept of word positions is ill-defined to represent the temporal nature of word hypotheses, which are not generally word-aligned with each other and often partially overlap.
- *No concept of posteriors:* For text indexers, there is no uncertainty whether a word occurred at a given position or not—they only consider "presence" of query words or phrases. Therefore, they neither store posteriors nor allow specifying thresholds at query time.

Besides, lattice size affects execution time—lattices can be up to 100 times or more the size of a text transcript.

We attempt to solve the above conundra by a series of transformations and approximations that simplify the lattice and transform the *numeric problem* of retrieving documents that match the query with a certain probability into a *symbolic text-based* one that can be implemented by a commercial full-text indexer.

3.1. Word-Position Mapping: Time-Anchored Lattice Expansion (TALE)

The goal of "Time-Anchored Lattice Expansion" (TALE) is to "get away" with not being able to store the precise lattice graph structure, yet *matching phrases as accurately as still possible*. TALE is a heuristic to approximately align lattice arcs to word positions, forming a "sausage"-like lattice [12]. Because the standard phrase matcher requires words belonging to phrases to be in consecutive word positions, TALE *aligns some words to multiple slots* (overgeneration) in order to guarantee that all phrases *up to M words* are retained in consecutive word positions. The resulting posterior distributions $P(w|i, O)$ for word w at position i are assigned in a way that retains the expected term frequencies $E_{w|O}\{\text{TF}_W(w)\}$ [12]:

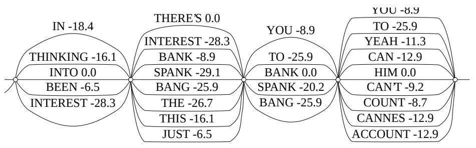

Fig. 2. The lattice of Fig. 1 after TALE processing.

$$P(w|i, O) \quad = \quad \sum_{\delta=1}^{M} \lambda_\delta \cdot \sum_{\forall n:i_n+\delta=i} P(w|n, \delta, O) \cdot P(*\text{-}n\text{-}*|O)$$

$$P(w|n, \delta, O) \quad = \quad \frac{\sum_{\substack{\forall n_k, w_k: \\ k=1...\delta \wedge w_\delta=w}} P(*\text{-}n\text{-}w_1\text{-}n_1\text{-}...\text{-}w\text{-}n_\delta\text{-}*|O)}{P(*\text{-}n\text{-}*|O)}$$

$P(w|n, \delta, O)$ is the posterior for word w to happen *as the δ-th path token after a given node n* (read "$*\text{-}n\text{-}w_1\text{-}n_1\text{-}...\text{-}w\text{-}n_\delta\text{-}*$" as a template for a lattice path that contains a δ-gram that starts at node n and ends in word w, and "$*\text{-}n\text{-}*$" for all paths passing through node n). λ_δ are interpolation weights ($\sum \lambda_\delta = 1$), and i_n denotes the word position *on the best path* that is nearest node n ("binning").

In this paper, we use TALE in conjunction with our "TMI" method (Time-based Merging for Indexing) introduced in [10]—a technique for merging similar lattice arcs that likely represent the same spoken word [10]. This is not strictly necessary for TALE, but the size reductions make data handling significantly easier, while, as experiments show, not harming word and phrase-matching accuracy.

Fig. 2 shows the lattice in Fig. 1 after TALE processing.

3.2. Indexing and Searching TALE Lattices with Standard Text Indexers

Lastly, in order to store the posterior values $P(w|i, O)$ in the index, we must use a trick: We "decorate" the word symbols themselves to somehow encode the posterior value. E.g. `computer@7` denotes a word hypothesis for "computer" where `@7` stands for a posterior quantization level (we found 16 levels to be sufficient).

Finally, to *search* such an index *for all documents matching the query with a minimum probability $P_{\min}$*, we must express this somehow in the full-text indexer's *symbolic query language*—as full-text indexers have no facility to specify $P_{\min}$. For this, we (rather crudely) approximate the products in Eq. (2) by minima:

$$\begin{aligned} P(R(Q, D)|O) \quad &\approx \quad \min_{q\in Q} \left[1 - \min_{\forall(t_s,t_e)} \left\{ 1 - P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O) \right\} \right] \\ &= \quad \min_{q\in Q} \max_{\forall(t_s,t_e)} P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O) \end{aligned}$$

Now, cutting off document relevance probabilities at a given threshold $P_{\min}$ becomes a *boolean expression*:

$$\begin{aligned} &P(R(Q, D)|O) > P_{\min} \\ &\quad \Leftrightarrow \quad \forall q \in Q, \exists(t_s, q, t_e), P(*\text{-}t_s\text{-}q\text{-}t_e\text{-}*|O) > P_{\min} \end{aligned}$$

which can indeed be implemented as a combination of symbolic AND and OR clauses.

Table 1. Spoken-document search results for phrase, single-word, and two-term AND queries. All numbers in percent.

query type:	phrase		single-word		x AND y		index
configuration	mAP	R_{75}	mAP	R_{50}	mAP	R_{75}	size
STT transcript	42.6	43.4	44.3	45.2	26.1	26.1	1.0
raw lattice	67.2	52.9	56.3	46.1	63.3	61.6	1617
+TMI, pruning[10]	67.1	55.6	55.6	46.0	60.4	60.9	9.9
+TALE	67.6	55.5	55.3	46.2	61.5	61.1	11.5
+min approx.	67.3	54.1	55.2	46.1	61.2	58.4	11.5
+quantization	66.2	54.2	52.1	46.1	60.9	57.4	11.5
rel. improvement	**+55%**	**+25%**	**+18%**	**+2%**	**×2.3**	**×2.2**	

We have actually implemented all of the above steps with an *unmodified out-of-the-box* Microsoft SQL Server 2005, only using publicly documented extensibility APIs originally intended for parsing proprietary binary formats and language extension. Here is an example of an actual thresholded SQL full-text AND query into audio: `SELECT id FROM files WHERE CONTAINS (audio, 'FORMSOF (INFLECTIONAL, "barack@10 obama@10") AND FORMSOF (INFLECTIONAL, "hillary@10 clinton@10")')`

4. RESULTS

We have evaluated our method on the 170-hour MIT iCampus lecture set [3] with 160 lectures. A speaker-independent Large Vocabulary Continuous Speech Recognition (LVCSR) system generated rich word lattices using an acoustic model trained on the 1700-hour Switchboard "Fisher" telephone-speech corpus [2] and a set of language models trained on the transcript using 10-way jackknifing (training and test sets kept disjunct). The word accuracy is 53.4%.

We evaluate search accuracy with multi-word (phrase), single-word, and two-term AND queries (x AND y where x and y can be single terms or phrases). The keyword set is synthetic and consists of noun phrases chosen from the transcripts such that for each query there are at most two matching lectures.

Table 1 shows search-accuracy results for exact and approximate lattice-based search according to two accuracy metrics:

- mAP: mean average precision common in information-retrieval research, where documents are ranked by their document-level matching posterior as defined in Eq. (2);
- R_{75}/R_{50}: document Recall at after-the-fact Precision 75% (multi-word queries) and 50% (single words), respectively.

The first and second result rows show the improvement we can obtain by searching raw lattices as obtained from the recognizer. Compared to "naïvely" indexing speech-to-text (STT) transcripts (like full-text indexers could do without our method), improvements are significant—more than 2× on AND queries (from 26% to 62%).

The next row shows the effect of reducing lattice size using the TMI method [10] and pruning away arcs that have posterior below -8.0. The subsequent three rows show the result for TALE processing, the min/max approximation, and posterior quantization. The resulting lattices—now *indexable by standard full-text indexers*—nearly retain the dramatic accuracy improvement obtained for the raw lattices, with a largest loss of 4 points for AND queries.

Next, we evaluated the accuracy for the primary task of this paper—retrieving documents that contain all query terms with a probability greater than a given Precision threshold. We consider 75% a realistic choice for the target Precision—with anything less, users may consider the system broken! For single-word queries, we had to reduce it to 50%. The cutoff threshold should be tuned on a development set, but in our case we split our test set in two and applied a threshold measured on one half to the other half and vice versa. Table 2 shows that for phrase queries and AND queries, "naïvely" searching STT transcripts suffers from a poor Recall despite high Precision. Raw lattices achieve significantly better Recall through using alternates, while the precision is within the target range. For single-word queries, STT transcripts fail to achieve the targeted Precision level of 50%, while using lattices achieves this while trading Recall. Again, the loss caused by the proposed approximations is less than 3 points.

Table 2. Recall (R) and precision (P) in percent when using a cutoff threshold targeting precision 50% for single-word queries and 75% for phrase and AND queries.

query type:	phrase		single word		x AND y	
setup	P	R	P	R	P	R
STT transcript	80.9	43.4	41.7	48.6	97.9	26.1
raw lattice	75.6	54.7	50.4	46.1	75.0	61.0
all approximations	75.9	55.9	49.5	45.8	75.0	58.1

5. CONCLUSION

We have examined the problem of indexing the spoken content of audio recordings. While simply indexing the speech-to-text transcripts results in poor accuracy, we have significantly improved search accuracy for "web-search style" (AND/phrase) queries by making correct use of speech-recognition probabilities—in particular by utilizing recognition alternates and word posterior probabilities (confidence scores) based on word lattices.

We have further presented an end-to-end approach to doing so with *standard full-text indexers*, despite the fact that by design text indexers cannot handle probabilities and unaligned alternates. We presented a sequence of approximations that transform the numeric lattice-matching problem into a symbolic text-based one that can be implemented by a commercial full-text indexer (Microsoft SQL Server 2005).

Experiments on a 170-hour lecture set have shown a relative accuracy improvement of 30-130% compared to indexing linear text.

6. REFERENCES

[1] M. Padmanabhan, G. Saon, J. Huang, B. Kingsbury, and L. Mangu, Automatic Speech Recognition Performance on a Voicemail Transcription Task. *IEEE Trans. on Speech and Audio Processing, Vol. 10, No. 7, 2002*.

[2] G. Evermann, H. Y. Chan, M. J. F. Gales, B. Jia, X. Liu, D. Mrva, K. C. Sim, L. Wang, P. C. Woodland, K. Yu, Development of the 2004 CU-HTK English CTS Systems Using More Than Two Thousand Hours of Data. *Proc. Fall 2004 Rich Transcription Workshop (RT-04), 2004*.

[3] J. Glass, T. J. Hazen, L. Hetherington, C. Wang, Analysis and Processing of Lecture Audio data: Preliminary investigation. *Proc. HLT-NAACL'2004 Workshop: Interdisciplinary Approaches to Speech Indexing and Retrieval*, Boston, 2004.

[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] T. Kawahara, C. H. Lee, B. H. Juang, Combining key-phrase detection and subword-based verification for flexible speech understanding, *Proc. ICASSP'1997*, Munich,1997.

[6] P. Yu, K. J. Chen, C. Y. Ma, F. Seide, Vocabulary-Independent Indexing of Spontaneous Speech, IEEE Transactions on Speech and Audio Processing, Vol.13, No.5.

[7] P. Yu, F. Seide, A hybrid word / phoneme-based approach for improved vocabulary-independent search in spontaneous speech. *Proc. ICLSP'04*, Jeju, 2004.

[8] M. Saraclar, R. Sproat, Lattice-based search for spoken utterance retrieval. *Proc. HLT'2004*, Boston, 2004.

[9] C. Chelba and A. Acero, Position specific posterior lattices for indexing speech. *Proc. ACL'2005*, Ann Arbor, 2005.

[10] Z. Y. Zhou, P. Yu, C. Chelba, F. Seide, Towards Spoken- Document Retrieval for the Internet: Lattice Indexing For Large-Scale Web-Search Architectures. *Proc. HLT'06*, 2006.

[11] L. Mangu, E. Brill, A. Stolcke, Finding Consensus in Speech Recognition: Word Error Minimization and Other Applications of Confusion Networks. *Computer, Speech and Language*, 14(4).

[12] F. Seide, P. Yu, Y. Shi, Towards Spoken-Document Retrieval for the Enterprise: Approximate Word-Lattice Indexing with Text Indexers. *Proc. ASRU'2007*, Kyoto.

[13] F. Wessel, R. Schlüter, and H. Ney, Using posterior word probabilities for improved speech recognition. *Proc. ICASSP'2000*, Istanbul, 2000.