A Survey on Spoken Document Indexing and Retrieval

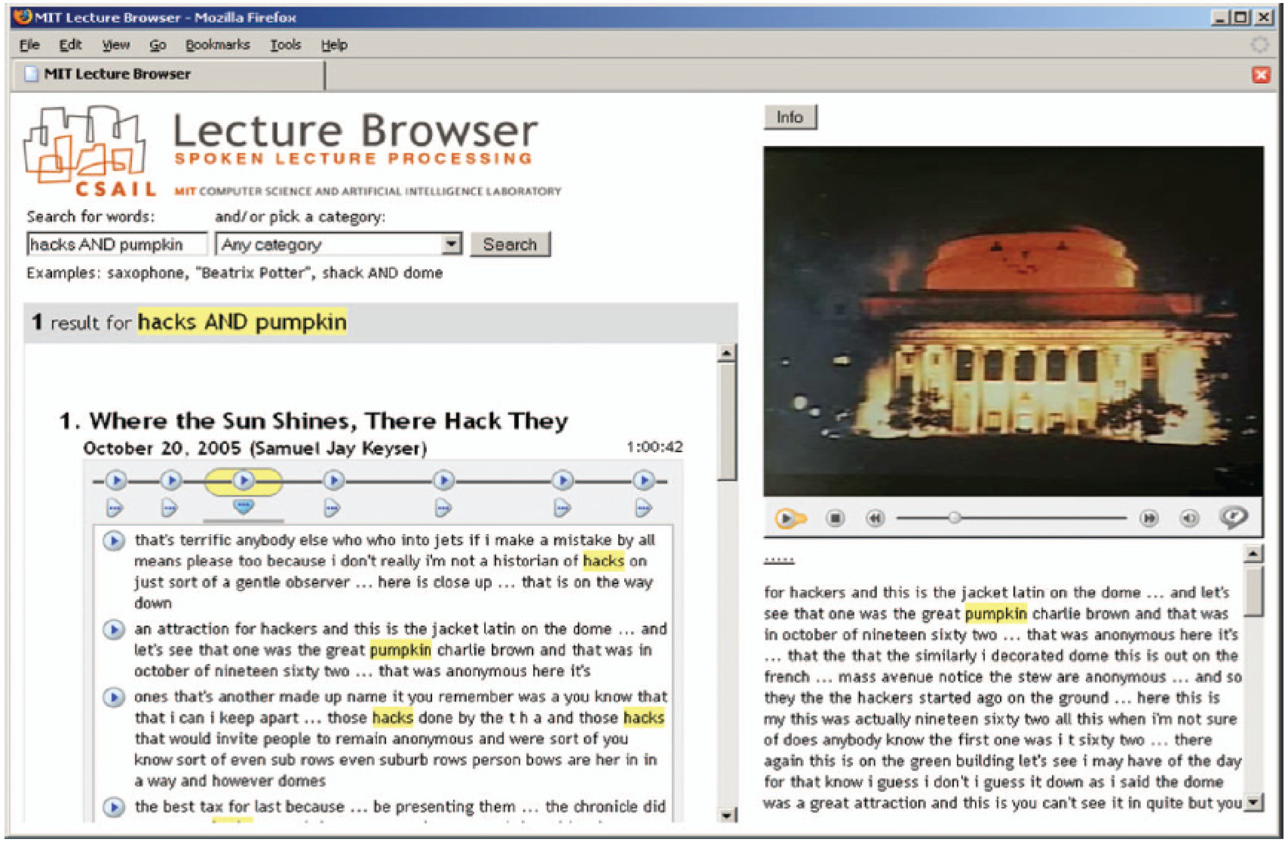
Berlin Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

Introduction (1/5)

- Ever-increasing volumes of audio-visual content have been accumulated on the Internet and in the enterprise
  - Broadcast news, lecture/meeting recordings, podcasts, etc.
  - How to efficiently search these contents just like we do for text ?
- Only a few repositories provided limited metadata or manual annotations (YouTube, Google, Yahoo, Apple, etc.)
  - Most of these audio-visual materials are still raw data and difficult to search and brows through
  - Manual transcription of the associated spoken documents (audio parts) is expensive in terms of time and cost
    - Even with automatic transcripts from automatic speech recognition (ASR) results, they are still difficult to scan through
      - Lack of structure, punctuations, sentence boundaries, etc.

Introduction (2/5)

- Many prototype systems have been built for retrieval and browsing of lecture or meeting recordings
  - E.g., MIT Lecture Browser[1]



[1] http://web.sls.csail.mit.edu/lectures

Introduction (3/5)

- There also are several research projects conducting on related spoken document processing tasks, e.g.,
  - Rich Transcription Project[2] in the United States (2002-)
    - Creation of recognition technologies that will produce transcriptions which are more readable by humans and more useful for machines
  - TC-STAR Project[3] (Technology and Corpora for Speech to Speech Translation) in Europe (2004-2007)
    - Translation of speeches recorded at European Parliament, between Spanish and English, and of broadcast news by Voice of America, from Mandarin to English
  - Spontaneous Speech Corpus and Processing Project in Japan (1999-2004)
    - 700 hours of lectures, presentations, and news commentaries
    - Automatic transcription, analysis (tagging), retrieval and summarization of spoken documents

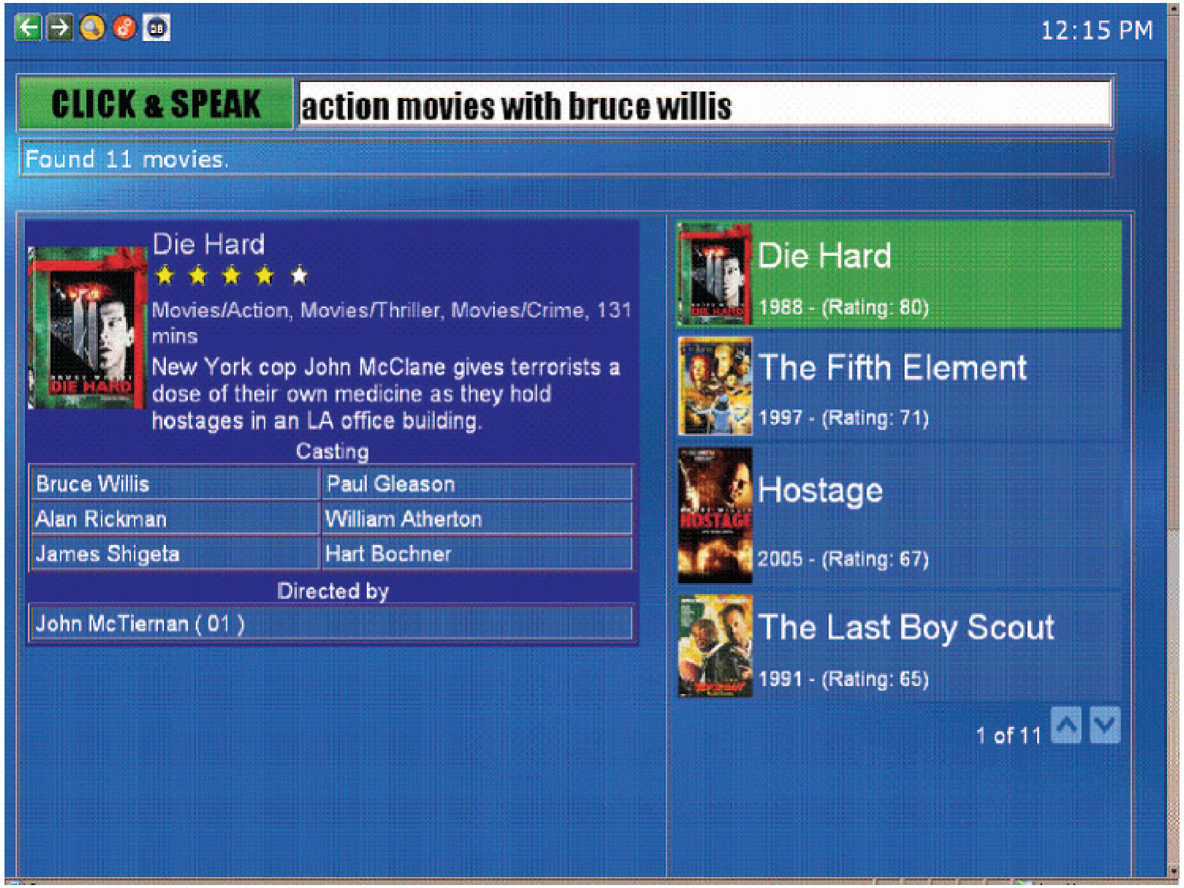
[2] http://www.nist.gov/speech/tests/rt/ [3] http://www.tc-star.org

Introduction (4/5)

- TC-STAR Demo System[3]

Introduction (5/5)

- A Prototype System Using Multimodal Voice Search for IPTV Movie-on-Demand[4]

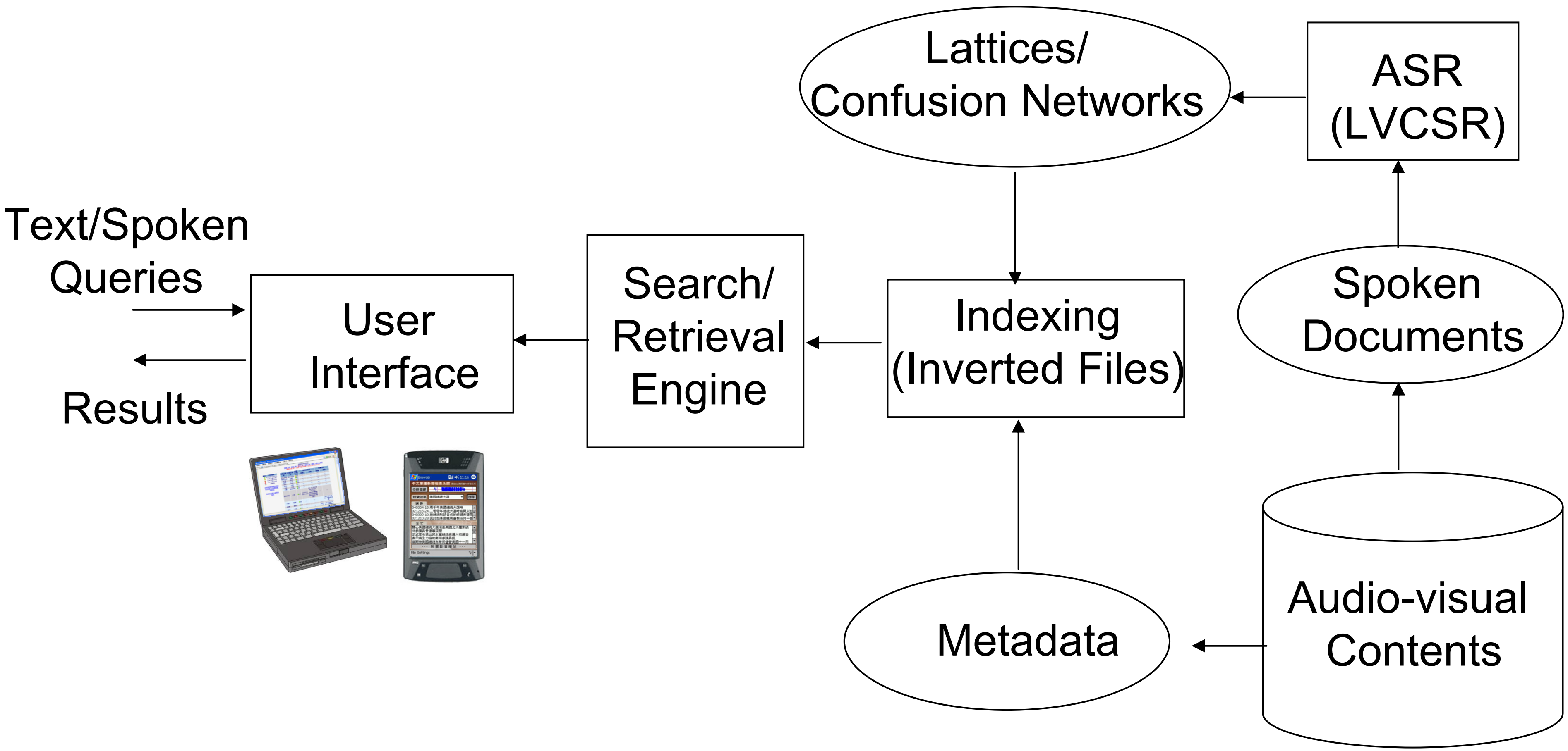


[4] Cf. [R7] Speech and Language Processing over the Web. IEEE Signal Processing Magazine 25 (3), May 2008

Evaluations of the Rich Transcription Project

- GALE (Global Autonomous Language Exploitation) Translation: 2006 – present
  - Translates language data from a input source language (either Arabic or Chinese, in audio or text) into a target one (English in text).
- Spoken Term Detection: 2006 – present
  - Facilitate research and development of technology for finding short word sequences rapidly and accurately in large heterogeneous audio archives (three languages: Arabic, English, and Mandarin)
- TRECVid Event Detection: 2008 –
- Language Recognition Evaluation: 1996 –
- …

A Typical Scenario for Spoken Document Search



Categorization of Spoken Document Search Tasks

- Spoken Document Retrieval (SDR)
  - Find spoken documents that are "relevant" to a given query
  - Queries usually are very long topic descriptions
  - Exploit LVCSR and text IR technologies
  - SDR is already regarded as a "solved" problem, especially for broadcast news (even with WER of more than 30%, retrieval using automatic transcripts are comparable to that using reference transcripts)
- Spoken Term Detection (STD)
  - Much like Web-style search
  - Queries are usually short (1-3 words), and find the "matched" documents where all query terms should be present
  - Then, relevance ranking are performed on the "matched" documents
  - Have drawn much attention recently in the speech processing community

TREC SDR Evaluation Plan

- A Series of SDR tracks conducted during 1996-2000 (TREC-6 ~ TREC-9)
  - Focus on using broadcast news from various sources: Voice of America, CNN, ABC, PRI, etc., comprising more than 5 hundred hours of speech (≧20,000 manually segmented documents, 250 words per document on average)
  - The queries are long and stated in plain English (e.g., a text news story) rather then using the keyword (Web) search scenario

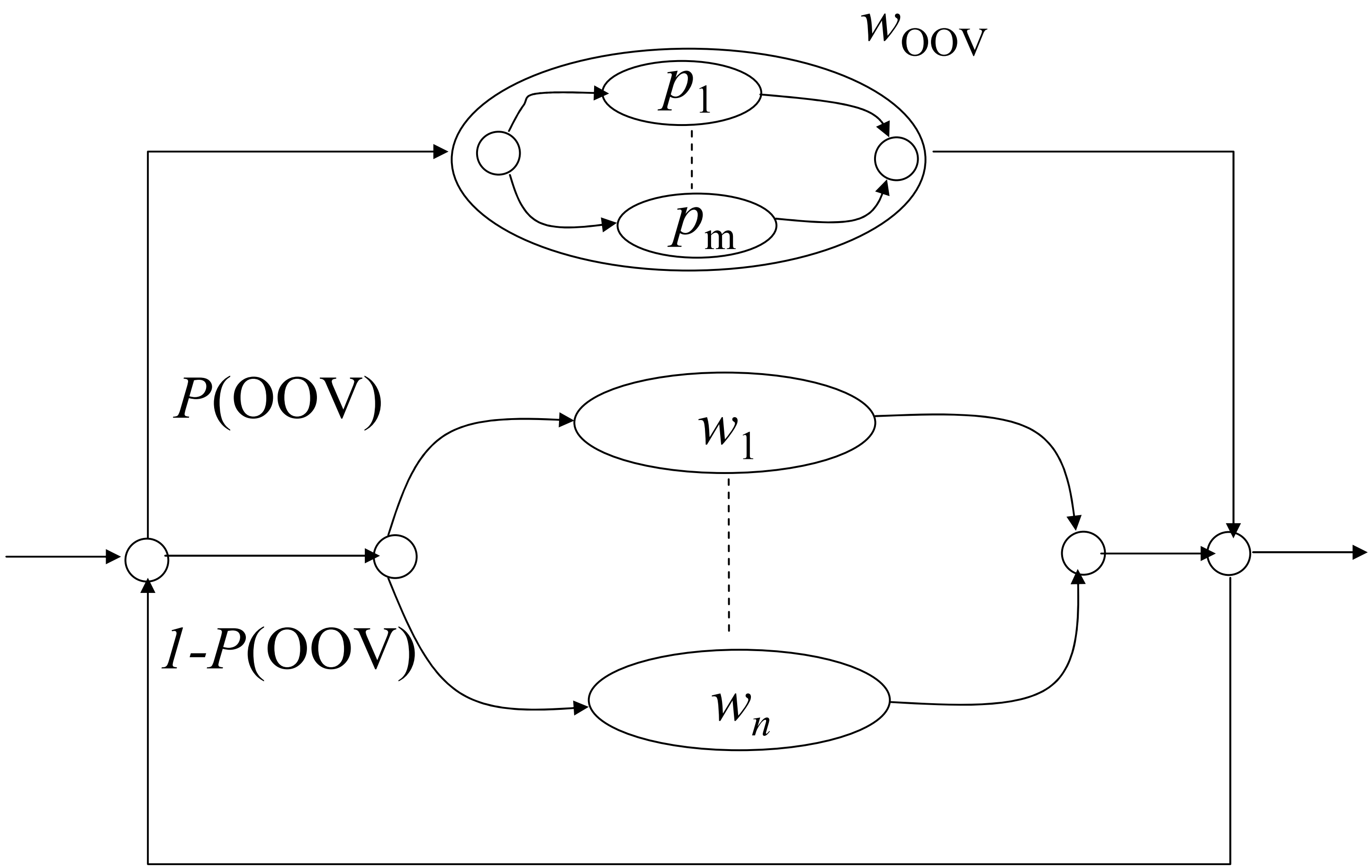
- Findings
  - Retrieval performance is quite flat with ASR WER variations in the range of 10~35% (roughly ≦5% degradation in performance in comparison with the "approximately" manual transcriptions)
  - SDR of broadcast news speech has been thought of as "a successful story"

Types of ASR Transcriptions (1/2)

- Word Sequences Produced by LVCSR
  - More accurate for audio indexing
  - Faced with the “OOV-word” problems (query terms are often less-frequent topic-specific words)
  - Tend to have lower recall
- Phonetic-Unit (or subword) Sequences Produced by Phone Recognizer
  - Bypass the “OOV-word” problems by locating spoken documents containing the phonetic sequences that match the pronunciations of the query words
  - Complicate the post-processing of the spoken documents for other IR-related applications
  - Tend to have higher recall at the expanse of lower precision
- Hybrid Approach Blending Word and Phonetic Information

Types of ASR Transcriptions (2/2)

- Represent the OOV region by a network of phonetic units

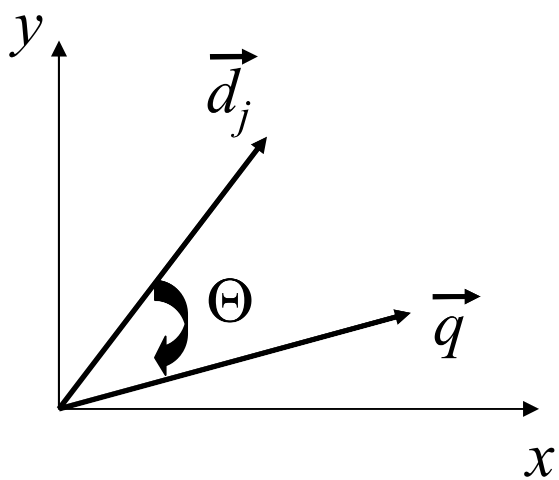


Vector Model

SMART system
Cornell U., 1968

- Also called Vector Space Model (VSM)
- Use vector representation for the query and each document
  - Each dimension represents the weight (e.g., TF-IDF score) associated with a word in the query or document
  - Use cosine measure to estimate the degree of similarity (relevance) between a query and a document

$$
\begin{aligned}
& sim(d_j, q) \\
& = \text{cosine}(\Theta) \\
& = \frac{\vec{d}_j \bullet \vec{q}}{|\vec{d}_j| \times |\vec{q}|} \\
& = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}
\end{aligned}
$$



TF-IDF Score

- A good weight must take into account two effects:
  - Quantification of intra-document contents (similarity)
    - TF factor, the term frequency for a word i within a document j
    - High term frequency is needed

$$TF_i = (1 + \log(freq_{i,j})) \text{ or } TF_i = \frac{freq_{i,j}}{\max_l freq_{l,j}}$$

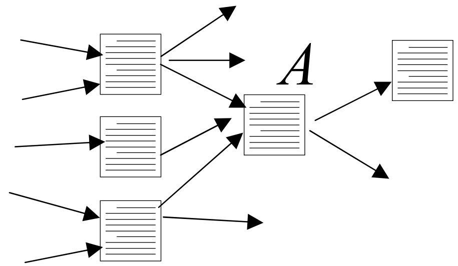
  - Quantification of inter-documents separation (dissimilarity)
    - IDF factor, the inverse document frequency
    - Low document frequency (or high IDF) is preferred

$$IDF_i = \log \frac{N}{n_i}$$

Sparck Jones, 1972

Page Rank

- Query-independent, derived from the WWW connective graph
- Notations
  - A page A has pages $T_1 \ldots T_n$ which point to it (citations)
  - d range from 0~1, a damping factor (Google sets to be 0.85)
  - C(A)：Number of links going out of page A



- PageRank of a page A

$$PR(A) = (1-d) + d\left(\frac{PR(T_1)}{C(T_1)} + \cdots + \frac{PR(T_n)}{C(T_n)}\right)$$

  - PageRank of each page is randomly assigned at the initial iteration and its value tends to be saturated through iterations

- A page with a high PageRank value
  - Many pages pointing to it (old pages tend to have high scores)
  - Or, there are some pages that point to it and have high PageRank values

Early Google Ad-Hoc Approach

Brin and Page, 1998

- For each given query term q_i , one retrieves the list of hits corresponding to q_i in document D
  - Hits can be of various types depending on the context in which the hit occurred: title, anchor text, metadata, etc.
    - Hit: an occurrence of a query word in a document
  - Each type of hit has its own type-weight and the type-weights are indexed by type
- This approach considers both word proximity and context information
- The resulting score is then combined with PageRank in a final relevance score

Evaluation Metrics

- SDR and STD
  - Recall
  - Precision
  - F-measure (a harmonic mean of recall and precision)
  - *R*-precision
  - Precision at *N* document cutoff level
  - Mean Average Precision (MAP)
  - Actual Term-Weighted Value (ATWV)
  - …
- ASR
  - WER
  - Lattice WER
  - OOV Rate
  - Query OOV Rate
  - …

Inverted Files (1/4)

- A word-oriented mechanism for indexing a text collection in order to speed up the searching task
  - Two elements:
    - A vector containing all the distinct words (called vocabulary) in the text collection
      - The space required for the vocabulary is rather small:

        $\sim O(n^{\beta})$, n: the text size, $0<\beta<1$ (Heaps' law)
    - For each vocabulary word, a list of all docs (identified by doc number in ascending order) in which that word occurs (hits)
      - Space overhead: 30~40% of the text size (for text position addressing)
- Distinction between inverted file and inverted list
  - **Inverted file**: occurrence points to documents or file names (identities)
  - **Inverted list**: occurrence points to word positions

Inverted Files (2/4)

- **Example**

1	6	9	11	17	19	24	28	33	40	46	50	55	60
This	is	a	text.	A	text	has	many	words.	Words	are	made	from	letters.

Text

Vocabulary	Occurrences
letter	60 ...
made	50 ...
many	28 ...
Text	11, 19, ...
word	33, 40, ...
....	....

<u>An inverted list</u>

Each element in a list points to a text position

<u>An inverted file</u>

Each element in a list points to a doc number

difference: indexing granularity

Inverted Files: Addressing Granularity

- Text (word/character) positions (full inverted indices)

  (document id, position)

- Documents
  - All the occurrences of a word inside a document are collapsed to one reference

- (Logical) blocks
  - The blocks can be of fixed or different size
  - All the occurrences of a word inside a single block are collapsed to one reference
  - Space overhead: ~5% of the text size for a large collection

Inverted Files: Some Statistics

- Size of an inverted file as approximate percentages of the size of the text collection

	Index	Small Collection (1 Mb)		Medium Collection (200 Mb)		Large Collection (2 Gb)	
4 bytes/pointer	Addressing Words	45%	73%	36%	64%	35%	63%
1,2,3 bytes/pointer	Addressing Documents	19%	26%	18%	32%	26%	47%
2 bytes/pointer	Addressing 64K blocks	27%	41%	18%	32%	5%	9%
1 byte/pointer	Addressing 256 blocks	18%	25%	1.7%	2.4%	0.5%	0.7%

Stopwords are removed   Stopwords are indexed

Inverted Files (3/4)

- Document Addressing
  - Assume that the vocabulary (control dictionary) can be kept in main memory. Assign a sequential word number to each word
  - Scan the text database and output to a temporary file containing the record number and its word number
  - Sort the temporary file by word number and use record number as a minor sorting field
  - Compact the sorted file by removing the word number. During this compaction, build the inverted list from the end points of each word. This compacted file (postings file) becomes the main index

…..
d_5 w_3
d_5 w_{100}
d_5 w_{1050}
…..
d_9 w_{12}
…..

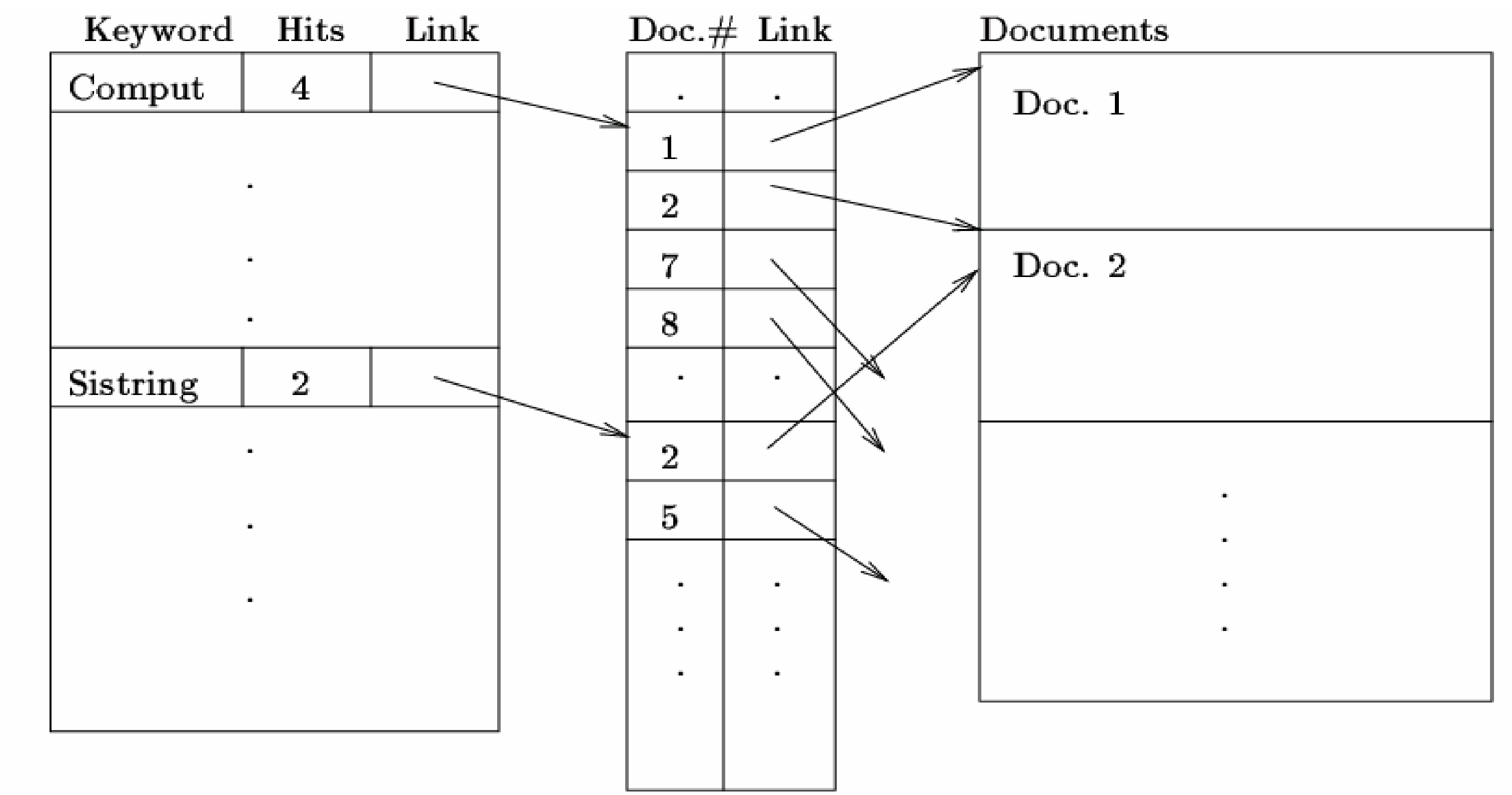
Inverted Files (4/4)

- Document addressing (count.)

Vocabulary

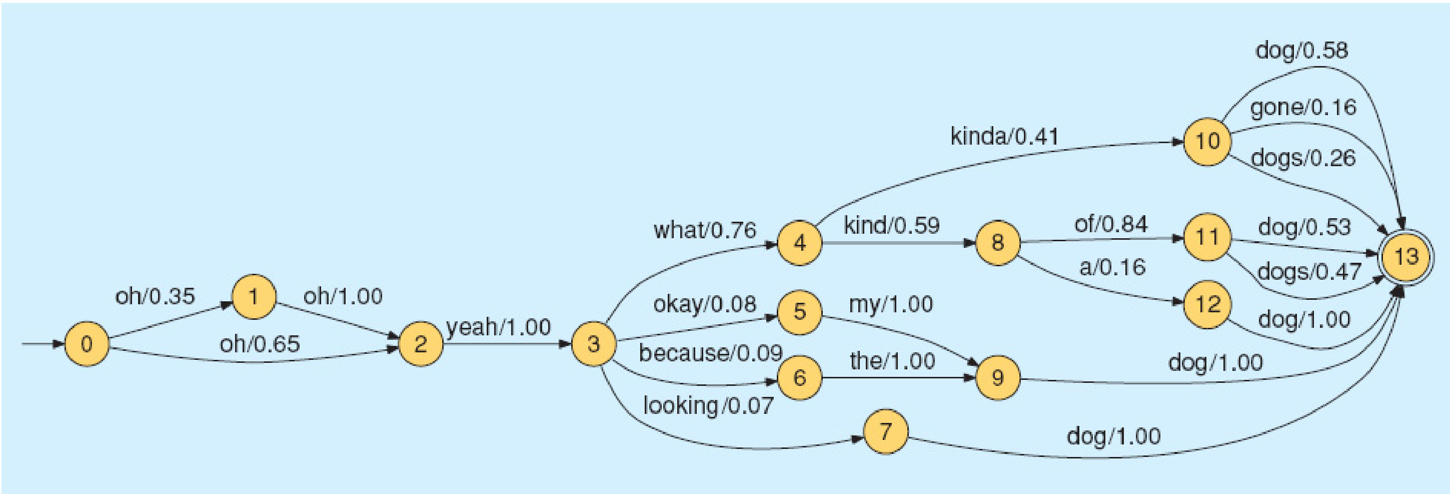
Occurrences (posting file)

Physical Documents



Indexing: 1-bset Sequences vs. Lattices (1/3)

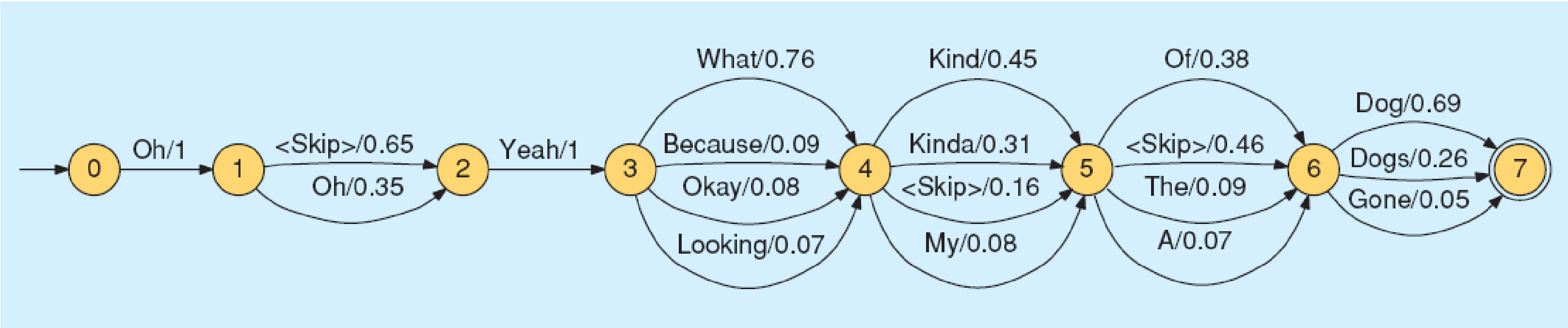
- Use of 1-best ASR output as the transcription to be indexed is suboptimal due to the high WER, which is likely to lead to low recall
- ASR lattices do provide much better WER, but the position information is not readily available (uncertainty of word occurrences) ? (document id, position, posterior prob.)
- An example ASR Lattice[5]



[5] Cf. [R3], Retrieval and Browsing of Spoken Content. IEEE Signal Processing Magazine 25 (3)

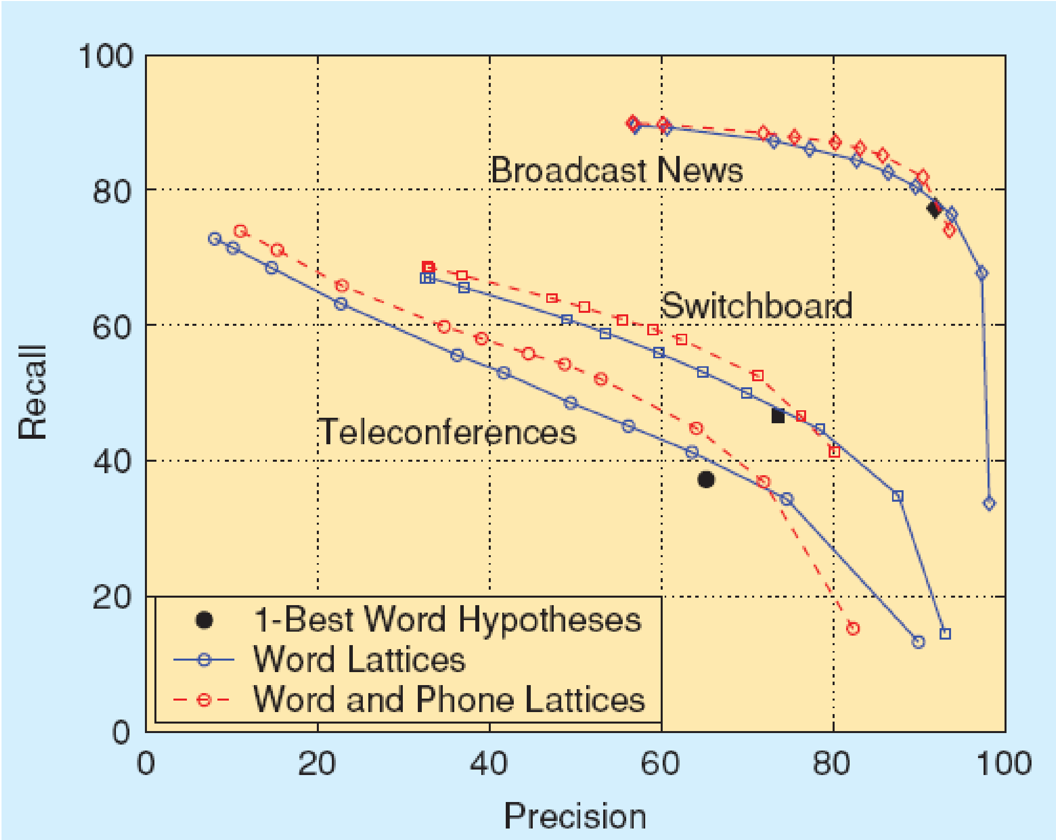
Indexing: 1-bset Sequences vs. Lattices (2/3)

- Confusion/Consensus Networks (CN, also called “Sausages”) derived from the Lattice[5]



Indexing: 1-bset Sequences vs. Lattices (3/3)

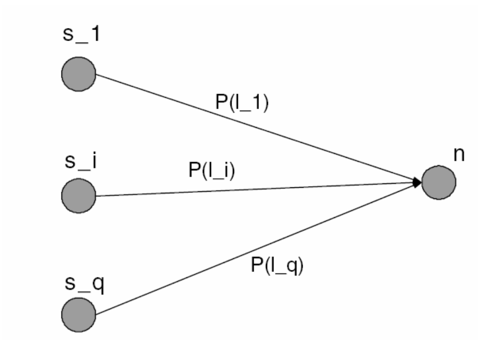
- Comparison between indexing with 1-bset sequences and lattices[5]



Position-Specific Posterior Probability Lattices (1/6)

- Position information is crucial for being able to evaluate proximity when assigning a relevance score to a given document
- Soft-hit: indexing of the occurrence of each word *n* in the lattice[6]

$$\alpha_n[l] = \sum_{\pi:end(\pi)=n, length(\pi)=l} P(\pi)$$



position/length along the partial path traversed

- A modified forward procedure

$$\alpha_n[l+1] = \sum_{i=1}^{q} \alpha_{s\_i}\left[l + \delta\left(l_{s\_i}, \varepsilon\right)\right] P\left(l_{s\_i}\right)$$

$$\log P\left(l_{s\_i}\right) = \eta \cdot \left(\frac{1}{\kappa} \log P_{AM}\left(l_{s\_i}\right) + \log P_{LM}\left(l_{s\_i}\right) - \frac{1}{\kappa} \log P_{IP} \right)$$

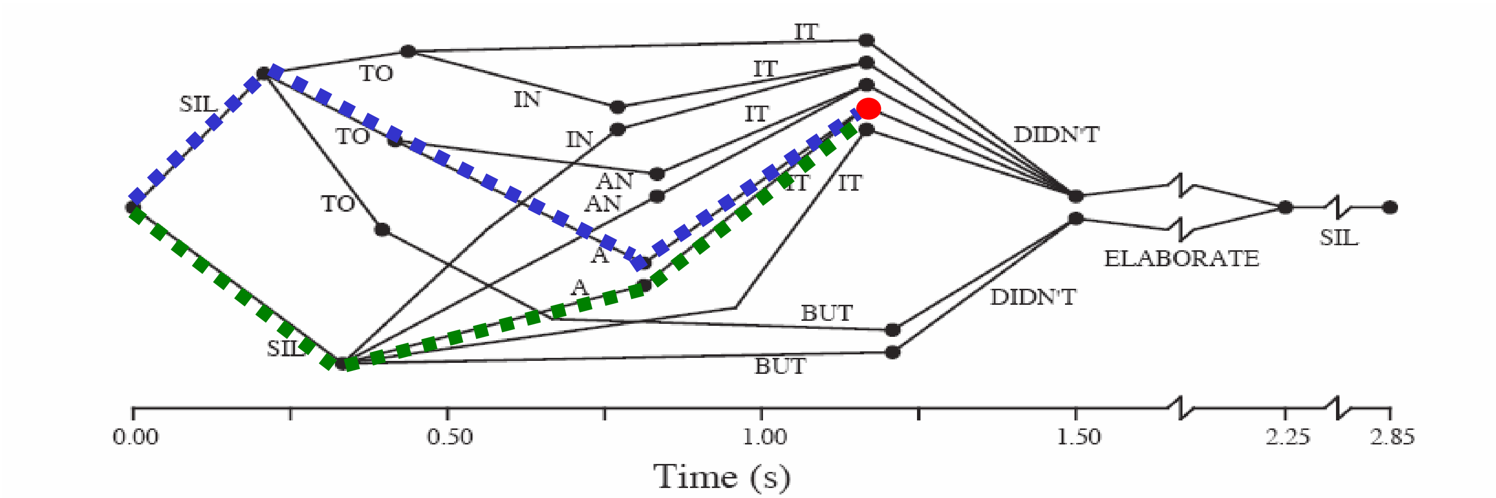
[6] Cf. [R4], Soft Indexing of Speech Content for Search in Spoken Documents. *Computer Speech & Language* 21(3), 2007

Position-Specific Posterior Probability Lattices (2/6)

- The backward procedure follows the original definition

- The posterior probability for a word w at position l is expressed as

$$P(w,l \mid LAT) = \sum_{n \;\; s.t. \alpha_n[l]\cdot\beta_n > 0} \frac{\alpha_n[l]\cdot\beta_n}{\beta_{start}} \delta(w, word(n))$$



Position-Specific Posterior Probability Lattices (3/6)

- An example Position-Specific Posterior Probability Lattice (PSPL)[5]

0	1	2	3	4	5	6	7
Oh 1.0	**Yeah** .65	**What** .46	**Kind** .27	Dog .26	EOS .34	**EOS** .44	EOS .16
—	Oh .35	Yeah .35	What .27	**Of** .23	**Dog** .29	Dog .09	—
	—	Because .06	Kinda .19	Kind .16	Dogs .13	Dogs .06	
		Okay .05	The .06	Kinda .11	Of .13	—	
		Looking .05	My .05	Dogs .05	A .03		
		—	Dog .05	EOS .05	Gone .02		
			....	....	—		

Position-Specific Posterior Probability Lattices (5/6)

- A document *D* can be first divided into several segments
- Then Calculate the expected count of a given query term according to the PSPL probability distribution
  for each segment *s* of document *D*

$$Q = q_1, q_2, \cdots, q_M$$

unigram matching

$$\begin{cases} S(D, q_i) = \log\Big[1 + \sum_s \sum_l P(w_l(s) = q_i \mid D)\Big] \\ S_{1-\mathrm{gram}}(D, Q) = \sum_{i=1}^{M} S(D, q_i) \end{cases}$$

N-gram (or phrase) matching

$$\begin{cases} S(D, q_i \ldots q_{i+N-1}) = \log\Big[1 + \sum_s \sum_l \prod_{r=0}^{N-1} P(w_{l+r}(s) = q_{i+r} \mid D)\Big] \\ S_{N-\mathrm{gram}}(D, Q) = \sum_{i=1}^{M-N+1} S(D, q_i \ldots q_{i+N-1}) \end{cases}$$

$$S(D, Q) = \sum_{N=1}^{M} \lambda_N \cdot S_{N-\mathrm{gram}}(D, Q)$$

Position-Specific Posterior Probability Lattices (5/6)

- "Relative Pruning" of PSPL lattices
  - For a given position bin *l*, the relative pruning first finds the most likely word entry given by

$$w_l^* = \arg\max_{w \in V} p(w_l(s) = w \mid D)$$

  - Word entries have test values lower than or equal to the threshold are retained in the position bin of the PSPL lattice

$$W_l = \{w \in V : \log \frac{P(w_l(s) = w_l^* \mid D)}{P(w_l(s) = w \mid D)} \leq \tau_r\} \qquad \tau_r \in [0, \infty)$$

    - As the threshold decreased to zero, the pruned PSPL is reduced "approximately" to the 1-best output
    - The posterior probability of words (bin entries) W_l in each are renormalized

Position-Specific Posterior Probability Lattices (6/6)

- "Absolute Pruning" of PSPL lattices
  - Retrain the word entries in each bin *l* that have log posterior probability higher than an absolute threshold

$$\overline{P}\left(w_k(s) = q \middle| D\right) = P\left(w_k(s) = q \middle| D\right) \cdot 1_{\left\{\log P\left(w_k(s) = q \middle| D\right) \geq \tau_{abs}\right\}}$$

$$\tau_r \in \left(-\infty, 0\right]$$

"absolute Pruning" can be performed at query run-time

Experiments on Indexing Using PSPL Lattices[5] (1/6)

- Corpus: *i*Campus Corpus (169 h, recorded using lapel microphone)
  - 20 Introduction to Computer Programming Lectures (21.7 h)
  - 35 Linear Algebra Lectures (27.7 h)
  - 35 Electro-magnetic Physics Lectures (29.1 h)
  - 79 Assorted MIT World seminars covering a wide variety of topics (89.9 h)

- Two Kinds of Lattices
  - 3-gram ASR lattices
  - PSPL lattices

Experiments on Indexing Using PSPL Lattices (2/6)

- Analysis of (116) Test Queries
  - Query out-of-vocabulary rate (Q-OOV) was 5.2%
  - The average query length was 1.97 words.
  - The queries containing OOV words were removed from the test

- Three Kinds of Evaluation Metrics
  - Standard Precision/Recall and Precision@N documents
  - Mean Average Precision (MAP)
  - *R*-precision (*R*=number of relevant documents for the query)

Experiments on Indexing Using PSPL Lattices (3/6)

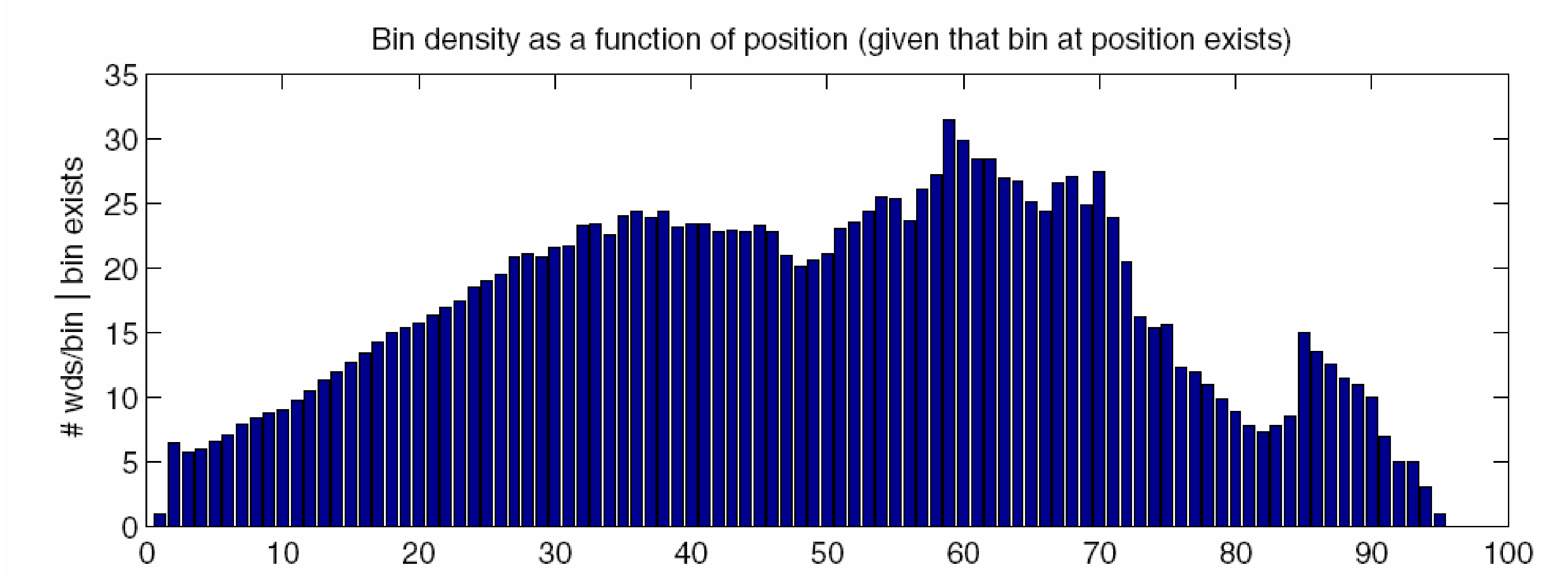


Table 1
Comparison between 3-gram and PSPL lattices for lecture L01 of the iCampus corpus: node and link density, 1-best and ORACLE WER, size on disk

Lattice type	3-gram	PSPL
Size on disk (MB)	11.3	3.2
Link density	16.3	14.6
Node density	7.4	1.1
1-best WER (%)	44.7	45
ORACLE WER (%)	26.4	21.7

Experiments on Indexing Using PSPL Lattices (4/6)

- 17% MAP relative improvement achieved by using the ASR lattice with respect to the one-best ASR result

Table 2
Retrieval performance on indexes built from transcript, ASR 1-best and PSPL lattices, respectively

	trans	1-best	lat
# docs retrieved	1411	3206	4971
# relevant docs	1416	1416	1416
# rel retrieved	1411	1088	1301
MAP	0.99	0.53	0.62
R-precision	0.99	0.53	0.58

manual transcriptions

ASR results

Experiments on Indexing Using PSPL Lattices (5/6)

- Experiments on various PSPL probability assigments

Table 3
Retrieval performance on indexes built from PSPL lattices under various PSPL probability assignments

	lat	raw	noP	unif	1-best
# docs retrieved	4971	4971	4971	4971	3206
# relevant docs	1416	1416	1416	1416	1416
# rel retrieved	1301	1301	1301	1301	1088
MAP	0.62	0.60	0.47	0.57	0.53
R-precision	0.58	0.56	0.42	0.52	0.53

without flattening of word prob.

without using posterior prob. (hard-index, more than one word occurs at the same position)

Uniform posterior prob. 1.0/#entries in each position

$$\log P(l_{s\_i}) = \eta \cdot \left(\frac{1}{\kappa} \log P_{AM}(l_{s\_i}) + \log P_{LM}(l_{s\_i}) - \frac{1}{\kappa} \log P_{IP} \right)$$

Experiments on Indexing Using PSPL Lattices (6/6)

Table 5

Retrieval performance on indexes built from pruned PSPL lattices using the relative thresholding technique, along with index size; 0 threshold represents the result for the 1-best approach

τ_r Pruning threshold	MAP	R-precision	Index size (MB)
0.0 (1-best)	0.53	0.54	16
0.1	0.54	0.55	21
0.2	0.55	0.56	26
0.5	0.56	0.57	40
1.0	0.58	0.58	62
2.0	0.61	0.59	110
5.0	0.62	0.57	300
10.0	0.62	0.57	460
1000000	0.62	0.57	540

$$W_l = \{w \in V : \log \frac{P(w_l(s) = w_l^* \mid D)}{P(w_l(s) = w \mid D)} \le \tau_r\}$$

Discussions

- The use of PSPL and CN has been an active area of research for robust audio indexing and retrieval
- Most of the research efforts were devoted to spoken document retrieval using "text" queries but not "spoken" queries
- Simply finding "matched" spoken terms vs. retrieving "relevant" documents in response to user information needs

References

[1] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval, Addison Wesley Longman, 1999

[2] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008

[3] C. Chelba, T.J. Hazen, and M. Saraclar. Retrieval and Browsing of Spoken Content. *IEEE Signal Processing Magazine* 25 (3), May 2008

[4] C. Chelba, J. Silva and A. Acero. Soft Indexing of Speech Content for Search in Spoken Documents. *Computer Speech & Language* 21(3), 2007

[5] C. Chelba and T. J. Hazen. Automatic Spoken Document Processing for Retrieval and Browsing. Tutorial, *ICASSP 2007*

[6] F. Casacuberta, M. Federico, H. Ney, and E. Vidal. Recent Efforts in Spoken Language Translation. *IEEE Signal Processing Magazine* 25 (3), May 2008

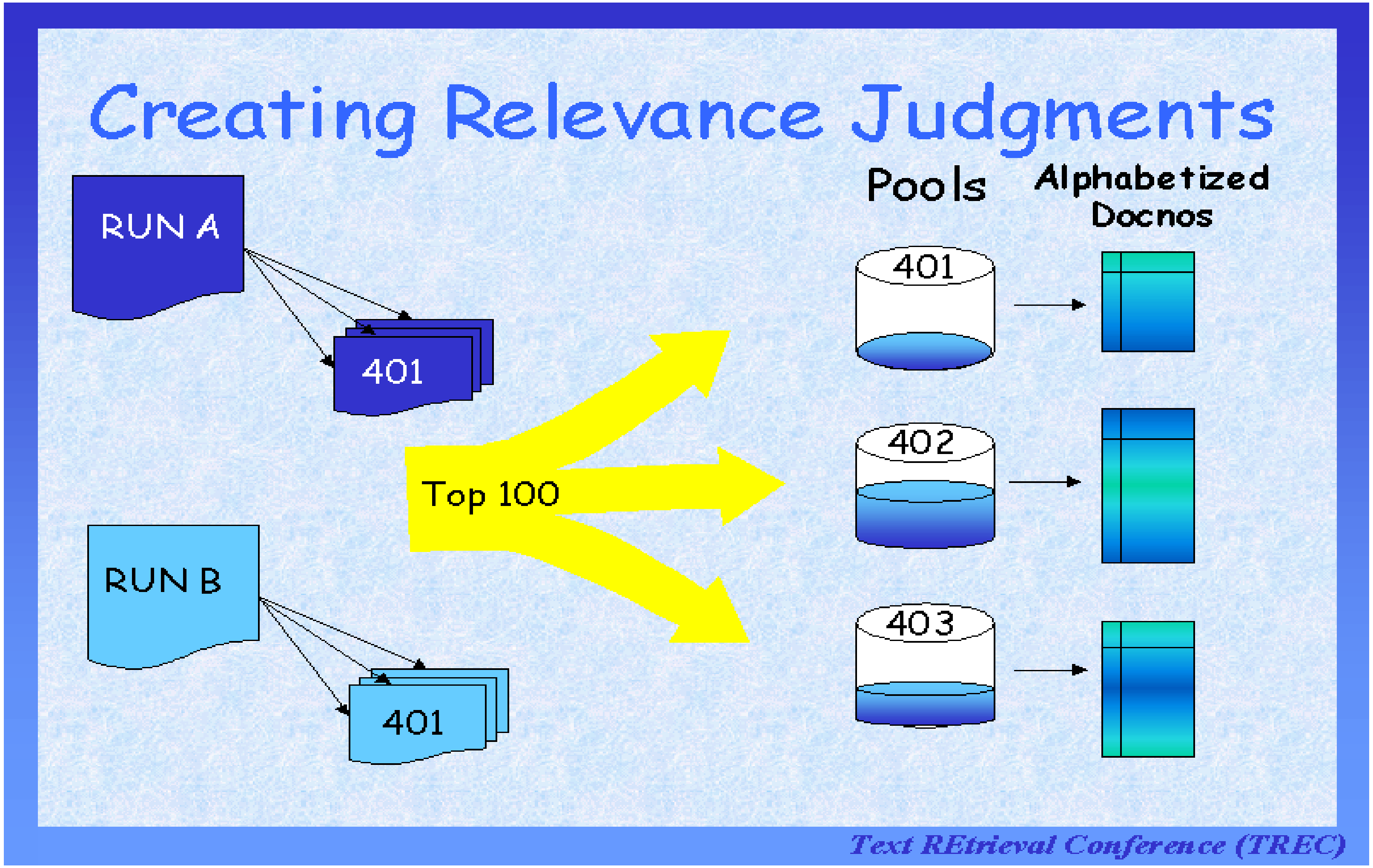
[7] M. Gilbert and J. Feng. Speech and Language Processing over the Web. *IEEE Signal Processing Magazine* 25 (3), May 2008

Appendix A: TREC-Creating Relevance Judgments (1/3)

- For each topic (example information request)
  - Each participating systems created top *K* (e.g. *K*=100) docs and put in a pool
  - Human "assessors" decide on the relevance of each doc

- The so-called "**pooling method**"
  - Two assumptions
    - Vast majority of relevant docs is collected in the assembled pool
    - Docs not in the pool were considered to be irrelevant
  - Such assumptions have been verified to be accurate !

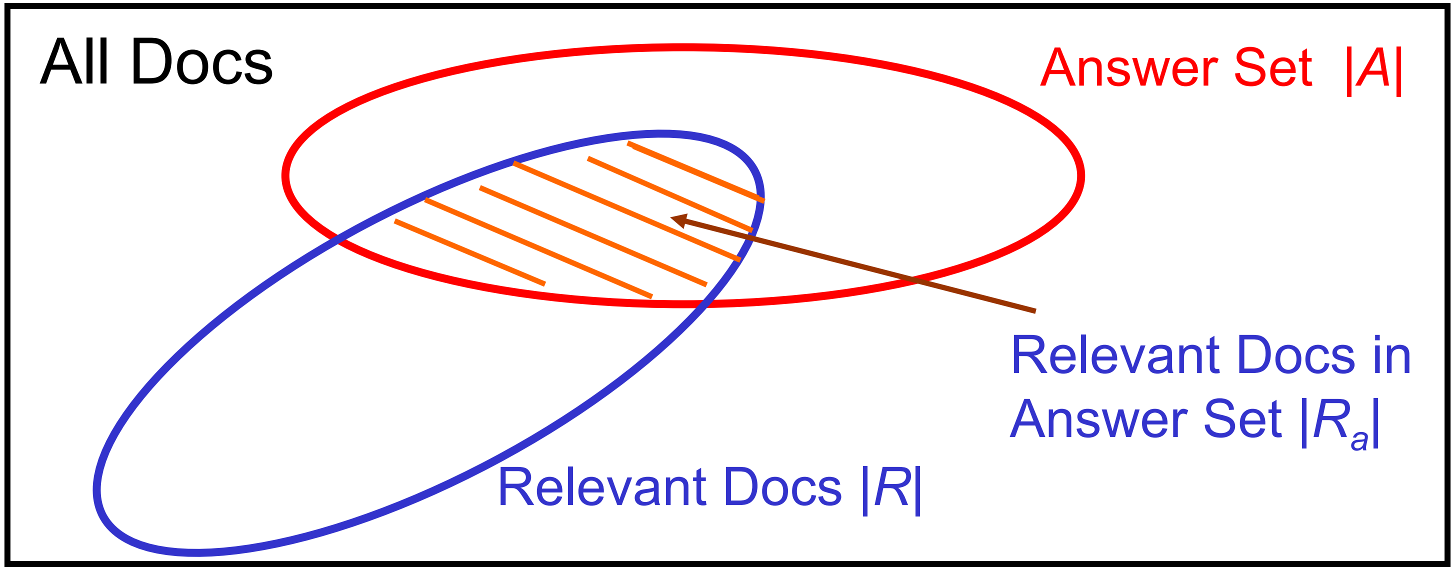
Appendix A: TREC-Creating Relevance Judgments (2/3)



Appendix A: TREC-Creating Relevance Judgments (3/3)

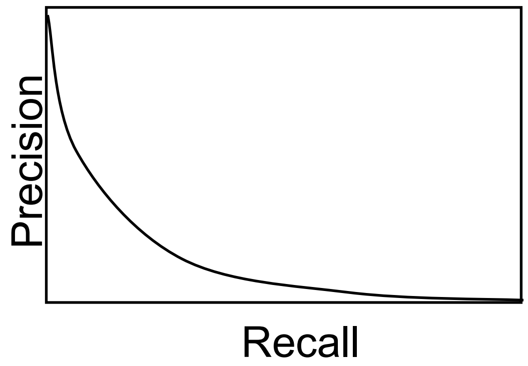
Appendix B: Recall and Precision (1/2)

- Recall ($\dfrac{|R_a|}{|R|}$)
  - The fraction of the relevant documents which has been retrieved
- Precision ($\dfrac{|R_a|}{|A|}$)
  - The fraction of the retrieved documents which is relevant



Appendix B: Recall and Precision (2/2)

- Recall and precision assume that all the documents in the answer set have been examined (or seen)
- However, the user is not usually presented with all the documents in the answer set A at once
  - Sort the document in A according to a degree of relevance
  - Examine the ranked list starting from the top document (increasing in recall, but decreasing in precision)
    - Varying of recall and precision measures
    - A precision versus recall curve can be plotted

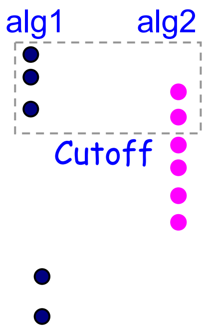


Appendix C: Mean Average Precision (mAP) (1/2)

- Average Precision at Seen Relevant Documents
  - A single value summary of the ranking by averaging the precision figures obtained after each new relevant doc is observed

1. d_{123} • (P=1.0)	6. d_{9} • (P=0.5)	11. d_{38}
2. d_{84}	7. d_{511}	12. d_{48}
3. d_{56} • (P=0.66)	8. d_{129}	13. d_{250}
4. d_{6}	9. d_{187}	14. d_{113}
5. d_{8}	10. d_{25} • (P=0.4)	15. d_{3} • (P=0.3)

(1.0+0.66+0.5+0.4+0.3)/5=0.57



  - It favors systems which retrieve relevant docs quickly (early in the ranking)
  - But when doc cutoff levels were used
    - An algorithm might present a good average precision at seen relevant docs but have a poor performance in terms of overall recall

Appendix C: Mean Average Precision (*m*AP) (2/2)

- Averaged at relevant docs and across queries
  - E.g. relevant docs ranked at 1, 5, 10, precisions are 1/1, 2/5, 3/10,
    - non-interpolated average precision (or called *Average Precision at Seen Relevant Documents* in textbook) =(1/1+2/5+3/10)/3
  - Mean average Precision (*m*AP)

$$\frac{1}{|Q|}\sum_{q=1}^{|Q|}(\text{non}-\text{interpolated average precision})_q$$

- Widely used in IR performance evaluation

Appendix C: Actual Term Weighted Value (2/2)

- Actual Term Weighted Value (ATWV) is a metric defined in the NIST Spoken Detection (STD) 2006 evaluation plan

$$\mathrm{ATWV} = 1 - \frac{1}{Q}\sum_{q=1}^{Q}\left\{P_{\mathrm{miss}}(q) + \beta P_{\mathrm{FA}}(q)\right\}$$

$$P_{\mathrm{miss}}(q) = 1 - \frac{C(q)}{R(q)} \qquad P_{\mathrm{FA}}(q) = \frac{A(q) - C(q)}{n_{tps} \times T_{\mathrm{speech}} - C(q)}$$

T_{speech} = duration of speech (in sec.)

n_{tps} = number of trials per sec. of speech

$R(q)$ = total number of times examples of a specific term (phrase) q actually appears

$C(q)$ = total number of times examples of a specific term (phrase) q detected by the system tha t are actually correct

$A(q)$ = total number of times examples of a specific term (phrase) q detected by the system

β : empiricall y set parameter (e.g., 1000)