Estimating Document Frequencies in a Speech Corpus

Damianos Karakos [#*1], Mark Dredze [#†2], Ken Church [#†3], Aren Jansen [#*4], Sanjeev Khudanpur [#*5]

[#] *Human Language Technology Center of Excellence*
[*] *Department of Electrical and Computer Engineering*
[†] *Department of Computer Science*
Johns Hopkins University, Baltimore, MD
{[1]damianos, [2]mdredze, [3]Kenneth.Church, [4]aren, [5]khudanpur}@jhu.edu

***Abstract*—Inverse Document Frequency (IDF) is an important quantity in many applications, including Information Retrieval. IDF is defined in terms of document frequency, $df(w)$, the number of documents that mention w at least once. This quantity is relatively easy to compute over textual documents, but spoken documents are more challenging. This paper considers two baselines: (1) an estimate based on the 1-best ASR output and (2) an estimate based on expected term frequencies computed from the lattice. We improve over these baselines by taking advantage of repetition. Whatever the document is about is likely to be repeated, unlike ASR errors, which tend to be more random (Poisson). In addition, we find it helpful to consider an ensemble of language models. There is an opportunity for the ensemble to reduce noise, assuming that the errors across language models are relatively uncorrelated. The opportunity for improvement is larger when WER is high. This paper considers a pairing task application that could benefit from improved estimates of df. The pairing task inputs conversational sides from the English Fisher corpus and outputs estimates of which sides were from the same conversation. Better estimates of df lead to better performance on this task.**

I. Introduction

Inverse Document Frequency, $\text{IDF}(w) = -\log_2 df(w)/D$, depends on the number of documents D and the document frequency $df(w)$, the number of documents that mention w one or more times. Computing IDF is critical for many downstream applications in information retrieval. It is easy to compute document frequency over textual documents, but spoken documents are challenging. Two standard methods for doing so serve as baselines in this paper: (1) an estimate based on the 1-best ASR output and (2) an estimate based on expected term frequencies computed from the lattice. Baseline (1) is easy to implement and works well if the word error rate (WER) is low. Baseline (2) is inspired by the literature on spoken document retrieval [1].

Despite the general acceptance of these baselines as sufficient, Figure 1 shows that there is plenty of room to improve over the 1-best baseline as compared to using the transcript. The plot highlights some words with large residual errors between the baseline estimates of df and the gold standard reference, the true df as computed from the transcript. Table I shows that many of these words are acoustically short and difficult to recognize. In addition, some of them such as "TV" (television) and "cat" are highly associated with certain topics in the Fisher corpus, which can cause trouble, especially when the testing and training data are limited and not well matched (discuss different content). The columns labeled 3hr and 20hr in Table I are computed from the 1-best ASR output with 3 hours and 20 hours of training data, respectively. Not surprisingly, there is "no data like more data" (Mercer, personal communication); we also find the training set should be large and well matched to the test set. However, this may not be realistic in many applications of interest.

Consider the effectiveness of these methods for a single word: "work." Table II and Figure 2 compare the 20hr and 3hr baselines to the reference for this word. Of the $D = 284$ documents (conversational sides), there were 143 sides with no instances of "work" ($k = 0$) in the reference gold standard, and 58 sides with $k = 1$ and so on. The next two lines are the two baselines mentioned above (we again see that more training data leads to better estimates) and the bottom line shows estimates from a Poisson with $\theta = 1.36$. The Poisson is even worse than the 1-best baselines because documents are not random bags of words. Whatever the document is about is likely to be repeated, unlike ASR errors, which tend to be more random (Poisson). Chapter 14 of [2] describes a number of adaptive language models such as cache models that exploit this fact; repetition can be viewed as self-triggering [3]. Thus, repetition provides a clear opportunity for achieving improved document frequency estimates.

In this work, we consider new approaches to IDF for spoken documents by taking advantage of repetition. We use multiple language models and a log-linear model to create new df estimators, and show that these lead to improvements on an information retrieval task (document pairing).

II. Feature Extraction and Model Training

The discussion above makes it clear that one should be able to do better at estimating document frequencies using a variety of cues, such as probability of repetition. In this section we describe how we compute our various features from speech lattices, and how they are used in a log-linear framework in order to estimate the df statistics.

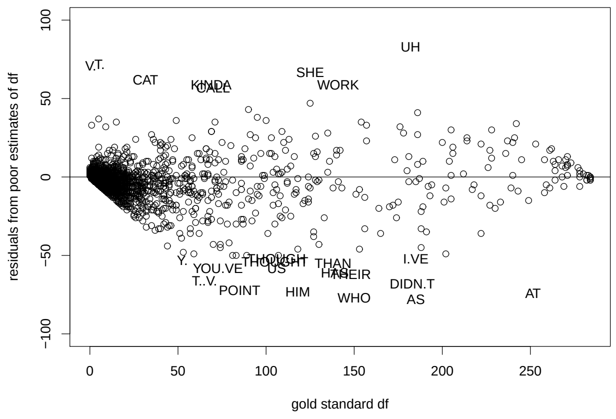

Fig. 1. There is considerable opportunity to improve over the baseline estimates of df. Each point shows the reference along the x-axis and the residual between the estimate and the reference along the y-axis. The reference is computed from the transcript, and the estimate is computed from the 1-best ASR output with just 3 hours of training data. Words with large errors are highlighted, and described in Table I.

TABLE I
WORDS WITH LARGE ERRORS

residual	3hr	20hr	reference	word
-78	107	140	185	AS
-77	73	112	150	WHO
-74	178	208	252	AT
-73	45	64	118	HIM
-72	13	61	85	POINT
-68	115	148	183	DIDN'T
-62	86	121	148	THEIR
-61	78	97	139	HAS
57	127	112	70	CALL
59	128	144	69	KINDA
59	200	176	141	WORK
62	94	78	32	CAT
67	192	188	125	SHE
71	72	79	1	V.
72	78	86	6	T.
83	265	262	182	UH

A. Expected counts

The most well-known features extracted from lattices are expected counts of words. Simply put, for each word w and utterance *utt*, the expected count of w (we call it $tf_{utt}(w)$ in the rest of this paper, to signify the connection between the notion of expected counts and term frequencies from clean text—the former equals the latter when there are no probabilities involved) is given by

$$tf_{utt}(w) = \sum_{n=1}^{\infty} n P_{utt}(w, n),$$

where $P_{utt}(w, n)$ is the probability that w appears exactly n times in *utt*. This probability can be computed by summing together the probabilities of all paths in a lattice that have the same number n of occurrences of w. (See Figure 3 for an automaton which selects only those paths with 2 occurrences of w.) These expected counts can be computed efficiently with the *grm* library [4], assuming that the lattices are in the *log-semiring*.

The above notion of expected count can be extended to the case of multiple utterances in a conversation by summing together the individual terms. Thus, for a conversation c,

$$tf_c(w) = \sum_{utt \in c} tf_{utt}(w).$$

Note that the above formula is applicable no matter whether the existence of w in an utterance is independent of other utterances; this is a consequence of the additivity of expectation: $E[X_1 + \cdots + X_n] = E[X_1] + \cdots + E[X_n]$, which holds irrespective of whether the random variables $X_1, \ldots, X_n$ are independent or not. As a way of quantizing the term frequencies (and because it results in better performance in our experiments) we further cap the term frequencies at 1:

$$\widetilde{tf}_c(w) = \min\{1, tf_c(w)\}. \quad (1)$$

B. Probabilities of multiple occurrences

For each word w, this is the probability that w occurs *at least k times* in an utterance or conversation. For each

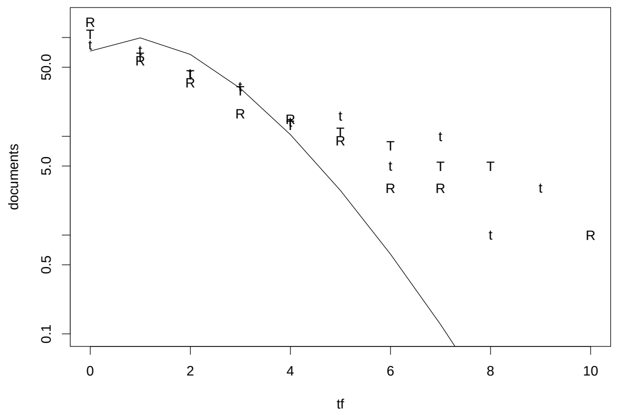

Fig. 2. How many documents mention "work" exactly k times? Values are shown in Table II. "R" is the reference. "T" is 1-best baseline with more training data (20 hours), and "t" is 1-best baseline with less training data (3 hours). In general, estimates based on more training data are closer to the reference than estimates based on less data. It is common practice to introduce various Poisson/independence assumptions but such assumptions can be disastrous because the Poisson (line) is even farther from the reference than the baseline estimates.

TABLE II
HOW MANY DOCUMENTS MENTION "WORK" EXACTLY k TIMES?

k	0	1	2	3	4	5	6	7
reference	143	58	35	17	15	9	3	3
ASR 1-best (20hr)	108	63	42	29	13	11	8	5
ASR 1-best (3hr)	84	74	43	32	14	16	5	10
Poisson	73	99	67	31	10	2.8	0.64	0.12

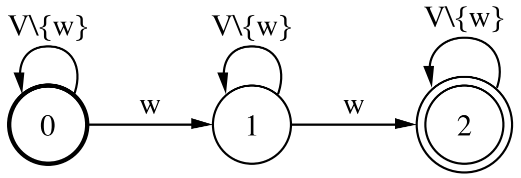

Fig. 3. An example of a finite-state-machine which accepts all strings of the lattice which contain exactly 2 occurrences of w. V represents the vocabulary, and $V \setminus \{w\}$ represents all the words except w.

utterance, it can be computed by summing together the probabilities of all paths in the lattice which contain w at least k times. In practice, it is computed efficiently using the formula

$$po_{utt}(w,k) = 1 - \sum_{j=0}^{k-1} P_{utt}(w,j), \qquad (2)$$

where $P_{utt}(w,j)$ was defined above. To compute $P_{utt}(w,j)$, an efficient implementation using finite-state machines was used: the lattice was composed[1] with a simple "filtering" machine, such as the one shown in Figure 3. This machine simply "accepts" all paths which contain w exactly j times; the total probability of these paths can then be computed easily with the operations of projection and ε-removal.

To extend this formula to the case of a whole conversation, simple summation of po_{utt} is not the right approach (for one thing, it can return a number greater than one). Instead, one can use the well-known formula for computing the distribution of a sum of *independent* random variables, which is given by a convolution [5] of the probability mass functions of the individual variables. Thus, for a conversation c,

$$P_c(w,\cdot) = P_{utt_1}(w,\cdot) \circledast \cdots \circledast P_{utt_N}(w,\cdot).$$

where $utt_1, \ldots, utt_N$ are the utterances of c. Similar to (2), the probability of occurrence of w *at least* k *times* in a conversation c can be calculated by

$$po_c(w,k) = 1 - \sum_{j=0}^{k-1} P_c(w,j), \qquad (3)$$

which makes it unnecessary to compute $P_c(w,j)$ for any $j > k-1$.

C. Other features

The above features can get thresholded, giving rise to "quantized" binary versions. In a log-linear framework, such binary features are usually very effective, as they provide a way of obtaining non-linear behavior. For example, for a word w and probability of at least k occurrences $po_c(w,k)$, T new features $\hat{po}_c^{(t_1)}(w,k), \ldots, \hat{po}_c^{(t_T)}(w,k)$ can be generated by setting

$$\hat{po}_c^{(t_i)}(w,k) \triangleq \mathbf{1}(po_c(w,k) \geq t_i).$$

[1]We have used OpenFst, the open-source library by Google. Its tutorial can be found at http://www.openfst.org/twiki/bin/view/FST/FstHltTutorial

where $\{t_1, \ldots, t_T\}$ is a set of user-defined thresholds.

The 1-best output from the lattice (the minimum-cost path) can also be used to provide an additional feature, the "baseline" feature. This is included as a "safety net", to make sure that the Machine Learner falls back to that output, especially when the test data have uninformative features. Thus,

$$q_c^{1best}(w) = \mathbf{1}(w \text{ exists in the 1-best output of } c).$$

D. The log-linear model

The features described in the previous subsections are generated with several recognizers (each using a different language model) and combined together in a log-linear model (logistic regression). For each word w and conversation c, the model tries to predict a binary variable which indicates whether w truly exists in the manual transcript of c or not.

The model learns a joint probability $p(\text{label, features}) \propto \exp\{\sum_i \lambda_i f_i\}$, where λ_i is the weight of the i-th feature learned by the model, and f_i is the value of the feature. Then, the probability that the word exists is given by the posterior

$$p_M(w \text{ exists}|\text{features}) = \frac{\exp\{\sum_i \lambda_i f_i\}}{1 + \exp\{\sum_i \lambda_i f_i\}},$$

where M represents the set of learned weights $\{\lambda_i\}$. Thus, for each conversation c, and for each word w which appears in the lattice, a separate training example is generated. Many words are less frequent and necessarily have very few positive examples (e.g., they truly exist in at most 1-2 conversations); on the other hand, the same words may result in lots of negative examples, depending on whether the recognizer tends to over-recognize such words. Although the number of examples runs into the millions, the number of features we have used does not exceed 100-150 and the log-linear model is usually trained (using the MEGAM toolkit [6]) without requiring special computational resources.

E. Estimation of df statistics

Once the log-linear model is trained, it is applied to an unseen set (validation or test). Thus, for each conversation c and word w, the model will predict whether w exists in c or not, and will also provide a confidence (posterior probability of existence) $p_M(w|c)$. Then, the df estimate can be provided either using thresholding, i.e.,

$$\widehat{df}_M^{thr}(w) \triangleq \sum_c \mathbf{1}(p_M(w|c) \geq thr), \qquad (4)$$

or, by summing together the confidences, i.e.,

$$\widehat{df}_M^{(soft)}(w) \triangleq \sum_c p_M(w|c). \qquad (5)$$

III. Experimental Setup

We have experimented with the English Fisher conversational corpus. The recognition/training/validation is done on non-overlapping parts of the corpus. For that purpose, we use the corpus split provided by [7], which was originally designed for topic identification. For the purposes of our paper, we have designated the various parts as follows:

Model	Description
ENG02	Do either of you have a pet? If so, how much time each day do you spend with your pet? How important is your pet to you?
ENG03	What do each of you think is the most important thing to look for in a life partner?
ENG04	Do each of you feel the minimum wage increase - to << $5.15 an hour - is sufficient?
ENG05	How do you each draw the line between acceptable humor and humor that is in bad taste?
ENG24	What changes, if any, have either of you made in your life since the terrorist attacks of Sept 11, 2001?
"combined"	*Concatenated text of the above 5 topic-based language models*

Fig. 4. Description of the topics of the language models used in the experiments. These topics were chosen because they contain at least 10 hours of audio each in set B we used for learning the log-linear model.

1. Set A: this is used as training material for building the recognizer(s). We have chosen three training sets from A, based on different sizes: a 1-hour set (denoted as A1), a 3-hour set (denoted as A2) and a 20-hour set (denoted as A3), so that we can study how data scarcity affect our estimates.
2. Set B (train & dev): after decoded with the recognizer(s) trained on set A, this set is subsequently used for learning the log-linear model of Section II. This is further split into a training part (B-train) and a development part (B-dev); the latter is used for making sure that the log-linear model is not over-fitted.
3. Set C (dev & test): after decoded with the recognizer(s) trained on set A, this set is used for testing the log-linear models, and reporting *df* statistics. The dev part is used for validation, and the test part is used for performing completely blind experiments.

To study the effect of topic mismatch, we further restrict sets B and C to a predefined set of 5 topics from Fisher. These 5 topics were chosen so that they provide at least 10 hours of audio per topic on set B (all other topics have less audio). The description of these topics appears in Figure 4. The language models used in the recognizer came from the transcripts of set A, but confined to the set of 5 topics mentioned above. Thus, 5 topic-based language models were estimated; also, a 6th language model (the "combined" model) was trained on the concatenated text of the 5 topics. The number of words per topic varied between 200K and 330K. The total number of words over the 5 topics was 1.5M.

BBN Technologies speech recognition system, BYBLOS, was used in all ASR experiments. It is a multi-pass LVCSR system that uses state-clustered Gaussian tied-mixture models at the triphone and quinphone levels [8]. The audio features are transformed using cepstral mean subtraction, HLDA and VTLN. Only maximum likelihood estimation was used. Decoding performs three passes: a forward and backward pass with a triphone acoustic model and a trigram language model, and rescoring using quinphone acoustic models and a fourgram language model. These three steps are repeated after speaker adaptation using CMLLR.

The logistic regression uses 132=22 $\times$ 6 features. There are 22 features for each language model (LM), and the 6 LMs described in Fig. 4. The 22 features consist of 4 primary features: $\widetilde{tf}_c(w), q_c^{1best}(w), po_c(w,1), po_c(w,2),$ plus 18 derived features. The 18 derived features are computed by thresholding $po_c(w,1)$ and $po_c(w,2)$ at $\{0.1,\ldots,0.9\}$.

IV. Estimation Results

In order to motivate the results reported in this section, we provide a list of questions that we would like to answer.

1. How does the acoustic model training size affect the estimation of *df* statistics?
2. Does one obtain better estimates when the topic of the language model and the topic of the speech match, even if the language model is trained on much less data?
3. Is it best to apply a threshold to the estimated existence probabilities, as returned by the log-linear model, or is it better to sum them up?

The following 3 subsections present short descriptions of the various estimators of *df* statistics. Presentation of results appears in the following subsection.

A. Baseline 1: using the 1-best output

This is the simplest baseline: simply count the number of documents (conversations) which contain a word w in the 1-best output:

$$\widehat{df}^{(\text{1best})}(w) = \sum_c \mathbf{1}(w \text{ is contained in the 1-best output of } c).$$

The third row of Table III shows the mean square error of this estimator for all words whose true *df* is shown in the first column. As can be easily seen, the error increases significantly with the WER of the recognition.

B. Baseline 2: using truncated tf statistics from lattices

This is the next baseline, which follows the approach of [9]: for each word w, make a decision whether it is contained in a conversation c based on a threshold *thr*, and then sum together the decisions over all conversations:

$$\widehat{df}^{(\text{from } tf \geq thr)}(w) = \sum_c \mathbf{1}(\widetilde{tf}_c(w) \geq thr)$$

Note that, in the case of clean text data, $\widehat{df}^{(\text{from } tf \geq 1)}(w)$ is exactly equal to the usual definition of df.

The fourth row of Table III shows the error of this estimator, based on lattices that were produced with the "combined" language model. The threshold was set to 0.7, based on performance on the dev part of set C.

TABLE III
TABLE WITH MEAN SQUARE ERRORS FOR VARIOUS ESTIMATORS OF df, AVERAGED OVER ALL WORDS WHICH HAVE THE TRUE df MENTIONED IN THE SECOND ROW. THE BEST RESULT IN EACH COLUMN IS SHOWN IN BOLDFACE. THE WERS ON SET B ARE SHOWN IN THE FIRST ROW.

	Acoust. Model →	1-hr AM (WER: 51.8%)			3-hr AM (WER: 44.6%)			20-hr AM (WER: 36.1%)		
	true df group →	$df = 1$	$df = 2$	$df \geq 3$	$df = 1$	$df = 2$	$df \geq 3$	$df = 1$	$df = 2$	$df \geq 3$
Baseline 1	$\widehat{df}^{(1best)}$ (combined LM)	1.10	2.93	88.59	0.96	2.51	62.88	0.84	2.13	40.67
Baseline 2	$\widehat{df}^{(\text{from } tf \geq 0.7)}$ (combined LM)	**0.92**	2.96	130.68	0.87	2.78	90.72	0.82	2.46	64.21
	$\widehat{df}^{(\text{from } po)}$ (combined LM)	1.64	3.60	87.42	1.28	2.71	62.60	1.04	2.10	48.32
Proposed (eq. 5)	$\widehat{df}_M^{(soft)}$	1.00	**2.25**	**69.12**	**0.84**	**1.97**	**46.42**	**0.70**	**1.63**	**30.95**
Contrastive	$\widehat{df}_M^{(soft)}$ (combined LM)	1.12	2.43	79.62	0.93	2.03	51.17	0.79	1.71	32.49
	$\widehat{df}^{(1best)}$ (oracle topic)	1.00	2.64	89.39	0.91	2.48	65.27	0.77	2.15	42.59
	$\widehat{df}^{(\text{from } tf)}$ (oracle topic)	1.41	2.92	99.79	1.11	2.56	85.04	0.89	2.10	66.06
	$\widehat{df}^{(\text{from } po)}$ (oracle topic)	1.30	2.74	70.60	1.03	2.42	54.56	0.84	2.01	39.89

C. Using the probability of at least one occurrence

The most natural way of computing a "soft" df from lattices is through the probability of at least one occurrence; this is $po_c(w, 1)$ of (3). Thus the df estimate for word w is given as the sum over all conversations

$$\widehat{df}^{(\text{from } po)}(w) = \sum_c po_c(w, 1).$$

Similar to $\widehat{df}^{(\text{from } tf)}(w)$, $\widehat{df}^{(\text{from } po)}(w)$ is also equal to the usual definition of df, when computed over clean data. However, it is better justified than $\widehat{df}^{(\text{from } \widetilde{tf})}(w)$, as it does not involve the "artificial" thresholding of $\widetilde{tf}_c(w)$.

The fifth row of Table III shows the error of this estimator, which was, as above, based on lattices that were produced with the "combined" language model.

D. Using the log-linear model

This is estimator $\widehat{df}_M^{(soft)}$, provided by (5). The corresponding results appear in the sixth row of Table III. We also tried $\widehat{df}_M^{thr}(w)$, but it gave significantly worse results than $\widehat{df}_M^{(soft)}(w)$.

E. Analysis of the results

The following trends are clear from Table III: (i) Better acoustic models result in better estimation of df, no matter which of the studied estimators is used. (ii) The learned log-linear model results in the best estimation in 8 out of 9 conditions (columns in the table). Moreover, we performed several regression tests with R using the features mentioned in Section III and found that almost all features were significant at $p < 0.001$. The existence of multiple language models is also important; the $po_c(\cdot, 1)$ features, computed under the multiple language models, were assigned the largest coefficients.

To analyze the effect of the multiple recognizers, we also learned a "contrastive" log-linear models that only contained features resulting from decoding with the "combined" language model only (that is, no features generated from a multitude of topic-specific language models were fed to train the model). The results appear in Table III, in the row $\widehat{df}_M^{(soft)}$ (combined LM). As is clear from these results, just using a single language model (despite being trained on five times more data than each of the individual topic-specific models) does not provide rich enough features for the log-linear model.

We also performed oracle experiments with topic-specific models. To see the effect of using language models which are *matched* to the topic of the speech, we computed the above estimators $\widehat{df}^{(1best)}$, $\widehat{df}^{(\text{from } tf)}$, $\widehat{df}^{(\text{from } po)}$ assuming that we know the topic of each utterance. That is, the various statistics for each conversation were collected only from the lattices of the *matched* recognizer. The bottom three rows of Table III summarize the results. As we can easily conclude from these results, recognizing speech with a matched topic does not give better results than recognizing the speech with a multitude of topic-specific language models and then combining their predictions together using the log-linear model.

V. CONVERSATION SIDE MATCHING

While we observe substantial gains in intrinsic evaluations of our df estimator, we seek an extrinsic evaluation to demonstrate that these improved estimates yield gains in a target application. Since frequencies of word types are frequently used in information retrieval (IR) tasks, we evaluate the effects of each estimator in an IR task.

Speech information retrieval [10], [11] searches over a corpus of audio documents for the best matches to a given spoken or written query. As in text based IR systems, the concept of query document relevance is based on the vector space model, in which each document is represented as a vector of word scores. Typically, these scores are based on a term-frequency inverse document-frequency (TFIDF) representation of the text, in which common words receive lower weight than rare words. For spoken document retrieval, documents are stored as either ASR 1-best transcripts, which leads directly to TFIDF computation, or as lattices in which fractional counts are used to compute term-frequency. In an analysis of term weightings effect on TFIDF, [12] found that IDF scores provide an effective upper bound on TFIDF scores, namely, that IDF alone can be an informative measure for IR. Since our focus is on computing document frequencies, we select their method for computing document representations: the ith position in the document vector representing the ith word is defined as IDF $= -\log_2 df_i/D$, where D is the number of documents in the corpus and df_i is the estimate of the document frequency of word i.

Using the English Fisher conversational corpus, we construct a simple query-by-example based IR task: conversation side matching. We split each conversation side into a separate document. The first document (side A) is the query, and the second document (side B) is added to the document collection. For each query (side A document), we return a ranking over all documents (side B documents) in the collection by the score of the inner product of the document vector with the query vector, both of which are based on the IDF representation above. We compute the mean reciprocal rank ($\sum_d \frac{1}{\text{rank}_{\text{correct}}}$), where higher scores are better. We ran this experiment for both the 142 conversations in dev and 141 in test of set C.

Using this framework, we evaluate the following methods:

- $\widehat{df}^{(1best)}$ (combined LM): A word is included in the document's vector if it appears a single time in the transcript. IDF is measured from the ASR 1-best transcript.
- $\widehat{df}^{(\text{from } tf)}$ (combined LM): A word is included in the vector if it has an expected count $>= \alpha$. IDF is measured by counting the number of document vectors in which each word type occurs.
- $\widehat{df}_M^{(soft)}$: A word is included in the vector if it has an expected count $>= \alpha$. IDF is measured based on the estimator of eq. (5).
- Reference: A word is included in the document's vector if it appears a single time in the transcript. IDF is measured from the reference transcript.

A simple value for the parameter α would be 1, but this dramatically undercounts words in each document since words that were spoken a single time will likely have expected counts below 1. Instead, we select a value for α using the development data that ensures that the resulting document vectors have (on average) the same number of non-zero elements as the 1-best vectors. For all of our experiments, this value was $\alpha = 0.5$.

We evaluated each of the 4 *df* methods (3 estimators and the true statistics) by running the conversation side matching evaluation on the 1, 3 and 20 hour acoustic model on development and test data (Table IV.) As expected, the 20 hour model outperforms the 1 and 3 hour model. For all models, our *df* estimator obtains substantial improvements over the baseline methods. In particular, for the 20-hr model, moving from 1-best output to lattices gives a small improvement (1% absolute) on development data (and hurts on test data), but our estimator yields a 5% absolute improvement. Overall, our estimator obtains a 20% and 17% relative error reduction on the 3-hr and 20-hr model respectively on development data. This suggests that our improvements in estimating type frequency statistics can have a significant impact on speech applications.

VI. Conclusions

Document frequency $df(w)$, the number of documents that mention w one or more times, is relatively easy to compute over text documents, but spoken documents are more challenging. We proposed $\widehat{df}_M^{\text{soft}}$, a log-linear combination of more than a hundred features. Some of these features take advantage of repetition (self-triggering). Whatever the document is about is likely to be repeated, unlike ASR errors, which tend to be more random. Other features use an ensemble of language models to reduce noise, since errors tend to be relatively uncorrelated across language models. Table III shows that $\widehat{df}_M^{\text{soft}}$ reduced the error over the baselines in most cases.

Better estimates of *df* lead to better performance, at least on a pairing task, as reported in Table IV. We are hopeful that these results will generalize to many other tasks as well, since Inverse Document Frequency, IDF$(w) = -\log_2 df(w)/D$, has proven effective across a wide range of applications including Information Retrieval.

TABLE IV
Mean reciprocal rank for conversation side matching on development and test data for a 1, 3 and 20 hour acoustic model. The best number in each column is shown in boldface.

	Dev			Test		
IDF Estimator	1-hr	3-hr	20-hr	1-hr	3-hr	20-hr
$\widehat{df}^{(1best)}$ (comb. LM)	0.347	0.408	0.542	0.337	0.448	0.491
$\widehat{df}^{(\text{from } tf)}$ (comb. LM)	0.371	0.457	0.555	0.382	0.438	0.491
$\widehat{df}_M^{(soft)}$	**0.409**	**0.494**	**0.593**	**0.414**	**0.503**	**0.537**
Reference	0.846			0.847		

Acknowledgments

We are grateful to Scott Novotney for providing his timely expertise in the use of the Byblos recognizer.

References

[1] M. Bacchiani and B. Roark, "Unsupervised language model adaptation," in *Proceedings of ICASSP-03*, vol. 1, April 2003, pp. I–224–227.

[2] F. Jelinek, *Statistical Methods for Speech Recognition*. MIT Press, 1997.

[3] R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modelling," *Computer Speech and Language*, vol. 10, no. 3, p. 187, 1996.

[4] C. Allauzen, M. Mohri, and B. Roark, "A general weighted grammar library," in *Proc. of the 9th Intl. Conf. on Automata (CIAA 2004)*, 2004.

[5] G. R. Grimmett and D. R. Stirzaker, *Probability and Random Processes*. Oxford Science Publications, 1992.

[6] H. Daumé III, "Notes on CG and LM-BFGS optimization of logistic regression," August 2004, paper available at http://pub.hal3.name#daume04cg-bfgs, implementation available at http://hal3.name/megam/.

[7] T. Hazen, F. Richardson, and A. Margolis, "Topic identification from audio recordings using word and phone recognition lattices," in *Proceedings of ASRU-2007*, December 2007, pp. 659 –664.

[8] R. Prasad, S. Matsoukas, C.-L. Kao, J. Z. Ma, D.-X. Xu, T. Colthurst, O. Kimball, R. Schwartz, J.-L. Gauvain, L. Lamel, H. Schwenk, G. Adda, and F. Lefevre, "The 2004 BBN/LIMSI 20xRT english conversational telephone speech recognition system," in *Proceedings of Interspeech-2005*, 2005, pp. 1645–1648.

[9] S. Olson, "Combining evidence from unconstrained spoken term frequency estimation for improved speech retrieval," PhD Dissertation, University of Maryland, 2008.

[10] C. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge University Press Cambridge, UK, 2008.

[11] J. Foote, "An overview of audio information retrieval," *Multimedia Systems*, vol. 7, no. 1, pp. 2–10, 1999.

[12] K. Umemura and K. Church, "Empirical term weighting and expansion frequency," in *Proceedings of the 2000 Joint SIGDAT conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, 2000, pp. 117–123.