NLP on Spoken Documents without ASR

Mark Dredze, Aren Jansen, Glen Coppersmith, Ken Church
Human Language Technology Center of Excellence
Center for Language and Speech Processing
Johns Hopkins University
mdredze,aren,coppersmith,Kenneth.Church@jhu.edu

Abstract

There is considerable interest in interdisciplinary combinations of automatic speech recognition (ASR), machine learning, natural language processing, text classification and information retrieval. Many of these boxes, especially ASR, are often based on considerable linguistic resources. We would like to be able to process spoken documents with few (if any) resources. Moreover, connecting black boxes in series tends to multiply errors, especially when the key terms are out-of-vocabulary (OOV). The proposed alternative applies text processing directly to the speech without a dependency on ASR. The method finds long (~ 1 sec) repetitions in speech, and clusters them into pseudo-terms (roughly phrases). Document clustering and classification work surprisingly well on pseudo-terms; performance on a Switchboard task approaches a baseline using gold standard manual transcriptions.

1 Introduction

Can we do IR-like tasks without ASR? Information retrieval (IR) typically makes use of simple features that count terms within/across documents such as term frequency (tf) and inverse document frequency (IDF). Crucially, to compute these features, it is sufficient to count repetitions of a term. In particular, for many IR-like tasks, there is no need for an automatic speech recognition (ASR) system to label terms with phonemes and/or words.

This paper builds on Jansen et al. (2010), a method for discovering terms with zero resources. This approach identifies long, faithfully repeated patterns in the acoustic signal. These acoustic repetitions often correspond to terms useful for information retrieval tasks. Critically, this method does not require a phonetically interpretable acoustic model or knowledge of the target language.

By analyzing a large untranscribed corpus of speech, this discovery procedure identifies a vast number of repeated regions that are subsequently grouped using a simple graph-based clustering method. We call the resulting groups pseudo-terms since they typically represent a single word or phrase spoken at multiple points throughout the corpus. Each pseudo-term takes the place of a word or phrase in bag of terms vector space model of a text document, allowing us to apply standard NLP algorithms. We show that despite the fully automated and noisy method by which the pseudo-terms are created, we can still successfully apply NLP algorithms with performance approaching that achieved with the gold standard manual transcription.

Natural language processing tools can play a key role in understanding text document collections. Given a large collection of text, NLP tools can classify documents by category (classification) and organize documents into similar groups for a high level view of the collection (clustering). For example, given a collection of news articles, these tools can be applied so that the user can quickly see the topics covered in the news articles, and organize the collection to find all articles on a given topic. These tools require little or no human input (annotation) and work across languages.

Given a large collection of speech, we would like

tools that perform many of the same tasks, allowing the user to understand the contents of the collection while listening to only small portions of the audio. Previous work has applied these NLP tools to speech corpora with similar results (see Hazen et al. (2007) and the references therein.) However, unlike text, which requires little or no preprocessing, audio files are typically first transcribed into text before applying standard NLP tools. Automatic speech recognition (ASR) solutions, such as large vocabulary continuous speech recognition (LVCSR) systems, can produce an automatic transcript from speech, but they require significant development efforts and training resources, typically hundreds of hours of manually transcribed speech. Moreover, the terms that may be most distinctive in particular spoken documents often lie outside the predefined vocabulary of an off-the-shelf LVCSR system. This means that unlike with text, where many tools can be applied to new languages and domains with minimal effort, the equivalent tools for speech corpora often require a significant investment. This greatly raises the entry threshold for constructing even a minimal tool set for speech corpora analysis.

The paper proceeds as follows. After a review of related work, we describe Jansen et al. (2010), a method for finding repetitions in speech. We then explain how these repetitions are grouped into pseudo-terms. Document clustering and classification work surprisingly well on pseudo-terms; performance on a Switchboard task approaches a baseline based on gold standard manual transcriptions.

2 Related Work

In the *low resource* speech recognition regime, most approaches have focused on coupling small amounts of orthographically transcribed speech (10s of hours) with much larger collections of untranscribed speech (100s or 1000s of hours) to train accurate acoustic models with semi-supervised methods (Novotney and Schwartz, 2009). In these efforts, the goal is to reduce the annotation requirements for the construction of competent LVCSR systems. This semi-supervised paradigm was relaxed even further with the pursuit of self organizing units (SOUs), phone-like units for which acoustic models are trained with completely unsupervised methods (Garcia and Gish, 2006). Even though the move away from phonetic acoustic models improves the universality of the architecture, small amounts of orthographic transcription are still required to connect the SOUs with the lexicon.

The segmental dynamic time warping (S-DTW) algorithm (Park and Glass, 2008) was the first truly zero resource effort, designed to discover portions of the lexicon directly by searching for repeated acoustic patterns in the speech signal. This work implicitly defined a new direction for speech processing research: unsupervised spoken term discovery, the entry point of our speech corpora analysis system. Subsequent extensions of S-DTW (Jansen et al., 2010) permit applications to much larger speech collections, a flexibility that is vital to our efforts.

As mentioned above, the application of NLP methods to speech corpora have traditionally relied on high resource ASR systems to provide automatic word or phonetic transcripts. Spoken document topic classification has been an application of particular interest (Hazen et al., 2007), for which the recognized words or phone n-grams are used to characterize the documents. These efforts have produced admirable results, with ASR transcript-based performance approached that obtained using the gold standard manual transcripts. Early efforts to perform automatic topic segmentation of speech input without the aid of ASR systems have been promising (Malioutov et al., 2007), but have yet to exploit the full the range of NLP tools.

3 Identifying Matched Regions

Our goal is to identify pairs of intervals within and across utterances of several speakers that contain the same linguistic content, preferably meaningful words or terms.

The spoken term discovery algorithm of Jansen et al. (2010) efficiently searches the space of $\binom{n}{2}$ intervals, where n is the number of speech frames.[1] Jansen et al. (2010) is based on dotplots (Church and Helfman, 1993), a method borrowed from bioinformatics for finding repetitions in DNA sequences.

[1] Typically, each frame represents a 25 or 30 ms window of speech sampled every 10 ms

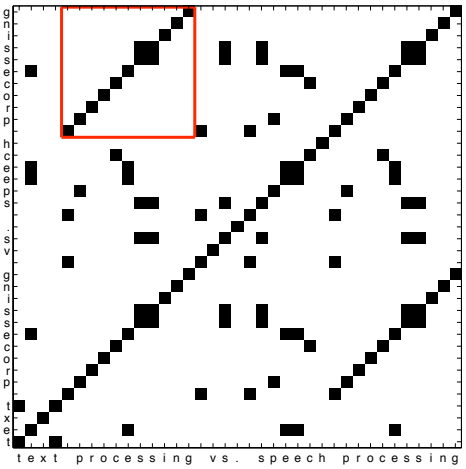

Figure 1: An example of a dotplot for the string "text processing vs. speech processing" plotted against itself. The box calls out the repeated substring: "processing."

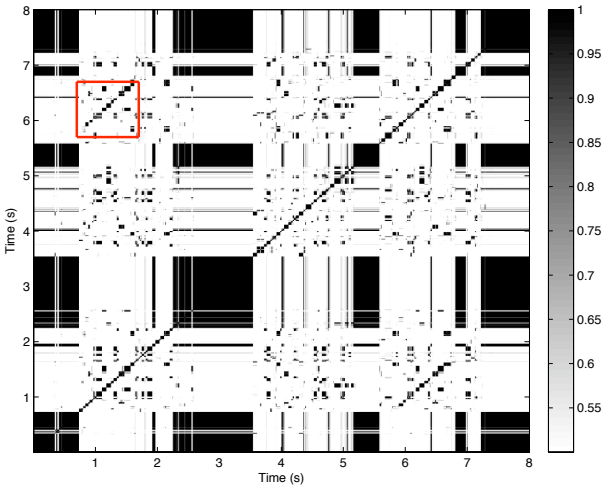

Figure 2: An example of an acoustic dotplot for 8 seconds of speech (posteriorgrams) plotted against itself. The box calls out a repetition of interest.

3.1 Acoustic Dotplots

When applied to text, the dotplot construct is remarkably simple: given character strings s_1 and s_2, the dotplot is a Boolean similarity matrix $K(s_1, s_2)$ defined as

$$K_{ij}(s_1, s_2) = \delta(s_1[i], s_2[j]).$$

Substrings common to s_1 and s_2 manifest themselves as diagonal line segments in the visualization of K. Figure 1 shows an example text dotplot where both $s1$ and $s2$ are taken to be the string "text processing vs. speech processing." The boxed diagonal line segment arises from the repeat of the word "processing," while the main diagonal line trivially arises from self-similarity. Thus, the search for line segments in K off the main diagonal provides a simple algorithmic means to identify repeated terms of possible interest, albeit sometimes partial, in a collection of text documents. The challenge is to generalize these dotplot techniques for application to speech, an inherently noisy, real-valued data stream.

The strategy is to replace character strings with frame-based speech representations of the form $\mathbf{x} = x_1, x_2, \ldots x_N$, where each $x_i \in \mathbb{R}^d$ is a d-dimensional vector space representation of the i^{th} overlapping window of the signal. Given vector time series $\mathbf{x} = x_1, x_2, \ldots x_N$ and $\mathbf{y} = y_1, y_2, \ldots y_M$ for two spoken documents, the acoustic dotplot is the real-valued $N \times M$ cosine similarity matrix $K(\mathbf{x}, \mathbf{y})$ defined as

$$K_{ij}(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\left[1 + \frac{\langle x_i, y_j \rangle}{\|x_i\| \|y_j\|}\right]. \qquad (1)$$

Even though the application to speech is a distinctly noisier endeavor, sequences of frames repeated between the two audio clips will still produce approximate diagonal lines in the visualization of the matrix. The search for matched regions thus reduces to the robust search for diagonal line segments in K, which can be efficiently performed with standard image processing techniques.

Included in this procedure is the application of a diagonal median filter of duration κ seconds. The choice of κ determines an approximate threshold on the duration of the matched regions discovered. Large κ values ($\sim$ 1 sec) will produce a relatively sparse list of matches corresponding to long words or short phrases; smaller κ values ($<$ 0.5 sec) will admit shorter words and syllables that may be less informative from a document analysis perspective. Given the approximate nature of the procedure, shorter κ values also admit less reliable matches.

3.2 Posteriorgram Representation

The acoustic dotplot technique can operate on any vector time series representation of the speech signal, including a standard spectrogram. However, at the individual frame level, the cosine similarities between frequency spectra of distinct speakers producing the same phoneme are not guaranteed to be high.

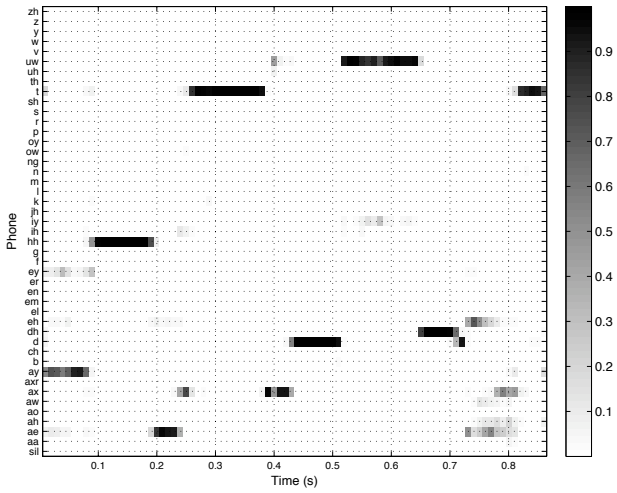

Figure 3: An example of a posteriorgram.

Thus, to perform term discovery across a multi-speaker corpus, we require a *speaker-independent* representation. Phonetic posteriorgrams are a suitable choice, as each frame is represented as the posterior probability distribution over a set of speech sounds given the speech observed at the particular point in time, which is largely speaker-independent by construction. Figure 3 shows an example posteriorgram for the utterance "I had to do that," computed with a multi-layer perceptron (MLP)-based English phonetic acoustic model (see Section 5 for details). Each row of the figure represents the posterior probability of the given phone as a function of time through the utterance and each column represents the posterior distribution over the phone set at that particular point in time.

The construction of speaker independent acoustic models typically requires a significant amount of transcribed speech. Our proposed strategy is to employ a speaker independent acoustic model trained in a high resource language or domain to interpret multi-speaker data in the zero resource target setting.[2] Indeed, we do not need to know a language to detect when a word of sufficient length has been repeated in it.[3] By computing cosine similarities of phonetic posterior distribution vectors (as opposed to reducing the speech to a one-best phonetic token sequence), the phone set used need not be matched to the target language. With this approach, a speaker-independent model trained on the phone set of a reference language may be used to perform speaker independent term discovery in any other.

In addition to speaker independence, the use of phonetic posteriorgrams introduces representational sparsity that permits efficient dotplot computation and storage. Notice that the posteriorgram displayed in Figure 3 consists of mostly near-zero values. Since cosine similarity (Equation 1) between two frames can only be high if they have significant mass on the same phone, most comparisons need not be made. Instead, we can apply a threshold and store each posteriorgram as an inverted file, performing inner product multiplies and adds only when they contribute. Using a grid of approximately 100 cores, we were able to perform the $O(n^2)$ dotplot computation and line segment search for 60+ hours of speech (corresponding to a 500 terapixel dotplot) in approximately 5 hours.

Figure 2 displays the posteriorgram dotplot for 8 seconds of speech against itself (i.e., $\mathbf{x} = \mathbf{y}$). The prominent main diagonal line results from self-similarity, and thus is ignored in the search. The boxed diagonal line segment results from two distinct occurrences of the term *one million dollars*. The large black boxes in the image result from long stretches of silence of filled pauses; fortunately, these are easily filtered with speech activity detection or simple measures of posteriorgram stability.

4 Creating Pseudo-Terms

Spoken documents will be represented as bags of pseudo-terms, where pseudo-terms are computed from acoustic repetitions described in the previous section. Let $\mathcal{M}$ be a set of matched regions (m), each consisting of a pair of speech intervals contained in the corpus ($m = [t_1^{(i)}, t_2^{(i)}], [t_1^{(j)}, t_2^{(j)}]$ indicates the speech from $t_1^{(i)}$ to $t_2^{(i)}$ is an acoustic match to the speech from $t_1^{(j)}$ to $t_2^{(j)}$). If a particular term occurs k times, the set $\mathcal{M}$ can include as many as $\binom{k}{2}$ distinct elements corresponding to that term, so we require a procedure to group them into clusters. We call the resulting clusters pseudo-terms since each

[2] A similarly-minded approach was taken in Hazen et al. (2007) and extended in Hazen and Margolis (2008), where the authors use Hungarian phonetic trigrams features to characterize English spoken documents for a topic classification task.

[3] While in this paper our acoustic model is based on our evaluation corpus, this is not a requirement of our approach. Future work will investigate performance of other acoustic models.

cluster is a placeholder for a term (word or phrase) spoken in the collection. Given the match list $\mathcal{M}$ and the pseudo-term clusters, it is relatively straightforward to represent spoken documents as bags of pseudo-terms.

To perform this pseudo-term clustering we represented matched regions as vertices in a graph with edges representing similarities between these regions. We employ a graph-clustering algorithm that extracts connected components. Let $G = (V, E)$ be an unweighted, undirected graph with vertex set V and edge set E. Each $v_i \in V$ corresponds to a single speech interval ($[t_1^{(i)}, t_2^{(i)}]$) present in $\mathcal{M}$ (each $m \in \mathcal{M}$ has a pair of such intervals, so $|V| = 2|\mathcal{M}|$) and each $e_{ij} \in E$ is an edge between vertex v_i and v_j.

The set E consists of two types of edges. The first represents repeated speech at distinct points in the corpus as determined by the match list $\mathcal{M}$. The second represents near-identical intervals in the same utterance (i.e. the same speech) since a single interval can show up in several matches in $\mathcal{M}$ and the algorithm in Section 3 explicitly ignores self-similarity. Given the intervals $[t_1^{(i)}, t_2^{(i)}]$ and $[t_1^{(j)}, t_2^{(j)}]$ contained in the same utterance and with corresponding vertices $v_i, v_j \in V$, we introduce an edge e_{ij} if fractional overlap f_{ij} exceeds some threshold τ, where $f_{ij} = \max(0, r_{ij})$ and

$$r_{ij} = \frac{(t_2^{(i)} - t_1^{(i)}) + (t_2^{(j)} - t_1^{(j)})}{\max(t_2^{(i)}, t_2^{(j)}) - \min(t_1^{(i)}, t_1^{(j)})} - 1. \quad (2)$$

From the graph G, we produce one pseudo-term for each connected component. More sophisticated edge weighting schemes would likely provide benefit. In particular, we expect improved clustering by introducing weights that reflect acoustic similarity between match intervals, rather than relying solely upon the term discovery algorithm to make a hard decision. Such confidence weights would allow even shorter pseudo-terms to be considered (by reducing κ) without greatly increasing false alarms. With such a shift, more sophisticated graph-clustering mechanisms would be warranted (e.g. Clauset et al. (2004)). We plan to pursue this in future work.

Counts	*Terms*
5	keep track of
5	once a month
2	life insurance
2	capital punishment
9	paper; newspaper
3	talking to you

Table 1: Pseudo-terms resulting from a graph clustering of matched regions ($\kappa = 0.75$, $\tau = 0.95$). Counts indicate the number of times the times the pseudo-terms appear across 360 conversation sides in development data.

Table 1 contains several examples of pseudo-terms and the matched regions included in each group. The orthographic forms are taken from the transcripts in the data (see Section 5). Note that for some pseudo-terms, the words match exactly, while for others, the phrases are distinct but phonetically similar. However, even in this case, there is often substantial overlap in the spoken terms.

5 Data

For our experiments we used the Switchboard Telephone Speech Corpus (Godfrey et al., 1992). Switchboard is a collection of roughly 2,400 two-sided telephone conversations with a single participant per side. Over 500 participants were randomly paired and prompted with a topic for discussion. Each conversation belongs to one of 70 pre-selected topics with the two sides restricted to separate channels of the audio.

To develop and evaluate our methods, we created three data sets from the Switchboard corpus: a `development` data set, a held out `tuning` data set and an `evaluation` data set. The development data set was created by selecting the six most commonly prompted topics (recycling, capital punishment, drug testing, family finance, job benefits, car buying) and randomly selecting 60 sides of conversations evenly across the topics (total 360 conversation sides.) This corresponds to 35.7 hours of audio. Note that each participant contributed at most one conversation side per topic, so these 360 conversation sides represent 360 distinct speakers. All algorithm development and experimentation was conducted exclusively on the development data.

For the tuning data set, we selected an additional 60 sides of conversations evenly across the same six topics used for development, for a total of 360 con-

versations and 37.5 hours of audio. This data was used to validate our experiments on the development data by confirming the heuristic used to select algorithmic parameters, as described below. This data was not used for algorithm development. The evaluation data set was created once parameters had been selected for a final evaluation of our methods. We selected this data by sampling 100 conversation sides from the next six most popular conversation topics (family life, news media, public education, exercise/fitness, pets, taxes), yielding 600 conversation sides containing 61.6 hours of audio.

In our experiments below, we varied the match duration κ between 0.6 s and 1.0 s and the overlap threshold τ between 0.75 and 1.0. We measured the resulting effects on the number of unique pseudo-terms generated by the process. In general, decreasing κ results in more matched regions increasing the number of pseudo-terms. Similarly, increasing τ forces fewer regions to be merged, increasing the total number of pseudo-terms. Table 2 shows how these parameters change the number of pseudo-terms (features) per document and the average number of occurrences of each pseudo-term. The user could tune these parameters to select pseudo-terms that were long and occurred in many documents. In the next sections, we consider how these parameters effect performance of various learning settings.

To provide the requisite speaker independent acoustic model, we compute English phone posteriorgrams using the multi-stream multi-layer perceptron-based architecture of Thomas et al. (2009), trained on 300 hours of conversational telephone speech. While this is admittedly a large amount of supervision, it is important to emphasize our zero resource term discovery algorithm does not rely on the phonetic interpretability of this reference acoustic model. The only requirement is that the same target language phoneme spoken by distinct speakers map to similar posterior distributions over the reference language phoneme set. Thus, even though we evaluate the system on matched-language Switchboard data, it can be just as easily applied to any target language with no language-specific knowledge or training resources required.[4]

[4]The generalization of the speaker independence of acoustic models across languages is not well understood. Indeed, the performance of our proposed system would depend to some extent on the phonetic similarity of the target and reference language. Unsupervised learning of speaker independent acoustic models remains an important area of future research.

κ	τ	*Features*	*Feat. Frequency*	*Feat./Doc.*
0.6	0.75	5,809	2.15	34.7
0.6	0.85	23,267	2.22	143.4
0.6	0.95	117,788	2.38	779.8
0.6	1.0	333,816	2.32	2153.4
0.75	0.75	8,236	2.31	52.8
0.75	0.85	18,593	2.36	121.7
0.75	0.95	48,547	2.36	318.2
0.75	1.0	90,224	2.18	546.9
0.85	0.75	5,645	2.52	39.5
0.85	0.85	8,832	2.44	59.8
0.85	0.95	15,805	2.24	98.3
0.85	1.0	24,480	2.10	142.4
1.0	0.75	1,844	2.39	12.3
1.0	0.85	2,303	2.24	14.4
1.0	0.95	3,239	2.06	18.6
1.0	1.0	4,205	1.93	22.7

Table 2: Statistics on the number of features (pseudo-terms) generated for different settings of the match duration κ and the overlap threshold τ.

6 Document Clustering

We begin by considering document clustering, a popular approach to discovering latent structure in document collections. Unsupervised clustering algorithms sort examples into groups, where each group contains documents that are similar. A user exploring a corpus can look at a few documents in each cluster to gain an overview of the content discussed in the corpus. For example, clustering methods can be used on search results to provide quick insight into the coverage of the returned documents (Zeng et al., 2004).

Typically, documents are clustered based on a bag of words representation. In the case of clustering conversations in our collection, we would normally obtain a transcript of the conversation and then extract a bag of words representation for clustering. The resulting clusters may represent topics, such as the six topics used in our switchboard data. Such groupings, available with no topic labeled training data, can be a valuable tool for understanding the contents of a speech data collection. We would like to know if similar clustering results can be obtained without the use of a manual or automatic transcript. In our case, we substitute the pseudo-terms discovered in a conversation for the transcript, representing

the document as a bag of pseudo-terms instead of actual words. Can a clustering algorithm achieve similar results along topical groups with our transcript-free representation as it can with a full transcript?

In our experiments, we use the six topic labels provided by Switchboard as the clustering labels. The goal is to cluster the data into six balanced groups according to these topics. While Switchboard topics are relatively straightforward to identify since the conversations were prompted with specific topics, we believe this task can still demonstrate the effectiveness of our representation relative to the baseline methods. After all, topic classification without ASR is still a difficult task.

6.1 Evaluation Metrics

There are numerous approaches to evaluating clustering algorithms. We consider several methods: Purity, Entropy and B-Cubed. For a full treatment of these metrics, see Amigó et al. (2009).

Purity measures the precision of each cluster, i.e., how many examples in each cluster belong to the same true topic. Purity ranges between zero and one, with one being optimal. While optimal purity can be obtained by putting each document in its own cluster, we fix the number of clusters in all experiments so purity numbers are comparable. The purity of a cluster is defined as the largest percentage of examples in a cluster that have the same topic label. Purity of the entire clustering is the average purity of each cluster:

$$\text{purity}(C, L) = \frac{1}{N} \sum_{c_i \in C} \max_{l_j \in L} |c_i \cap l_j| \qquad (3)$$

where C is the clustering, L is the reference labeling, and N are the number of examples. Following this notation, c_i is a specific cluster and l_j is a specific true label.

Entropy measures how the members of a cluster are distributed amongst the true labels. The global metric is computed by taking the weighted average of the entropy of the members of each cluster. Specifically, entropy(C, L) is given by:

$$-\sum_{c_i \in C} \frac{N_i}{N} \sum_{l_j \in L} P(c_i, l_j) \log_2 P(c_i, l_j) \qquad (4)$$

where N_i is the number of instances in cluster i, $P(c_i, l_j)$ is the probability of seeing label l_j in cluster c_i and the other variables are defined as above.

B-Cubed measures clustering effectiveness from the perspective of a user's inspecting the clustering results (Bagga and Baldwin, 1998). B-Cubed precision can be defined as an algorithm as follows: suppose a user randomly selects a single example. She then proceeds to inspect every other example that occurs in the same cluster. How many of these items will have the same true label as the selected example (precision)? B-Cubed recall operates in a similar fashion, but it measures what percentage of all examples that share the same label as the selected example will appear in the selected cluster. Since B-Cubed averages its evaluation over each document and not each cluster, it is less sensitive to small errors in large clusters as opposed to many small errors in small clusters. We include results for B-Cubed F1, the harmonic mean of precision and recall.

6.2 Clustering Algorithms

We considered several clustering algorithms: repeated bisection, globally optimal repeated bisection, and agglomerative clustering (see Karypis (2003) for implementation details). Each bisection algorithm is run 10 times and the optimal clustering is selected according to a provided criteria function (no true labels needed). For each clustering method, we evaluated several criteria functions. Additionally, we considered different scalings of the feature values (the number of times the pseudo-terms appear in each document). We found that scaling each feature by the inverse document frequency, effectively TFIDF, produced the best results, so we use that scaling in all of our experiments. We also explored various similarity metrics and found cosine similarity to be the most effective.

We used the Cluto clustering library for all clustering experiments (Karypis, 2003). In the following section, we report results for the optimal clustering configuration based on experiments on the development data.

6.3 Baselines

We compared our pseudo-term feature set performance to two baselines: (1) Phone Trigrams and

(2) Word Transcripts. The Phone Trigram baseline is derived automatically using an approach similar to Hazen et al. (2007). This baseline is based on a vanilla phone recognizer on top of the same MLP-based acoustic model (see Section 5 and the references therein for details) used to discover the pseudo-terms. In particular, the phone posteriorgrams were transformed to frame-level monophone state likelihoods (through division by the frame-level priors). These state likelihoods were then used along with frame-level phone transition probabilities to Viterbi decode each conversation side. It is important to emphasize that the reliability of phone recognizers depends on the phone set matching the application language. Using the English acoustic model in this manner on another language will significantly degrade the performance numbers reported below.

The Word Transcript baseline starts with Switchboard transcripts. This baseline serves as an upper bound of what large vocabulary recognition can provide for this task. n-gram features are computed from the transcript. Performance is reported separately for unigrams, bigrams and trigrams.

6.4 Results

To optimize parameter settings, match duration (κ) and overlap threshold (τ) were swept over a wide range ($0.6 < \kappa < 1.0$ and $0.75 < \tau < 1.0$) using a variety of clustering algorithms and training criteria. Initial results on development data showed promising performance for the default $\mathcal{I}_2$ criteria in Cluto (repeated bisection set to maximize the square root of within cluster similarity). Representative results on development data with various parameter settings for this clustering configuration appear in Table 3.

A few observations about results on development data. First, the three evaluation metrics are strongly correlated. Second, for each κ the same narrow range of τ values achieve good results. In general, settings of $\tau > 0.9$ were all comparable. Essentially, setting a high threshold for merging matched regions was sufficient without further tuning. Third, we observed that decreasing κ meant more features, but that these additional features did not necessarily lead to more useful features for clustering. For example, $\kappa = 0.70$ gave a small number of reasonably good features, while $\kappa = 0.60$ can give an order of magnitude more features without much of a change in clustering performance. Finally, while pseudo-term results are not as good as with the manual transcripts (unigrams), they achieve similar results. Compared with the phone trigram features determined by the phone recognizer output, the pseudo-terms perform significantly better. Note that these two automatic approaches were built using the identical MLP-based phonetic acoustic model.

Pseudo-term Results					
κ	τ	*Features*	*Purity*	*Entropy*	B^3 F1
0.60	0.95	117,788	0.9639	0.2348	0.9306
0.60	0.96	143,299	**0.9750**	**0.1664**	**0.9518**
0.60	0.97	178,559	0.9667	0.2116	0.9366
0.60	0.98	223,511	0.9528	0.2717	0.9133
0.60	0.99	333,630	0.9583	0.2641	0.9210
0.60	1.0	333,816	0.9583	0.2641	0.9210
0.70	0.93	58,303	0.9528	0.3114	0.9105
0.70	0.94	66,054	**0.9667**	**0.2255**	**0.9358**
0.70	0.95	74,863	0.9583	0.2669	0.9210
0.70	0.96	86,070	0.9611	0.2529	0.9260
0.70	0.97	100,623	0.9639	0.2326	0.9312
0.70	0.98	117,535	0.9556	0.2821	0.9158
0.70	0.99	161,219	0.9056	0.4628	0.8372
0.70	1.0	161,412	0.9333	0.4011	0.8760

Phone Recognizer Baseline				
Type	*Features*	*Purity*	*Entropy*	B^3 F1
Phone Trigram	28,110	0.6194	1.3657	0.5256

Manual Word Transcript Baselines				
Type	*Features*	*Purity*	*Entropy*	B^3 F1
Word Unigram	7,330	**0.9917**	**0.0559**	**0.9839**
Word Bigram	74,216	0.9833	0.1111	0.9678
Word Trigram	224,934	0.9889	0.0708	0.9787

Table 3: Clustering results on development data using globally optimal repeated bisection and $\mathcal{I}_2$ criteria. The best results over the manual word transcript baselines and for each match duration (κ) are highlighted in bold. Pseudo-term results are better than the phonetic baseline and almost as good as the transcript baseline.

We sought to select the optimal parameter settings for running on the evaluation data using the development data and the held out tuning data. We defined the following heuristic to select the optimal parameters. We choose settings for κ, τ and the clustering parameters that independently maximize the performance averaged over all runs on development data. We then selected the single run corresponding to these parameter settings and checked the result on the held out tuning data. This setting was also the best performer on the held out set, so we used these parameters for evaluation. The best performing parameters were globally optimal repeated

κ	τ	*Features*	*Purity*	*Entropy*	B^3 F1
0.70	0.98	123,901	0.9778	0.1574	0.9568
Phone Trigram		28,374	0.6389	1.2345	0.5513
Word Unigram		7,640	0.9972	0.0204	0.9945
Word Bigram		77,201	0.9972	0.0204	0.9945
Word Trigram		233,744	0.9972	0.0204	0.9945

Table 4: Results on held out tuning data. The parameters (globally optimal repeated bisection clustering with $\mathcal{I}_2$ criteria, $\kappa = 0.70$ seconds and $\tau = 0.98$) were selected using the development data and validated on tuning data. Note that the clusters produced by each manual transcript test were identical in this case.

κ	τ	*Features*	*Purity*	*Entropy*	B^3 F1
0.70	0.98	279,239	0.9517	0.3366	0.9073
Phone Trigram		31,502	0.7000	1.0496	0.6355
Word Unigram		9939	0.9883	0.0831	0.9772
Word Bigram		110,859	0.9883	0.0910	0.9771
Word Trigram		357,440	0.9900	0.0775	0.9803

Table 5: Results on evaluation data. The parameters (globally optimal repeated bisection clustering with $\mathcal{I}_2$ criteria, $\kappa = 0.7$ seconds and $\tau = 0.98$) were selected using the development data and validated on tuning data.

bisection clustering with $\mathcal{I}_2$ criteria, $\kappa = 0.7$ s and $\tau = 0.98$. Note that examining Table 3 alone may suggest other parameters, but we found our selection method to yield optimal results on the tuning data.

Results on held out tuning and evaluation data for this setting compared to the manual word transcripts and phone recognizer output are shown in Tables 4 and 5. On both the tuning data and evaluation data, we obtain similar results as on the development data. While the manual transcript baseline is better than our pseudo-term representations, the results are quite competitive. This demonstrates that useful clustering results can be obtained without a full-blown word recognizer. Notice also that the pseudo-term performance remains significantly higher than the phone recognizer baseline on both sets.

7 Supervised Document Classification

Unsupervised clustering methods are attractive since they require no human annotations. However, obtaining a few labeled examples for a simple labeling task can be done quickly, especially with crowd sourcing systems such as CrowdFlower and Amazon's Mechanical Turk (Snow et al., 2008; Callison-Burch and Dredze, 2010). In this setting, a user may listen to a few conversations and label them by topic. A supervised classification algorithm can then be trained on these labeled examples and used to automatically categorize the rest of the data. In this section, we evaluate if supervised algorithms can be trained using the pseudo-term representation of the speech.

We set up a multi-class supervised classification task, where each document is labeled using one of the six Switchboard topics. A supervised learning algorithm is trained on a sample of labeled documents and is then asked to label some test data. Results are measured in terms of accuracy. Since the documents are a balanced sample of the six topics, random guessing would yield an accuracy of 0.1667.

We proceed as with the clustering experiments. We evaluate different representations for various settings of κ and τ and different classifier parameters on the development data. We then select the optimal parameter settings and validate this selection on the held out tuning data, before generating the final representations for the evaluation once the optimal parameters have been selected.

For learning we require a multi-class classifier training algorithm. We evaluated four popular learning algorithms: a) MIRA—a large margin online learning algorithm (Crammer et al., 2006); b) Confidence Weighted (CW) learning—a probabilistic large margin online learning algorithm (Dredze et al., 2008; Crammer et al., 2009); c) Maximum Entropy—a log-linear discriminative classifier (Berger et al., 1996); and d) Support Vector Machines (SVM)—a large margin discriminator (Joachims, 1998).[5] For each experiment, we used default settings of the parameters (tuning did not significantly change the results) and 10 online iterations for the online methods (MIRA, CW). Each reported result is based on 10-fold cross validation.

Table 6 shows results for various parameter settings and the four learning algorithms on development data. As before, we observe that values for $\tau > 0.9$ tend to do well. The CW learning algorithm performs the best on this data, followed by Maximum Entropy, MIRA and SVM. The optimal κ for classification is 0.75, close to the 0.7 value selected in clustering. As before, pseudo-terms do

[5]We used the "variance" formulation with $k = 1$ for CW learning, Gaussian regularization for the Maximum Entropy classifier, and a linear kernel for the SVM.

κ	τ	*MaxEnt*	*SVM*	*CW*	*MIRA*
0.60	0.99	0.8972	0.6944	0.8667	0.8972
0.60	1.0	0.8972	0.6944	0.8639	0.8944
0.70	0.97	0.9000	0.7722	0.8500	0.8056
0.70	0.98	0.8806	0.7417	0.8917	0.8167
0.70	0.99	0.9000	0.6556	0.9194	0.9056
0.70	1.0	0.8917	0.6556	0.9194	0.9083
0.75	0.94	0.8778	0.7806	0.8639	0.8056
0.75	0.95	0.8778	0.7694	0.8889	0.8111
0.75	0.96	0.9028	0.7778	0.9000	0.8778
0.75	0.97	**0.9111**	0.7722	**0.9250**	**0.9278**
0.75	0.98	0.9056	0.7417	0.9194	0.9167
0.85	0.85	0.8639	**0.7833**	0.8500	0.8167
0.85	0.90	0.8611	0.7528	0.8611	0.8583
0.85	0.91	0.8389	0.7500	0.8722	0.8556
0.85	0.92	0.8528	0.7222	0.8944	0.8556
Phone Trigram		0.6111	0.7139	0.9138	0.5000
Word Unigram		**0.9472**	**0.8861**	0.9861	**0.9306**
Word Bigram		0.9250	0.8833	**0.9917**	0.9278
Word Trigram		0.9278	0.8611	0.9889	0.9222

Table 6: The top 15 results (measured as average accuracy across the 4 algorithms) for pseudo-terms on development data. The best pseudo-term and manual transcript results for each algorithm are bolded. All results are based on 10-fold cross validation. Pseudo-term results are better than the phonetic baseline and almost as good as the transcript baseline.

well, though not as well as the upper bound based on manual transcripts. The performance for pseudo-terms and phone trigrams are roughly comparable, though we expect pseudo-terms to be more robust across languages.

Using the same selection heuristic as in clustering, we select the optimal parameter settings, validate them on the held out tuning data, and compute results on evaluation data. The best performing configuration was for $\kappa = 0.75$ seconds and $\tau = 0.97$. Notice these parameters are very similar to the best parameters selected for clustering. Results on held out tuning and evaluation data for this setting compared to the manual transcripts are shown in Tables 7 and 8. As with clustering, we see good overall performance as compared with manual transcripts. While the performance drops, results suggest that useful output can be obtained without a transcript.

8 Conclusions

We have presented a new strategy for applying standard NLP tools to speech corpora without the aid of a large vocabulary word recognizer. Built instead on top of the unsupervised discovery of term-

κ	τ	*MaxEnt*	*SVM*	*CW*	*MIRA*
0.75	0.97	0.8722	0.7389	0.8972	0.8750
Phone Trigram		0.7167	0.6972	0.9056	0.5083
Word Unigram		0.9500	0.9056	0.9806	0.9250
Word Bigram		0.9444	0.9111	0.9833	0.9250
Word Trigram		0.9417	0.8972	0.9778	0.9250

Table 7: Results on held out tuning data. The parameters ($\kappa = 0.75$ seconds and $\tau = 0.97$) were selected using the development data and validated on tuning data. All results are based on 10-fold cross validation. Pseudo-term results are very close to the transcript baseline and often better than the phonetic baseline.

κ	τ	*MaxEnt*	*SVM*	*CW*	*MIRA*
0.75	0.97	0.8683	0.7167	0.7850	0.7150
Phone Trigram		0.8600	0.7750	0.9183	0.6233
Word Unigram		0.9533	0.9317	0.9850	0.9267
Word Bigram		0.9467	0.9200	0.9900	0.9367
Word Trigram		0.9383	0.9233	0.9817	0.9367

Table 8: Results on evaluation data. The parameters ($\kappa = 0.75$ seconds and $\tau = 0.97$) were selected using the development data and validated on tuning data. All results are based on 10-fold cross validation. Pseudo-term results are very close to the transcript baseline and often better than the phonetic baseline.

like units in the speech, we perform unsupervised topic clustering as well as supervised classification of spoken documents with performance approaching that achieved with the manual word transcripts, and generally matching or exceeding that achieved with a phonetic recognizer. Our study identified several opportunities and challenges in the development of NLP tools for spoken documents that rely on little or no linguistic resources such as dictionaries and training corpora.

References

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4).

A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 79–85. Association for Computational Linguistics.

A.L. Berger, V.J.D. Pietra, and S.A.D. Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical

Turk. In *Workshop on Creating Speech and Language Data With Mechanical Turk at NAACL-HLT*.

K. W. Church and J. I. Helfman. 1993. Dotplot: A program for exploring self-similarity in millions of lines of text and code. *Journal of Computational and Graphical Statistics*.

Aaron Clauset, Mark E J Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Physical Review E*, 70.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research (JMLR)*.

Koby Crammer, Mark Dredze, and Alex Kulesza. 2009. Multi-class confidence weighted algorithms. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Mark Dredze, Koby Crammer, and Fernando Pereira. 2008. Confidence-weighted linear classification. In *International Conference on Machine Learning (ICML)*.

Alvin Garcia and Herbert Gish. 2006. Keyword spotting of arbitrary words using minimal speech resources. In *ICASSP*.

J.J. Godfrey, E.C. Holliman, and J. McDaniel. 1992. SWITCHBOARD: Telephone speech corpus for research and development. In *ICASSP*.

Timothy J. Hazen and Anna Margolis. 2008. Discriminative feature weighting using MCE training for topic identification of spoken audio recordings. In *ICASSP*.

Timothy J. Hazen, Fred Richardson, and Anna Margolis. 2007. Topic identification from audio recordings using word and phone recognition lattices. In *IEEE Workshop on Automatic Speech Recognition and Understanding*.

Aren Jansen, Kenneth Church, and Hynek Hermansky. 2010. Towards spoken term discovery at scale with zero resources. In *Interspeech*.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML)*.

George Karypis. 2003. CLUTO: A software package for clustering high-dimensional data sets. Technical Report 02-017, University of Minnesota, Dept. of Computer Science.

Igor Malioutov, Alex Park, Regina Barzilay, and James Glass. 2007. Making Sense of Sound: Unsupervised Topic Segmentation Over Acoustic Input. In *ACL*.

Scott Novotney and Richard Schwartz. 2009. Analysis of low-resource acoustic model self-training. In *Interspeech*.

Alex Park and James R. Glass. 2008. Unsupervised pattern discovery in speech. *IEEE Transactions of Audio, Speech, and Language Processing*.

R. Snow, B. O'Connor, D. Jurafsky, and A.Y. Ng. 2008. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263. Association for Computational Linguistics.

S. Thomas, S. Ganapathy, and H. Hermansky. 2009. Phoneme recognition using spectral envelope and modulation frequency features. In *Proc. of ICASSP*.

H.J. Zeng, Q.C. He, Z. Chen, W.Y. Ma, and J. Ma. 2004. Learning to cluster web search results. In *Conference on Research and development in information retrieval (SIGIR)*.