Towards Spoken Term Discovery At Scale With Zero Resources

Aren Jansen[1,2], *Kenneth Church*[1,3], *Hynek Hermansky*[1,2]

[1]Human Language Technology Center of Excellence,
[2]Department of Electrical and Computer Engineering, [3]Department of Computer Science
Johns Hopkins University, Baltimore, Maryland
aren@jhu.edu, kenneth.church@jhu.edu, hynek@jhu.edu

Abstract

The spoken term discovery task takes speech as input and identifies terms of possible interest. The challenge is to perform this task efficiently on large amounts of speech with zero resources (no training data and no dictionaries), where we must fall back to more basic properties of language. We find that long ($\sim$ 1 s) repetitions tend to be contentful phrases (e.g. University of Pennsylvania) and propose an algorithm to search for these long repetitions without first recognizing the speech. To address efficiency concerns, we take advantage of (i) sparse feature representations and (ii) inherent low occurrence frequency of long content terms to achieve orders-of-magnitude speedup relative to the prior art. We frame our evaluation in the context of spoken document information retrieval, and demonstrate our method's competence at identifying repeated terms in conversational telephone speech.

Index Terms: spoken term discovery, zero resource speech recognition, dotplots

1. Introduction

The current stable of large vocabulary speech recognition systems have been developed on the assumption that large orthographically transcribed speech corpora are available for constructing detailed acoustic and language models. As the global need for speech technologies rises, there has been considerable recent interest in developing comparable speech technologies for lower resource languages where the available annotated corpora are of insufficient size to apply standard methods. We are interested in the limit of this low resource paradigm, a zero resource setting where we assume no prior knowledge of the linguistic structure of the target language, no transcribed training data, and (in extreme cases) no untranscribed training data.

In the zero resource setting, we require algorithms for unsupervised discovery of the categorical linguistic structure (e.g. phonological and lexical) and models for the full range of acoustic realizations of this structure. From the bottom up, we require a discrete characterization of an acoustic feature vector space, either in terms of a predefined inventory of phonological features (e.g. distinctive features or phonemes) or some automatically learned inventory that emerges from unsupervised clustering and continuity constraints (e.g. see [1]). From the top down, we are tasked with identifying the lexical structure of the language (e.g. identifying words and terms) and relating the discovered lexicon to the underlying phonological system.

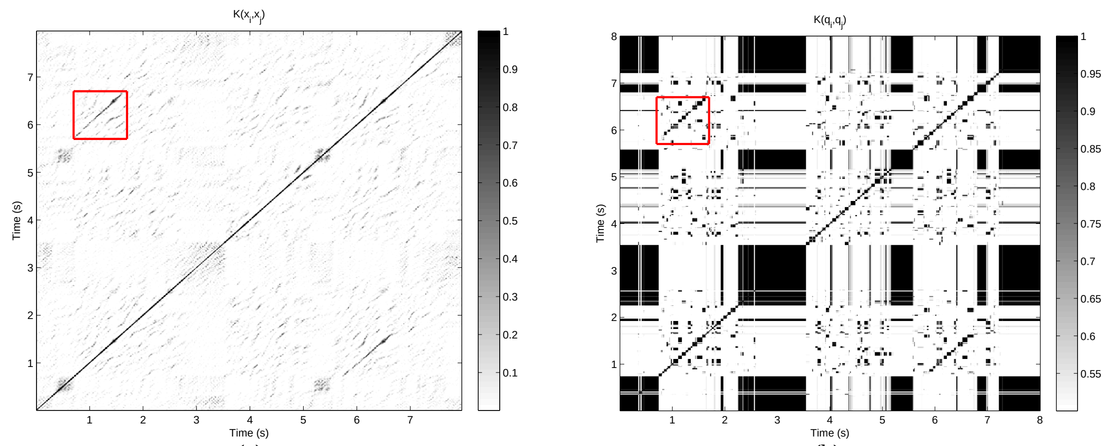
In this paper, we take as given a phonetic acoustic model (though, in general, the phonetic interpretability it provides will not be a requirement of our approach) and focus solely on the task of discovering intervals of the speech signal that may contain words or terms of possible interest. In this pursuit, we are faced with a fundamental computational complication: in n frames of continuous speech, there are $\binom{n}{2}$ possible intervals that can correspond to some word or term in the speech. Thus, we require a heuristic to sort out the potential relevance of these $O(n^2)$ possibilities. Derived from similar efforts in text information retrieval [2, 3], we root our search in the notion that interval length, repetition, and burstiness (inhomogeneity of occurrence frequency) are each strong relevance cues.

The lossiness of speech communication offers two additional cues that can be powerful indicators of interval relevance. First, contentful information must be conveyed clearly and, as a result, repeats of important terms are typically produced with relatively high fidelity. Thus, the more accurate the acoustic match, the more likely it was important. Second, speech communication often occurs over a noisy/lossy channel (e.g. telephone, loud restaurant), often prompting adjacent repeats of terms by the speaker if the listener missed something they deemed relevant from context. With these motivations, we reduce the problem of unsupervised spoken term discovery to a search for long, faithfully repeated intervals of speech.

Our solution to this problem is based on the graphical method of dotplots for comparing sequences, first applied to speech processing by Park and Glass [4] in the form of the segmental dynamic time warping (S-DTW) algorithm. The search for repeated intervals is inherently an $O(n^2)$ problem, and while the S-DTW algorithm provides an elegant solution, it is not scalable to many applications of interest. We significantly improve the constants by implementing a coarse first-pass that relies on representational and dotplot sparsity. Using this first pass to limit the ultimate S-DTW search space, we achieve an orders-of-magnitude speedup relative to the prior work.

2. The Search Algorithm

Dotplots are a graphical method of comparing sequences, initially developed for identifying recurring protein sequences in the bioinformatics community and later extended to large scale text processing more generally [5]. Given a character string $X = x_1x_2 \ldots x_n$, where each x_i is an element of some alphabet, the traditional dotplot on X is an $n \times n$ Boolean matrix M defined as $M_{ij} = (x_i == y_j)$. Substrings repeated at different points in X manifest themselves as diagonal line segments in the visualization of M; substrings consisting of a single repeated token will manifest themselves as large blocks. Recent work [4] extended these techniques to acoustic processing in order to identify repeated intervals of speech or audio. In this section, we define how this extension is accomplished and, using standard image processing techniques, motivate a new efficient search algorithm to discover repeated intervals.



(a) (b)

Figure 1: Acoustic (a) and posteriorgram (b) dotplots computed on 8 seconds of Fisher English speech.

2.1. Acoustic Dotplots

The traditional dotplot defined on strings can be extended to audio by (i) representing the signal as an acoustic feature vector time series and (ii) replacing the Boolean-valued similarity between pairs of characters with a real-valued similarity between pairs of feature vectors. Formally, let $X = x_1 x_2 \dots x_n$ now be a vector time series computed over an audio signal, where each $x_i \in \mathbb{R}^d$. Then, we may define the acoustic dotplot M as the generalized Gram matrix $M_{ij} = K(x_i, x_j)$, where $K : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is a symmetric similarity function between feature vectors. In this study, we consider cosine similarity

$$K(x, y) = \frac{1}{2}\left[1 + \frac{\langle x, y\rangle}{\|x\|\|y\|}\right], \tag{1}$$

which takes a value of 1 when x and y point in the same direction, 0.5 when they are orthogonal, and 0 when they point in opposite directions. Fig. 1(a) displays an example acoustic dotplot, computed using standard 39-dimensional mel frequency cepstral coefficient features (25 ms windows, 10 ms step) on 8 seconds of speech from the Fisher English corpus. Since K is symmetric in its arguments, the dotplot is symmetric and the main diagonal line arises from self-similarity. The boxed line segment off the main diagonal arises from a repeat of the term *one million dollars*, occurring around the 1 second mark and then again at the 6 second mark.

2.2. Efficient Line Segment Detection

Given an acoustic dotplot computed for a pair of utterances, we can reduce the problem of finding repeated intervals to a search for diagonal line segments. This is a common image processing task that can be accomplished efficiently using the following:

1. Transform the dotplot matrix M into binary matrix M' by applying a similarity threshold δ such that $M'_{ij} = 1$ if $M_{ij} > \delta$ and 0 otherwise.
2. Apply to the thresholded image a diagonal μ-percentile filter of length Δ with orientation *parallel* to the target line segments (45° relative to the x-axis of the image), allowing only diagonal line structures of sufficient length (and thus sufficient repeated interval duration) and density to pass.
3. Apply a Gaussian filter of width σ with orientation *orthogonal* to the target line segments (-45° relative to the x-axis of the image) to the thresholded image, transversely smearing the remaining line segments. This filter accommodates variation in speaking rate by allowing deviation from the 45° assumption. The result of these first three steps as applied to the dotplot of Fig. 1(a) is shown in Fig. 2(a).
4. Apply a classical one-dimensional Hough transform (r is varied, θ fixed at -45°) to the filtered image, amounting to a projection of the image onto the line $y = -x$ (as demonstrated in the red overlay of Fig. 2(a)). The result is shown in Fig. 2(b). Notice there are three peaks. The large central peak, about which the transform is symmetric, is a result of the self-similarity line (main diagonal).
5. Use the peaks of the Hough transform to define rays through which to search for line segments. In our example of Fig. 2, we ignore the center peak because it corresponds to the main diagonal, which in this example results from self similarity. Since the Hough transform is symmetric, there is only one search ray (outlined in green in Fig. 2(a)).

3. Computational Optimizations

As defined above, computing an $n \times n$ acoustic dotplot with d-dimensional feature vectors requires $O(n^2 d)$ computation and $O(n^2)$ memory space. With the typical 100 Hz frame rate, computing the dotplot for just 1 hour of speech would require computing and storing a 129 gigapixel image. In [6], the authors report a run time of over 33 machine-hours to process 1 hour of speech with dotplot-based techniques, an already prohibitive number that would become quadratically worse as the input size is further increased. To improve the viability of acoustic dotplot techniques for larger problem sizes, we can exploit sparsity at multiple stages of the processing to construct an alternative $O(n^2)$ algorithm with significantly improved constants.

Generally speaking, the role of an acoustic model for speech is to map acoustic feature vectors to a more categorical representation. While the original feature vectors may be broadly distributed in $\mathbb{R}^d$, a good acoustic model will concentrate most of the posterior probability mass on a few confusable categories. Thus, the posterior vectors will be generally sparser (most values close to zero) than their acoustic vector counterparts. Our approach, also advocated in [6], is to replace the acoustic feature vector time series $X = x_1 x_2 \dots x_n$ with its posteriorgram image $Q = q_1 q_2 \dots q_n$. Here, each $q_i = \langle P(c_1|x_i), \dots P(c_p|x_i)\rangle \in \mathbb{R}^p$ is the posterior distribution for acoustic feature vector x_i over a set of p categories $\mathcal{C} = \{c_j\}_{j=1}^{p}$ (e.g. the set of English phonemes). Given this representation, we notice that we can approximate the inner

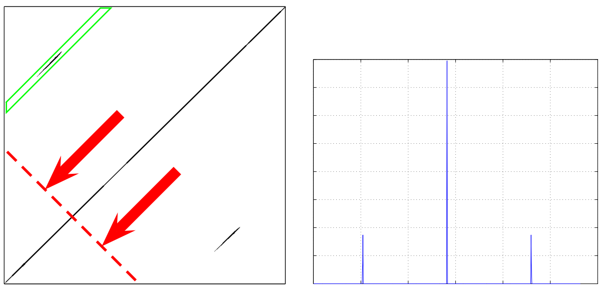

(a) (b)

Figure 2: (a) Dotplot after steps 1-3. (b) Hough transform.

product in Eq. 1 by performing the multiply and add only where the posteriorgrams exceed some small threshold ρ. Thus, we can compute the dotplot to a precision determined by ρ using the common inverted file technique:

1. Transform Q into p postings lists of the form $L_{c_j} = \{i|P(c_j|x_i) > \rho\}$ for each $c_j \in \mathcal{C}$, each consisting of the set of frame indices that threshold is exceeded.
2. For each c_j and each $a, b \in L_j$, increment M_{ab} by $P(c_j|x_a)P(c_j|x_b)$.

This approximation has time complexity $O(\sum_{j=1}^{p} |L_j|^2)$; in the limit of zero entropy posteriorgrams (maximally sparse) and assuming uniform class priors, this reduces to $O(n^2/p)$, a factor of p^2 times faster than the full computation. It should be noted that nothing in this prescription depends on the nature of the class set $\mathcal{C}$; while a phonetic acoustic model matched to the target language will have relatively low entropy, we expect mismatched/unsupervised acoustic models to be reasonably effective in the approximation as well.

Fig. 1(b) displays the posteriorgram dotplot counterpart to the acoustic dotplot in Fig. 1(a). Here, we have used a posteriogram over the English phone set computed using multi-layer perceptrons (MLP). Notice the resulting soft vector quantization introduce a complication: a block structure emerges from every run of a particular speech sound. In fluent speech, where each phone lasts around 20-100 ms, these blocks are relatively small; indeed, the boxed diagonal line segment, corresponding again to the repeat of *one million dollars*, can still be detected using the search method described in Sec. 2. However, when we are presented with long silence intervals or filled pauses, the large resulting blocks can produce a huge number of false alarms. Fortunately, these are easily filtered using speech activity detection and simple measures of posteriorgram stability.

Since the thresholded dotplots tend to be exceedingly sparse, sparse matrix storage is a reasonable way to loosen the $O(n^2)$ space constraints. Towards this end, we developed extremely efficient (both in time and space) sparse matrix versions of the percentile filter, Gaussian filter, and Hough transform algorithms required by the line segment detection procedure defined in Sec. 2.2. The details of these algorithms will be presented in a forthcoming article.

4. From Line Segments to Curves

Natural prosodic variation across repeated utterances of the same word or term can easily break the line segment assumption we have been working with. For this reason, the segmental dynamic time warping algorithm (S-DTW) of [4] was designed to perform the more computationally intensive search for curves in acoustic dotplots. If the curve deviation from the diagonal is reasonably constrained, this search can identify term repeats produced with different speaking rates and stress patterns.

While it is desirable to have this extra capacity, it comes at significant computational cost. Our solution is to adopt a two-pass strategy. First, we apply our optimized line segment search algorithm with a diagonal percentile filter length Δ short enough to detect syllable-sized repeats. Second, for each detected line segment, we apply S-DTW to find the curve that (i) passes through the midpoint of the line segment, and (ii) maximizes the similarity path integral.

Our application of the S-DTW algorithm introduces three additional parameters to the algorithm. The first is the maximal allowable deviation R (in frames) of the optimal warp path from the diagonal. Next, to impose the constraint that our optimal warp path passes through the first-pass line segment, we start from the segment midpoint and apply S-DTW both forward and backward in time with a dissimilarity budget of B in each direction (i.e. we stop growing the match when the path integral of $[1 - K(x_i, x_j)]$ exceeds B). Finally, we trim the ends of the optimal path by removing all initial or final points in the path that have a similarity less than T.

5. Experiments

Unlike previous research efforts in this direction, we frame our spoken term discovery evaluation in an information retrieval setting. Our goal is to quantify the ability of our search algorithm to extract a set of predetermined target terms in the speech that we identified with the transcript as both long (in number of characters) and as repeated in the corpus. Moreover, unlike previous studies that were implemented for 16 kHz read or lecture speech [4, 6], we developed and evaluated our system on the more challenging task of conversational telephone speech.

5.1. Evaluation and Implementation Details

We constructed our development set using 50 conversation sides (a single speaker per conversation) from the Switchboard corpus, each of duration ranging from 5-10 minutes. Using the transcription, we defined one target term per utterance chosen such that (i) each target term is repeated in the conversation side, (ii) each target term has three non-consecutive uppercase letters, (iii) each target is at least one second long, and (iv) each target term is the longest term (as text) in the file satisfying (i)-(iii). Example targets include *Ed Sullivan Show* (sw02102) and *The Incredible Journey* (sw02499). For evaluation, we constructed another set of 50 conversation sides from the Fisher English corpus and used the same target selection procedure.

As mentioned above, our spoken term discovery algorithm does not rely on the interpretability of the acoustic model class set $\mathcal{C}$. The only requirements are that the model map each acoustic feature vector to a relatively low entropy posterior distribution over the classes and that this map reasonably preserve locality. However, to isolate the efficacy of our search algorithm from the quality of the acoustic model, we compute highly sparse English phone posteriorgrams using the multi-stream MLP architecture of [7] trained on 300 hours of conversational telephone speech. In this approach, two acoustic front ends are run in parallel: one computes short time, 351-dimensional PLP features (39×9 frame context), while the second computes modulation frequency features computed using frequency domain linear prediction (FDLP) techniques. Each feature stream serves as input to one of two MLPs, whose output phone posteriorgrams are combined according to Dempster-Schafer theory

Table 1: Parameter values used for the single- and two-pass versions of the discovery algorithm (tuned on development set).

Parameter	Single-Pass	Two-Pass
Posteriorgram threshold, ρ	0.1	0.1
Percentile filter threshold, μ	40%	55%
Dotplot threshold, δ	0.65	0.0
Percentile filter length, Δ	0.5 s	0.5 s
Gaussian filter width, σ	30 ms	30 ms
Max S-DTW deviation, R	–	10
Dissimilarity budget, B	–	15
Path trimming threshold, T	–	0.6

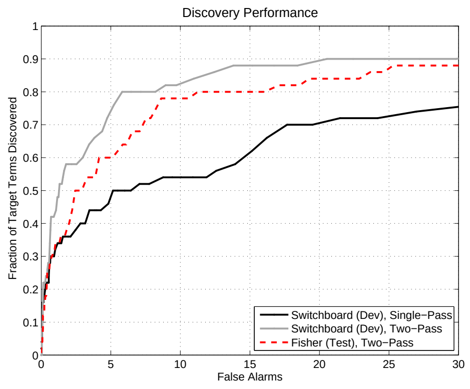


Figure 3: ROC curves for the development and test tasks. The second pass greatly improves performance.

of evidence. The incorporation of FDLP features, which model long range critical band trajectories, produce more accurate and significantly lower entropy posteriorgrams than if we use PLP features alone. This makes them the ideal input to our algorithm.

5.2. Results and Discussion

We proceeded by applying our search algorithms (both the single- and two-pass versions) to each utterance and labeled all non-target matches as false alarms regardless of their acoustic match quality. If we then sort our term discovery results by predicted duration, and threshold on the rank of the sorted list, we can sample a sort of ROC curve measuring target term recall versus the false positive count (i.e. the number of matches whose predicted duration was longer than the match we were targeting). The quality of this ROC curve on our Switchboard development task was used to manually tune the various algorithm parameters to the values shown in Table 1.

The Switchboard ROC curves using both the single-pass and two-pass algorithms are shown in Fig. 3. While, the single-pass search discovers 50% of the terms when making only 5 mistakes per 5-10 minute utterance, the significant improvement provided by the second pass demonstrates the importance of the tolerance to prosodic variability in conversational speech as provided by S-DTW. The two-pass algorithm achieves 75% recall with an average of 5 false alarms per utterance, a recall improvement of 25%. This means that a user would need to listen to only 6 seconds of each 5-10 minute utterance to have a 75% chance of hearing the target term. Note that this was accomplished unsupervised–the algorithm had no knowledge of the target term beyond the duration assumption.

It is important to note that even when our target was not ranked first, most higher ranking intervals were also reasonable acoustic matches. For example, in one instance (sw02499) a repeat of the phrase *critically acclaimed* is ranked higher than the target term *The Incredible Journey*. This is not a failure of our search algorithm; after all, *critically acclaimed* was faithfully repeated. Instead, it demonstrates the limitations of our assumptions regarding interval relevance. Still, the majority recall achieved with relatively low false alarm rates demonstrates that our relevance assumptions may be a good place to start.

Next, to test generalization, we applied the two-pass algorithm to the Fisher English evaluation set using the tuned parameter values from the development task. The resulting evaluation ROC curve is shown in Fig. 3. The small deviation in performance indicates that the choice of algorithm parameters is not highly sensitive to a variation in target set.

On a single grid node (one 3 GHz core, 2 GB RAM), the typical first-pass run time for a 10 minute utterance was approximately 10 seconds. This amounts to computing and searching for diagonal line segments on a 3.6 gigapixel dotplot. The second-pass run time for each first-pass match was approximately 3×10^{-4} seconds; thus, for a typical utterance, the second-pass added less than 1 additional second of processing time per 10 minutes of speech. Since the run-time grows quadratically with the input, we expect one hour of speech to take approximately *6 minutes* of processing time on a single machine. If we did not have the initial first pass filtering of the search space, we would be faced with the reported run time of over *33 machine-hours* using exhaustive S-DTW search on the same amount of speech [6].

6. Conclusions

We have presented an efficient two-pass search algorithm for discovering long, repeated acoustic patterns in speech. We find that our approach is proficient at discovering long ($\sim$ 1 s) repeated content terms in conversational telephone speech and runs over two orders of magnitude faster than the second pass alone. Our system has immediate relevance for information retrieval tasks for spoken documents, including unsupervised topic clustering, and may provide a more scalable first step in constructing speech recognition systems from zero resources.

7. Acknowledgments

The authors would like to thank Sriram Ganapathy and Samuel Thomas of the Johns Hopkins Department of Electrical and Computer Engineering for their help in computing the multi-stream MLP-based posteriorgrams used in the experiments.

8. References

[1] B. Varadarajan, S. Khudanpur, and E. Dupoux, "Unsupervised learning of acoustic subword units," in *ACL-08: HLT*, 2008.

[2] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513–523, 1988.

[3] K. Umemura and K. Church, "Repetition and language models and comparable corpora," in *ACL-IJCNLP Workshop*, 2009, http://www.fask.uni-mainz.de/lk/videoarchive/videos/2009-08-06-acl-ijcnlp-bucc-ken-church.html.

[4] A. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *IEEE T-ASLP*, vol. 16, no. 1, pp. 186–197, 2008.

[5] K. W. Church and J. I. Helfman, "Dotplot: A program for exploring self-similarity in millions of lines of text and code," *J. Comp. Graph. Stat.*, vol. 2, no. 2, pp. 153–174, 1993.

[6] Y. Zhang and J. R. Glass, "Towards multi-speaker unsupervised speech pattern discovery," in *Proc. of ICASSP*, 2010.

[7] S. Thomas, S. Ganapathy, and H. Hermansky, "Phoneme recognition using spectral envelope and modulation frequency features," in *Proc. of ICASSP*, 2009.