Spoken WordCloud: Clustering Recurrent Patterns in Speech

Rémi Flamary*
LITIS EA 4108, Université de Rouen
76801 Saint-Etienne-du-Rouvray, France
remi.flamary@insa-rouen.fr

Xavier Anguera and Nuria Oliver
Telefonica Research
Via Augusta 177, Barcelona, Spain
{xanguera, nuriao}@tid.es

Abstract

The automatic summarization of speech recordings is typically carried out as a two step process: the speech is first decoded using an automatic speech recognition system and the resulting text transcripts are processed to create a summary. However, this approach might not be suitable in adverse acoustic conditions or when applied to languages with limited training resources. In order to address these limitations, in this paper we propose an automatic speech summarization method that is based on the automatic discovery of recurrent patterns in the speech: recurrent acoustic patterns are first extracted from the audio and then are clustered and ranked according to the number of repetitions, creating an approximate acoustic summary of what was spoken. This approach allows us to build what we call a "Spoken WordCloud" *termed after similarity with text-based word-clouds. We present an algorithm that achieves a cluster purity of up to* 90% *and an inverse purity of* 71% *in preliminary experiments using a small dataset of connected spoken words.*

1 Introduction

Automatic speech recognition (ASR) systems have traditionally been the cornerstone modules to extract information from audio content. The transcript is first extracted and then analyzed to obtain information (*e.g. its topic, particular keywords, etc.*) or even generate a summary. Extensive research efforts have been devoted to improve ASR over the years. However, speech recognition results are still challenged by adverse acoustic conditions or when dealing with languages with limited training resources. Recent research addresses these limitations by analyzing the acoustic signal itself to find acoustic patterns that appear multiple times along one or multiple recordings. This approach was first proposed for speech by [11, 12] to augment ASR transcripts, and has also been used as a new computational model to avoid acoustic modeling by [13] or as a way to summarize the most important information in the audio by [6, 8].

In this paper we aim at automatically summarizing acoustic data by finding the most representative and recurrent sequences in the input speech. Our goal is to obtain, as a result of this process, the set of most often recurring short acoustic words[1] which represent the most prevalent terms in the content. This could be used, for example, to automatically classify the audio into topics or to group together several recordings with similar content. In previous similar research, Jansen *et al.* [6] consider that relevant recurrent information appears in long (1 second) sequences which are considered relevant even when appearing only twice in the audio. This assumption limits the representative segments only to long repetitions, which diverges from the WordCloud concept proposed here. Similar to our approach, they pay special attention to a fast implementation of the algorithm. Muscarielo *et al.* [8] incrementally perform a recurrent sequence discovery (which they call "motifs") by locally searching for repetitions in the audio around defined time intervals. Aside from local matches, they also keep a global library of long-term motifs. Like us, they aim at detecting short segments (> 0.6 seconds), but these are only stored as long-term library motifs if they occur at least twice within the local interval, thus being prone to missing segments that might still be relevant by occurring consistently over a long audio document but never frequently enough within a particular local segment.

The approach proposed in this paper is inspired by the algorithm proposed in [11, 12], where all recurrent short (~ 0.5 second) sequences are first found in the acoustic data, and then clustered to form homogeneous groups. As mentioned above, the main goal in [11] is to enhance speech

*This work was done while R. Flamary was visiting Telefonica Research.

[1] Although we refer to spoken words throughout the paper, as we do not use any knowledge about the language being processed, we are in fact detecting commonly repeated sequences which we hypothesize could correspond to actual words in that language, but might as well represent commonly occurring word sequences

recognition results towards information retrieval applications. Therefore their effort goes into maximizing cluster purity, but not paying special attention to inverse purity (*i.e.* how spread are acoustically similar segments among the final clusters). Instead, our goal is to obtain a set of the most representative spoken words in the acoustic data by building a list of the most recurrent acoustic segments, ranked by the number of times they occur. The proposed algorithm first extracts recurrent acoustic patterns from the speech by using an extension of the Unbounded-DTW algorithm proposed in [2]. Next, all found patterns are clustered to form what we refer to as a "spoken WordCloud" (given its similarity to text-based word-clouds). A comparison with the algorithm in [11] shows that the proposed approach is faster and more accurate when tested on this task by using a database composed of concatenated words spoken in isolation.

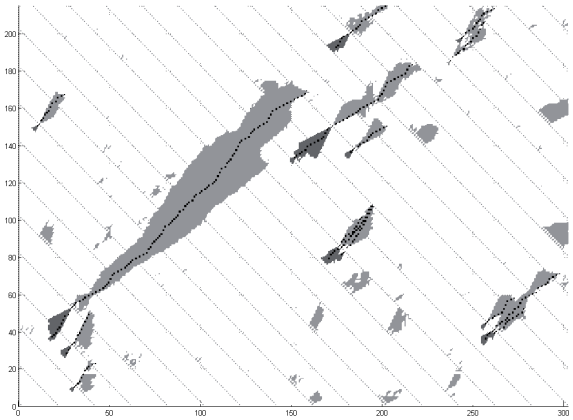

Figure 1. Resulting similarity matrix using U-DTW, where white areas are not computed

2 Segmental-DTW-Based Clustering

The segmental-DTW-based clustering in [11, 12] was shown to be able to find repeating sequences within an audio document that could be used to augment the speech recognition transcripts towards an information retrieval task. In this section we briefly review the main key points of their algorithm as we take is as the baseline for our system.

In a first step, unsupervised pattern discovery in speech is performed using the so-called *segmental-DTW* algorithm. Segmental-DTW works as follows: A distance matrix is first constructed by computing the Euclidean distance between all pairs of acoustic feature vectors from the two sequences to be compared (these can be the same sequence in case we are looking for repetitions in the same acoustic document). The choice of acoustic feature vectors varies between standard MFCC's in [11, 12] to phone posteriorgrams in more recent work by the same authors. Next, local optimum paths are discovered by applying standard DTW within each of several overlapping diagonal bands along the similarity matrix by uniformly setting start-end points along the matrix axes, and constraining the possible paths to a maximum deviation from the diagonal between these start-end points. Finally, for each band, the most similar subsequence – longer than a predefined minimum length – is returned.

In a second step, repeating sequences are clustered as follows: a similarity profile is first constructed by accumulating the resulting scores from the matching sequences with respect to time, and then selecting the most recurrent sequences at the local maxima of such profile. In their implementation they do not delimit any start-end times for the selected recurrent sequences. Finally in a third step, these sequences are set to be the nodes of a graph where its paths (and matching scores) correspond to the edges between nodes. This graph can also be represented as a sparse similarity matrix as later explained in section 3.3. The graph is clustered using the algorithm proposed in [9], which is fast to compute and has been shown to work well for community clustering.

Although the algorithm proposed in this paper closely resembles the general architecture proposed by [11, 12], their proposal favors the creation of many clusters with few sequences in each cluster, therefore with a high purity but low inverse purity (refer to Section 4.1 for a definition). This makes their algorithm not applicable to the creation of a "spoken WordCloud", where we are interested in a one-to-one relationship between the clusters and the actual number of unique repeating terms in the recordings.

3 Spoken WordCloud Algorithm

In this section we describe each of the building blocks that conform the proposed "spoken WordCloud" algorithm. As seen in Figure 2, the algorithm is composed of 3 main modules: (1) First we perform an automatic pattern discovery to detect audio segments longer than 0.5 seconds that repeat over time. We do this by using the previously proposed Unbounded Dynamic Time Warping (U-DTW) algorithm [2] and extend it to process long audio recordings; (2) Next, a similarity profile (see section 3.2) is created and the most repeated segments (which we will refer as *acoustic words* from now on) are obtained by finding the maxima of the profile and extracting their start-end points; (3) Finally, these words are clustered to form the resulting "spoken WordCloud", which is composed of a a ranked list of the most recurrent words in the audio, and the segments that appear in each of the clusters.

Note that the proposed approach differs from [11] in three aspects: the pattern discovery algorithm, the pro-

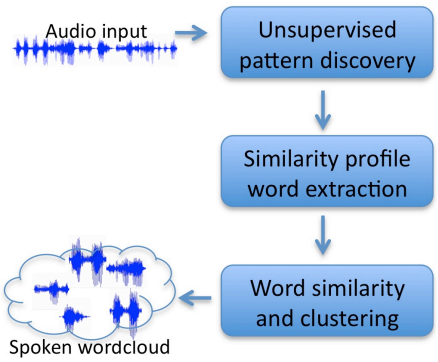

Figure 2. Spoken WorlClouds algorithm blocks diagram

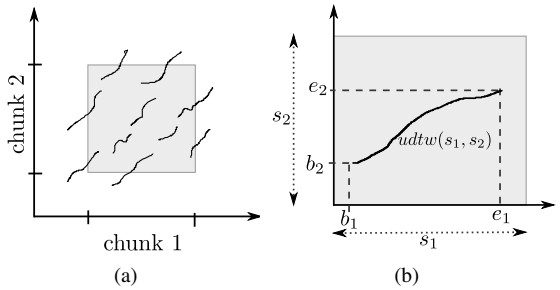

Figure 3. (a) U-DTW for large scale: paths are allowed to cross the borders of the chunk; (b) U-DTW similarity measure example between segment s_1 and s_2.

cess to obtain the start-end positions of selected acoustic words and the similarity metric between the different acoustic words in the clustering step (and the clustering algorithm itself). In this section we describe in detail each of the aforementioned differences.

3.1 Unbounded DTW (U-DTW) for Long Sequences

The first necessary step for unsupervised pattern discovery entails finding the acoustic sequences that repetitively appear in the acoustic document over time. In [2] we proposed the U-DTW algorithm that is able to find all repeated sequences in an acoustic signal longer than a given minimum duration. This algorithm differs from previous proposals [11, 7] in that it avoids the evaluation of the complete similarity matrix between the two sequences, hence significantly reducing its computational cost. Unbounded DTW leverages the minimum segment matching length set by design in order to avoid computing similarities between acoustic vectors which do not belong to a matching sequence. To do so, it computes an initial set of K similarities between both sequences whose locations need to be chosen to ensure that the path along the matrix of any pair of matching sequences will at least go through one of them. By carefully defining the locations where to compute these similarities (which we call synchronization points) we can achieve that $K << N \cdot M$, where N and M are the sizes of both compared sequences. As an example, Figure 1 shows the resulting similarity matrix for two instances of a sentence containing the word "Barcelona" with diagonal synchronization points. For every synchronization point that we suspect belongs to a matching segment (*i.e.* with a high similarity value) we perform a forward and backward path search until the entire matching path is found or the sequence is discarded because it is too short. Note that this approach ensures that the similarity between pairs other than the synchronization points is computed only when necessary. For more information on the algorithm, please refer to [2]. This algorithm automatically finds matching sequences in two acoustic segments but does not perform any clustering of the results, which we address in the rest of the paper.

A general problem of previous algorithms [2, 11, 7] is the quadratic memory requirements of the similarity matrices, which limit their efficiency and speed when processing very long recordings. In order to overcome this problem without constraining the possible patterns to be found, we automatically cut the sequence into manageable adjacent chunks and perform U-DTW on each of them with all the rest. As seen in Figure 3(a), an extra margin around each chunk is added to allow for any possible paths to be found even if crossing the chunk's borders. This way we ensure that no matching sequences will be lost in comparison to the original U-DTW algorithm.

3.2 Similarity Profile and Word Extraction

Given a list of matching paths and their scores, we wish to obtain the start-end times of the sequences in the speech segments that are matched. These will become the *acoustic words* that will be clustered and used to build the "Spoken WordCloud". We first compute a similarity profile, such as in [11], for all detected matching paths by adding their scores along time. This similarity profile contains, at any given time, the sum of similarities for all matching sequences that go through that instant. Therefore, the higher the values in the profile, the more recurrent the patterns are. Acoustic words are defined for each maxima in the similarity profile plot whose value is higher than a predefined minimum threshold. Note that we choose the maxima because we are using a similarity matrix, instead of choosing the minima if we used a distance matrix.

In order to determine the start-end times of the found

acoustic words, we gather all matching paths that cross such maxima and find the median values of all their starting and ending times. Although simple, this method turns out to be more robust than, for example, finding the local minima at either side of the maxima or taking the average (probably because the selected time corresponds to existing paths starting and ending times). In order to further constrain the acoustic words, we perform a simple energy-based speech activity detection on the input data and slightly adjust the start-end times or cut acoustic words in two when they interfere with a silence region.

3.3 Similarity Matrix Between Acoustic Words

All found acoustic words are clustered in order to find the most recurrent words. The simplest strategy would be to cluster together those acoustic words that share the most common matching segments. However, this would not return optimum results for our WordCloud as similar acoustic words are usually composed of an overlapping (but quite different) set of matching sequences. In this section we define an appropriate similarity measure between acoustic words and in the next section we explain possible clustering strategies that lead to more desirable WordClouds.

In [11], the authors use the similarity obtained between matching segments in the pattern discovery step, setting to 0 the similarity of all unmatched pairs. We shall denote this similarity K_p. Note that by ignoring the similarity between pairs that have not been matched during the discovery step, some important information might get lost. Hence and in addition to testing K_p, we propose to build a full similarity matrix $K_U(\mathbf{s}_1, \mathbf{s}_2)$ between any pair of acoustic words $\mathbf{s}_1$ and $\mathbf{s}_2$ by using the metric in Eq. 1.

$$K_U(\mathbf{s}_1, \mathbf{s}_2) = \frac{e_1 - b_1}{|\mathbf{s}_1|} \cdot \frac{e_2 - b_2}{|\mathbf{s}_2|} \cdot udtw(\mathbf{s}_1, \mathbf{s}_2) \qquad (1)$$

where $|\mathbf{s}_*|$ is the size (in frames) of sequence $\mathbf{s}_*$, b_* and e_* correspond to the optimum starting and ending points of the best U-DTW path computed between the two sequences, and $udtw(\mathbf{s}_1, \mathbf{s}_2)$ is the total U-DTW score similarity, averaged by the path length (See Figure 3(b)). Note that within U-DTW we use the dot product, therefore $udtw$ and the final similarity always fall within the $[0, 1]$ interval.

3.4 Clustering

In this paper, we evaluate two different clustering techniques: graph clustering and spectral clustering. The graph clustering technique was used in [11]. We have implemented it by means of the *Normalized Cuts* algorithm [4]. Note that this is different from the implementation in [11] because they require setting the value of a threshold (which leads to an unknown a priori number of final clusters) while in *Normalized Cuts* the final number of clusters is defined *a priori*. In general, graph clustering is commonly used for large scale clustering such as pixels in images. Its main interest with respect to standard spectral clustering techniques is that it is less computationally demanding as it does not require to perform a prior factorization of the similarity matrix.

Conversely, spectral clustering performs the classification of the segments in a smaller meaningful linear subspace [10]. In other words, a singular value decomposition of the similarity matrix is first performed in order to get its eigenvectors and eigenvalues. Then the matrix is projected on a small number of eigenvectors having the biggest eigenvalues in order to get the Principal Components of the matrix (PCA). Finally a standard clustering technique such as *k*-means can be applied on the projected segments. The possible limitation of this approach is an intractable dimensionality reduction when too many segments have to be clustered. In such cases one can use the large scale extension of this method as proposed in [5].

Note that the dimensionality reduction used in spectral clustering is useful to regularize the acoustic words by frequency. For instance, words that appear a lot will correspond to a principal direction, whereas unique or rare words are neglected by the dimensionality reduction. Such behavior is interesting when clustering a long speech segment if we want to extract only the most recurrent acoustic words. This dimensionality reduction can also be seen as a smoothing of the similarity matrix which helps in separating the acoustic words.

4 Experiments

4.1 Experimental Setup

In our experiments we used the MAMI dataset (see [3]), which we are making publicly available to the community[2]. It consists of recordings from 23 different speakers, each one uttering 47 different words, 5 times each, using a mobile phone in a noisy environment. Results using this database should be considered preliminary, given that they do not take into account the co-articulation effects of pronouncing several words together in natural speech. On the contrary, the background conditions in which each word is recorded are not ensured to be constant, unlike in other datasets like lecture recordings. We are planning on evaluating the proposed approach on larger datasets in our future work.

For each speaker we create a single long feature stream by concatenating the 235 spoken words, and apply a simple energy-based speech/silence detector to filter out silence.

[2] http://mm2.tid.es/mamidb/mamidb.tar.gz

Method	Pur.	IPur.	Fsc.	time (m)
Segmental DTW	0.757	0.556	0.645	110
Unbounded DTW	**0.802**	0.587	**0.675**	**70**

Table 1. Comparison between Segmental DTW and Unbounded DTW for Unsupervised Pattern Discovery.

The presence of silence is a big problem for unsupervised word discovery as all silence regions look very much alike. Note that no bias is introduced by the concatenation of words as the silence detector separates the words and the paths cannot cross the silences. For each file we extract 20 dimensional Mel-Frequency Cepstral Coefficients (MFCC) features every 20ms. As MFCC are speaker dependent we treat all speakers independently in our experiment, leaving the multi-speaker case to future work.

The performance measures used for the evaluation of the clustering are *purity*, *inverse purity* and *F-score* as explained in [1]. The *purity* focuses on the frequency of the most common category in each cluster. This is important as we want to prioritize having only one class of acoustic words in every given cluster. On the contrary, *purity* does not take into account how spread the acoustic words from a given class are in the clustering. For instance having one cluster per acoustic word would lead to a purity value of 1 but this is not desirable for a WordCloud application. Therefore, we also compute the *inverse purity*, which measures how accurate is the grouping of the acoustic words into the real classes. Finally, the *F-score* takes into account both the purity and the grouping of the true classes together.

One application of a spoken WordCloud is a summary of the spoken content with only the most representative/recurring acoustic words. Hence, it is of interest that these few acoustic words be as pure as possible. For this reason we also compute these metrics *only* on the n biggest clusters. In our experiments, we keep the $n = 40$ biggest clusters, which is slightly less than the real number of different words in the MAMI database for each speaker. Note that all results correspond to the average over the 23 subjects in the database.

4.2 Segmental vs. Unbounded DTW

In this section we compare Segmental-DTW and Unbounded-DTW for unsupervised pattern discovery and posterior clustering. We use a pipeline similar to the one used in [11]. All the parameters are exactly the same for both algorithms except for the path detection threshold that has been set to 0.7 for U-DTW and 0.6 for Segmental-DTW in order to obtain a similar number of paths in the pattern discovery step. The minimum path size is set to 0.4 seconds for both, and the minimum similarity for a maximum in the similarity profile to be considered as an acoustic word is set to 1. This means that at least 2 paths crossing the sequence are necessary to consider a sequence to be a word. An N_{cuts} graph clustering is used with 60 clusters in both cases.

Results are shown in Table 1. The purity, the inverse purity and the F-score are all better with U-DTW ($\sim 4\%$ absolute improvement). Furthermore, the last column of the table shows the execution time – in minutes – for the pattern recognition on all 23 subjects, showing that the U-DTW is 40% quicker than Segmental-DTW. Here the time needed for word detection and clustering is neglected due to the small number of words (≈ 1 min per subject). All these results show the advantage of using U-DTW for pattern detection and posterior clustering over the Segmental-DTW algorithm for the WordCloud task.

4.3 Similarity and Clustering

In this section we investigate the use of the proposed similarity measures and two different clustering strategies (graph and spectral) using the U-DTW algorithm. From the obtained acoustic words we first compute the standard similarity matrix K_P proposed by [11] and the U-DTW similarity matrix K_U proposed in section 3.3. We then compare the clusters using each similarity matrix and also using the average of the two. Both the graph and spectral clustering algorithms are set to find 60 clusters.

Results are shown on Table 2. First, we observe that on this dataset, the spectral clustering works better than the graph clustering. Concerning the similarity measure, the U-DTW measure alone is not as good as the standard one, but combining the two leads to a *purity* improvement of 2% absolute. Note that the three measures used (purity, inverse purity and F-score) show similar improvements for the different clustering approaches and similarity measures. This shows that we are not promoting one aspect of the clustering over others, but rather evaluating the global efficiency.

Previous results correspond to a fixed number of overall clusters, but in a real life application, the total number of spoken words is usually not known *a priori*. Figure 4 shows results for different numbers of clusters when using the best method ($K_U + K_P$ for similarity and spectral clustering). The dotted lines correspond to the results of keeping only $n = 40$ biggest clusters. We can see that when the number of clusters increases the purity increases and the inverse purity decreases. But when keeping only the 40 biggest clusters, the purity remains more constant while the inverse purity still increases, leading to an overall better F-score. This shows that regardless of the total number of clusters we use in the clustering of a speech recording, our final WordCloud (containing only the n-best clusters) will usually be a good summarization of that recording.

	Graph					Spectral				
Sim. Meas.	Pur.	Pur._{40}	IPur.	IPur._{40}	Fsc.	Pur.	Pur._{40}	IPur.	IPur._{40}	Fsc.
K_P	0.80	0.78	0.59	0.67	0.68	0.88	0.86	0.69	0.80	0.74
K_U	0.82	0.81	0.61	0.71	0.69	0.86	0.84	0.68	0.78	0.73
$K_P + K_U$	0.84	0.84	0.62	0.72	0.71	**0.90**	**0.89**	**0.71**	**0.82**	**0.77**

Table 2. Results obtained for different similarity measures and clustering (60 clusters).

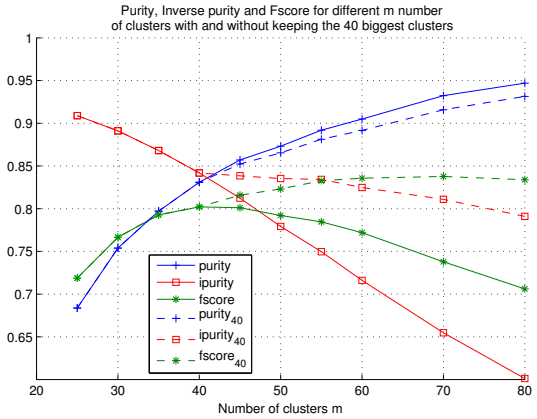

Figure 4. Purity, inverse Purity and F-score for different number of clusters with $K_U + K_P$ and spectral clustering.

5 Conclusions and Future Work

Automatic speech recognition (ASR) systems have traditionally been the cornerstone modules to extract information from audio content. However, ASR may be very challenging under adverse acoustic conditions or for languages with limited resources. Recent research address the problem by looking directly at the signal and finding repetitive content that can be later used, for example, to help augment and improve ASR results or to summarize the acoustic content with representative acoustic snippets. In this paper we are interested in obtaining a representation of the most prevalent spoken words in the acoustic input by building a list of the most recurrent acoustic segments, ranked by the number of times they occur. We refer to the output of our system as "Spoken WordClouds" given its similarity with text-based word-clouds. We have proposed important modifications to a well known pattern discovery and clustering algorithm, originally used to augment ASR transcripts, to make it more suitable for this application. First, we have used U-DTW for unsupervised pattern discovery with good results in terms of computational time and clustering performances. Next we have investigated the use of a new similarity measure between acoustic words and the use of two different clustering algorithm. We have evaluated the proposed approach in a small database and obtained up to 90% cluster purity and 71% inverse purity. Future work will entail testing the algorithm on recorded lectures and meetings, working on a speaker-independent version and evaluating the meaningfulness of the obtained WordClouds as a summary of the content.

Aknlowledgement

During the development of this work X. Anguera was partially funded by the Torres Quevedo Spanish program

References

[1] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486, 2009.

[2] X. Anguera, R. Macrae, and N. Oliver. Partial sequence matching using an unbounded dynamic time warping algorithm. In *Proc. ICASSP*, 2010.

[3] X. Anguera and N. Oliver. MAMI: Multimodal annotations on a mobile phone. In *Proc. Mobile-HCI*, 2008.

[4] I. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proc. SIGKDD*, pages 551–556. ACM, 2004.

[5] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:214–225, 2004.

[6] A. Jansen, K. Church, and H. Hermansky. Towards spoken term discovery at scale with zero resources. In *Proc. Interspeech*, Makuhari, Japan, 2010.

[7] M. Muller and D. Appelt. Path-constrained partial music synchronization. In *Proc. ICASSP*, 2008.

[8] A. Muscarielo, G. Gravier, and F. Bimbot. Audio keyword extraction by unsupervised word discovery. In *Proc. Interspeech*, 2009.

[9] M. E. J. Newman. Fast algorithm for detecting community structure. *Phys. Rev. E*, 69:066133, 2004.

[10] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[11] A. Park and J. R. Glass. Towards unsupervised pattern discovery in speech. In *Proc. ASRU*, 2005.

[12] A. Park and J. R. Glass. Unsupervised pattern discovery in speech. *IEEE Trans. Acoustics, Speech and Language Processing*, 8(1):186197, 2008.

[13] L. ten Bosch and B. Cranen. A computational model for unsupervised word discovery. In *Proc. Interspeech*, 2007.