Unsupervised Speech Segmentation

Jevgenija Aksjonova
jevaks@kth.se

Sebastian Bujwid
bujwid@kth.se

Abstract

The goal of unsupervised speech segmentation is to find boundaries between speech units, such as words, syllables and phonemes. In this project report we present an overview of the state of the art approach used to perform an unsupervised word segmentation. We also include the results of our own word segmentation experiments on TIDigits corpus [1].

1 Introduction

The unsupervised discovery of linguistic units from speech is an important research direction, since supervised learning algorithms require a large amount of labeled data, which is typically hard to obtain. Besides, children learn to recognize different acoustic units in an unsupervised way within the first year of their life, which provides an additional inspiration for the researchers and developers of speech processing systems.

1.1 The Zero-Resource Speech Challenge

The Zero-Resource Speech Challenge 2015 [2] was organized to encourage the development of unsupervised speech technologies by providing a common open-source evaluation scheme so that the proposed solutions could be easily compared. The challenge consisted of two tracks, subword modeling and spoken term discovery. The aim in the first track was to construct a good feature representation of speech units that maximizes phoneme discriminability. The second track addressed the problem of word segmentation, which is a subject of this report.

Initial approach to spoken term discovery relied on comparing speech sequences using dynamic time warping algorithm (DTW) and performing an exhaustive search over the whole data space [3]. The efforts has been made to improve the time complexity of this approach by utilizing more efficient computational techniques. One of such systems ([4]) was chosen as a baseline for the second track in the Zero-Resource Speech Challenge. The system proposed in [4] evaluates similarity between vectors of row acoustic features using Segmental-DTW and randomized approximation algorithms to reduce the search space.

Two research groups submitted their solutions for the second track of the challenge. Lyzinski et al. [5] used a Segmental-DTW-based term discovery procedure introduced by the authors of a baseline system [4]. Authors experimented with using different acoustic features as an input, but the main focus of their research was on utilizing advanced graph clustering methods to group discovered speech segments into word-representing clusters.

Räsänen et al. [6] proposed a solution, which slightly outperformed the opponent and was acknowledged as highly original [7]. Their system segmented speech signal into syllable like units by tracking changes of an amplitude of the signal. They represented all the segments with feature vectors of the same length by choosing subsamples from the Mel Frequency Cepstral Coefficient (MFCC) features of each segment. Then they grouped discovered segments into clusters using k-means algorithm and, finally, grouped recurring combinations of similar syllables into words.

All solutions were evaluated on English and Xitsonga datasets. English corpus consisted of speech records of 6 male and 6 female speakers (4h59m05s in total); Xitsonga corpus — of speech records of 12 male and 12 female speakers (2h29m07s in total) [2].

1.2 Segmental Bayesian model

The solution of Räsänen et al. showed great results at the Zero-Resource Speech Challenge. Though the second part oh their algorithm, which includes clustering with k-means and grouping, seems oversimplified. Kamper et al. [8] were able to achieve better results for the datasets of the Zero-Resource challenge by combining syllable detection with Bayesian Gaussian mixture model (GMM).

The Segmental Bayesian model was introduced by Kamper et al. in their earlier work [9]. Similarly as in [6] speech segments of arbitrary size were represented by fixed-length feature vectors. They modeled the distribution of feature vectors with Bayesian GMM and chose segmentation of the speech signal which was highly probable according to the model. This method was evaluated on TIDigits dataset and showed very promising results.

At first, quite sophisticated feature representation was used and all the possible segmentations of a speech signal were considered. Authors acknowledged that their system needed modifications before it could be applied to the larger datasets, therefore in their later work [8] they reused a syllable detection part from [6] and used less sophisticated downsampling of feature vectors.

1.3 The goal of this project

The goal of this project was to implement an unsupervised speech segmentation system and test it on the TIDigits dataset. We chose to implement a system, which combines syllable detection with Bayesian GMM as proposed in [8].

First of all, we wanted to verify that syllable detection proposed in [6] can produce a good set of potential segment boundaries. Further, we wanted to answer if it is beneficial to perform segmentation with the Segmental Bayesian model from [9, 8] applying it on already extracted syllables. Lastly, we wanted to try a deterministic approach instead of stochastic sampling used in [9, 8] in order to train a GMM model. We thought that sampling might be redundant in the case, when detected syllable boundaries are used as landmarks for potential segmentation.

In order to get a fair comparison, all systems were evaluated on TIDigits corpus.

2 Method

As defined in Kamper et al. [8, 9] in Segmental Bayesian Model, acoustic features $y_{1:M} = y_1, y_2, \ldots, y_M$ are extracted from an acoustic signal with function f_a. Then, segments of an arbitrary length are transformed into a fixed-dimensional space, using an embedding function f_e. Embeddings $\mathcal{X} = \{x_i\}, x_i = f_e(y_1, \ldots, y_{s_i})$ are clustered using the Bayesian GMM model. The objective is to obtain such model configuration, in which each cluster would represent a word. The model can be trained by applying an iterative procedure, in which segment boundaries and model parameters are simultaneously updated.

This general framework is illustrated in figure 1.

2.1 Acoustic features and embedding function

We used MFCC features as an output of function f_a.

A simple embedding function f_e can be defined as proposed in [10]. A segment is split into a fixed number of intervals of equal size (typically 5-10) and for each interval a mean feature vector is computed. Obtained mean vectors are concatenated to form an embedding x_i. In order to convey the information about the length of a segment we added one more element to the end of the vector x_i as in [6]. This element is equal to $n/3 \log(l)$, where n is the number of intervals and l is the length of a segment.

Any function that maps segments of an arbitrary length into fixed dimensional vectors can potentially be used as f_e. However, a good embedding function should ensure a good speech sound discrimination in the output space.

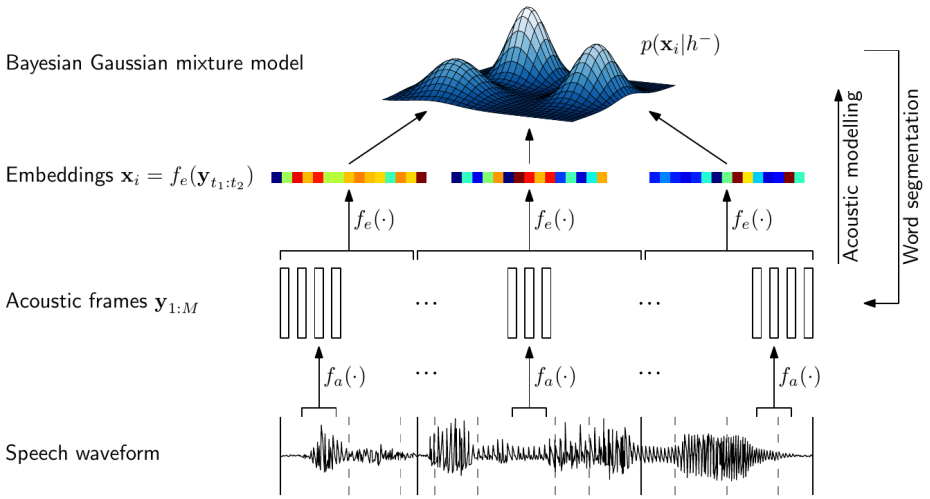


Figure 1: Segmental Bayesian Model overview (taken from [8])

2.2 Segmental Bayesian model

A Bayesian GMM model with Dirichlet prior is formally defined as:

$$\begin{aligned} \pi &\sim Dir(a/K\mathbf{1}) \\ z_i &\sim \pi \\ \mu_k &\sim \mathcal{N}(\mu_0, \sigma_0^2\mathbf{I}) \\ x_i &\sim \mathcal{N}(\mu_{z_i}, \sigma^2\mathbf{I}), \end{aligned} \tag{1}$$

where z_i indicates the component to which $\mathbf{x}_i$ is assigned.

Kamper et al. sample the component assignments z_i for embeddings $\mathcal{X} = \{x_i\}$ using a collapsed Gibbs sampler as explained in [11], section 24.2.4.1. Probability of assigning x_i to the GMM component k is

$$P(z_i = k|\mathbf{z}_{\setminus i}, \mathcal{X}; a, \beta) \propto P(z_i = k|\mathbf{z}_{\setminus i}; a)p(\mathbf{x}_i|\mathcal{X}_{k\setminus i}; \beta), \tag{2}$$

where $\beta = (\mu_0, \sigma_0^2, \sigma^2)$, $\mathbf{z}_{\setminus i}$ represent all component assignments excluding z_i and similarly $\mathcal{X}_{k\setminus i}$ is the set of embedding vectors assigned to component k except for $\mathbf{x}_i$. $p(\mathbf{x}_i|\mathcal{X}_{k\setminus i}; \beta)$ is the likelihood function, which can be estimated from the GMM model. $P(z_i = k|\mathbf{z}_{\setminus i}; a)$ is the prior probability, which can be calculated according to the formula

$$P(z_i = k|\mathbf{z}_{\setminus i}; a) = \frac{N_{k\setminus i} + a/K}{N + a - 1}, \tag{3}$$

where $N_{k\setminus i}$ is number of embeddings in mixture component k not counting $\mathbf{x}_i$.

Given the GMM model new segmentation boundaries are sampled using forward filtering backward sampling dynamic programming algorithm [12].

In our system we tried a different approach and used deterministic procedures instead of sampling. In this case, a component assignment z_i is defined as the most likely assignment according to the GMM model. Similarly, the most likely segmentation boundaries are obtained using a well-known Viterbi algorithm [13].

Further, we present a pseudocode of the algorithm almost in the same form as it was given in [9].

2.3 Syllable detection

Syllable detection algorithm can be incorporated into the described system to significantly reduce the number of potential segmentation boundaries.

Algorithm 1 Segmental Bayesian model

Choose an initial segmentation (e.g. random)
for $j = 1$ to J **do**
  for i = $randperm(1$ to $S)$ **do**
    Select utterance s_i.
    Remove embeddings $\mathcal{X}(s_i)$ from acoustic model.
    Update word boundaries for s_i, yielding new $\mathcal{X}(s_i)$.
    for embedding x_i in $\mathcal{X}(s_i)$ **do**
      Obtain z_i for embedding x_i.

In [6] Räsänen et al. evaluated 3 different syllable detection methods. The method, which showed the best results is Amplitude Envelope-driven Oscillator. The algorithm was inspired by neurophysiological models of speech perception.

First of all, an envelope of a speech signal has to be computed by downsampling the signal to 1000Hz and applying a low-pass finite impulse response filter. Then, the syllabic rhythm of speech is modeled as a simple damped harmonic oscillator, driven by the amplitude of speech envelope. Behavior of the oscillator is described with following equations:

$$\begin{aligned} f(t) &= e(t) - kx(t-1) - cv(t-1) \\ a(t) &= f(t)/m \\ v(t) &= v(t-1) + a(t)\Delta t \\ x(t) &= x(t-1) + v(t)\Delta t \end{aligned} \tag{4}$$

where $e(t)$, $f(t)$, $a(t)$, $v(t)$ and $x(t)$ denote amplitude of the envelope, force, acceleration, velocity, and displacement of the oscillator at time t. Parameters $k = 1$, $c = \Delta f\sqrt{m}/f_0$, $m = 1/(4\pi f_0^2)$, $f_0 = 4Hz$ and $\Delta f = 8Hz$ are set so that the model approximates the syllabic rhythm of speech.

The boundaries between syllables correspond to is to the local minimum points of the displacement $x(t)$ shifted backward by approximately 70 ms. An example of syllable detection is given in figure 2. This is a speech record of a woman pronouncing a sequence of digits "1393387" from the TIDigits corpus.

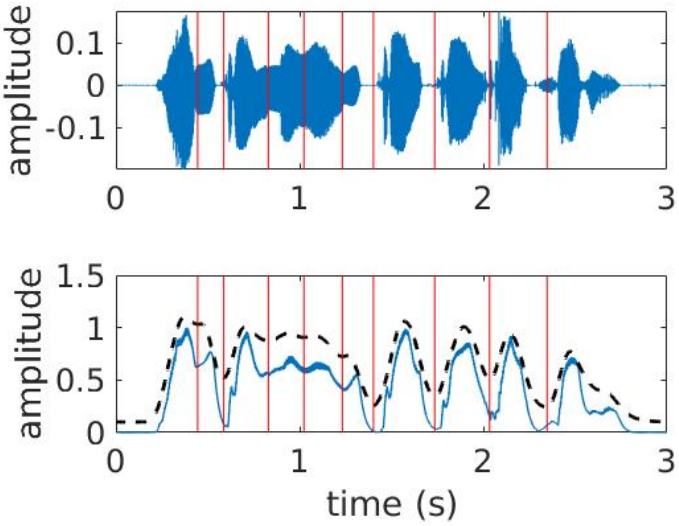


Figure 2: Syllable boundaries detected by the Amplitude Envelope-driven Oscillator. The upper plot shows an audio signal, the lower plot shows an amplitude envelope (blue line) and the oscillator displacement (dashed line). Red vertical lines show detected boundaries.

3 Experiments

3.1 Evaluation

In order to evaluate the segmentation systems, as a reference we obtained an almost correct word segmentation of TIDigits corpus [1] by performing forced alignment using GMM-HMM model. In further description recognized words will serve as labels to obtained ground truth segments.

We implemented three evaluation metrics also used in [8, 9].

Average cluster purity: To compute this measure every detected segment is assigned a label of the ground truth segment with which it overlaps the most. Then, every cluster is assigned a label, which has been assigned to the majority of it's segments. The segment is considered to be classified correctly if it's label matches a label of the cluster. The total amount of matching segments is then divided by the total number of segments. In our interpretation, if a detected segment overlaps with any other ground truth segment for less than one half, it corresponds to silence ("sil").

Word error rate (WER): To compute this measure word clusters are labeled as in average cluster purity. Then, each detected segment is assigned a label of the corresponding cluster. Labels of detected segments form word sequences, which are aligned with ground truth word sequences. The Levenshtein distance between two aligned sequences is defined as $dLev(a, b) = S + D + I$, where S, D and I is the minimal number of substitutions, deletions and insertions required to transform one sequence to another. WER was computed as

$$WER = \frac{1}{N} \sum_i dLev(a_i, b_i),$$

where N is the total number of words in the ground truth. Usually, to compute WER only one cluster is mapped to one label, other clusters, which are left without the mapping, are considered as errors. We also were interested in the results in the case of many-to-one mapping and computed a measure, which are further referred to as WER_m (as in [8]).

Word boundary precision, recall and F-score: These measures show how many detected segment boundaries match the true word boundaries from the forced alignment. A boundary is considered to be correct, if it falls within a small distance from a true boundary. We used a distance equal to 0.04s. Precision is a number of detected true boundaries divided by the number of detected boundaries, recall is the number of detected true boundaries divided by the number of true boundaries and F-score is a harmonic mean of precision and recall. This measure is computed by considering all segments except for those, which correspond to clusters labeled as silence ("sil").

3.2 Experimental setup

For the purpose of evaluation of the algorithms, the test subset of the TIDigits corpus was used. It consists of recordings from 113 different speakers (56 male and 57 female). We trained the models in a speaker independent setting, which means that all the samples were used simultaneously disregarding an information about the speaker identity or gender.

Having the MATLAB code for the system [6], which was made publicly available by the authors, we modified data input procedures to make it suitable for the TIDigits dataset. Additionally, we applied a threshold to the oscillator values in order to detect the beginning of the first word in the sequence and the ending of the last. As it can be seen in figure 2 side boundaries are not detected by the algorithm as they do not correspond to local minimums. This is particularly important for TIDigits dataset, as most of the sequences are very short. Further, this system is referred to as *SyllableSegOsc*. To make the algorithm more suitable for the TIDigits dataset, the number of syllable clusters was set to 25 and minimum token frequency for each n-gram order to [1,400]. In this case, the algorithm produced 28 clusters in most of the cases.

MFCC features and syllable boundaries obtained from *SyllableSegOsc* were used in our system as an input to the Segmental Bayesian model. We implemented a simple embedding function f_e as described in the subsection 2.1 with number of subintervals equal to 10 as in [8]. Each segment was represented by 121 feature coming from top 12 MFCC coefficients. We trained a GMM model with 25 components by using all syllable boundaries as initial segmentation and running the algorithm described in the subsection 2.2 for 15 iterations ($J = 15$). Values used for parameters were as

following: $\sigma = 0.1$, $a = 1$, $\mu_0 = 0$, $\sigma_0^2 = \sigma^2/0.005$. All the values were set as proposed in the literature, except for σ, which was increased. Further, the system is referred to as *SimpleSyllableGMM*.

The code for the system proposed in [8] was also obtained by us, as it was made publicly available by the authors. We provided the MFCC features and detected syllable boundaries as an input for this algorithm and set the number of clusters to 25. Although the general framework developed in [8] was utilized by us, due to high complexity of the original system, we did not aim to replicate it in every detail. However, it is interesting for us to include evaluation of this system for comparison. Further, this system is referred to as *SyllableGMM*.

4 Results

The results of our experiments are summarized in table 1. Presented average values and standard deviations were obtained in 5 independent trails for each system.

We can see that each method outperformed the others in one of the aspects. The method of Räsänen et al. *SyllableSegOsc* showed the lowest word error rates, our method *SimpleSyllableGMM* showed the best results in cluster purity and the method of Kamper et al. *SyllableGMM* showed the highest recall in detecting word boundaries.

Table 1: Evaluation for speaker-independent models.

Method	Cluster Purity	WER	WER_m	Word boundaries		
				Precision	Recall	F-measure
SyllableSegOsc	0.559 ± 0.009	$\mathbf{0.623 \pm 0.020}$	$\mathbf{0.507 \pm 0.017}$	0.297 ± 0.001	0.295 ± 0.001	0.296 ± 0.001
SimpleSyllableGMM	$\mathbf{0.656 \pm 0.010}$	0.684 ± 0.005	$\mathbf{0.507 \pm 0.018}$	$\mathbf{0.366 \pm 0.001}$	0.285 ± 0.007	0.320 ± 0.005
SyllableGMM	0.593 ± 0.012	0.724 ± 0.023	0.575 ± 0.020	0.358 ± 0.003	$\mathbf{0.338 \pm 0.004}$	$\mathbf{0.348 \pm 0.003}$

The metrics for word boundaries are of course highly depended on tolerance distance from the true boundaries. As shown in figure 3, recall is very sensitive to allowed distance, the higher it is the higher recall is. The figure was created for detected syllables and should be treated as a upper bound for word segmentation recall. For a tolerance, which was used in evaluation (0.4s) the upper bound is 0.397. However, for a tolerance distance 0.1s recall reaches 0.890.

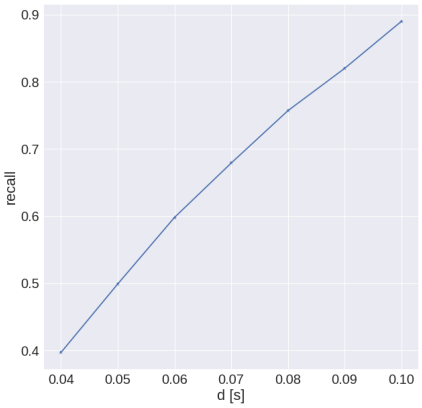


Figure 3: Recall of detected syllables for different allowed distance from the true boundaries.

Two examples of segmentation produced by *SimpleSyllableGMM* were presented on tables 2 and 3, as well as on figure 4.

Table 2: Segmentation example 1.

Filename:	$tidigits/disc\_4.2.1/tidigits/test/man/bn/11o646oa.wav$										
Syllable boundaries:	0.13	0.21	0.38	0.39	0.87	1.54	1.96	2.44	2.52	2.75	3.10
Word boundaries:		0.21		0.39			1.96	2.44		2.75	3.10
True word boundaries:		0.18	0.37		0.82	1.55	1.89	2.39		2.78	3.10
Detected classes:	11	18		5			24	3		19	
Detected words:	sil	1		sil			4	6		o	
True words:		1	1		o	6	4	6		o	

Table 3: Segmentation example 2.

Filename:	$tidigits/disc\_4.2.1/tidigits/test/woman/gk/7723439a.wav$										
Syllable boundaries:	0.44	0.61	0.72	0.92	1.06	1.26	1.98	2.36	2.76		3.40
Word boundaries:		0.61		0.92		1.26		2.36	2.76		3.40
True word boundaries:	0.49			0.85		1.19	1.99	2.33	2.64	2.97	3.44
Detected classes:	6	23		23		5		24	9		
Detected words:	sil	7		7		sil		4	9		
True words:	7			7		2	3	4	3	9	

First of all, we can see that most of the true word boundaries can be matched to detected syllable boundaries. One exception is a missing boundary between 3 and 9 (2.97s) in the example 2. This is the reason, why the 6^{th} digit in this sample was not detected by the algorithm.

Secondly, the algorithm tends to merge subsequent word segments into one. For instance, words "1o6" in the first example and words "23" in the second example were merged into one segment corresponding to cluster 5. Segments in this cluster consisted of 2.89 syllables on average. As these segments did not represent any particular pattern and were longer than true word segments, the cluster was labeled as "sil".

Nevertheless, in some cases syllables were successfully combined by the algorithm. For instance, both digits 7 in the second example were recognized as one word, despite the fact that a word "seven" consists of two syllables. Segments in the corresponding cluster (23) consisted of 1.66 syllables on average.

5 Discussion and Conclusions

Before conducting the experiments, we expected to see the increase in performance positively correlated with the increase in complexity of the method. This would mean that our system would show results, which are somewhere in between *SyllableSegOsc* and *SyllableGMM*. However, table 1 shows that all the compared systems performed quite similarly. Moreover, by changing parameters of the model it is possible to enhance the performance according to one criterion while sacrificing the other. Therefore, it could be possible to obtain even more similar results by changing parameters, especially between our method and *SyllableGMM*.

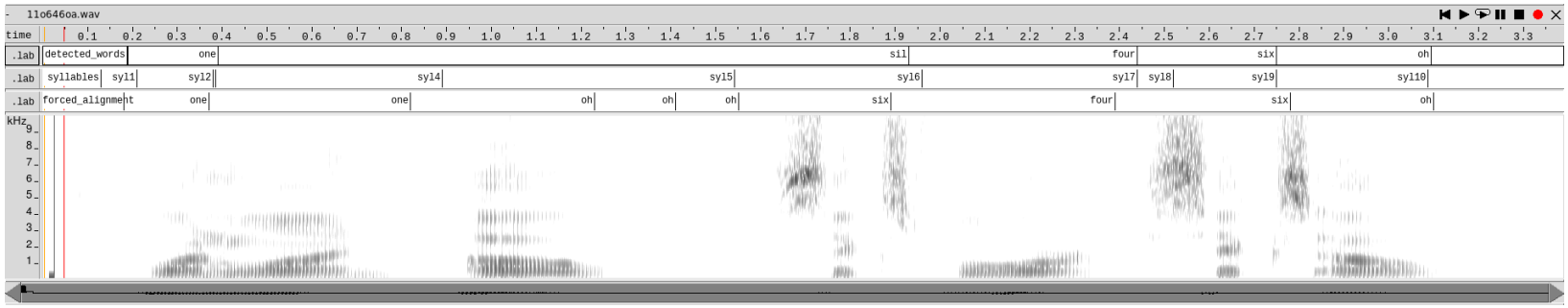
In our experiments a syllable segmentation algorithm proposed in [6] detected almost 90% of word boundaries within 0.1s tolerance. This might be considered as a good result, given that the reference segmentation was obtained using forced alignment and probably contained imprecisions.

Kamper et al. ([9], [8]) used sampling to obtain new segmentation boundaries and component assignments for the GMM model. Our method *SimpleSyllableGMM*, on the contrary, was implemented to make deterministic choices corresponding to the highest probability. The results obtained with *SimpleSyllableGMM* were similar (even slightly better) than the results obtained with *SyllableGMM*.

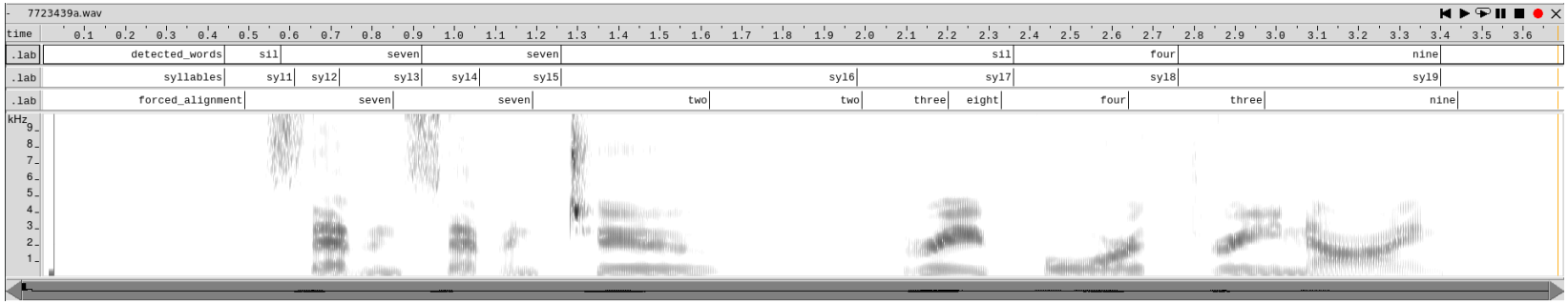
Furthermore, applying GMM model to group detected syllables into words did not lead to significant improvements comparing to a combination of k-means and n-grams used in [6]. Segmentation examples showed that the algorithm tended to over-merge the segments, causing errors. Partially, this could happen because of the poorly chosen embedding function f_e. This function took an average of MFCC features over a predefined number of intervals. If a segment is too long, too much valuable information can be lost in the averaging process.

On the other hand, a tendency to produce larger segments could force clusters to represent words instead of syllables. As the TIDigits corpus consists mostly of monosyllabic words, it is hard to tell if it indeed is the case. In order to answer this question, the method must be tested on a dataset containing more multisyllabic words.

When Kamper et al. proposed the Segmental GMM model in [9], they evaluated their method on the TIDigits dataset. The results obtained in their work were much better than the results of our experiments. The key advantage of their method was a sophisticated embedding function f_e based on DTW similarity with a reference set. This is another reason to believe that the performance of the word segmentation system developed in this project could be further improved by applying a more discriminative embedding function.



(a) *tidigits/disc_4.2.1/tidigits/test/man/bn/11o646oa.wav*



(b) *tidigits/disc_4.2.1/tidigits/test/woman/gk/7723439a.wav*

Figure 4: Example segmentation on spectrogram

References

[1] R. Leonard. A database for speaker-independent digit recognition. In *ICASSP '84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 9, pages 328–331, Mar 1984.

[2] Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan-Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. The zero resource speech challenge 2015. In *INTERSPEECH*, pages 3169–3173, 2015.

[3] Alex S Park and James R Glass. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197, 2008.

[4] Aren Jansen and Benjamin Van Durme. Efficient spoken term discovery using randomized algorithms. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pages 401–406. IEEE, 2011.

[5] Vince Lyzinski, Gregory Sell, and Aren Jansen. An evaluation of graph clustering methods for unsupervised term discovery. In *INTERSPEECH*, pages 3209–3213, 2015.

[6] Okko Räsänen, Gabriel Doyle, and Michael C Frank. Unsupervised word discovery from speech using automatic segmentation into syllable-like units. In *INTERSPEECH*, pages 3204–3208, 2015.

[7] Maarten Versteegh, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. The zero resource speech challenge 2015: Proposed approaches and results. *Procedia Computer Science*, 81:67–72, 2016.

[8] Herman Kamper, Aren Jansen, and Sharon Goldwater. A segmental framework for fully-unsupervised large-vocabulary speech recognition. *arXiv:1606.06950 [cs]*, June 2016. 00003 arXiv: 1606.06950.

[9] Herman Kamper, Aren Jansen, and Sharon Goldwater. Unsupervised word segmentation and lexicon discovery using acoustic word embeddings. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(4):669–679, 2016.

[10] Keith Levin, Katharine Henry, Aren Jansen, and Karen Livescu. Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 410–415. IEEE, 2013.

[11] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[12] Steven L. Scott. Bayesian methods for hidden markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97(457):337–351, 2002.

[13] G David Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.