Automatic segmentation of TIMIT by dynamic programming

Van Zyl van Vuuren*, Louis ten Bosch† and Thomas Niesler*

*Department of Electrical and Electronic Engineering
University of Stellenbosch, South Africa
Email: {15446204,trn}@sun.ac.za

†Department of Linguistics
Radboud University Nijmegen, The Netherlands
Email: l.tenbosch@let.ru.nl

***Abstract*—We propose an algorithm based on the principle of dynamic programming for the automatic segmentation of continuous speech into phoneme-like units. A measure of local dissimilarity among consecutive feature vectors is combined with a knowledge of the expected statistical distribution of the segment lengths within a dynamic programming framework to obtain an optimal placement of segment boundaries. We compare the performance of our algorithm with the performance of two recently-proposed alternatives by measuring how closely the hypothesised boundaries match the TIMIT phone boundaries. The results showed that we are able to improve on the performance of the two contrasting approaches. Furthermore, we show that a hybrid approach which combines aspects of all three algorithms leads to even better results.**

I. Introduction

The task of accurately segmenting a speech signal into phoneme-like units plays an important role in the speech processing field. Although accurate manual segmentation can be achieved by trained phoneticians, the task is tedious, expensive and intrinsically subjective. In HMM-based ASR systems, time-aligned phonetic transcriptions are often needed for the development of the pronunciation dictionary and acoustic models. This is not always feasible, and is a particular obstacle for the development of ASR systems for under-resourced languages, for which no, or very little, transcribed phonetic material is available. In these situations, automatic segmentation algorithms can accelerate the task of developing a pronunciation dictionary and obtaining suitable bootstrapping acoustic training data, thereby substantially reducing the time it would take to develop the ASR system. The availability of reliable automatic segmentation algorithms is also useful in technologies outside ASR, such as the study of pronunciation variation and the development of coherent large-scale dictionaries.

Several approaches to the automatic segmentation of speech have been proposed over the years. Some require prior training, relying for example on HMM forced alignments [1]. Others make use of previously stored speech segments for template matching by using the phonetic transcription [2]. A third and more prevalent class of algorithms rely solely on the acoustic information to detect transient events in the speech signal [3]–[7]. When considering a under-resourced setting in which speech corpora are unavailable or very small, model training may not be feasible. Under these circumstances this latter class of algorithms represents the most viable option.

In this paper we propose a new algorithm for the acoustic segmentation of speech based on the principle of dynamic programming (DP). DP-based segmentation has been proposed in [3], in which a distortion metric within segments is minimised by using prior knowledge of the number of phones in a sequence. The algorithm we propose requires no information regarding the number of phones and maximises the probability of a specific segment boundary sequence.

A well known class of speech segments are phonemes, the identification of which is the goal of most published segmentation algorithms. By using the annotated phoneme boundaries given in TIMIT, the acoustic characteristics in the vicinity of the phoneme boundaries as well as the lengths of the phonemes can be inspected. The proposed algorithm then uses this prior information to infer the probability of a boundary occurring at every specific point in time in a speech signal. Dynamic programming principles are then applied to detect the most probable sequence of boundary positions.

Section II gives a brief overview of the class of segmentation algorithms based on transient events in acoustic information, and includes a discussion on a few recent algorithms. Section III provides a detailed description of the proposed DP-based segmentation algorithm, and Section IV discusses the quality measures used to assess segmentations. The experimental setup is specified in Section V, and experimental results are given in Section VI. Finally concluding remarks are presented in Section VII.

II. Background

Many segmentation algorithms are based on the assumption that there are regions in speech, termed speech segments, where the acoustic features stay relatively constant, and that there are clear transitions between such regions. To detect these transitions, the algorithms employ some estimate of the local acoustic change in the signal. ‘Local’ in this context refers to temporal acoustic changes taking place at a specific time independent of any previous or future acoustic changes within the signal. A function that quantifies these local acoustic changes will be referred to as the **local score** function in the remainder of this text. The local score function is central to all acoustic segmentation methods, and therefore different types of local score functions and their application in the recent literature will briefly be reviewed.

A. Algorithms based on maximum local acoustic change

The most common approach used in speech segmentation is to hypothesise segment boundaries at the times at which local acoustic change is at a maximum. These local maxima are found by searching for the peaks in the local score. However, the local score may contain many small peaks, which are the result of small acoustic changes that do not necessarily indicate segment boundaries. These additional peaks can lead to *over-segmentation*, where more than one segment boundary is hypothesised while only one is truly present. Over-segmentation can be reduced by including a threshold below which peaks are ignored. A selection of segmentation algorithms falling into this category are reviewed in the following. They were specifically chosen to illustrate a diversity of local score functions, of which a selection will later be compared experimentally. The local score will henceforth be denoted as LS in equations.

1) Räsänen et al. [4]: The local score function used in this algorithm is the cross correlation between two FFT magnitude vectors. This is shown in Equation 1, where f and g represent the FFT magnitude vectors for the frames to the left and to the right respectively of the investigated frame, F_j.

$$LS(F_j) = \frac{f.g}{\parallel f \parallel \parallel g \parallel} \tag{1}$$

Feature vectors that are similar will give a score close to 1, and dissimilar vectors will give a score closer to 0. The algorithm applies a non-linear filter to the cross-correlation sequence in order to quantify the degree of uniformity in the region preceding and following the point of interest. In a similar way, the dissimilarity between these regions is also determined. The difference between the dissimilarity and uniformity values leads to a signal of which the valleys corresponds to probable segment boundaries. However, this signal is very noisy, and there are many small valleys. The number of these smaller valleys is reduced by application of a 'minmax' filter, which searches a fixed region (n_{mm}) around the point of interest to find the local maximum and minimum values. The difference between this maximum and minimum serves as the output of the filter at the position of the minimum. This filter is applied throughout the signal in non-overlapping regions. The filter output is a signal of which the peaks represents possible boundaries. Given that the 'minmax' filter region is usually very small and applied in non-overlapping intervals, many closely spaced peaks may still remain. Temporal peak masking is therefore applied in a subsequent step. Two peaks falling within a determined interval (t_d) of each other and which are above a chosen threshold (p_{min}) are identified, and the highest peak retained. The location of the highest peak is also shifted a small distance toward the eliminated smaller peak in proportion to their amplitudes.

2) Ten Bosch et al. [5]: This work uses the angle between the smoothed feature vectors just before and just after the point of interest to quantify the degree of local change. This is given by Equation 2, where f and g are the averages of the two feature vectors before and after the frame of interest F_j respectively.

$$LS(F_j) = arccos \frac{f.g}{(\parallel f \parallel \parallel g \parallel)^{\frac{1}{2}}} \tag{2}$$

12 MFCC and log energy together with their first and second derivatives are used as a 39-dimensional feature vector. All local maxima above a threshold (δ) are hypothesised as boundaries.

3) Estevan et al. [7]: This algorithm employs maximum margin clustering to detect points of change in a feature vector consisting of 12 MFCC coefficients, log energy and their first and second derivatives. A sliding window, N frames wide and centered about the frame of interest, sweeps through the signal. MMC clustering (using a RBF kernel) is applied to the frames within this window. The width of the RBF kernel, W, is estimated from a development set. The MMC clustering results in a cluster label for each frame within the window, and changes in these labels indicate possible segment boundaries. It was found that the best way to detect these changes is by using the Euclidean distance, as given by Equation 3, between the cluster labels and the cluster means. Let f be the cluster label of each frame within the sliding window, and g be the mean of the cluster throughout the signal. Peaks in the Euclidean distance will then indicate the segment boundaries.

$$LS(F_j) = [\sum_{l=1}^{T} (f_l - g_l)^2]^{\frac{1}{2}} \tag{3}$$

4) Sarkar et al. [6]: This method differs from the previous three by operating in the time domain rather than the frequency domain. The local score function used in this case is the average level crossing rate. The level crossing rate is closely related to the zero crossing rate, but with multiple additional levels other than $y = 0$, and among which the average crossing rate is taken. The levels can be distributed uniformly or non-uniformly. For this choice of local score, a boundary corresponds to a valley rather than a peak. As for some of the preceding algorithms, a threshold is required to prune out shallow valleys which lead to over-segmentation.

B. Algorithms based on minimising a distortion metric

Another approach to speech segmentation is to increase the uniformity within segments, i.e. to minimise some distortion metric within segments. In the work by Sharma et al. [3], the local score is the Euclidean distance applied to MFCC features. The distortion within a segment is calculated by Equation 5. This calculation employs the local score at frame j, given by Equation 3, and the mean of the local score from frame i to n, given by Equation 4. The segment stretching from frame i to frame n is denoted by $S_{i,n}$.

$$M_{i,n} = \frac{1}{n-i+1} \sum_{j=i}^{n} LS_j \tag{4}$$

$$distortion\_metric(S_{i,n}) = \sum_{j=i}^{n} (LS_j - M_{i,n})^2 \tag{5}$$

The overall distortion of the speech signal is a cumulative sum of the distortions of all the segments. The overall distortion can be minimised by applying a level-based DP algorithm to search for the optimal segmentation, assuming that the number of levels (segments) in the signal is known.

C. A proposed local score

For our formulation of the segmentation problem it is convenient if the local score lies between the values of 0 and 1. We propose the use of a normalised city block distance as shown in Equation 6,

$$LS(F_j) = \frac{\sum_{l=1}^{T} |f_l - g_l|}{\sum_{l=1}^{T} |f_l| + \sum_{l=1}^{T} |g_l|} \tag{6}$$

where f and g are the feature vectors before and after the frame of interest Fj. This proposed formulation of the local score will be compared with other candidates in the experimental evaluation. Note that parameterisations for f and g are not specified, allowing different feature vectors to be used during experimentation.

III. A DP-based segmentation algorithm

Most segmentation algorithms based on maximum local-acoustic changes are prone to over-segmentation because they hypothesise more than one segment boundary at a point of acoustic change. This occurs due to the presence of multiple local maxima in the local score. To counteract this, the algorithms include various types of thresholds to eliminate such very short segments. Several examples of such measures were given in Section II. These remedies are adhoc, however, and introduce additional parameters into the algorithm that require optimisation.

The algorithm we propose includes an explicit probabilistic model for the length of a segment. Segments that are either very short or very long are penalised by their associated low probability. The probability distribution of phoneme lengths for TIMIT can be estimated from the phonetic annotations, as illustrated in Figure 1. For illustrative purposes, the distribution is normalised with respect to its maximum probability.

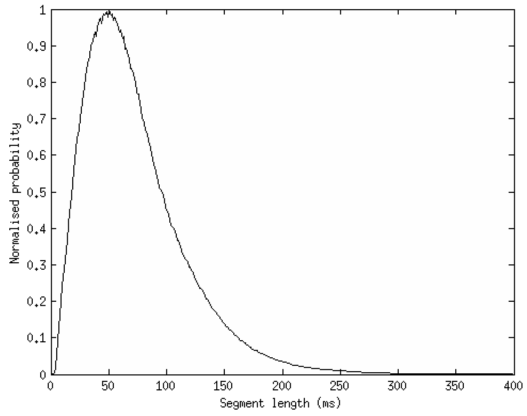

Fig. 1. Probability distribution of phoneme lengths in the TIMIT training set [8].

A. Segment probability

To gain some insight into the behaviour of local scores near segment boundaries, the local score in the close vicinity of phoneme boundaries, as given by the TIMIT annotations, is calculated and used to estimate a local score probability distribution given a boundary. A similar distribution is determined for the local score values taken far from boundaries, i.e. a local score probability distribution given that there is no boundary. Figure 2 shows these distributions, each normalised with respect to its maximum probability, for the local score calculated with Equation 6 when using FFT magnitudes as the feature vector. The distributions of the local score and the phoneme length can now be used to determine the probability of a boundary occurring at a specific frame in a speech signal.

Consider a signal consisting of N+1 frames. Now let the time of occurrence of each frame correspond to a state of a HMM as shown in Figure 3, where M is the maximum allowed number of frames per segment and S_0 is the time of occurrence of the first frame of the signal. The vertical dashed arrows between S_1 and S_1, S_2 and S_2, and between S_{N-1} and S_{N-1} indicate an expansion of the same HMM state.

When a state is visited by a path through the Markov model shown in Figure 3, a segment boundary is considered to occur at the corresponding speech frame. The transition and emission probabilities are calculated according to Equations 7 and 8 respectively, where SL refers to the segment length, LS to the local score, and SB to the occurrence of a segment boundary.

$$a_{i,j} = P(S_j|SL(S_j,S_i)) \tag{7}$$

$$b_j = P(SB|LS(S_j)) \tag{8}$$

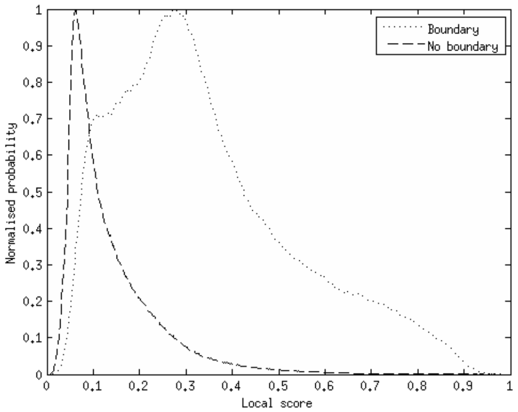

Fig. 2. Probability distribution estimates of local score values at, and away, from phoneme boundaries for Equation 6 applied to the FFT magnitudes.

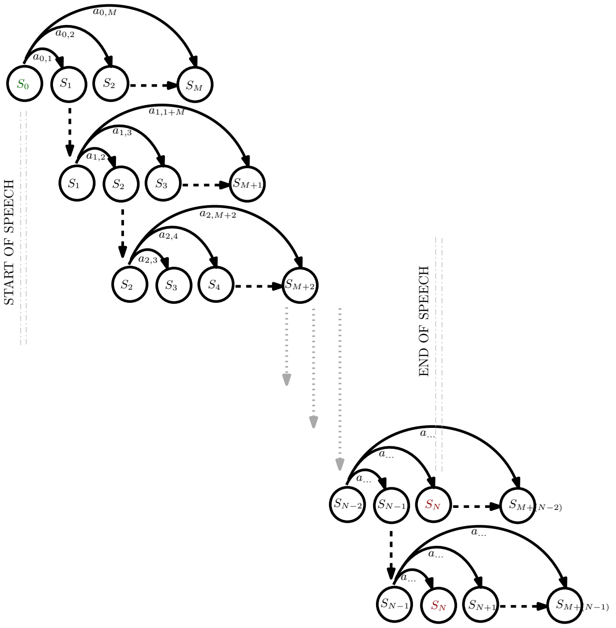

Fig. 3. DP-based segmentation cast as a HMM.

The segment length in Equation 7 is equal to the time step between two consecutive frames multiplied by the number of states separating the currently visited state and its parent state, as shown in Equation 9, where S_j is the current state, and S_i is the parent state.

$$SL(S_j,S_i) = (j-i) * step \tag{9}$$

Hence the transition probability is dependent only on the elapsed time between states. The emission probability at state Sj, as shown in Equation 8, is dependent on the local score $LS(Sj)$. To calculate the emission probability, Bayes rule is applied as shown in Equation 10, where $!SB$ refers to the absence of a segment boundary.

$$P(SB|LS(S_j)) = \frac{P(LS(S_j)|SB)P(SB)}{P(LS(S_j)|SB)P(SB) + P(LS(S_j)|!SB)P(!SB)} \tag{10}$$

The prior probability of a segment boundary can be estimated by dividing the number phoneme boundaries in the TIMIT annotations by the number of frames, as shown in Equation 11.

$$P(SB) = \frac{number\ of\ phoneme\ boundaries\ in\ TIMIT}{number\ of\ frames\ in\ TIMIT} \quad (11)$$

The probability that a boundary occurs at a particular frame can now be calculated by using Equations 9 and 10 in conjunction with estimates of the various probability distributions.

B. Optimal path

To find the globally optimal path from S_0 to S_N, all possible transitions shown in Figure 3 must be considered. This can be accomplished by using a DP algorithm. The states that were visited along the optimal path will identify the optimal segmentation. It is important to note that S_0 and S_N are always included in the path, and therefore the algorithm assumes that segment boundaries are always present at the start and the end of the speech signal. This means that any initial and final silence must be removed before applying the algorithm.

C. Normalising for path length

During the Viterbi decoding, many probabilities are multiplied together for any given path. When determining the optimal path, shorter paths (which contain fewer multiplications and thus longer segments) may be preferred, even when these have low associated emission and transition probabilities. We compensate for this effect by modifying the emission and transition probabilities as shown in Equations 12 and 13.

$$a_{i,j} = P(S_j|SL(S_j, S_i))^{SL(S_j,S_i)} \quad (12)$$

$$b_j = P(SB|LS(S_j))^{SL(S_j,S_i)} \quad (13)$$

These modifications normalise the path probability and remove the bias towards segmentations containing fewer segment boundaries.

IV. Assessing segmentation accuracy

In order to assess the quality of automatic-generated segmentations, we will determine how closely they correspond to the TIMIT phonetic segmentations. This provides a useful measure of segmentation accuracy. However it is dependent on the segmentation conventions used in TIMIT. For example, even though it is common practice for the /p/ to be segmented as a single phone in human annotations, the silence (closure) associated with the stop is considered a separate acoustic event in TIMIT. We found that the automatic segmentation algorithms could detect these closures quite accurately, and therefore decided to adhere to the original 61 TIMIT phone definitions without modification.

A. Comparing segmentations by DP

Comparing two sequences of segment boundary times can again be achieved by DP. We will proceed by first determining the best alignment between two sequences of boundary times. Then we will use this alignment to calculate a path cost. The alignment procedure uses a matrix of path costs as shown in Figure 4.

The first boundary in both sequences must coincide, and this corresponds to the bottom left cell of the matrix. Three alternative scenarios are then considered: (i) a hypothesised boundary $P_H(i)$ is paired (matches) a boundary $P_R(j)$ in the reference segmentation, (ii) a hypothesised boundary $P_H(i)$ is not paired with any boundary in the reference transcription (insertion) or (iii) there is no hypothesised boundary that can be paired with a boundary $P_R(j)$ in the reference transcription (deletion).

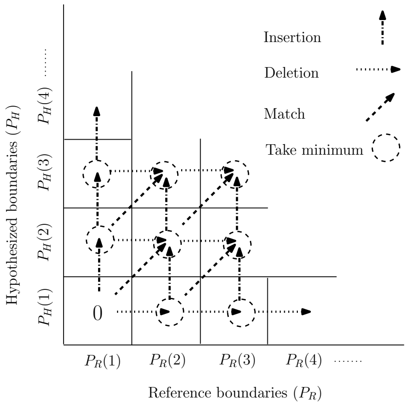

Fig. 4. Alignment matrix for segmentation scoring.

All possible paths from the bottom left to top right in the matrix shown in Figure 4 are computed recursively by dynamic programming. Starting from the bottom left of this matrix, each path can be extended upwards, to the right, or diagonally up and to the right, indicating an insertion, a deletion or a match between boundaries respectively. Each of these possibilities has a specific associated cost. When a reference boundary falls between two hypothesised boundaries, or vice versa, the cost is calculated by considering the distance to the nearest of the two boundaries. When paths meet, only the path with the lowest cost survives.

This procedure is applied iteratively, until all paths have reached the top right cell, which will then contain the final alignment cost between the two sequences. This cost reflects the difference between the hypothesised and reference sequences since it is the cumulative cost of every match, insertion, and deletion in the alignment. Furthermore, the cost has dimensions of time. By dividing it by the number of reference boundaries, the cost in seconds per reference boundary can be obtained. This is the average time difference between a paired hypothesised- and reference boundary and it will be used as a figure of merit in our later experiments. In addition, the number of insertions, deletions, and matches can be obtained by tracing back along the optimal path.

B. Fixed margin method

It appears to be standard practice in related research to consider a hypothesised and a reference segmentation boundary to be a match whenever they occur within 20ms of one another [4]. All non-matching boundaries are then either insertions or deletions. In order to make our results more directly compatible with those of others, this scoring framework has also been employed. An error measure termed the **average error** is defined, which is the average of the percentage insertions and deletions taken with respect to the number of reference boundaries in a speech signal. Furthermore, this interpretation of insertions, deletions and average error will be used.

V. Experimental setup

A. Data

Our experimental evaluations are based on the TIMIT database. The development set specified in [8] was used to optimise all parameters, and the core test set defined in [8] was used exclusively for final testing. There is no speaker overlap between these two sets. The use of an explicit development set avoids biased results which would be obtained if the performance of the algorithm was measured on the same data used to optimise its hyperparameters. In the literature dealing with automatic segmentation, the separation of development and testing data was found not to be common. Leading and trailing silences were removed to account for the assumption that each utterance begins and ends with a segment boundary.

B. Feature vectors

We have chosen three feature vector configurations popular in literature on automatic speech segmentation for comparative experimentation.

1) FFT: Unprocessed 128-point FFT magnitudes
2) MFCC: 12 MFCCs and log energy
3) MFCC+Δ+$\Delta\Delta$: MFCC with appended first and second derivatives

By considering the local scores separately for the MFCCs, for the delta and for the acceleration features, it was found that a peak for the MFCCs or the acceleration components always coincides with a valley for the delta component, and vice versa. To account for this, the overall local score was calculated by averaging the local scores calculated for MFCCs and acceleration components, and the negative of the local score for the deltas.

C. The local score

Three local scores were investigated:

1) The cosine distance (C) shown in Equation 1,
2) The Euclidean distance (E) shown in Equation 3, and
3) The normalised city block distance (NCB) shown in Equation 6.

In our experiments, f and g were taken to be the averages of two frames to the left and two to the right of the inspected frame respectively. Depending on the local score, boundaries are expected to occur at either peaks or valleys (local maxima or minima) of the local score. Equation 10 is therefore only calculated at frames which coincide with local maxima or minima of the local score and a probability of 0 is assigned to all other frames.

D. The probability weights

As it stands, the DP segmentation algorithm will give equal weight to the transition and emission probabilities, due to the segment length and local score respectively. However, it may be beneficial to shift the balance more strongly towards one or the other. By multiplying the log values of the emission and transition probabilities by positive constants that sum to one, this shift in balance can be achieved, and will allow deletions to be traded for insertions and vice versa. Optimal performance on the development set was achieved by assigning a heavier weight to the emission probability (0.6–0.7) than to the transition probability (0.3–0.4). This gives a stronger preference to higher emission probabilities and leads to a reduction in insertions.

VI. Experimental results

A. Smoothing window size

In the following experiments a frame size of 16ms and a frame shift of 4ms were used. Before calculating the local score, each resulting MFCC and FFT value were smoothed by taking the average within a window centered on the feature vector in question. Subsequently, the average DP cost (Section IV-A) and the average error (Section IV-B) was calculated on the development set for different smoothing window sizes applied to different local score and feature vector combinations. Figures 5, 6, and 7 respectively show these results for the cases in which the cosine distance is applied to the FFT, the normalised city block distance is applied to MFCCs, and the Euclidean distance is applied to MFCC with first and second derivatives. For each configuration, all other parameters were optimised on the development set.

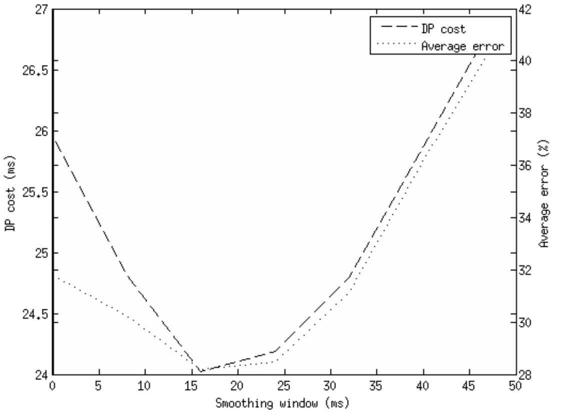

Fig. 5. DP cost (Section IV-A) and average error (Section IV-B) for different smoothing window sizes on the development set for the cosine distance applied to the FFT.

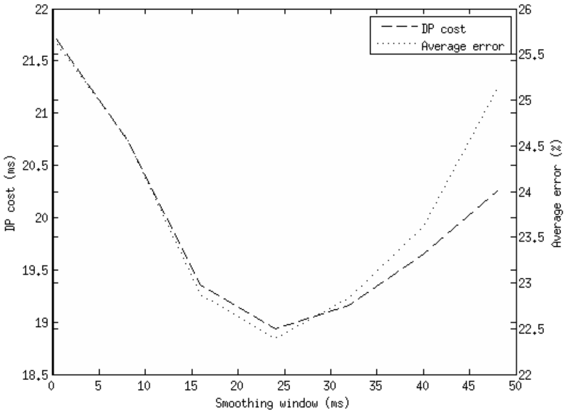

Fig. 6. DP cost (Section IV-A) and average error (Section IV-B) for different smoothing window sizes on the development set for the normalised city block distance applied to the MFCCs.

The results show that the optimal smoothing window sizes are similar for the FFT and MFCC parameterisations (16–24ms). A longer window (around 40ms) is required by the MFCC+Δ+$\Delta\Delta$ parameters, however. We believe that the introduction of first and second differentials introduces additional local maxima into the local score, which can lead to an increase in insertions. By lengthening the smoothing window, this is compensated for.

B. Choice of feature vector and local score

The performance of the DP segmentation algorithm when using the three different feature parameterisations and the three different local score formulations was compared experimentally, and results are shown in Table I. For each configuration, the length of the smoothing

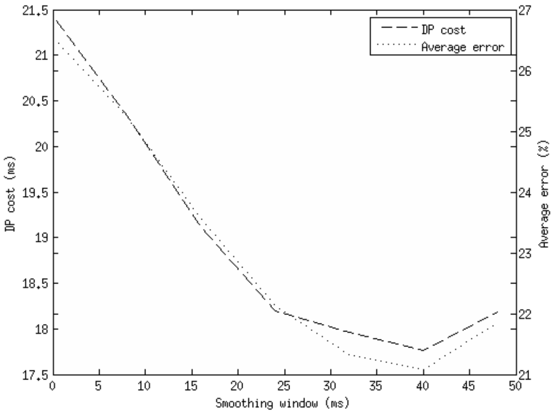

Fig. 7. DP cost (Section IV-A) and average error (Section IV-B) for different smoothing window sizes on the development set for the Euclidean distance applied to the MFCC with their first and second derivatives.

window as well as the probability weights are optimised on the development set, and segmentation accuracies determined on the test set. Both the DP path cost, in milliseconds per reference boundary, and the fixed margin average percentage error are shown.

TABLE I

DEVELOPMENT- AND TEST-SET PERFORMANCE OF THE DP SEGMENTATION ALGORITHM FOR THREE CHOICES OF FEATURE VECTOR AND FOR THE NORMALISED CITY BLOCK (NCB), EUCLIDEAN (E) AND COSINE (C) LOCAL SCORE (LS) FORMULATIONS.

Configuration		**DP Cost (ms)**		**%ERR**	
Feature	**LS**	**Dev**	**Test**	**Dev**	**Test**
FFT	NCB	18.62	18.42	20.21	19.80
MFCC	NCB	18.94	18.82	22.40	22.80
MFCC+Δ+ΔΔ	NCB	19.28	18.83	21.81	22.07
FFT	C	24.02	24.13	28.20	27.62
MFCC	C	19.01	18.98	22.53	22.85
MFCC+Δ+ΔΔ	C	18.94	18.72	21.56	21.72
FFT	E	28.06	27.93	33.27	33.13
MFCC	E	18.49	18.22	22.67	22.85
MFCC+Δ+ΔΔ	E	17.77	17.58	21.08	21.40

The normalised city block distance delivers the best overall performance. When applied to the MFCC+Δ+ΔΔ parameterisation, the Euclidean distance achieved similar performance. A configuration that stands out from the rest is the normalised city block distance applied to the FFT, which greatly outperforms all other combinations with the FFT feature vector. The FFT in general is the feature which is most sensitive to the remaining parameters, and was seen to be prone to over-segmentation. The FFT also has a higher dimensionality than the other parameterisation. It appears from the results that the normalised city block distance is most robust to this variation in dimensionality. Thus, the the normalised city block distance with a weighting leaning towards the emission probability (0.7) to reduce insertions gives very promising results. Among the feature parametrisations, the MFCC and the MFCC+Δ+ΔΔ are most competitive.

When comparing performance on the development and on the test sets, it is evident that the same patterns emerge from both. In the experiments that follow, each local score's best overall performing configuration will be used. These are the normalised city block distance for the FFT, the cosine distance for the MFCC+Δ+Δ, and the Euclidean distance for MFCC+Δ+Δ. These will henceforth be referred to as configuration C1, C2, and C3 respectively.

C. Silence removal

Many TIMIT sentences contain regions of silence in which temporal changes nevertheless occur. In order to avoid the hypotheses of segment boundaries in these regions, all boundaries were removed at frames when the ratio of the average energy content from 30ms before to 30ms after the frame in question, to the mean energy of the signal fall below a certain threshold. Different threshold values were investigated, and a typical result is shown in Figure 8. A threshold of 0.2% (i.e. a value of 0.002) delivered optimal performances for all configurations.

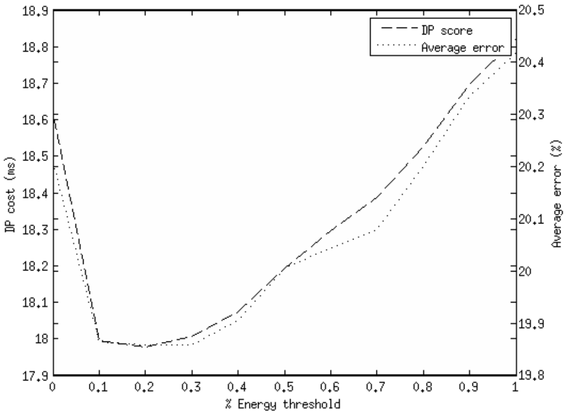

Fig. 8. DP cost (Section IV-A) and average error (Section IV-B) against % energy threshold on the development set for configuration C1.

D. Comparison with other segmentation algorithms

In the previous sections, an optimal configuration for the DP segmentation algorithm proposed in this paper is determined by experimentation. In this section we will benchmark the performance of this optimal configuration against two recent approaches to speech segmentation found in literature [4] [5]. Both approaches belong to the class of segmentation algorithms that rely on transient events in the acoustical information, as described in Section II-A. The method described in [4] claimed to achieve the same or better performance than many earlier approaches, while [5] is an algorithm with which the authors have had good prior experience.

Each method compensates for silences in its own way. The algorithm given in [5] scales the local score by the log frame energy to attenuate points of low energy, while the algorithm in [4] uses a similar approach to that proposed in this paper, but uses the average energy measured over the interval from -8ms to +30ms about the point of interest, and a threshold which is a multiple of the minimum energy for the signal. In the evaluation presented in the following, the parameters of each method were optimised on the development set.

Table II presents the DP cost in milliseconds per reference boundary, the percentage insertions and deletions with respect to the number of reference boundaries, and the average error for the optimised cases on the development set. The values shown for configurations C1, C2 and C3 are those achieved after silence removal.

When applying these parameter values to the core test set, the results shown in Table III are obtained.

TABLE II

PERFORMANCE COMPARISONS ON THE DEVELOPMENT SET AFTER SILENCE REMOVAL.

Method	DP Cost (ms)	% Ins	% Del	%ERR
DP (C1)	17.98	15.56	24.16	19.86
DP (C2)	18.12	15.28	26.97	21.13
DP (C3)	17.04	18.15	23.05	20.60
Räsänen	18.91	17.92	26.99	22.46
ten Bosch	25.07	26.19	27.37	26.78

TABLE III

PERFORMANCE COMPARISONS ON THE CORE TEST SET AFTER SILENCE REMOVAL.

Method	DP Cost (ms)	% Ins	% Del	%ERR
DP (C1)	17.92	14.49	24.53	19.51
DP (C2)	18.23	14.80	28.04	21.42
DP (C3)	17.13	17.14	24.93	21.03
Räsänen	19.40	17.18	28.19	22.68
ten Bosch	25.17	25.36	28.28	26.82

For illustrative purposes, the segmentations produced by the three algorithms for the same sentence, dr6-fbch0-sa1, are shown in Figures 9, 10, and 11, where configuration C3 was used for the DP algorithm. Each figure shows the first two seconds of the sentence as well as the TIMIT phone boundaries. The dashed vertical lines show the hypothesised boundaries, and the solid vertical lines show the reference boundaries.

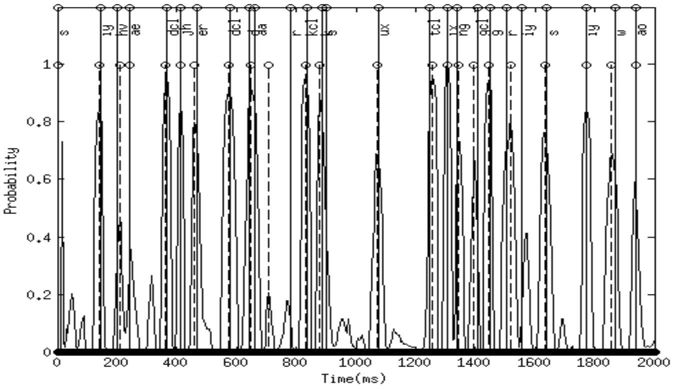

Fig. 9. Segmentation results for the DP algorithm on dr6-fbch0-sa1.

The vertical axis in Figure 9 for the DP algorithm shows the emission probabilities. Unlike the other two approaches, there is no threshold under which boundaries are ignored. Thus, even when the local score results in a low emission probability, a boundary can be hypothesised if the transition probability is high. This is clear, for example, at the boundary that is hypothesised at the 'aa' phoneme. The converse may also be true, i.e. even when the emission probability is high, a segment boundary may be suppressed by a low transition probability, as illustrated at the second 'iy'.

Figure 10 shows the output of the 'minmax' filter described in Section II-A of the Räsänen algorithm. Notice that all peaks falling within 32ms of each other have been combined by temporal peak masking, and that the threshold in this case is 0.07, below which all peaks are ignored. These parameter values were determined to be optimal for the development set.

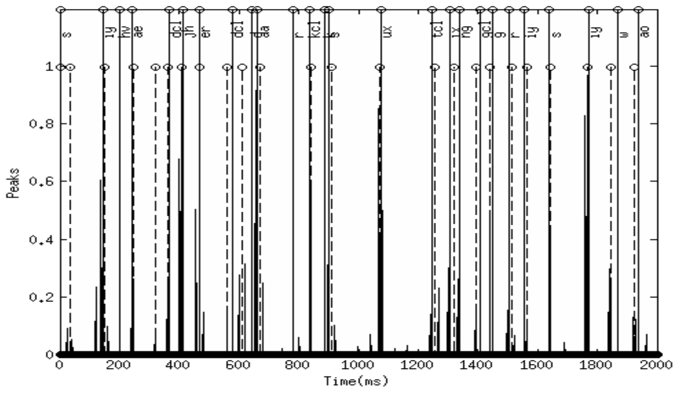

Fig. 10. Segmentation results for the Räsänen algorithm on dr6-fbch0-sa1.

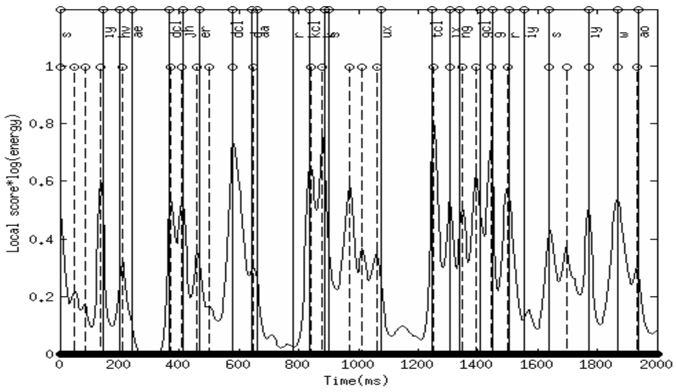

Fig. 11. Segmentation results for the ten Bosch algorithm on dr6-fbch0-sa1.

The local score of ten Bosch's algorithm has been multiplied by the log energy to reduce the insertion of boundaries in regions of silence, which are characterised by very low energy. Unfortunately this also results in the introduction of unwanted small peaks when the log energy increases while the local score decreases, or when the energy decreases while the local score increases. Because the segment boundaries usually coincide with the peaks of the local score, these newly added peaks lead to insertions as shown, for example, in Figure 11 at 'er' and at each 's'. This leads to over segmentation, which is clear when looking at the higher percentage insertions in Table III. From the development set it was found that the optimal threshold for the ten Bosch algorithm is 0.13.

E. Combined methods

By inspection of the segmentation results produced by the DP algorithm, it was found that there regularly are small emission probability peaks present between the boundaries of very long segments. When these peaks coincide with high probability segment lengths, as determined by the segment length distribution, boundaries are hypothesised at these locations, resulting in unwanted insertions. With some experimentation it was found that better results can be obtained by applying a threshold to the emission probability (Equation 8) before searching for the optimal path by DP. All probabilities above the threshold are unchanged, and the probabilities below the threshold are reduced to 0. A variety of threshold values were investigated on the development set for each of the chosen three DP configurations, with all other parameters fixed at their previously found optimal values. Figures 12 and 13 show the resulting effect on the DP cost and on the average error for configuration C1. By inspecting the average error graph, there is a point at which the reduction in insertions is greater than the rise in deletions. However, the average error can only be reduced to a certain point, after which the hypothesised and reference boundaries rapidly become misaligned. This is indicated at the point of DP cost increase. The DP cost is therefore the best way to determine the optimal threshold.

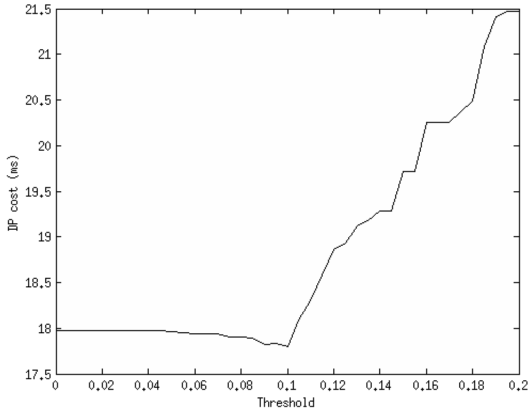

Fig. 12. DP cost (Section IV-A) against emission probability threshold on the development set for configuration C1.

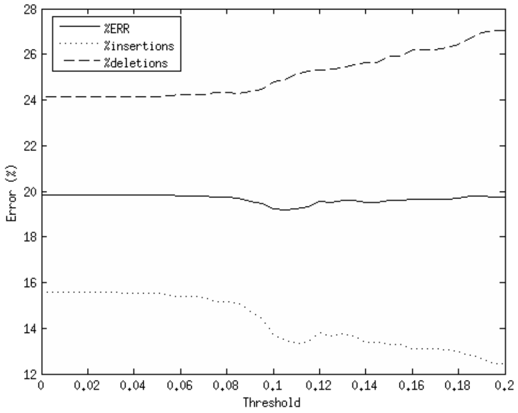

Fig. 13. Average error (Section IV-B) against emission probability threshold on the development set for configuration C1.

It was found that thresholds of 0.1, 0.5, and 0.1 lead to optimal performance on the development set for configurations C1, C2 and C3 respectively. When applied to the core test set, this leads to the results in Table IV.

TABLE IV
METHOD COMPARISONS, AFTER EMISSION PROBABILITY THRESHOLD WERE APPLIED, ON THE CORE TEST SET.

Method	DP Cost (ms)	% Ins	% Del	%ERR
DP (C1)	17.68	12.83	24.96	18.89
DP (C2)	18.08	13.91	28.44	21.17
DP (C3)	16.99	16.65	25.02	20.83
Räsänen	19.40	17.18	28.19	22.68
ten Bosch	25.17	25.36	28.28	26.82

By comparing the results in Tables III and IV, improvements in performance for all three configurations are seen. Two key values that stand out from Table IV are the small DP cost obtained by configuration C3, and the small average error obtained by configuration C1. The overall best, and most consistent configuration thus far, is configuration C1, which has the normalised city block distance and the FFT.

VII. Summary and conclusion

We have proposed an algorithm based on the principle of dynamic programming for the automatic segmentation of continuous speech into phoneme-like units. A measure of the local dissimilarity between feature vectors is combined with a statistical description of the expected segment lengths within the dynamic programming framework in order to determine the optimal locations of segment boundaries within the speech utterance. We find that this approach leads to performance improvements relative to two alternative methods drawn from the literature. Analysis of the strengths of the individual techniques revealed that further improvements can be obtained by a hybrid approach employing aspects of each. We conclude that the use of dynamic programming as a basis for speech segmentation is a successful approach. In future work we plan to analyse the occurrence of insertion and deletion errors more carefully with respect to the type of phoneme within which they occur, as well as the role of context in the placement of segment boundaries. The effectiveness of our DP-based segmentation will also be tested on other languages using the distributions created from TIMIT to see how universal the segment boundary behaviour is. Furthermore, we will investigate the sensitivity of the segmentation algorithms to parameter changes, and the effect of increased parameters.

VIII. Acknowledgements

We gratefully acknowledge the Department of Arts and Culture (DAC) for their financial support.

References

[1] Y. jun Kim and A. Conkie, "Automatic segmentation combining an hmm-based approach and spectral boundary correction," in *Proceedings of the International Conference on Spoken Language Processing, ICSLP*, 2002, pp. 145–148.

[2] T. Svendsen and F. Soong, "On the automatic segmentation of speech signals," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 12, apr 1987, pp. 77 – 80.

[3] M. Sharma and R. Mammone, "'blind' speech segmentation: automatic segmentation of speech without linguistic knowledge," in *Proceedings of the Fourth International Conference on Spoken Language Processing, ICSLP*, vol. 2, oct 1996, pp. 1237 –1240.

[4] Okko Räsänen, U. K. Laine, and T. Altosaar, "Blind segmentation of speech using non-linear filtering methods," in *Ipsic I. (Ed.): Speech Technologies*. InTech Publishing, 2011, pp. 105 –124.

[5] L. ten Bosch and B. Cranen, "A computational model for unsupervised word discovery," in *Order A Journal On The Theory Of Ordered Sets And Its Applications*, 2007, pp. 1 – 4.

[6] A. Sarkar and T. Sreenivas, "Automatic speech segmentation using average level crossing rate information," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 1, 2005, pp. 397 – 400.

[7] Y. P. Estevan, V. Wan, and O. Scharenborg, "Finding Maximum Margin Segments in Speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, 2007, pp. 937 –940.

[8] A. K. Halberstadt, "Heterogeneous Acoustic Measurements and Multiple Classifiers for Speech Recognition ," Ph.D. dissertation, Massachusetts Institute of Technology, MIT, 1998.