3

DYNAMIC TIME WARPING

Rubita Sudirman
Khairul Nadiah Khalid

INTRODUCTION

Template matching is an alternative to perform speech recognition. However, the template matching encountered problems due to speaking rate variability, in which there exist timing differences between the two utterances. Speech has a constantly changing signal, thus it is almost impossible to get the same signal for two same utterances. The problem of time differences can be solved through DTW algorithm: warping the template with the test utterance based on their similarities. So, DTW algorithm actually is a procedure, which combines both warping and distance measurement. DTW is considered as one effective method in speech pattern recognition, however the bad side of this method is that it requires a long processing time plus large storage capacity, especially for real time recognitions. Thus, it is only suitable for application with isolated words, small vocabularies, and speaker dependent with/without multi-speaker, which has yielded a good recognition under these circumstances (Liu, *et al.*, 1992).

Human speeches are never at the same uniform rate and there is a need to align the features of the test utterance before computing a match score. Dynamic Time Warping (DTW), which is a Dynamic Programming technique, is widely used for solving time-alignment problems.

DYNAMIC TIME WARPING

In order to understand Dynamic Time Warping, two procedures need to be dealt with. The first one is the information in each signal that has to be presented in some manner, called features. (Rabiner and Juang, 1993). One of the features is the LPC-based Cepstrum. The LPC-based Cepstrum procedure is the calculation of the distances because some form of metric has to be used in the DTW in order to obtain a match between the database and the test templates. There are two types of distances, which are local distances and global distances. Local distance is a computational different between a feature of one signal and another feature. Global distance is the overall computational difference between an entire signal and another different length signal.

The ideal speech feature extractor might be the one that produces the word that match the meaning of the speech. However, the method to extract optimal feature from the speech signal is not trivial. Thus separating the feature extraction process from the pattern recognition process is a sensible thing to do, since it enables the researchers to encapsulate the pattern recognition process according to (Rabiner and Juang, 1993).

Feature extraction process outputs a feature vector at every regular interval. For example, if an MFCC analysis is performed, then the feature vector consists of the Mel-Frequency Cepstral Coefficients over every fixed tempo. For a LPC analysis the feature vector consists of prediction coefficients while the LPC-based Cepstrum analysis outputs Cepstrum coefficients.

Because the feature vectors could have multiple elements, a method of calculating local distances is needed. The distance measure between two feature vectors can be calculated using the Euclidean distance metric. (Rabiner and Juang, 1993) Therefore, the local distance between two feature vectors x and y is given by,

$$d(x,y)=\sqrt{\sum_{j=1}^{P}\left(x_j-y_j\right)^2} \tag{3.9}$$

Although the Euclidean metric is computationally more expensive than some other metrics, it gives more weight to large differences in a single feature.

For example, let consider two feature vectors $A = a_1, a_2, a_3, ..., a_i, ..., a_I$ and $B = b_1, b_2, b_3, ..., b_j, ..., b_J$, let *A* be the template/reference speech pattern while *B* be the unknown/test speech pattern. Translating sequences *A* and *B* into Fig. 3.1, the warping function at each point is calculated. Calculation is done based on Euclidean distance measure as a mean of recognition mechanism. It takes the smallest distance between the test utterance and the templates as the best match. For each point, the distance called local distance, *d* is calculated by taking the difference between two feature-vectors a_i and b_j:

$$d(i, j) = \left\| b_j - a_i \right\| \tag{3.2}$$

Every frame in a template and test speech pattern must be used in the matching path. If a point (*i,j*) is taken, in which *i* refers to the template pattern axis (x-axis), while *j* refers to the test pattern axis (y-axis), a new path must continue from previous point with a lowest distance path, which is from point (*i-1, j-1*), (*i-1, j*), or (*i, j-1*) of warping path shown in Fig. 3.2.

If *D(i,j)* is the global distance up to (*i,j*) with a local distance at (*i,j*) given as *d(i,j)*, then

$$D(i,j) = min[D(i-1,j-1), D(i-1,j), D(i,j-1)] + d(i,j) \tag{3.3}$$

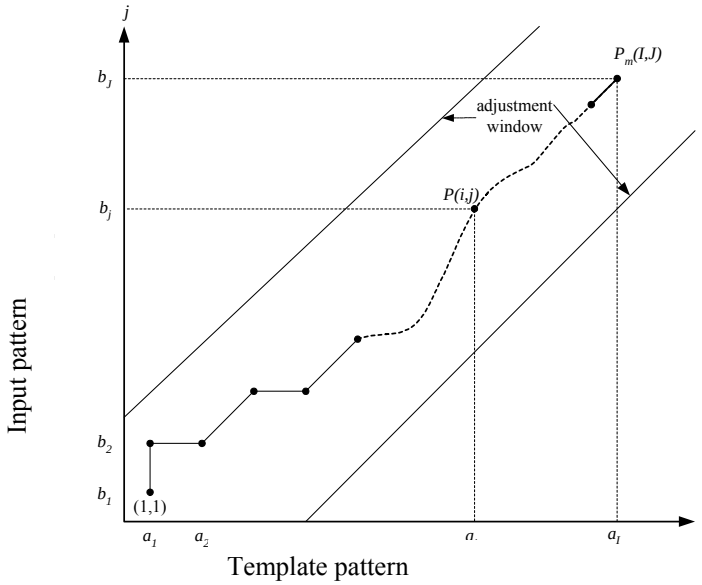

Fig. 3.1 Fundamental of warping function

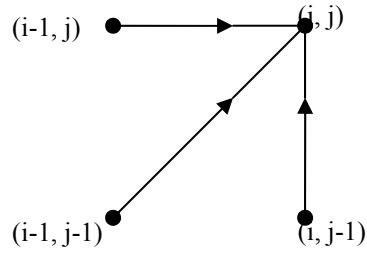

Fig. 3.2 DTW heuristic path type 1

Back to reference pattern *A* and *B*, if their feature vector *B* and an input pattern with feature vector *A*, which each has N_A and N_B frames, the DTW is able to find a function *j=w(i)*, which maps the

time axis *i* of *A* with the time axis *j* of *B*. The search is done frame by frame through *A* to find the best frame in *B*, by making comparison of their distances. After the warping function is applied to *A*, distance *d(i,j)* becomes

$$d(i, j(i)) = \left\| b_j' - a_i \right\| \tag{3.4}$$

Then, distances of the vectors are summed on the warping function. The weighted summation, *E* is:

$$E(F) = \sum_{i=1}^{I} d(i, j(i)) * w(i) \tag{3.5}$$

where *w(i)* is a nonnegative weighting coefficient. The minimum value of *E* will be reached when the warping function optimally aligned the two pattern vectors.

A few restrictions have to be applied to the warping function to ensure close approximation of properties of actual time axis variations. This is to preserve essential features of the speech pattern. Rabiner and Juang (1993) outlined the warping properties as follows for DTW path Type I:

1. Monotonic conditions imposed: $j(i-1) \le j(i)$
2. Continuity conditions imposed: $j(i) - j(i-1) \le 1$
3. Boundary conditions imposed: $j(i) = 1$ and $j(J) = I$
4. Adjustment window implementation: $|i - j(i)| \le r$, r is a positive integer
5. Slope condition: to hold this condition, say if $b'_{j(i)}$ moves forward in one direction *m* times consecutively, then it must also step *n* times diagonally in that direction. This is to make sure a realistic relation between *A* and *B*, in which short

segments will not be mapped to longer segments of the other. The slope is measured as: $M = \frac{n}{m}$.

The warping function slope is more rigidly restricted by increasing M, but if slope is too severe then time normalization is not effective, so a denominator to time normalized distance, N is introduced, however it is independent of the warping function.

$$N = \sum_{i=1}^{I} w(i) \tag{3.6}$$

So, the time normalized distant becomes

$$D(A,B) = \frac{1}{N} \underset{F}{Min} \left[\frac{\sum_{i=1}^{I} d(i, j(i)) * w(i)}{\sum_{i=1}^{I} w(i)} \right] \tag{3.7}$$

Having this time normalized distant, minimization can be achieved by dynamic programming principles.

There are two typical weighting coefficients that permit the minimization (Rabiner and Juang, 1993):

1. ***Symmetric time warping***
   The summation of distances is carried out along a temporary defined time axis $l=i+j$.
2. ***Asymmetric time warping***
   Previous discussion has described the asymmetric type, in which the summation is carried out along i axis warping B to be of the same size as A. The weighting coefficient for asymmetric time warping is defined as:

$$w(i) = j(i) - j(i-1) \tag{3.8}$$

When the warping function attempts to step in the direction of the j axis, the weighting coefficient is reduce to 0 because $j(i) = j(i-1)$, thus $w(i) = 0$. Meanwhile, when the warping function steps in the direction of i axis or diagonal, then $w(i) = 1$, so $N = I$.

The asymmetric time warping algorithm only provides compression of speech patterns. Therefore, in order to perform speech pattern expansion, a linear algorithm has to be employed.

SYMMETRICAL DTW ALGORITHM

In speech signal, different speeches have different durations. Ideally, when comparing different length of utterances of the same word, the speaking rate and the utterance duration should not contribute to the dissimilarity measurement. Several utterances of the same word are possibly to have different durations while utterances with the same duration differ in the middle because different parts of the words have been spoken in different rates. Thus a time alignment must be done in order to get the global distance between two speech patterns.

This problem is illustrated in Fig. 3.3, in which a "time to time" matrix is used to visualize the alignment. The reference pattern goes up the side and the input pattern goes along the bottom. As shown in Fig. 3.3, "KOSsONGg" is the noisy version of the template "KOSONG". The idea is 's' is closer match to "S" compared with other alphabets in the template. The noisy input is matched against all the templates. The best matching template is the one that has the lowest distance path aligning the input pattern to template. A simple global distance score for a path is simply the sum of local distances that make up the path.

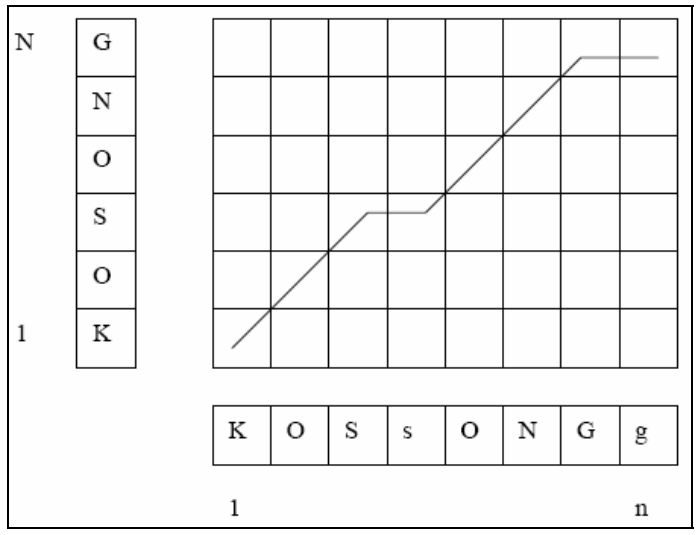

Fig. 3.3 Illustration of time alignment between pattern "KOSONG" and a noisy input "KOSsONGg"

Now the lowest global distance path (or the best matching) between an input and a template can be evaluated by all possible paths. However, this is very inefficient as the possible number of path increases exponentially as the input length increases. So some constraints have to be considered on the matching process and using these constraints as efficient algorithm.

There are many types of local constraints imposed, but they are very straightforward and not restrictive. The constraints are:

1) Matching path cannot go backwards in time.
2) Every frame in the input must be used in a matching path.
3) Local distance scores are combined and added to give a global distance.

For now every frame in the template and input must be used in a matching path. If a point (i,j) is taken in the time-time

matrix(where i indexes the input pattern frame, j indexes the template frame), then previous point must be (i-1,j-1), (i-1,j) or (i,j-1). The key idea in this dynamic programming is that at point (i,j) we can only continue from the lowest distance path that is from (i-1,j-1),(i-1,j) or (i,j-1).

If D(i,j) is the global distance up to (i,j) and the local distance at (i,j) is given by d(i,j), thus,

$$D(i,j) = \min\left[D(i-1,j-1), D(i-1,j), D(i,j-1)\right] + d(i,j) \tag{3.10}$$

Given that D(1,1)=d(1,1), the efficient recursive formula for computing D(i,j) can be found (Rabiner and Juang, 1993). The final global distance D(n, N) is the overall score of the template and the input. Thus, the input word can be recognized as the word corresponding to the template with the lowest matching score. The N value is normally different for every template.

The symmetrical DTW requires very small memory because the only storage required is an array that holds every column of the time-time matrix. The only direction that the match path can move when at (i,j) in the time-time matrix are as shown in Fig. 3.4.

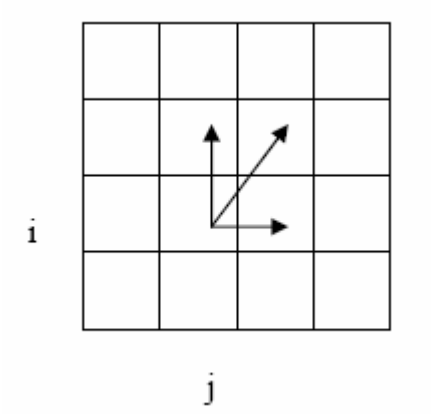

Fig. 3.4 The three possible directions the best matched may move

IMPLEMENTATION DETAILS

The pseudo code for calculating the least global cost (Rabiner and Juang, 1993) is:

```
calculate first column (predCol)
for i=1 to number of input feature vector
        curCol[0]=local cost at (i,0) + global cost at (i-1,0)
        for j=1 to number of template feature vectors
                curCol[j]=local cost at (i,j)+minimum of global
                        costs at (i-1,j),(i-1,j-1) or (i,j-1)
        end for j
        predCol=curCol
end for i
minimum global cost is value in curCol[number of templater
feature vectors]
```

VARIOUS LOCAL CONSTRAINTS

Although the Symmetrical DTW algorithm has benefit of symmetry, this has the side effect of penalizing horizontal and vertical transitions compared to the diagonal ones (Rabiner and Juang, 1993). To ensure proper time alignment while keeping any potential loss of information to a minimum, the local continuity constraints need to be added to the warping function. The local constraints can have many forms. According to Rabiner and Juang (1993), the local constraints are based on heuristics. The speaking rate and the temporal variation in speech utterances are difficult to model. Therefore the significance of these local constraints in speech pattern comparison cannot be assessed analytically. Only the experimental results can be used to determine their utility in various applications.

BIBLIOGRAPHIES

Rabiner, L. and Juang, B. H. (1993). *Fundamentals of Speech Recognition.* Englewood Cliffs, N.J.: Prentice Hall.

Liu, Y., Lee, Y. C., Chen, H. H., and Sun, G. Z. (1992). Speech Recognition using Dynamic Time Warping with Neural Network Trained Templates. *International Joint Conference in Neural Network*. 2: 7-11.