DSP Lab Report

Introduction

Sound is very common in the nature. Animals use sound to communicate with each other, such as birds, elephants and monkeys. Also, humans use sound to communicate with each other. The sound human use follows particular rules, which is called language. It is nature for humans to communicate with each other using sound. People can understand with each other quickly using sound, or in other words, speech.

For a long time, the way people interact with computers is through input devices such as keyboard and mouse. People need to type the word they want to express into the input text field. As the technology develops, computer is becoming more powerful. The power of modern computer enables us to realize a system to recognize the word human speaks. This system is called Speech Recognition System. Nowadays there have occurred a lot of speech recognition systems, such as Siri, Cortana, Google Assistant etc. You can awake Siri by saying “Hey Siri!” by default. The sound wave you make will first be transformed into digital signal by your phone. Then, by applying algorithms, the digital signal will be transformed into words or commands.

There are many algorithms to implement the speech recognition system, and my paper mainly focuses on a basic but important method, Dynamic Time Warping (DTW). It is the fundamental of many modern speech recognition systems. It compares the input signal with a standard signal pattern to get the most similar word.

Dynamic Time Warping (DTW)

Dynamic Time Warping algorithm calculates the similarity between two digital signals in time domain. The advantage of Dynamic Time Warping algorithm is that it can calculate two signals in different time domain. For example, two people saying the same sentence may have different speaking speed, but they can still have very high-level similarity in using Dynamic Time Warping. How to handle different time domain is the major difficulty in Speech Recognition System.

DTW algorithm approaches this difficulty by calculating an "optimal warping path" between the two digital signals. Rather than directly compare two signals, DTW will find a best match for the input signal. The figure below shows how "optimal warping path" works. For figure (a), it compares all the corresponding points of the two signals. For figure (b), the input signal will find a best match from the signal pattern to compare. You can see that although two signals vary in time, they have similar pattern. By applying this method, DTW can calculate the similarity between two different signals in time domain.

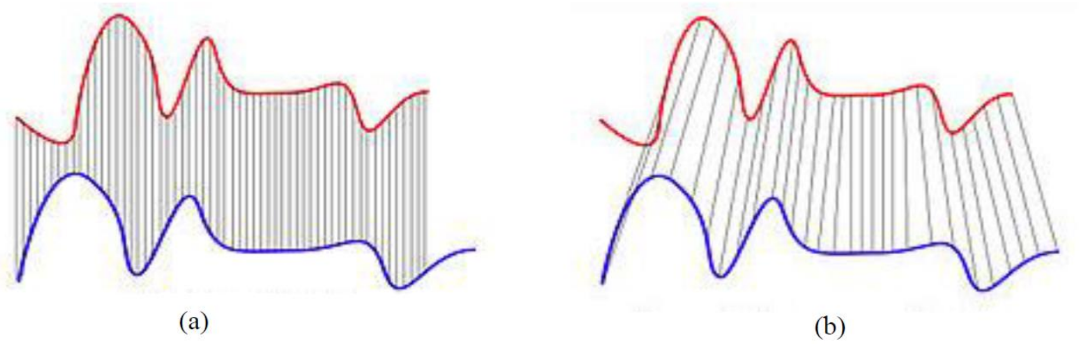


Figure 1. (a) The original alignment of two sequences (b) alligments with DTW

Euclidean Distance

$$Dist(x,y) = |x-y| = [(x_1-y_1)^2 + (x_2-y_2)^2 + \cdots + (x_n-y_n)^2]^{\frac{1}{2}}$$

Dynamic Programming

Dynamic programming is a programming method to solve an optimized problem. It decomposes the problem into subproblems in steps. The final solution depends on the previous results. With a solution to the initial state of the problem, we get a final solution step by step. Usually we store all the previous solutions inside an array.

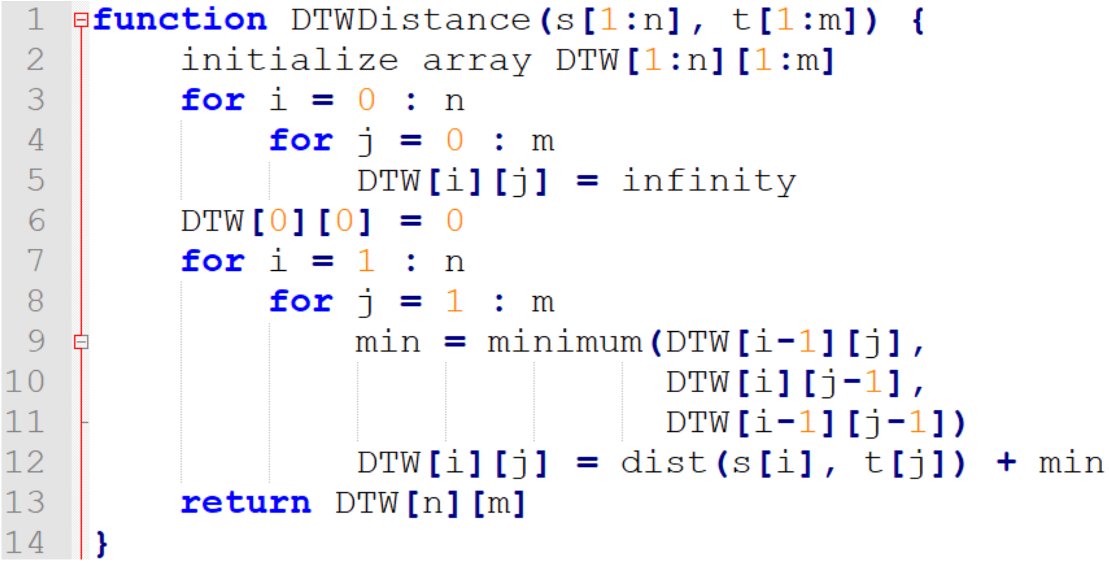
Implementation

DTW's implementation depends on Dynamic Programing. We will use a two-dimensional array DTW[m][n]. DTW[m][n] represents the optimal distance between two input signals, lets say audioIn[m] and sampleAudio[n]. DTW[m][n] can be calculated by the following recursive formula:

$$\mathrm{DTW[m][n]} = Dist(m,n) + min\begin{pmatrix} \mathrm{DTW[m-1][n]} \\ \mathrm{DTW[m][n-1]} \\ \mathrm{DTW[m-1][n-1]} \end{pmatrix}$$

It means that current optimized distance depends on the minimal distance of previous sub-sequence of signals plus the current distance of input value. This shows how the "optimal warping path" is calculated.

Pseudocode



```
function DTWDistance(s[1:n], t[1:m]) {
    initialize array DTW[1:n][1:m]
    for i = 0 : n
        for j = 0 : m
            DTW[i][j] = infinity
    DTW[0][0] = 0
    for i = 1 : n
        for j = 1 : m
            min = minimum(DTW[i-1][j],
                          DTW[i][j-1],
                          DTW[i-1][j-1])
            DTW[i][j] = dist(s[i], t[j]) + min
    return DTW[n][m]
}
```

By applying DTW, we can calculate the similarity and output an optimal word.

Future Application

DTW algorithm for now is widely used in speech recognition. Many projects are built on it, for example the ESP system (https://github.com/damellis/ESP).

In my point of view, I think it can also be used in safety area. It can be used to unlock some system. It can be used to compare the similarity between two sound pattern and return lock or unlock.

References

[1] Yurika Permanasari et al 2019 J. Phys.: Conf. Ser. 1366 012091