Speech Recognition dengan Pendekatan Algoritma Dynamic Programming

Thomas Ferdinand Martin - 13519099
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
thomasferdinandm00@gmail.com

***Abstract*—*Speech recognition* atau metode pengenalan ucapan merupakan salah satu fitur pemrosesan audio yang sering dijumpai pada alat-alat elektronik saat ini, teknologi terkenal yang menggunakannya adalah Google dan Alexa. Dalam menerima masukkan perintah dari suara, karakteristik suara pengguna setiap saat tidak selalu sama, terkadang pengucapan cepat, lambat, keras, lembut dan sebagainya tergantung dengan kondisi pengguna saat itu. Oleh karena itu, teknologi pengenalan ucapan memerlukan suatu metode untuk mengenali suatu perintah dari berbagai karakteristik audio yang berbeda-beda tiap saatnya. Dengan menggunakan pendekatan algoritma Dynamic Time Warping, masalah proses pengenalan ucapan pada tempo audio yang berbeda-beda dapat terpecahkan. Algoritma Dynamic Time Warping ini menggunakan dasar konsep dynamic programming dalam implementasinya.**

Keywords—Dynamic Programming; Dynamic Time Warping; Speech Recognition; Word Identifier

I. PENDAHULUAN

Gambar 1.1 Pemrosesan Ucapan Manusia oleh Google API
sumber: https://glaforge.appspot.com/article/a-poor-man-assistant-with-speech-recognition-and-natural-language-processing

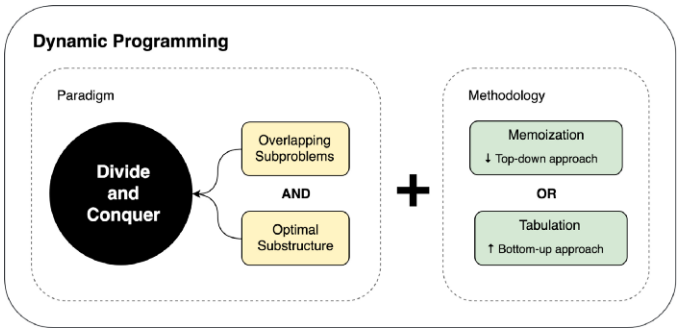
Saat ini dunia sedang terpengaruh oleh keberadaan IoT yang mempermudah kehidupan sehari-hari. Salah satu teknologi IoT yaitu pada teknologi pemrosesan sinyal yang salah satunya adalah *speech recognition*. *Speech recognition* merupakan suatu kemampuan yang memungkinkan sebuah perangkat untuk menerima dan memproses masukan berupa sebuah audio atau sebuah ucapan kata dari manusia. Teknologi ini mengubah audio yang masuk menjadi sekumpulan sinyal digital dan melakukan pencocokan dengan pola tertentu untuk memproses perintah masukan. Melalui teknologi ini, segala bentuk pekerjaan dapat terselesaikan dengan mudah. Misalnya saja jika seseorang memiliki *smartphone*, android terkhususnya, manusia tidak lagi harus repot-repot utnuk mengambil smartphonenya untuk menyalakan sebuah lagu, hanya dengan mengatakan "Oke Google, putar playlist favoritku" maka *playlist* musik favorit langsung diputarkan melalui smartphone itu. Salah satu perusahaan yang memiliki produk yang menerapkan teknologi ini adalah Google.

Pada teknologi *speech recognition*, dikenal beberapa algoritma yang membangun fitur tersebut. Salah satu masalah dari pengolahan *speech recognition* adalah ketika membandingkan dua audio dengan kecepatan berbeda. Misal saja seseorang saat terburu mengucapkan "Ok Google" dalam waktu 2 detik dan ketika baru bangun tidur akan mengucapkan "Ok Google" dalam waktu 5 detik. Perbandingan jarak euclidian biasa tidak dapat menangani hal ini karena kedua ukuran waktu audio yang berbeda. Salah satu algoritma klasik yang dapat memecahkan permasalahan perhitungan jarak antara dua data dengan ukuran waktu yang berbeda adalah algoritma *Dynamic Time Warping*.

II. LANDASAN TEORI

A. *Dynamic Programming*

Dynamic programming atau program dinamis merupakan algoritma yang menguraikan solusi masalah menjadi sekumpulan atau beberapa tahapan sedemikian sehingga solusi akhir dipandang sebagai seruntutan keputusan yang saling berkaitan dan berurutan. Istilah dinamis muncul karena pencarian solusinya melakukan perhitungan denagn menggunakan tabel yang dapat berkembang. *Dynamic programming* digunakan untuk memecahkan persoalan-persoalan optimasi. Ciri lain dari pemrograman dinamis adalah dapat memetakan persoalan rekursif menjadi permasalahan linear. Perbedaan mendasar dari *Greedy* dan Program Dinamis adalah pada greedy hanya digunakan satu rangkaian keputusan yang dihasilkan sedangkan pada program dinamis digunakan lebih dari satu rangkaian keputusan yang dipertimbangkan.



Gambar 2.1 Konsep Pemrograman Dinamis

(Sumber: https://itnext.io/dynamic-programming-vs-divide-and-conquer-2fea680becbe)

Pada dynamic programming dikenal prinsip optimalitas yaitu jika solusi total optimal, maka bagian solusi sampai tahap ke-n juga optimal. Hal ini berimplikasi kepada aturan jika bekerja pada tahap ke-n maka digunakan hasil optimal pada tahap n-1 tanpa harus kembali melakukan tahap sebelumnya.

Karakterisitik pada persoalan dinamis:

a. Persoalan dapat dibagi menjadi beberapa tahap, yang pada setiap tahapnya diambil satu keputusan

b. Masing-masing tahap terdiri dari sejumlah status (state) yang berhubungan dengan tahap tersebut. Status merupakan bermacam kemungkin masukan yang ada pada suatu tahap.

Pada program dinamis juga dikenal dua jenis pendekatan yaitu program dinamis maju dan program dinamis mundur. Pada program dinamis maju, perhitungan dilakukan dari posisi 1, 2, 3, hingga ke n. Sedangkan pada program dinamis mundur berlaku sebaliknya, yaitu evaluasi dilakukan dari tahap ke n, n-1 hingga tahap 1.

Dalam aplikasinya dalam memecahkan permasalahan, beberapa langkah pengembangan program dinamis yang dapat dilakukan yaitu:

a. Karakteristikan struktur solusi optimla seperti pada pemetaan tahap, variabel keputusan, dan status

b. Definisikan secara rekursif nilai optimal dengan mencari hubungan dengan tahapan sebelumnya

c. Hitung nilai solusi optimal secara maju atau mundur menggunakan tabel yang dapat terekspan

d. Rekonstruksi solusi optimal untuk mendapatkan minimum atau maksimum path.

B. Speech Recognition

Speech recognition atau dikenal sebagai automatic speech recognition (ASR) merupakan sebuah teknologi yang memiliki kemampuan untuk memproses ucapan manusia untuk didefinisikan sebagai sebuah masukan perintah. Berbagai macam jenis algoritma digunakan untuk menyelesaikan permasalahan ini. Beberapa jenis algoritma yang umum digunakan antara lain:

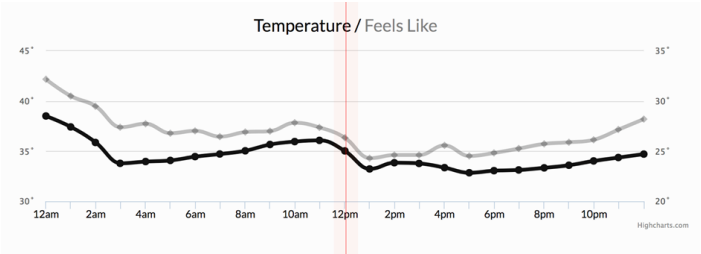
a. *Natural Language Processing* : Merupakan area dalam artificial intelligence yang memfokuskan pada interaksi antar manusia dan mesin melalui ucapan dan tulisan

b. *Hidden Markov Models*: Model ini dibangun atas dasar Markov chain model, yang memiliki pendekatan dynamic programming

c. *Dynamic Time Warping*: Dynamic Time Warping mengukur kemiripan antara dua buah data sequences yang memiliki ukuran waktu dan kecepatan yang berbeda. Algoritma ini menggunakan pendekatan dynamic programming untuk memperoleh solusinya yang akan dibahas pada makalah ini.

C. Time Series Data

Time Series Data adalah kumpulan pengamatan yang diperoleh melalui pengukuran berulang sepanjang waktu. Data yang dikumpulkan dapat diubah menjadi sebuah plot grafik dan nantinya salah satu sumbu grafik itu berlaku sebagai waktu.

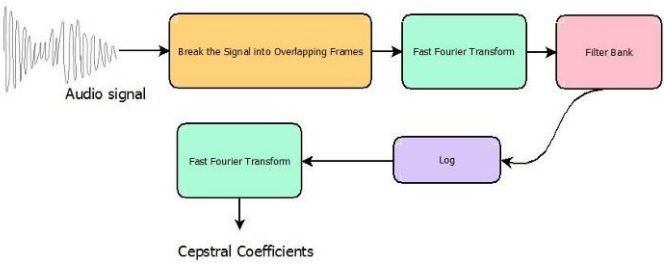


Gambar 2.2 Contoh *Time Series Data* dalam Perubahan Suhu

(Sumber: https://www.influxdata.com/what-is-time-series-data/)

D. Mel Frequency Cepstral Co-efficients

*Mel Frequency Cepstral Co-efficient*s atau biasa disingkat MCCF merupakan salah satu fitur dari ekstraksi sinyal audio. MCCF adalah karakteristik suatu sinyal audio yang diukur sebagai frekuensi suatu sinyal. Skala ini menghubungkan frekuensi yang dirasakan dari suatu nada dengan frekuensi terukur yang sebenarnya. Skala ini diturunkan dari serangkaian eksperimen pada subjek manusia. Pada makalah ini, sinyal audio akan diproses kedalam bentuk MCCF sebagai bentuk representasi dari *time series data* sebuah audio.



Gambar 2.3 Diargam Alir Perhitungan MCCF

(Sumber: https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd)

III. DYNAMIC TIME WARPING

Dynamic Time Warping atau biasa disingkat DTW, merupakan algoritma yang mengukur kemiripan antara dua data *time series* khususnya saat kedua data tersebut memiliki rentang waktu atau kecepatan perubahan yang berbeda. Misalnya ketika antara dua orang yang berlari dan berjalan dapat saja memiliki kemiripan jumlah langkah yang sama meskipun bergerak dala interval dan kecepatan yang berbeda. Hal ini sama seperti ketika orang berbicara, terkadang dalam pelafalan sebuah kata, tempo berbicara seseorang tidak menentu. Oleh karena itu dalam teknologi *speech recognition*, algoritma DTW dapat menjadi salah satu solusi untuk mengidentifikasikan jenis ucapan yang ukuran waktunya berbeda-beda.

Terdapat beberapa pendekatan pada penyelesaian menggunakan DTW, salah satunya program dinamis. Bagian ini akan menjelaskan bagaimana ilustrasi pengerjaan algoritma DTW secara garis besar dengan pendekatan program dinamis. Misalkan terdapat dua buah data *time series* yaitu S dan Q serta matriks M.

$$S = s_1, s_2, s_3, ..., s_i, ..., s_n$$
$$Q = q_1, q_2, q_3, ..., q_j, ... ,q_m$$

M = berupa tabel dari kumpulan perhitungan *cost* (*cost matrix)*

Variabel n dan m menyatakan panjang *time series* dalam satuan waktu sama dan panjang yang berbeda. Inti dari DTW adalah *dynamic programming*, DTW membagi beberapa persoalan menjadi beberapa sub persolaan dan kemudian setiap sub persoalan saling berkaitan menyusun solusi optimal. Persamaan (1) menunjukkan bagaimana sub persoalan dibentuk untuk mencari solusi optimal di tahap berikutnya.

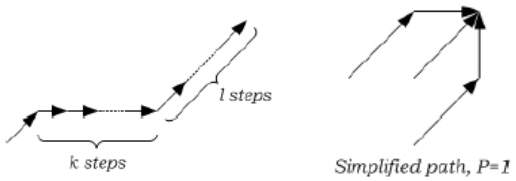
$$M(i,j) = distance(i,j) + min\begin{cases} M(i-1,j)\,; i-1 \geq 0 \\ M(i,j-1)\,; j-1 \geq 0 \\ M(i-1,j-1); i-1 \geq 0 \; dan \; j-1 \geq 0 \end{cases} \quad (1)$$

Berdasarkan persamaan tesebut, maka DTW dikategorikan sebagai persoalan dengan pendekatan maju atau *forward dynamic programming* karena mengevaluasi dari tahap 0 hingga tahap n.

Fase pertama algoritma ini adalah mengisi sebuah larik *distance* yang menyatakan jarak antara dua titik pada *time series*, variabel i menyatakan indeks pada *time series* S dan variabel j menyatakan indeks pada time series Q. Pada contoh ini, dimisalkan menggunakan *Euclidian Distance* untuk melakukan pengukuran jarak. Fase kedua melakukan pengisian *cost matrix* atau *warping matrix* menggunakan prinsip program dinamis sesuai dengan persamaan (1). Pengisian dilakukan hingga iterasi ke m,n atau sampai pasangan terakhir dari kedua data.

Ketika *warping matrix* telah seluruhnya terisi, dilakukan konstruksi jalur optimal dan *cost* optimal dari hasil perhitungan. Jalur solusi dinyatakan sebagai $p = p_1, p_2, p_3, ..., p_h$ dengan batasan yaitu:

a. Kondisi batas: $p_1 = (i, j)$ dengan $i = 1$ dan $j = 1$ dan $p_h = (i, j)$ dengan $i = m$ dan $j = n$. Hal ini menyatakan bahwa bagian awal jalur harus merupakan pasangan dari awal *time series* dan bagian akhir jalur merupakan pasangan dari akhir *time series*.
b. *Monotonicity condition*: Pada pasangan jalur solusi $p(m, n)$ harus berlaku $m_1 \leq m_2 \leq ... \leq m_h$ dan $n_1 \leq n_2 \leq ... \leq n_h$ untuk menjaga keterurutan waktu.
c. *Step size condition*: *Constraint* ini diimplementasikan sebagai hubungan antara beberapa titik berurutan di jalur warping. Sebagai contoh ketika jalur bergerak ke arah yang sama dalam k titik berturut-turut secara horizontal, titik *warping path* tidak diperbolehkan untuk melanjutkan ke arah yang sama sebelum langkah l menuju ke arah diagonal. Ilustrasi dapat dilihat pada gambar di bawah. [2]



Gambar 3.1 Step Function

Secara lengkap, pseudocode DTW dapat dinyatakan sebagai berikut:

Pseudocode untuk mengisi *warping matrix*

```
Algorithm 1 : FillCostMatrix
Input: S: Sequence of length n, Q: Sequence of length m.
Output: DTW distance.
1: Initialize D(i, 0) ⇐ infinite for each i
2: Initialize D(0, j) ⇐ infinite for each j
3: for all i such that 1 ≤ i < n do
4:   for all j such that 1 ≤ j < m do
5:       Use Equation 1 to compute D(i, j)
6:   end for
7: end for
8: return D(n, m)
```

Pseudocode untuk menyusun jalur optimal

```
Algorithm 2 OptimalWarpingPath
Input: dtw: Cost matrix of DTW
1: path[] ⇐ new array
2: i = rows(dtw)
3: j = columns(dtw)
4: while (i > 1) & (j > 1) do
5:  if i == 1 then
6:    j = j - 1
7:  else if j == 1 then
8:    i = i - 1
9:  else
10:   if dtw(i-1, j) == min {dtw(i-1, j); dtw(i, j-1); dtw(i-1; j-1)} then
```

```
11:      i = i - 1
12:    else if dtw(i, j-1) == min {dtw(i-1, j); dtw(i, j-1);
dtw(i-1; j-1)} then
13:      j = j - 1
14:    else
15:      i = i - 1; j = j - 1
16:    end if
17:    path:add((i, j))
18:  end if
19: end while
20: return path
```

Perhitungan *cost* minimum dilakukan dengan persamaan berikut, dengan d adalah jalur yang dihasilkan dan k adalah total titik yang harus ditempuh:

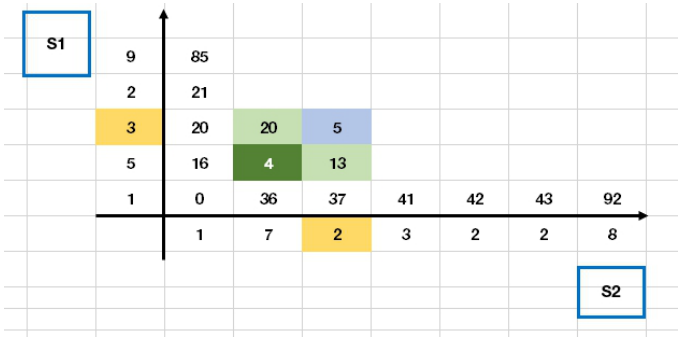
$$D = \frac{\sum_{i=1}^{k} d(i)}{\sum_{i=1}^{k} k} \quad (2)$$

Pada algoritma DTW, terdapat kemungkinan pengecekan setiap *cell* yang mungkin diekspan sehingga untuk *time-complexity* dan *space-complexity* sama-sama bernilai O(MN)

Untuk memperjelas proses pengisian matriks, ilustrasi pengisian matrix dan penyusunan jalur dapat dilihat pada gambar di bawah ini. Misalkan terdapat dua buah *time series data* S1 dan S2.

S1 = 1,5,3,2,9

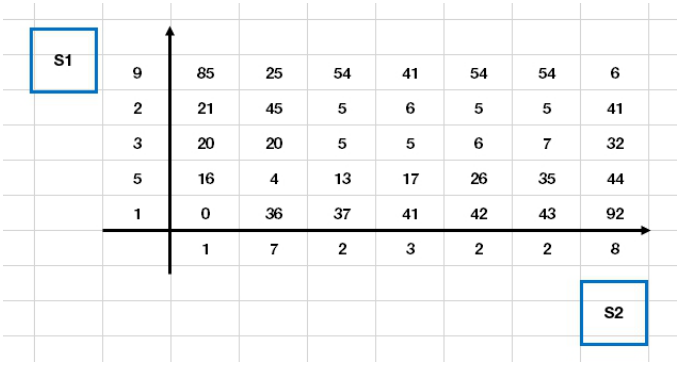
S2 = 1,7,2,3,2,2,8



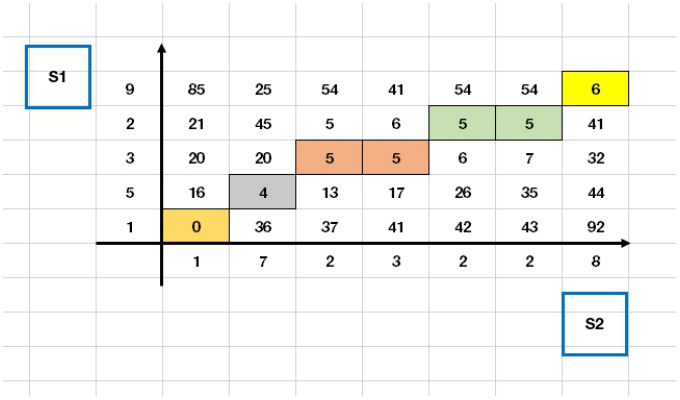
Gambar 3.2 Ilustrasi Perhitungan Sub Persoalan DTW

Untuk memperoleh nilai 5 pada sub-persoalan di atas, berdasarkan persamaan (1).

5 = euclidian distance(3, 2) + min (20,4,13) (3)



Gambar 3.3 Ilustrasi *Warping Matrix* Penuh Terisi



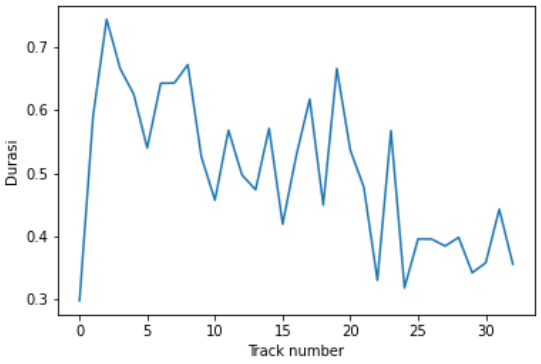
Gambar 3.4 Hasil Konstruksi Jalur Optimal

IV. Implementasi DTW pada Word Identifier

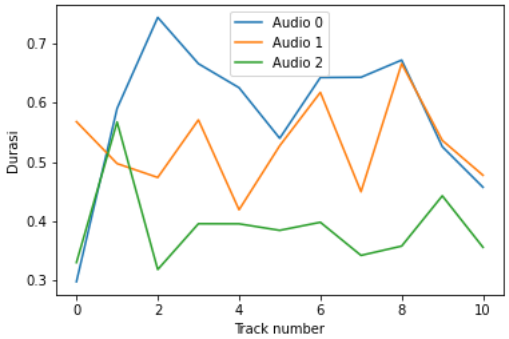
Saat ini terdapat beberapa algoritma yang dapat digunakan untuk pemrosesan *speech* recognition, salah satu algoritma yang cukup tua sudah dikenal adalah *Dynamic Time Warping*. Salah satu fitur dari *speech recognition* adalah *word identifier* yang digunakan dalam proses *speech-to-text*. Sama seperti namanya, *word identifier* mengidentifikasi kata yang diterima dari sinyal audio untuk kemudian dilanjutkan prosesnya sesuai perintah yang diterima. Pada makalah ini, penulis mencoba melakukan uji coba identifikasi kata menggunakan algoritma DTW, untuk *source code* akan dilampirkan di bagian bawah makalah. Uji coba dilakukan menggunakan jupyter notebook dan beberapa library pendukung.

Pada pengujian ini penulis mencoba beberapa audio untuk dilakukan testing identifikasi kata yang diucapkan. Untuk langkah dalam pengujian ini, pertama penulis menyiapkan 3 buah tipe audio. Setiap tipe audio mewakili sebuah kata yang diucapkan pada audio tersebut sehingga ada 3 klasifikasi kata. Audio tipe pertama memiliki kata “zero”, tipe kedua memiliki kata “one” dan tipe ketiga memiliki kata “two”. Setiap tipe audio memiliki 11 buah rekaman dengan tipe sama (11 buah rekaman yang mengucapkan kata “zero” dan sebagainya) sehingga total ada 33 buah track audio untuk dibandingkan dengan audio uji coba. Setiap audio merupakan rekaman dari orang yang sama hanya saja terdapat durasi pengucapan yang berbeda. Perbedaan durasi bukan dari hasil *editing* melainkan dari sumber suara yang melakukan *re-take* audio berkali-kali.

Perbedaan durasi audio sekaligus untuk membuktikan bahwa DTW digunakan untuk melihat kemiripan antara dua buah data dengan panjang waktu berbeda.



Gambar 4.1 Grafik Perbedaan Durasi Tiap Audio



Gambar 4.2 Grafik Perbedaan Durasi Berdasarkan Tipe Audio

Tipe audio :

Audio 0 = "**z e r o**"

Audio 1 = "**o n e**"

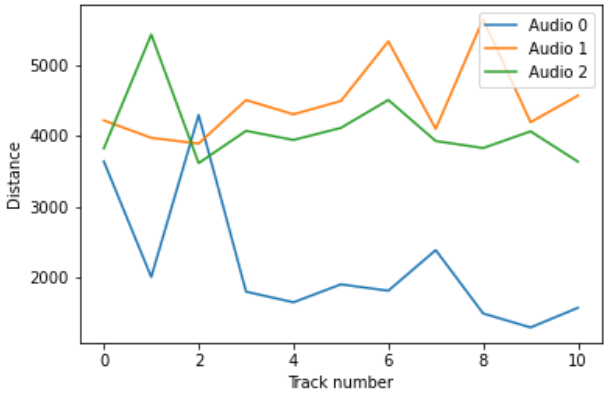
Audio 2 = "**t w o**"

Berdasarkan grafik, secara intuitif dapat terlihat perbedaan yang cukup signifikan pada durasi pengucapan tiap tipe kata.

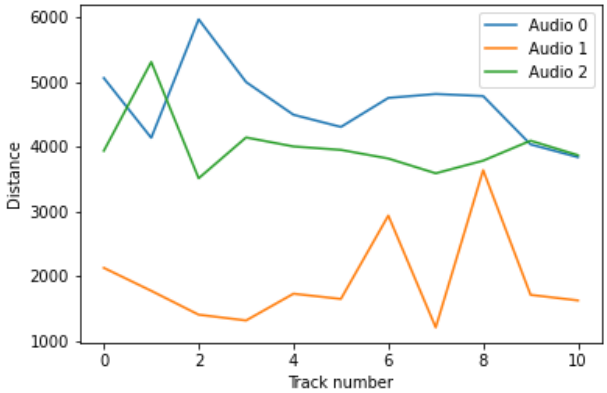
Langkah selanjutnya adalah mentransformasikan sinyal audio ke dalam bentuk matriks koefisien MCCF dengan bantuan library librosa pada python. Matriks MCCF tiap audio ini nantinya akan dihitung kemiripannya menggunakan DTW dengan audio uji coba.

Setiap audio masing-masing dilakukan perhitungan terhadap *minimum cost* terhadap audio lain dan disimpan ke dalam sebuah array untuk membuat unit uji sehingga terbentuk matriks berdimensi 33x33. Uji coba ini juga menggunakan library sklearn untuk memproses model yang akan membantu proses identifikasi dari 33 data training dengan perhitungan *classifier* menggunakan algoritma Gaussian Naive Bayes.

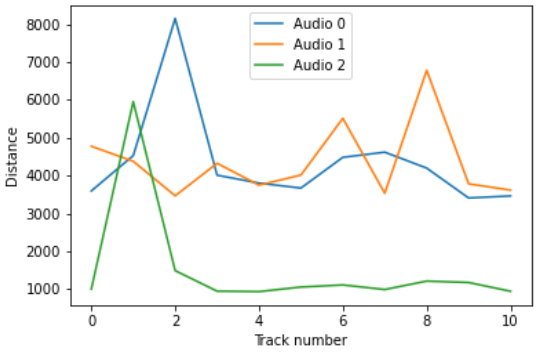
Setelah dilakukan pemodelan pada data menggunakan algoritma Gaussian Naive Bayes, 3 buah audio *testing* dengan nama file "0_george_test.wav", 1_george_test.wav, 2_george_test.wav" diuji coba dengan melakukan prediksi pada kata yang diucapkan. Hasil prediksi adalah identifikasi dari kata yang diucapkan oleh audio testing. Selanjutnya dilakukan plot pada jarak tiap tipe audio dengan audio testing. Jarak antara tiap tipe audio masing-masing dibedakan untuk melihat perbedaan jarak yang terjadi dari setiap tipe audio setelah melakukan uji prediksi.



Gambar 4.3 Grafik Jarak Kemiripan untuk Audio "0_george_test.wav"



Gambar 4.3 Grafik Jarak Kemiripan untuk Audio "1_george_test.wav"



Gambar 4.3 Grafik Jarak Kemiripan untuk Audio "2_george_test.wav"

```
pre,distance = predict_result('test/0_george_test.wav')
arr = ["0", "1", "2"] # Hasil prediksi
print(f"Audio menyebutkan angka {arr[int(pre)]}")
```

```
Audio menyebutkan angka 0
```

Gambar 4.4 Hasil Test Uji Audio 0

```
pre,distance = predict_result('test/1_george_test.wav')
arr = ["0", "1", "2"]
print(f"Audio menyebutkan angka {arr[int(pre)]}")
```

```
Audio menyebutkan angka 1
```

Gambar 4.5 Hasil Test Uji Audio 1

```
pre,distance = predict_result('test/2_george_test.wav')
arr = ["0", "1", "2"]
print(f"Audio menyebutkan angka {arr[int(pre)]}")
```

```
Audio menyebutkan angka 2
```

Gambar 4.6 Hasil Test Uji Audio 2

Hasil uji coba menunjukkan prediksi yang sesuai dari uji coba identifikasi kata menggunakan pendekatan DTW. Secara intuitif, dari seluruh grafik di atas juga dapat terlihat bahwa pada setiap kecocokan yang terjadi, plot audio yang sesuai berada di posisi paling bawah atau memiliki jarak terkecil, sehingga dapat diidentifikasikan kemiripan antara data test dan data training (data yang dibandingkan dengan data test) meskipun terdapat beberapa bias pada data.

V. KESIMPULAN

Speech recognition merupakan fitur yang sangat bermanfaat untuk mempermudah kehidupan sehari-hari. Salah satu algoritma yang digunakan untuk memproses hal ini adalah *Dynamic Time Warping*. *Dynamic Time Warping* digunakan untuk mengukur tingkat kemiripan antara dua buah data dengan pendekatan program dinamis. Berdasarkan uji coba yang penulis lakukan dengan melakukan identifikasi pada kata yang diucapkan oleh seseorang, dapat disimpulkan DTW memberi hasil akurat meskipun terdapat beberapa hasil yang bias. Oleh karena itu, dalam pengembangan teknologi *speech recognition* saat ini lebih memanfaatkan algortima lain seperti HMM dan *natural language processing* (NLP) yang menghasilkan tingkat kemiripan yang lebih akurat.

LINK SOURCE CODE

Untuk kode program dapat diakses pada link berikut: https://github.com/thomas-fm/project-stima

LINK YOUTUBE

Untuk penjelasan mengenai makalah ini, dapat melihat video pada link berikut: https://youtu.be/--Tq7lCPdGs

UCAPAN TERIMA KASIH

Penulis hendak mengucapkan syukur dan terima kasih kepada Tuhan Yang Maha Esa atas segala rahmat-Nya yang diberikan dalam bentuk ilmu pengetahuan sehingga makalah ini dapat selesai tepat pada waktunya. Penulis juga turut berterima kasih kepada Ibu Dr. Nur Ulfa Maulidevi, S.T., M.Sc sebagai dosen yang membimbing saya dalam kelas IF2211 Strategi Algoritma, Bapak Dr. Ir. Rinaldi Munir, Bapak Ir. Rila Mandala, M.Eng.,Ph.D., dan Bapak Prof.Ir. Dwi Hendratmo Widyantoro, M.Sc.,Ph.D. sebagai dosen pengampu dalam mata kuliah IF2211Strategi Algoritma.

DAFTAR PUSTAKA

[1] Munir, Rinaldi, informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Program-Dinamis-2020-Bagian1.pdf, diakses pada 10 Mei 2021

[2] Senin, Pavel, "Dynamic Time Warping Algorithm Review", pp. 3-11, Desember 2008

[3] Tim Penulis IBM. September,2020. "Speech Recognition", https://www.ibm.com/cloud/learn/speech-recognition diakses pada 10 Mei 2021

[4] Prathena. 2020. "The Dummy Guide to MFCC", https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd diakses pada 11 Mei 2021

[5] Mishra, Abhisek. 2020. "Time Series Similarity Using Dynamic Time Warping-Explained", https://medium.com/walmartglobaltech/time-series-similarity-using-dynamic-time-warping-explained-9d09119e48ec diakses pada 11 Mei 2021

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 26 April 2021

Thomas Ferdinand Martin 13519099