Penerapan Algoritma Viterbi untuk Memberikan Rekomendasi Genre Lagu sesuai Suasana Hati Pendengar

Rici Trisna Putra - 13522026[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13522026@std.stei.itb.ac.id

***Abstract*— Fenomena kemunculan Artificial Intelligence (AI) saat ini bukanlah sekadar tren belaka, melainkan sebuah revolusi besar dalam teknologi yang akan membentuk masa depan dunia. Sebagai salah satu Algoritma yang digunakan pada subbidang Artificial Intelligence yakni Natural Language Processing, Algoritma Viterbi memiliki potensi untuk dimanfaatkan sebagai pemberi rekomendasi lagu. Oleh karena itu makalah ini bertujuan untuk membuktikan hal tersebut dengan memodelkan permasalahan perekomendasian lagu sebagai sebuah Hidden Markov Model sehingga dapat diselesaikan oleh Algoritma Viterbi. Hasil menunjukkan bahwa memang Algoritma Viterbi memiliki potensi sebagai sistem perekomendasi lagu meskipun kemampuannya terbatas.**

***Kata Kunci*—Algoritma Viterbi, Hidden Markov Model, Graf, Artificial Intelligence**

I. PENDAHULUAN

Fenomena kepopuleran *Artificial Intelligence* (AI) dewasa ini merupakan bukti bahwa *Artificial Intelligence* bukan hanya sebuah tren belaka melainkan sebuah revolusi besar teknologi yang akan membentuk arah masa depan dunia. AI bukan hanya terbatas pada gambaran futuristik yang sering kita jumpai dalam karya fiksi ilmiah. Sebaliknya, AI akan menjadi realitas yang kita jumpai dalam kehidupan sehari-hari. Potensi AI yang sangat luas di berbagai bidang menunjukan seberapa pentingnya pengembangan dan pengaplikasian AI. Dengan pengaplikasian yang tepat dan bijaksana maka AI dapat membawa kesejahteraan bagi umat manusia. Untuk mewujudkan hal tersebut dibutuhkan pemahaman akan bagaimana AI bekerja.

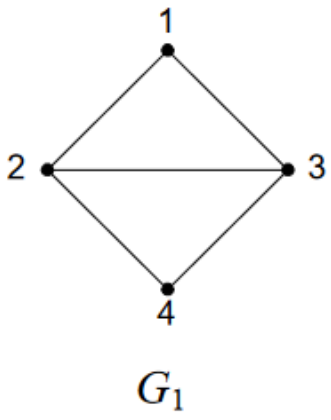
Algoritma Viterbi merupakan salah satu algoritma yang digunakan pada aplikasi Pemrosesan Bahasa Alami atau *Natural Language Processing* (NLP), yakni suatu subbidang pada *Artificial Intelligence* yang berfokus pada pengembangan kemampuan komputer untuk mengolah dan memahami bahasa manusia. Algoritma ini ditemukan oleh Andrew Viterbi pada tahun 1967. Pada *Natural Language Processing* algoritma ini diperkenalkan sebagai *part-of-speech tagging* (POST) yakni bertugas melabeli kata sesuai kedudukannya di dalam tata bahasa. Selain pada NLP, Algoritma Viterbi juga mempunyai peran dalam beberapa bidang lain seperti BioInformatika, *Motion Capture*, Parsing, dll. Algoritma Viterbi ternyata tidak lepas dari ranah ilmu Matematika Diskrit. Pemahaman terhadap Algoritma Viterbi merupakan suatu pengantar yang ideal tentang bagaimana Matematika Diskrit memiliki kontribusi besar pada berbagai bidang ilmu seperti *Artificial Intelligence*.

Untuk itu makalah ini bertujuan membahas pengaplikasian sederhana Algoritma Viterbi dalam suatu pencarian urutan genre lagu musik terbaik yang akan diputar sehingga sesuai dengan suasana hati pendengar. Algoritma Viterbi akan digunakan untuk menentukan solusi dengan mencari *Viterbi Path* dari Hidden Markov Model permasalahan tersebut.

II. LANDASAN TEORI

A. Graf

Graf dalam Matematika Diskrit digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut. Suatu Graf G didefinisikan sebagai sebuah pasangan himpunan (V,E) yang biasa ditulis dalam notasi G = (V,E). V merupakan himpunan tidak kosong yang berisi simpul-simpul (*vertices* atau *node*) sehingga V = { $v_1,v_2,v_3, \dots , v_n$}. Sedangkan E adalah himpunan semua sisi (*edges* atau *arcs*) yang menghubungkan sepasang simpul dalam graf sehingga E = { e_1,e_2,e_3 } . Misalkan graf G_1 memiliki simpul v_1,v_2,v_3,v_4maka V = { v_1,v_2,v_3,v_4 }. dan misalkan graf G memiliki sisi e_1,e_2,e_3, e_4 maka E = { e_1,e_2,e_3,e_4}



G_1

***Gambar** 1 Graf Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*

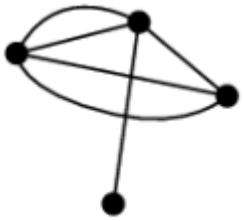
Graf digolongkan menjadi dua jenis berdasarkan ada atau tidaknya gelang (sisi yang menghubungkan suatu simpul dengan simpul itu sendiri) atau sisi ganda(terdapat lebih dari satu sisi yang menghubungkan dua buah titik) pada suatu graf yakni :

1. Graf Sederhana (*Simple Graph)*
   Merupakan graf yang tidak memiliki gelang maupun sisi ganda

simple graph

Gambar 2 *Graf Sederhana Sumber :* *https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*

2. Graf tak-sederhana (*unsimple-graph)*
   Merupakan graf yang memiliki gelang atau sisi ganda. Graf ini dibedakan lagi menjadi dua yakni graf ganda (*multi-graph)* atau graf yang memiliki sisi ganda dan graf semu (*pseudograph)* atau graf yang memiliki gelang.

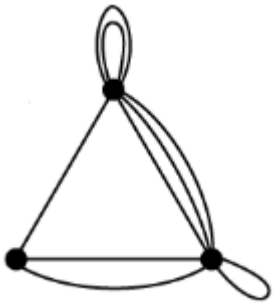


nonsimple graph with multiple edges

Gambar 3 *Graf tak-sederhana Sumber :* *https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*

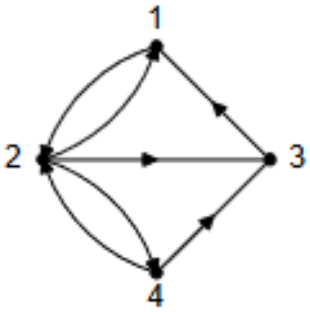
Selain berdasarkan ada atau tidaknya sisi ganda dan gelang, graf juga digolongkan berdasarkan orientasi arah pada sisi yakni :

1. Graf tak-berarah (*undirected graph*)
   Merupakan graf yang sisinya tidak mempunyai orientasi arah dan biasa digambarkan tanpa tanda panah.



Gambar 3 *Graf tak-berarah Sumber :* *https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*

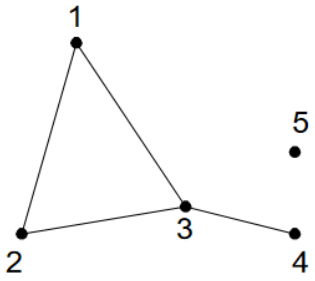
2. Graf Berarah (*directed graph*)
   Merupakan graf yang pada setiap sisinya mempunyai orientasi arah dimana arah tersebut biasa direpresentasikan oleh tanda panah.



Gambar 4 *Graf Berarah Sumber :* *https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*

B. Terminologi Graf

1. Ketetanggaan (*Adjacent*)
   Apabila terdapat dua buah simpul yang dihubungkan oleh sebuah sisi maka kedua simpul tersebut dikatakan bertetangga.

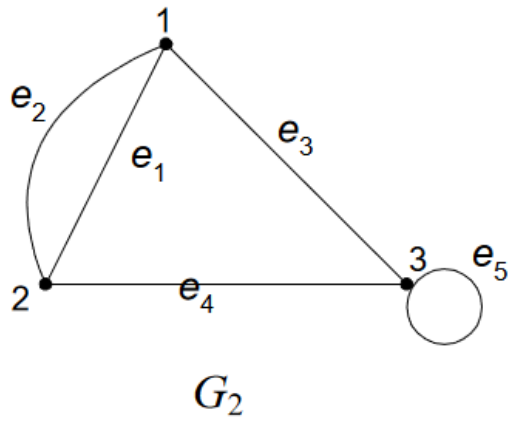


Gambar 5 *Ketetanggaan Sumber :* *https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*

Tinjau contoh graf di atas maka didapatkan bahwa simpul 1 bertetangga dengan simpul 2 dan 3, namun simpul 1 tidak bertetangga dengan simpul 4 maupun 5.

2. Bersisian (*Incidency*)

Apabila sebuah sisi e menghubungkan dua buah simpul v_1 dan v_2 maka dapat dikatakan sisi e bersisian dengan simpul v_1 atau e bersisian dengan simpul v_2.

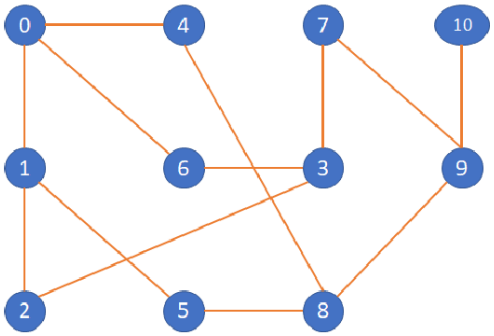


Gambar 6 *Bersisian Sumber :* https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf

Tinjau contoh graf di atas maka didapatkan bahwa sisi e1 dan e2 bersisian dengan simpul 1 dan simpul 2, akan tetapi tidak bersisian dengan simpul 3.

3. Lintasan (Path)

Lintasan dengan panjang n merupakan barisan berselang-seling simpul-simpul dan sisi-sisi yang berbentuk $v_0,e_1,v_1,e_2,v_2,\ldots, v_{n-1},e_n,v_n$ sedemikian sehingga $e_1 = (v_0,v_1)$, $e_2 = (v_1,v_2)$, … , $e_n = (v_{n-1},v_n)$ adalah sisi-sisi dari graf.

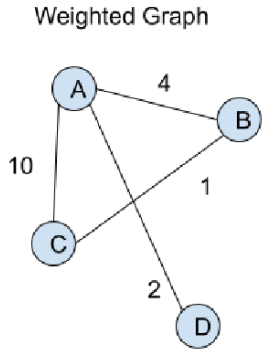


Gambar 7 *Lintasan Sumber :* https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf

Apabila ditinjau dari gambar di atas maka lintasan 0,6,3,7,9,10 melewati sisi (0,6), (6,3), (3,7),(7,9),(9,10) yang totalnya berjumlah 5 sisi sehingga lintasan tersebut memiliki panjang 5.

4. Graf Berbobot (*Weighted Graph*)

Merupakan jenis graf yang memiliki harga pada setiap sisinya.

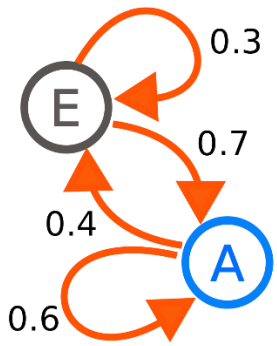


Gambar 8 *Graf Berbobot Sumber :* https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf

C. Markov Chain

1. Markov Chain

Markov Chain merupakan sebuah urutan variabel acak (*random variables)* dimana nilai variabel hanya bergantung pada variabel saat ini dan tidak bergantung pada variabel-variabel sebelumnya (*memoryless*) atau dengan kata lain urutan variabel acak tersebut memenuhi kaidah Markov. Markov Chain juga dapat di definisikan sebagai graf berarah dimana simpul merepresentasikan variabel acak dengan keadaan (*states)* dan sisi berarah yang memiliki bobot merepresentasikan transisi keadaan (*state transitions*).[2]



Gambar 8 *Markov Chain Sumber :* https://en.wikipedia.org/wiki/Discrete-time_Markov_chain

Tinjau gambar contoh di atas, Markov Chain tersebut memiliki dua buah *states* yakni A dan E dimana pada setiap states terdapat masing-masing dua buah sisi yang memiliki bobot. Bobot tersebut merepresentasikan kemungkinan transisi dari simpul asal sisi ke simpul akhir sisi tersebut.

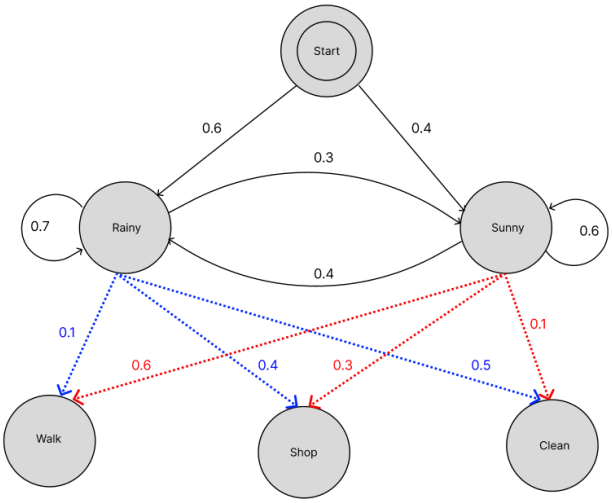
2. Random Walk

Random walk pada suatu Markov Chain merupakan lintasan yang ditempuh oleh sistem ketika berpindah dari satu *state* ke *state* yang lainnya dan

perpindahan antar *state* ini hanya dipengaruhi oleh *state* saat ini dan peluang transisi *state*. Misalkan dari Markov Chain pada contoh gambar 8 kita memulai dari *state* E, maka kita bisa berpindah menuju *state* A dengan peluang 0.7 atau tetap berada pada *state* E dengan peluang 0.3. Dengan berpindah secara random sesuai peluang yang ada pada sisi graf tersebut maka dapat diperoleh suatu lintasan E, A, A, A, E, A dengan peluang total mendapatkan lintasan (*random walk)* tersebut sebesar 0.7 x 0.6 x 0.6 x 0.4 x 0.7 = 0.07056 atau 7.056%.

D. Hidden Markov Model

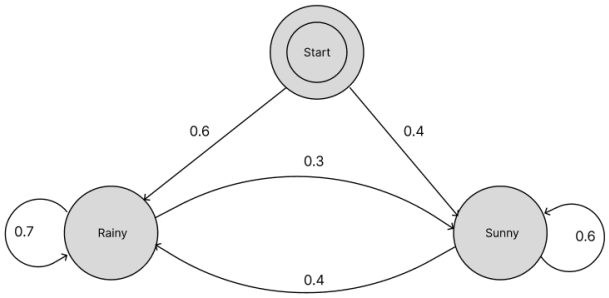
Hidden Markov Model (HMM) merupakan sebuah model statistika dimana sistem yang dimodelkan diasumsikan merupakan sebuah Markov Chain (sebuah urutan variabel acak yang memenuhi kaidah Markov) yang biasa disebut *X* dengan beberapa *state* teramati. Dalam suatu HMM akan ada proses *Y* yang dapat diamati dan dipengaruhi oleh proses *X* yang tidak dapat diamati (*hidden states)* namun pengaruh *X* terhadap *Y* terdefinisi dengan jelas. Karena *X* yang tidak dapat diamati namun mempengaruhi *Y*, maka tujuan utama kita adalah mempelajari *X* melalui pengamatan terhadap *Y*.



Gambar 9 *Hidden Markov Model*
Sumber : dokumentasi penulis

Pada contoh HMM di atas terdapat sebuah Markov Chain *X* = {*Rainy*, *Sunny*, *Start*} yang prosesnya (*Random walk*) tersembunyi. Selain *X*, terdapat pula kumpulan *state* yang dapat diamati yakni *Y* = {*Walk*, *Shop*, *Clean*}. Pada setiap *state* di *X* terdapat suatu sisi yang menghubungkan *state* tersebut dengan setiap *state* di *Y*. Sisi tersebut juga memiliki bobot yang merepresentasikan peluang, namun peluang ini bukan peluang perpindahan *state* melainkan peluang apabila berada pada salah satu *state* di *X* maka seberapa mungkin setiap *state* di *Y* tersebut muncul sebagai *state* yang teramati. Peluang ini disebut sebagai Peluang Emisi (*Emission Probability*) dan merupakan hal yang dimaksud sebagai pengaruh *X* terhadap *Y* yang terdefinisi secara jelas.

Ketika terjadi suatu *random walk* pada Markov Chain *X*, maka urutan *state* di random walk tersebut tidak akan terlihat, sedangkan urutan *state Y* hasil emisi akan terlihat.



Gambar 10 *Markov Chain X tanpa state yang teramati*
Sumber : dokumentasi penulis

perlu diperhatikan bahwa suatu urutan hasil emisi dihasilkan oleh lebih dari satu kemungkinan *random walk*. Misalnya urutan *state* teramati *walk, shop, walk, clean* dapat di hasilkan oleh *random walk* berikut: *rainy, sunny, rainy, sunny*. Namun juga dapat dihasilkan oleh *random walk* lainnya seperti: *sunny, rainy, sunny, sunny* dan berbagai *random walk* lainnya. Perbedaan *random walk* ini mengakibatkan perbedaan peluang kejadian suatu urutan *state* hasil emisi yang sama. Pada contoh sebelumnya peluang kejadian urutan *walk, shop, walk, clean* yang dihasilkan oleh *random walk* pertama adalah (0.6 x 0.1) x (0.3 x 0.3) x (0.4 x 0.1) x (0.3 x 0.1) = 0.0000648. Sedangkan peluang kejadian urutan tersebut yang dihasilkan oleh *random walk* kedua adalah (0.4 x 0.6) x (0.4 x 0.4) x (0.3 x 0.6) x (0.6 x 0.1) = 0.0041472 atau 64 kali lebih mungkin terjadi dari kejadian sebelumnya.

E. Algoritma Viterbi

Algoritma Viterbi merupakan algoritma pemrograman dinamis yang digunakan untuk menemukan urutan paling mungkin dari *hidden states* (*X*) yang selanjutnya dapat disebut sebagai Lintasan Viterbi (*Viterbi Path*) berdasarkan dari suatu urutan *states* yang teramati (*Y*). Algoritma ini dinamakan atas nama penemunya yakni Andrew Viterbi yang mengajukan algoritma ini pada tahun 1967 sebagai algoritma dekoding untuk *convolutional codes* pada *noisy digital communication links.*

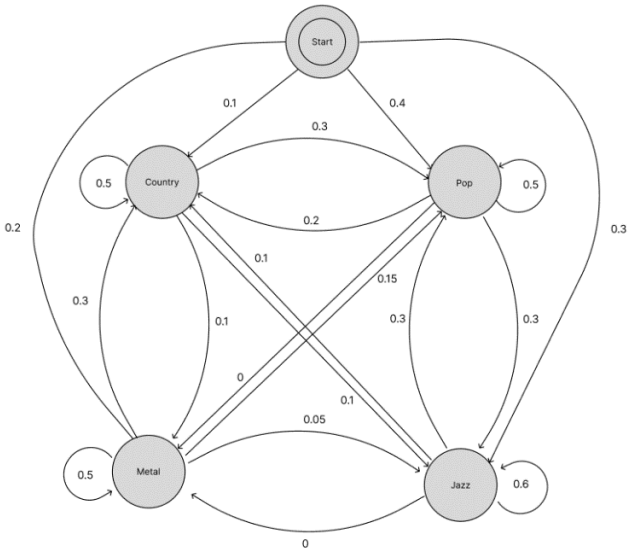
Seperti yang sudah dijelaskan pada bagian HMM, terdapat banyak kemungkinan *random walk* pada *Markov Chain X* yang hasil emisinya sama dan sudah kita cek pula bahwa untuk setiap *random walk* tersebut menghasilkan emisi urutan *states* teramati yang sama dengan peluang kejadian yang berbeda. Oleh karena itu Algoritma Viterbi disini bertugas untuk mencari *random walk* paling mungkin (memiliki peluang terbesar) yang menghasilkan emisi suatu urutan *states* teramati (Lintasan Viterbi).

Untuk sebuah HMM dengan sebuah urutan observasi $O = \{O_1, O_2, \ldots, O_N\}$, *Hidden states* $S = \{s_1, s_2, \ldots, s_K\}$, sebuah larik peluang awalan (dari *start state* menuju *state* lain) $\Pi = (\pi_1, \pi_2, \ldots, \pi_K)$, sebuah Matriks Transisi A dengan ukuran K x K sehingga semua peluang transisi dapat termuat di dalamnya dan Matriks emisi B dengan ukuran K x N sehingga semua peluang teramatinya *state* teramati dapat disimpan. Maka secara umum Algoritma Viterbi dapat menemukan Lintasan Viterbi dengan Langkah-langkah sebagai berikut:

```
function Viterbi(O, S, Π, Tm, Em): best_path
    # Untuk menampung peluang setiap state apabila diberi
sebuah pengamatan
    trellis = matrix(len(S), len(O))

    # Untuk menampung backpointer ke state terbaik sebelumnya
    pointers = matrix(len(S), len(O))

    # Menghitung peluang emisi setelah keluar dari start state
    for s in range(len(S)):
        trellis[s][0] = Π[s] * Em[s][O[0]]  #

    # Mencarikan setiap state sebuah state tepat sebelumnya
yang paling mungkin
    for o in range(1, len(O)):
        for s in range(len(S)):
            k = argmax(trellis[k][o-1] * Tm[k][s] * Em[s][O[o]]
for k in range(len(S)))
            trellis[s][o]= trellis[k][o-1] * Tm[k][s]*
Em[s][O[o]]
            pointers[s][o] = k

    # Mencari k dari final state yang paling bagus
    k = argmax(trellis[k][len(O)-1] for k in range(len(S)))

    # backtrack untuk mencari semua state sebelumnya dan
memasukkan ke best path.
    best_path = []
    for o in range(len(O)-1, -1, -1):
        best_path.insert(0, S[k])
        k = pointers[k][o]

    return best_path
```

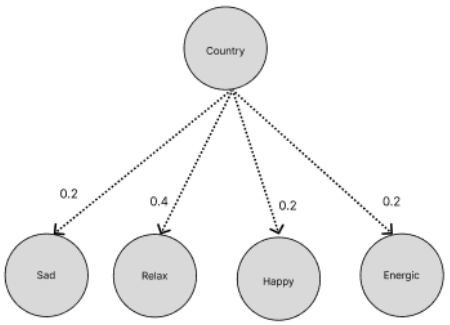
III. Aplikasi Algoritma Viterbi Dalam Menentukan Genre Musik Terbaik sesuai suasana hati

A. Model Permasalahan

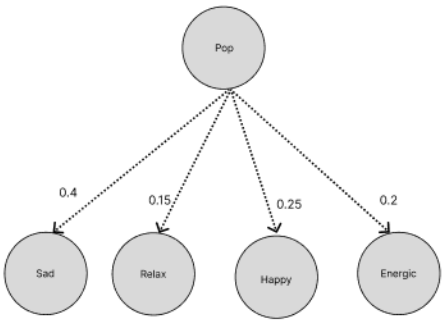


Gambar 11 *Hidden Markov Model dari Permasalahan*
Sumber : dokumentasi penulis

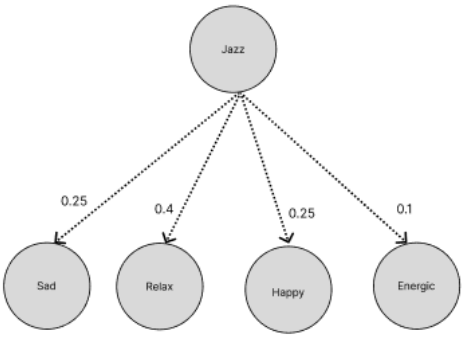
Pada permasalahan ini terdapat sebuah Hidden Markov Model dimana Markov Chainnya memiliki 5 *hidden states* beserta *state* awal (*start*). Setiap *state* tersebut melambangkan sebuah genre lagu kecuali *start* dan setiap genre lagu memiliki peluang transisi ke *state* genre lagu lainnya. Selain itu terdapat pula state yang dapat diamati yakni *state* yang berisi suasana hati pendengar lagu. *State* yang dapat diamati tersebut tentunya memiliki peluang emisi (*emission probability*) pada setiap *hidden states*.



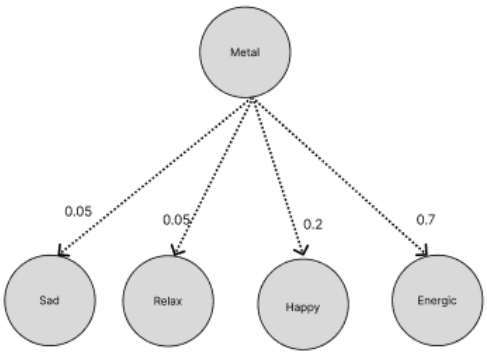
Gambar 12 *Peluang emisi untuk genre country*
Sumber : dokumentasi penulis



Gambar 13 *Peluang emisi untuk genre pop*
Sumber : dokumentasi penulis



Gambar 14 *Peluang emisi untuk genre jazz*
Sumber : dokumentasi penulis



Gambar 15 *Peluang emisi untuk genre metal*
Sumber : dokumentasi penulis

Maka dengan menggunakan Algoritma Viterbi sesuai dengan definisi HMM tersebut kita dapat mencari Lintasan Viterbi untuk setiap urutan *state* teramati.

Oleh karena itu Algoritma Viterbi diimplementasikan dalam bahasa Python sebagai berikut ini dengan sedikit tambahan fungsi print untuk menampilkan peluang terbesar pada tiap step. Maka dengan definisi HMM seperti yang telah dijelaskan, Lintasan Viterbi pada permasalahan ini dapat dicari.

```
def viterbi(obs, states, start_p, trans_p, emit_p):
    V = [{}]
    for st in states:
        V[0] [st] = {"prob": start_p[st] * emit_p[st] [obs[0]],
"prev": None}
    for t in range(1, len(obs)):
        V.append({})
        for st in states:
            max_tr_prob = V[t - 1] [states[0]] ["prob"] *
trans_p[states[0]] [st] * emit_p[st] [obs[t]]
            prev_st_selected = states[0]
            for prev_st in states[1:]:
                tr_prob = V[t - 1] [prev_st] ["prob"] *
trans_p[prev_st] [st] * emit_p[st] [obs[t]]
                if tr_prob > max_tr_prob:
                    max_tr_prob = tr_prob
                    prev_st_selected = prev_st

            max_prob = max_tr_prob
            V[t] [st] = {"prob": max_prob, "prev":
prev_st_selected}

    for line in dptable(V):
        print(line)

    opt = []
    max_prob = 0.0
    best_st = None
    # Get most probable state and its backtrack
    for st, data in V[-1].items():
        if data["prob"] > max_prob:
            max_prob = data["prob"]
            best_st = st
    opt.append(best_st)
    previous = best_st

    # Follow the backtrack till the first observation
    for t in range(len(V) - 2, -1, -1):
        opt.insert(0, V[t + 1] [previous] ["prev"])
        previous = V[t + 1] [previous] ["prev"]

    print ("Urutan states yang paling mungkin adalah " + "
".join(opt) + " dengan kemungkinan kejadian sebesar %s" %
max_prob)

def dptable(V):
    # Print a table of steps from dictionary
    yield " " * 5 + "     ".join(("%3d" % i) for i in
range(len(V)))
    for state in V[0]:
        yield "%.7s: " % state + " ".join("%.7s" % ("%lf" %
v[state] ["prob"]) for v in V)
```

Misalkan teramati bahwa pendengar mengalami urutan suasana hati sebagai berikut: *Energic*, *Sad*, *Relax*, *Relax*. Maka dengan menggunakan Algoritma Viterbi didapatkan:

```
      0       1       2       3
Country: 0.02000 0.00840 0.00168 0.00033
Pop: 0.08000 0.01600 0.00120 0.00009
Jazz: 0.03000 0.00600 0.00192 0.00046
Metal: 0.14000 0.00350 0.00008 0.00000
Urutan states yang paling mungkin adalah Pop Pop Jazz Jazz dengan
kemungkinan kejadian sebesar 0.0004608000000000002
```

Sehingga urutan genre lagu yang paling pas dan paling mungkin untuk didengar pendengar sesuai urutan suasana hati *Energic*, *Sad*, *Relax*, *Relax* adalah *Pop*, *Pop*, *Jazz*, *Jazz* dengan peluang kejadian sebesar 0.0004608.

Misalkan lagi pada lain waktu pendengar mengalami urutan suasana hati sebagai berikut: *Energic*, *Relax*, *Relax*, *Sad*. Maka akan didapat:

```
      0       1       2       3
Country: 0.02000 0.01680 0.00336 0.00033
Pop: 0.08000 0.00600 0.00075 0.00040
Jazz: 0.03000 0.00960 0.00230 0.00034
Metal: 0.14000 0.00350 0.00008 0.00001
Urutan states yang paling mungkin adalah Metal Country Country Pop dengan
kemungkinan kejadian sebesar 0.0004032000000000004
```

Sehingga urutan genre lagu yang paling pas dan paling mungkin untuk didengar pendengar sesuai urutan suasana hati *Energic*, *Relax*, *Relax*, *Sad* adalah *Metal*, *Country*, *Country*, *Pop* dengan peluang kejadian sebesar 0.0004032.

IV. Kesimpulan

Matematika Diskrit merupakan disiplin ilmu yang memiliki aplikasi yang luas dalam berbagai bidang ilmu. Salah satu pengaplikasian tersebut adalah dalam bidang *Artificial Intelligence*. Hal ini dapat kita jumpai di dalam Algoritma Viterbi yang menerapkan konsep Graf yang ada pada Matematika Diskrit. Algoritma Viterbi ini terbukti memiliki potensi yang besar, karena meskipun belum sepenuhnya digunakan dengan semua kapabilitasnya, Algoritma ini terbukti memiliki potensi sebagai Sistem Rekomendasi Lagu seperti yang telah penulis bahas. Tentunya permasalahan yang penulis bawa masih bisa dikembangkan lagi untuk bisa mendapatkan rekomendasi lagu dengan sistem yang lebih rumit asalkan masih memenuhi kriteria Markov. Dengan adanya pembahasan mengenai Algoritma ini penulis juga berharap agar semua pembaca bisa dapat lebih memahami bagaimana pentingnya Matematika Diskrit dan bagaimana *Artificial Intelligence* bekerja.

V. Ucapan Terima Kasih

Penulis ingin mengucapkan terima kasih kepada beberapa pihak. Pertama, penulis mengucapkan puji syukur kepada Allah SWT karena atas rahmat dan hidayah-Nya, penulis dapat menyelesaikan makalah berjudul “Penerapan Algoritma Viterbi untuk Memberikan Rekomendasi Genre Lagu sesuai Suasana Hati Pendengar” dengan lancar dan baik. Selain itu tidak lupa terima kasih saya ucapkan kepada dosen pengampu mata kuliah Matematika Diskrit yakni, Dr. Nur Ulfa Maulidevi S. T, M.Sc, Dr. Ir. Rinaldi Munir, M. T. dan Dr. Fariska Zakhralativa Ruskanda, S.T. yang telah sabar memberikan bimbingan penulis selama menjalani mata kuliah ini. Tidak lupa juga ucapan terima kasih kepada seluruh sumber yang telah dijadikan referensi pada pembuatan makalah ini.

References

[1] *https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf*, diakses pada 11 Desember 2023 pukul 07.00

[2] Dabrowski Christopher, Hunt Fern, “Using markov chain and graph theory concepts to analyze behavior in complex distributed systems,”, diakses pada 11 Desember 2023 pukul 07.46

[3] https://www.geeksforgeeks.org/hidden-markov-model-in-machine-learning/ , diakses pada 11 Desember 2023 pukul 08.00

[4] https://www.stat.purdue.edu/~junxie/topic3.pdf, diakses pada 11 Desember pukul 22.10

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, 11 Desember 2023

Rici Trisna Putra 13522026