Rancang Bangun Aplikasi MusicMoo dengan Metode MIR (Music Information Retrieval) pada Modul Fingerprint dan Song Recommendation

Mochammad Faris Ponighzwa Rizkanda, Riyanarto Sarno, dan Dwi Sunaryo
Departemen Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS)
e-mail: riyanarto@if.its.ac.id

***Abstrak*—Industri musik sudah mulai merambah ke bidang komputer. Salah satunya adalah aplikasi Soundhound, Shazam, dan masih banyak lagi. Namun semua aplikasi tersebut hanya melakukan deteksi dari potongan suara yang direkam. Semua aplikasi tersebut bekerja dengan cara melakukan ekstrak *fingerprint* hanya dari beberapa segmen sinyal *audio* yang direkam. Pada penelitian yang dijadikan sebagai studi ini akan dibangun suatu sistem aplikasi untuk melakukan proses terhadap suatu lagu menggunakan ekstraksi fitur sesuai dengan standar MPEG-7. Pertama, penulis membangun *database* yang berisi fitur lagu. Di dalam fitur tersebut terdapat kumpulan nilai yang mengidentifikasikan suatu lagu. Dari deskripsi ini digunakan untuk melakukan pencarian pada suatu lagu. Kedua, melakukan proses pada fitur audio yang terkait. Ketiga, melakukan klasifikasi dan hasilnya adalah detail informasi *fingerprint* dan *rekomendasi* pada suatu lagu.**

***Kata Kunci—Analisa Audio, MIR,* MPEG-7.**

I. PENDAHULUAN

AKHIR-akhir ini, industri musik berkembang sangat pesat. Tidak terkecuali, industri musik di Indonesia juga mulai terkena dampaknya. Aplikasi seperti Shazam dan Soundhound dapat mendeteksi potongan *audio* yang direkam dan menebak judul lagu dari potongan *audio* tersebut. Sehingga, masyarakat tidak perlu merasa kesusahan ketika menginginkan judul lagu dari potongan *audio* yang terdengar di tempat umum.

Namun aplikasi tersebut hanya sebatas mengidentifikasi dan melakukan pencocokan dari suatu *audio*. Padahal, telah ditemukan suatu istilah MIR dengan kepanjangan *Music Information Retrieval*, yang berarti pencarian keterangan informasi dari suatu musik. *MIR* merupakan istilah yang digunakan ketika ingin mendapatkan informasi dari suatu musik. Informasi yang didapat dapat berupa judul lagu, genre, tempo, dan lainnya.

Pada penelitian sebelumnya yang terkait dengan studi ini, identifikasi musik dengan memanfaatkan *fingerprint* berdasarkan standar MPEG-7 sudah dilakukan dengan menggunakan *sliding algorithm* [1]. Penelitian tersebut yang dijadikan acuan dalam implementasi pada modul *fingerprint*. Pada penelitian tersebut pengujian dilakukan dengan cara membandingkan lagu berdasarkan *file* asli terhadap *file* uji yang sudak dilakukan pemotongan bagian. Pada penelitian ini, penulis akan merancang sebuah sistem sehingga pengujian dilakukan dengan cara merekam potongan lagu dari perangkat bergerak yang akan dideteksi oleh sistem. Penelitian ini menggunakan acuan MPEG-7 dikarenakan hasil ekstraksi fitur dari MPEG-7 berupa metadata. Metadata ini akan digunakan sebagai bahan dasar untuk melakukan klasifikasi suara, sehingga dalam penelitian ini tidak menggunakan bahan dasar dalam klasifikasi yang berbasis konten. Penelitian ini menggunakan ekstraksi fitur dari *file* berekstensi .wav dan dilakukan klasifikasi. Sehingga dapat dikatakan bahwa penelitian ini menggunakan metadata dalam pengolahannya, dan bukan melakukan klasifikasi berbasis konten.

Pada penelitian ini, penulis akan melakukan implementasi MIR. Penulis akan melakukan penelitian metode dan langkah-langkah yang tepat, agar istilah MIR dapat diimplementasikan. Fokus dalam penelitian kali ini adalah, pada modul *fingerprint*, dan *recommendation*. *Fingerprint* yang berarti *audio* akan diidentifikasi dengan hanya potongan *audio* yang dicari. *Reccomendation*, yang berarti sistem akan mencoba merekomendasikan lagu dari lagu yang sudah dicari saat ini.

Standar yang digunakan penulis adalah MPEG-7 yang sudah menjadi standar dalam konten multimedia berdasarkan ISO/IEC 15938 yang berekstensi XML. XML ini akan diperoleh dari *library MPEG7AudioEnc* yang bersifat *open source*. XML tersebut akan dilakukan *query* untuk diambil fitur-fiturnya dalam bentuk matriks. Fitur-fitur inilah yang digunakan untuk melakukan pemanggilan informasi musik/MIR. Kumpulan fitur dalam *file* XML ini akan disimpan dalam *database* untuk pemanggilan informasi (*fingerprint* dan *recommendation*) yang dimaksud.

Diskusi pada penelitian ini dibagi dalam struktur sebagai berikut. Bab II membahas materi dan metode yang digunakan dalam percobaan ini. Bab III membahas hasil dan diskusi percobaan yang dilakukan. Terakhir pada Bab IV membahas kesimpulan dari hasil yang didapatkan dalam percobaan ini.

II. MATERI DAN METODE

Pada modul *fingerprint* dan *recommendation* digunakan fitur *Audio Signature type* yang merupakan fitur yang menjadi ciri khas pada sebuah lagu. Sehingga dapat digunakan untuk mencari kemiripan sebuah lagu beserta lagu yang menjadi rekomendasi bagi para pendengarnya. *Audio Signature Type*

$$\begin{bmatrix} A_{1,1} & A_{1,2} & A_{1,3} & \cdots & A_{1,M-1} & A_{1,M} \\ A_{2,1} & A_{2,2} & A_{2,3} & \cdots & A_{2,M-1} & A_{2,M} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ A_{N-1,1} & A_{N-1,2} & A_{N-1,3} & \cdots & A_{N-1,M-1} & A_{N-1,M} \\ A_{N,1} & A_{N,2} & A_{N,3} & \cdots & A_{N,M-1} & A_{N,M} \end{bmatrix}$$

Gambar 1. Sinyal dalam Bentuk Matriks

dapat dicari dengan cara melakukan pengolahan dari *Audio Signature Type*. Pengolahan secara umum subbab berikut akan membahas materi dan metode yang digunakan dalam penelitian ini.

A. Fingerprint

Fingerprint merupakan suatu istilah yang memiliki definisi sebagai ciri khas dari sebuah lagu. *Fingerprint* digunakan untuk mengidentifikasi pada sebuah lagu. Bentuk dari *fingerprint* yang dimiliki pada sebuah lagu adalah sebuah sinyal. Sinyal ini bersifat unik, sehingga antara satu lagu dengan lagu lain memiliki *fingerprint* yang berbeda. Sehingga dari sinyal inilah yang digunakan untuk mengidentifikasi suatu musik (*audio*). *Fingerprint* suatu musik didapatkan dengan cara melakukan pengambilan fitur *Audio Signature Type* [2].

Gambar 1 merupakan contoh dari sinyal yang tersimpan dalam bentuk matriks. Matriks tersebut memiliki ukuran $n \times m$ dengan nilai *m* adalah 16 untuk *Audio Signature Type* dan nilai *n* yang tergantung dari ukuran atau durasi dari *file* musik (*audio*) yang terkait. Matriks inilah yang merupakan ciri khas dari suatu musik yang diolah sistem untuk melakukan pendeteksian sebuah lagu.

B. Audio Signature Type

Audio Signature (AS) adalah suatu fitur yang menjadi mencerminkan identik suatu lagu [2]. *AS* bisa juga disebut sebagai *fingerprint* dari sebuah lagu. Tujuannya untuk menghitung similaritas dari suatu lagu dengan lagu yang lain.

C. Xquery

Xquery adalah bahasa untuk melakukan *query*/ pemanggilan data di dalam suatu dokumen XML [3]. Dalam penelitian kali ini *Xquery* akan dieksekusi pada bahasa pemrograman Java dengan menggunakan *library BaseX*.

Gambar 2 merupakan salah satu contoh implementasi *Xquery* dalam pengambilan suatu fitur pada dokumen XML. *Library* BaseX akan menerima input berupa varibel *string* untuk eksekusi Xquery pada dokumen XML. Hasil dari proses tersebut adalah variabel *string* yang merupakan isi dari *tag* dalam suatu dokumen XML.

D. MIR (Music Information Retrieval)

MIR merupakan suatu istilah yang melakukan pengolahan sinyal dari suatu musik untuk tingkat yang lebih lanjut. Dari MIR dapat dilakukan identifikasi *fingerprint*, *genre identification*, *cover identification*, dan sebagainya [4].

```
1.  String AudioSpectrumFlatnessType(String string){
2.      String query =
3.        "declare default element namespace \"urn:mpe
    g:    mpeg7:schema:2001\";" +
4.                          "declare namespace mpeg7
    = \"urn:mpeg:mpeg7:schema:2001\";" +
5.                          "declare namespace xsi =
    \"http://www.w3.org/2001/XMLSchema-
    instance\";" +
6.                          "for $x in doc(\"C:/Users
    /ponighzwa/Desktop/CoverSong2/Test/XML/"+ string+
    ".XML\")/Mpeg7/Description/MultimediaContent/" +
7.                          "Audio/AudioDescriptor\n
    return if($x/@xsi:type=\"AudioSpectrumFlatnessTyp
    e\")then data($x/SeriesOfVector/Raw) else \"\"";
8.          //System.out.println(new Xquery(query).ex
    ecute(context));
9.          String hasil = new Xquery(query).execute(
    context);
10.         return hasil;
11. }
```

Gambar 2. Aplikasi Xquery

E. Play Framework

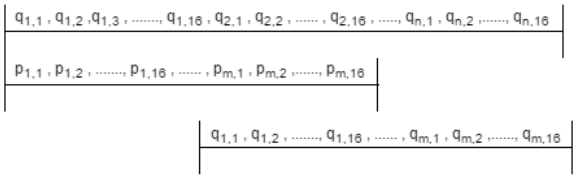
Suatu *framework* dalam pemrograman bahasa Java yang digunakan sebagai aplikasi berbasis web [5]. *Play Framework* digunakan sebagai server yang mengolah data *audio* / musik untuk dilakukan ekstrak data dan memperolah sinyal-sinyal yang terkait. Dalam penelitian ini *Play Framework* akan dipanggil melalui *file routes*. Sehingga ketika alamat yang tersimpan dalam *file routes* dipanggil, *Play Framework* akan melakukan kompilasi ekstraksi fitur sesuai dengan alamat yang dipanggil.

F. Flask

Suatu *web application framework* yang terdapat pada bahasa pemrograman Python [6]. *Flask* digunakan sebagai server yang melakukan perhitungan dan klasifikasi data uji terhadap dataset. Eksperimen ini memilih *Flask*, sebab *Flask* memiliki tingkat fleksibel yang sangat tinggi untuk digunakan bersamaan dengan *library* Python pada umumnya. Sama halnya dengan *Play Framework*, *Flask* memiliki *routes* yang dapat dipanggil. Ketika *routes* dalam *Flask* dipanggil, maka *Flask* akan melakukan kompilasi sesuai dengan *routes* yang bersangkutan.

G. Sliding Algorithm

Suatu algoritma yang digunakan dalam penelitian ini untuk mencari *distance* dari *subband* matriks. Dalam eksperimen ini, perhitungan *distance* yang digunakan adalah metode *Euclidian Distance*. Hasil *distance* yang didapat akan dilakukan klasifikasi menggunakan KNN. *Sliding algorithm* diusulkan, karena ditemukan masalah dalam melakukan klasifikasi. Masalah tersebut adalah fitur dalam MPEG-7 tidak berupa satu data, melainkan berbentuk matriks.



Gambar 3. *Sliding Algorithm*

Data Latih Bunga Iris

Sepal Width	Sepal Length	Petal Width	Petal Height	Iris Class
5.1	3.5	1.4	0.2	I.Setosa
4.9	3.0	1.4	0.2	I.Setosa
….	….	….	….	….
….	….	….	….	….
….	….	….	….	

Data Uji Bunga Iris

Sepal Width	Sepal Length	Petal Width	Petal Height	Iris Class
4.0	3.3	2.0	0.5	??

Data Latih Sinyal

Feature A	Feature B	Label
$[A_1, A_2, ..., A_n]$	$[B_1, B_2, ..., B_n]$	Label A
$[A_1, A_2, ..., A_n]$	$[B_1, B_2, ..., B_n]$	Label B
…. ….	…. ….	….
…. ….	…. ….	….
…. ….	…. ….	….

Data Uji Sinyal

Feature A	Feature B	Label
$[A_1, A_2, ..., A_m]$	$[B_1, B_2, ..., B_m]$	???

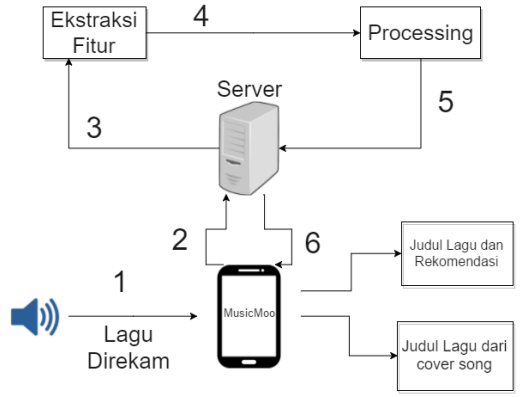
Gambar 4. Perbandingan Klasifikasi Sinyal dan Klasifikasi Bunga Iris

$$d(p,q) = \sqrt{\sum_i^n (p_i - q_i)^2} \quad (1)$$

Gambar 3 menjelaskan bagaimana *sliding algorithm* bekerja [1]. *Sliding algorithm* akan membandingkan setiap bagian dari matriks, hasil *distance* terkecil dari metode ini dianggap mewakili nilai *similarity* dari matriks tersebut. Perhitungan *distance* pada *sliding algorithm* yang dipakai menggunakan (1), yang merupakan rumus perhitungan jarak menggunakan metode *Euclidian Distance* [7].

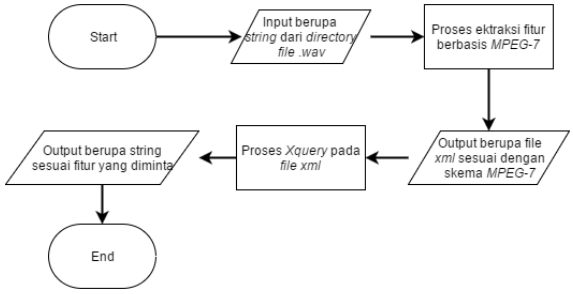
H.KNN

Dalam penelitian ini implementasi klasifikasi KNN tidak dapat langsung diaplikasikan kedalam studi kasus. Hal ini disebabkan karena perbedaan dimensi matriks pada fitur antara data latih dan data uji. Sebagai suatu contoh pada kasus bunga *iris*, data latih dan data uji merupakan suatu data yang *single*.



Keterangan Proses :
Proses 1 : Perekaman musik
Proses 2 : Musik diupload pada server
Proses 3 : Ekstraksi Fitur pada potongan Musik
Proses 4 : Hasil ekstraksi dikirimkan pada server *Python*
Proses 5 : Kalkulasi pada fitur dari potongan Musik
Proses 6 : Hasil dikirimkan kembali pada aplikasi android untuk ditampilkan

Gambar 5. Deskripsi Umum Sistem



Gambar 6. Diagram Alur Ekstraksi Fitur

Namun pada implementasi dalam penelitian ini, data latih dan data uji berbentuk suatu matriks yang berbeda ukurannya. Perbedaan ukuran sebabkan oleh perbedaan ukuran dan panjang *file* dari suatu musik. Pengolahan sinyal dengan klasifikasi KNN sudah pernah dilakukan sebelumnya pada bidang EEG [8], [9]. Penelitian ini akan mengacu pada penelitian tersebut untuk melakukan klasifikasi pada sinyal.

Sebagai contoh, pada Gambar 4 kasus klasifikasi bunga *iris* dapat langsung diaplikasikan klasifikasi KNN dengan implementasi perhitungan rumus jarak yang ada. Namun pada Gambar 4 kasus klasifikasi pada penelitian ini tidak dapat langsung diimplementasikan karena terdapat perbedaan dimensi antara data uji dan data latih. Untuk menangani kasus ini, maka klasifikasi KNN perlu dikombinasikan dengan algoritma *Sliding Algorithm* yang telah dibahas pada subbab sebelumnya.

III. HASIL DAN DISKUSI

Aplikasi yang akan dibuat pada Studi ini adalah program aplikasi *mobile*. Alur kerja pada sistem akan digambarkan pada Gambar 5.

A. Ekstraksi Fitur

Gambar 6 merupakan diagram alur mengenai proses ekstraksi fitur. Proses ini menerima *input* berupa *file* musik

dengan format .wav dan menghasilkan *output* berupa nilai dari fitur sesuai dengan modul yang diminta. Proses implementasi ekstraksi fitur terjadi pada sisi server Java. Secara umum setiap kelas pada server Java memiliki tujuan untuk melakukan ekstraksi fitur pada *file* musik dengan format .wav menjadi format dokumen XML sesuai dengan standar MPEG-7. Namun perbedaan yang paling utama adalah *Xquery* yang diimplementasikan pada dokumen XML. Hasil keseluruhan dari proses ekstraksi fitur ini adalah nilai *string* yang berisi tentang fitur yang digunakan pada masing-masing modul.

B. Processing

Tahapan *processing* secara garis besar terdapat 2 tahapan utama yaitu tahapan *pre-processing* sinyal dan tahapan *processing* sinyal. Tahapan *pre-processing* sinyal meliputi aplikasi dan konversi dari data *string* yang tersimpan dalam *database* menjadi *list* pada Python. Tahapan *processing* sinyal meliputi aplikasi KNN dengan dimodifikasi bersama dengan *sliding algorithm*.

Proses pada modul *fingerprint* dilakukan dengan melakukan rekaman dari potongan suara berupa lagu asli. Potongan lagu yang direkam akan disimpan dalam penyimpanan lokal pada perangkat bergerak yang digunakan. Rekaman akan disimpan dalam ekstensi *audio* .wav. Ekstensi *audio* .wav akan dilakukan *upload* kepada server dan dilakukan perhitungan. Server akan mengembalikan nilai berupa tipe data *string* yang berisi tentang judul lagu dari potongan lagu yang dicari.

Proses pemberian rekomendasi pada lagu akan mengikuti pada modul *fingerprint*. Sebab modul *recommendation* merupakan modul yang *include* pada modul induk, yaitu *fingerprint*. Salah satu *output* dari modul *fingerprint* selain lagu asli adalah, modul *recommendation*. Modul *recommendation* akan memberikan lagu rekomendasi yang sesuai berdasarkan nilai kemiripan potongan lagu yang dicari pada saat ini.

Secara keseluruhan aplikasi Android akan merekam musik dari lingkungan sekitar. Hasil dari proses rekam aplikasi Android akan menghasilkan *file* berupa format .wav. *File* tersebut akan dikirim pada server untuk dilakukan ekstraksi fitur. Proses ekstraksi fitur terjadi pada server Java dan menghasilkan dokumen XML sesuai dengan standar ISO MPEG-7. *Xquery* akan diaplikasikan pada dokumen XML untuk diambil fitur yang sesuai dengan modul *fingerprint*. Fitur yang sesuai dengan modul *fingerprint* adalah fitur *Audio Signature Type*. Fitur ini digunakan dalam perhitungan pada server Python untuk mencari kemiripan sebuah musik. Hasil dari perhitungan akan ditampilkan pada aplikasi Android sesuai dengan tujuan modul masing-masing. Modul *fingerprint* memiliki keluaran berupa judul lagu yang dicari. Modul *recommendation* memiliki keluaran berupa daftar dari judul dari lagu yang diperdengarkan.

Perancangan data latih dalam penelitian pada studi ini berasal dari lagu yang terdapat pada Youtube.com. Video lagu pada situs tersebut akan dilakukan proses *convert* menjadi *file* musik dengan format .wav. *File* musik .wav inilah yang akan digunakan sebagai data latih. Namun *file* .wav tidak disimpan pada *database*, melainkan hanya fitur yang sesuai dengan masing-masing modul. Data latih merupakan lagu yang dipotong 1 menit sedangkan untuk data uji menggunakan lagu yang dipotong secara acak sepanjang 30 detik.

Tabel 1.
Hasil pada Modul *Fingerprint*

Judul Lagu	Hasil
Careless Whisper	x
Everybody Knew	v
Faded	v
Get Low	v
Its My Life	v
Livin on A Prayer	x
Middle	v
Saxobeats	v
Thing Will Get Better	v
Tremor	v
Turn Up the Speakers	v

Keterangan:
x: Potongan lagu yang tidak terdeteksi.
v: Potongan lagu yang berhasil dideteksi.

C. Hasil Percobaan

Pada percobaan kali ini, *testing* yang digunakan adalah 11 sampel untuk modul *fingerprint* dan dilakukan perancangan beserta pengujian sesuai dengan perancangan yang telah dibahas pada subbab sebelumnya. Akurasi dihitung menggunakan (2) dengan rincian sebagai berikut:

$$ACC = \frac{TRUE}{TOTAL\ DATA} \times 100\% \quad (2)$$

Tabel 1. adalah hasil uji coba pada modul. Dengan menggunakan Persamaan 2, maka akurasi yang didapatkan untuk modul *fingerprint* 81,81%.

IV. KESIMPULAN

Fingerprint dapat didapatkan dengan fitur *Audio Signature Type* pada fitur MPEG-7. Klasifikasi *Fingerprint* dapat menggunakan KNN, namun harus dimodifikasi. Hal tersebut sangatlah penting, sebab terjadi perbedaan dimensi antara sinyal untuk data uji dan sinyal untuk data latih. *Recommendation* lagu dapat dicapai hanya dengan melihat modul *fingerprint*. Sebab pada modul *fingerprint* sudah menghitung kemiripan dari suatu lagu. Rekomendasi lagu dapat diberikan ketika lagu tersebut mirip dengan lagu yang lain. Akurasi untuk modul *recommendation* akan mengacu pada *fingerprint* karena setiap pengguna berhak menolak maupun menerima rekomendasi dari lagu yang telah diberikan oleh sistem. Akurasi pada modul *fingerprint* adalah 81,81%.

DAFTAR PUSTAKA

[1] S. D. You, W.-H. Chen, and W.-K. Chen, "Music Identification System Using MPEG-7 Audio Signature Descriptors," *Sci. World J.*, vol. 2013, pp. 1–11, Mar. 2013.

[2] H.-G. Kim, N. Moreau, and T. Sikora, "MPEG-7 Audio and Beyond Audio Content Indexing and Retrieval," John Wiley & Sons Ltd, 2005, p. 217.

[3] H.-G. Kim, N. Moreau, and T. Sikora, "MPEG-7 Audio and Beyond Audio Content Indexing and Retrieval," John Wiley & Sons Ltd, 2005, pp. 13–58.

[4] "XQuery Tutorial." [Online]. Available: http://www.w3schools.com/xml/xquery_intro.asp. [Accessed: 18-Dec-2016].

[5] "why_mir." [Online]. Available: http://musicinformationretrieval.com/why_mir.html. [Accessed: 21-Dec-2016].

[6] "Play Framework - Build Modern & Scalable Web Apps with Java and Scala." [Online]. Available: https://www.playframework.com/. [Accessed: 31-Dec-2016].
[7] "Welcome | Flask (A Python Microframework)." [Online]. Available: http://flask.pocoo.org/. [Accessed: 21-Dec-2016].
[8] M. M. Deza and E. Deza, *Encyclopedia of Distances*. Springer Science & Business Media, 2009.
[9] M. N. Munawar, R. Sarno, D. A. Asfani, T. Igasaki, and B. T. Nugraha, "Significant preprocessing method in EEG-Based emotions classification," *J. Theor. Appl. Inf. Technol.*, vol. 87, no. 2, pp. 176–190, May 2016.
[10] R. Sarno, B. T. Nugraha, M. N. Munawar, R. Sarno, B. T. Nugraha, and M. N. Munawar, "Real Time Fatigue-Driver Detection from Electroencephalography Using Emotiv EPOC+," *Int. Rev. Comput. Softw. IRECOS*, vol. 11, no. 3, pp. 214–223, Mar. 2016.