TUGAS AKHIR – K141502

Implementasi MIR (Music Information Retrieval) pada Modul Genre Recognition dan Deep Learning Classification untuk Aplikasi Musicmoo

Muhammad Nezar mahardika
NRP 5114100001

Dosen Pembimbing
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dwi Sunaryono, S.Kom, M.Kom.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – K141502

Implementasi MIR (Music Information Retrieval) pada Modul Genre Recognition dan Deep Learning Classification untuk Aplikasi Musicmoo

Muhammad Nezar Mahardika
NRP 5114100001

Dosen Pembimbing
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dwi Sunaryono, S.Kom, M.Kom.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – K141502

IMPLEMENTATION OF MIR (MUSIC INFORMATION RETRIEVAL) METHOD GENRE RECOGNITION AND DEEP LEARNING CLASSIFICATION ON MUSICMOO APPS

Muhammad Nezar Mahardika
NRP 5114100001

Supervisor
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dwi Sunaryono, S.Kom, M.Kom.

Department of Informatics
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

IMPLEMENTASI MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL GENRE RECOGNITION DAN DEEP LEARNING CLASSIFICATION UNTUK APLIKASI MUSICMOO

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Manajemen Informasi
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
MUHAMMAD NEZAR MAHARDIKA
NRP: 5114 100 001

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
NIP: 19590803 198601 1 001 ..... (pembimbing 1)

Dwi Sunaryono, S.Kom., M.Kom.
NIP: 19720528 199702 1 001 ..... (pembimbing 2)

SURABAYA
JANUARI 2018

[Halaman ini sengaja dikosongkan]

IMPLEMENTASI MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL GENRE RECOGNITION DAN DEEP LEARNING CLASSIFICATION UNTUK APLIKASI MUSICMOO

Nama Mahasiswa	: Muhammad Nezar Mahardika
NRP	: 5114 100 001
Jurusan	: Teknik Informatika FTIf-ITS
Dosen Pembimbing 1	: Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Dosen Pembimbing 2	: Dwi Sunaryono, S.Kom., M.Kom.

ABSTRAKSI

Saat ini, perkembangan dunia musik mengalami peningkatan. Peningkatan ini terjadi di dalam sisi industri musik maupun dalam sisi perkembangan *genre* musik itu sendiri. Dalam perjalanannya, *genre* musik mengalami perkembangan dengan terbentuknya bagian-bagian dalam sebuah *genre*, yaitu *subgenre.* Selain itu, perkembangan di industri musik juga membuat semakin banyaknya lagu-lagu baru bermunculan. Seringkali kita tidak mengetahui judul lagu-lagu tersebut.

Metode pemanggilan kembali informasi suatu musik atau yang sering disebut *Music Information Retrieval* (MIR) bisa diterapkan untuk pendeteksian lagu dan prediksi *subgenre* pada suatu lagu. Langkah pertama yang dilakukan adalah melakukan ekstraksi fitur audio berbasis MPEG-7. Selanjutnya adalah menerapkan algoritma *Bhattacharyya Distance* untuk pendeteksian lagu dan *Deep Neural Network* (DNN) untuk klasifikasi suatu *subgenre.* Hasil akhirnya adalah dapat dilakukannya pendeteksian dan prediksi *subgenre* suatu lagu.

Kata Kunci: MIR, MPEG-7, *subgenre, DNN, Bhattacharyya*.

[Halaman ini sengaja dikosongkan]

IMPLEMENTATION OF MIR (MUSIC INFORMATION RETRIEVAL) METHOD GENRE RECOGNITION AND DEEP LEARNING CLASSIFICATION ON MUSICMOO APPS

Student Name	:	Muhammad Nezar Mahardika
Student ID	:	5114 100 001
Major	:	Informatics Department FTIf-ITS
Advisor 1	:	Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.
Advisor 2	:	Dwi Sunaryono, S.Kom, M.Kom.

ABSTRACT

Currently, the development of the music world has increased. This increase occurs within the music industry as well as in the developmental side of the music genre itself. In its journey, the music genre has developed with the formation of parts in a genre, the subgenre. In addition, developments in the music industry also made more and more new songs emerging. Often we do not know the title of the songs.

The method of calling back information of a music or often called Music Information Retrieval (MIR) can be applied to the detection of songs and subgenre predictions on a song. The first step is to extract the MPEG-7 based audio features. the second step is to apply Bhattacharyya Distance algorithm for song detection and Deep Neural Network (DNN) for the classification of a subgenre. The result is the detection and prediction of a subgenre of a song.

Keywords: MIR, MPEG-7, *subgenre, DNN, Bhattacharyya*.

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Puji syukur kepada Allah SWT atas rahmat dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul:

IMPLEMENTASI MIR (MUSIC INFORMATION RETRIEVAL) PADA MODUL GENRE RECOGNITION DAN DEEP LEARNING CLASSIFICATION UNTUK APLIKASI MUSICMOO

Melalui lembar ini, penulis ingin menyampaikan ucapan terimakasih dan penghormatan yang sebesar-besarnya kepada:

1. Sri Suntani, S.pd., selaku Ibu saya tercinta yang merupakan sumber motivasi terbesar bagi saya dalam mengerjakan tugas akhir ini. Sekaligus sumber bantuan moril dan materil terbesar untuk saya selama 4 tahun kebelakang.
2. Ahmadiah Reza Lukman, A.Md., selaku kakak saya yang selalu memberikan dukungan pada saya baik moril maupun materil.
3. Prof. Drs. Ec. Ir. Riyanarto Sarno, M.sc., Ph.D., selaku dosen pembimbing 1 yang telah banyak membantu, membimbing, bahkan memotivasi penulis untuk menyelesaikan Tugas Akhir ini.
4. Bapak Dwi Sunaryo, S.Kom, M.Kom., selaku dosen pembimbing 2 yang telah banyak memberikan semangat, motivasi, serta arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.
5. Johanes Andre dan Faris Ponighzwa, selaku kakak tingkat yang membantu dalam pengerjaan Tugas Akhir ini.

6. Irfan, Dimmy, Imam, Abror, Dewi, Ela, dan kawan Es Cincau lainnya yang mendukung saya dari awal masa-masa kuliah saya. Terlebih untuk Irfan yang menemani saya tidur dalam 3,5 tahun terakhir.
7. Aldo, Glleen, Sani, Faris, Ambon, Lian, Buyung, Paul, Oing, serta kawan WarkopXJojoran lainnya yang terlalu banyak jika saya sebutkan satu per satu yang sudah membantu saya sejak masa awal perkuliahan.
8. Kawan-kawan administrator Laboratorium Manajemen Informasi Jurusan Teknik Informatika ITS yang telah menyediakan fasilitas selama masa pengerjaan tugas akhir ini.
9. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS lainnya yang telah banyak menyampaikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
10. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu per satu.

Penulis menyadari masih terdapat banyak kekurangan baik dalam pelaksanaan Tugas Akhir maupun penyusunan buku laporan ini, namun penulis berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi. Penulis mengharapkan kritik dan saran yang membangun untuk kesempurnaan penulisan buku Tugas Akhir ini.

Surabaya, Desember 2017

Muhammad Nezar Mahardika

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

DAFTAR PERSAMAAN

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1. Latar Belakang

Perkembangan musik akhir-akhir ini sangatlah pesat. Perkembangan yang terjadi pada sisi produksi musik maupun perkembangan jenis musik itu sendiri, atau yang biasa ktia sebut dengan genre. Pada sisi produksi musik pada saat ini para musisi makin produkif menghasilkan lagu-lagu baru. Sedangkan dalam sisi genre musik, genre musik mulai membentuk bagian-bagian dari suatu genre, atau disebut dengan *subgenre*.

Sebagai hasil dari ledakan terbaru dalam media tersebut, muncul juga suatu kebutuhan pokok untuk kalangan masyarakat agar bisa mengetahui informasi lagu yang lebih lengkap pada suatu lagu. Menjawab kebutuhan masyarakat tersebut, ditemukanlah metode *Music Information Retrieval* atau yang disingkat menjadi MIR. MIR adalah metode pemanggilan informasi suatu musik agar dapat memberikan informasi lagu yang kompleks[1].

Pada kali ini, Tugas Akhir ini akan menjawab implementasi kebutuhan masyarakat mengenai MIR. Penelitian yang dilakukan terkait dengan analisis dan implementasi MIR pada modul *song* dan, *subgenre recognition*. Tujuan dari Tugas Akhir ini adalah memperkaya detail informasi suatu lagu dengan memberikan informasi mengenai *subgenre* dari sebuah lagu.

Konsep yang digunakan pada Tugas Akhir ini adalah menggunakan ekstraksi fitur berbasis MPEG-7 yang sudah menjadi standar dalam konten multimedia berdasarkan ISO/IEC 15938 [2]. Hasil ekstraksi fitur ini berupa metadata dalam format XML. Untuk mengimplementasikan ekstraksi fitur berbasis MPEG-7 ini menggunakan *library* Java bernama

MPEG7AudioEnc yang bersifat *open source*. Di dalam file metadata XML tersebut berisi terdapat fitur-fitur dalam bentuk angka-angka digital yang merepresentasikan karakteristik suatu sinyal. Setelah mendapatkan fitur ini, tahap berikutnya adalah melakukan pemilihan fitur yang dipakai dan diambil menggunakan *Xquery*. Fitur yang dipilih ini nantinya digunakan untuk pemrosesan sesuai dengan masing-masing modul. Pemrosesan yang dimaksud adalah melakukan *Discrete Wavelet Transform* (DWT) beserta level dekomposisi terbaik menggunakan *library pywt*. Pemrosesan selanjutnya adalah pencarian jarak terpendek dari 2 buah potongan lagu menggunakan perhitungan *Bhattacharya Distance* untuk *song recognition*. Kemudian melakukan penggabungan fitur pada suatu list beserta penyamaan panjang fitur untuk proses klasifikasi. Tahap terakhir untuk adalah melakukan klasifikasi dengan menggunakan *Deep Neural Network* (DNN). Terdiri dari 2 tahap yaitu tahap *training* dan predisi. Tahap *training* adalah melakukan pembelajaran karakteristik sinyal sesuai dengan label yang dimaksud, sedangkan tahap prediksi adalah memprediksi data baru yang belum diketahui dan akan memberikan suatu penilaian sesuai dengan tahap *training.* Hasilnya adalah detail informasi *subgenre* pada suatu lagu berdasarkan karakteristik sinyal.

1.2. Tujuan

Tujuan dari pembuatan tugas akhir ini adalah:

1. Mengetahui fitur-fitur lagu apakah yang dimiliki suatu lagu berbasis MPEG-7.
2. Mengetahui fitur lagu apakah yang mempengaruhi dalam *song recognition*.
3. Mengetahui fitur lagu apakah yang mempengaruhi dalam menentukan *subgenre* sebuah lagu.

1.3. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini antara lain:

1. Bagaimana fitur musik yang dihasilkan lewat ekstraksi fitur audio berbasis MPEG -7?
2. Bagaimana cara mengidentifikasi suatu musik dari *fingerprint* ?
3. Bagaimana cara identifikasi *subgenre* pada suatu musik?
4. Bagaimana hasil implementasi MIR terhadap *song recognition* ?
5. Bagaimana hasil implementasi MIR dan *Deep Learning classification* terhadap klasifikasi *subgenre ?*

1.4. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, antara lain:

1. Hasil dari tugas akhir ini adalah sebuah aplikasi MusicMoo dengan modul *Song*, dan *Genre Recognition.*
2. Audio yang digunakan berekstensi .wav.
3. Identifikasi lagu hanya berupa judul dari lagu tersebut.
4. *Genre* yang digunakan adalah dangdut, dengan *subgenre* dangdut klasik, dangdut koplo, dan dangdut remix.
5. Pada saat pengujian, teknik perekaman suatu lagu bisa berubah-ubah karena dipengaruhi oleh *speaker* sumber suara maupun sumber perekam.
6. Pengujian untuk penentuan *subgenre* sebuah lagu, harus melakukan perekaman selama minimal 45 detik.

1.5. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

a. Penyusunan proposal tugas akhir

Proposal tugas akhir ini berisi latar belakang pembuatan tugas akhir, rumusan masalah, batasan masalah, tujuan pembuatan, manfaat, metodologi hingga jadwal kegiatan pembuatan tugas akhir. Selain itu proposal tugas akhir ini memberikan ringkasan dari tugas akhir. Proposal tugas akhir juga berisi tinjauan pustaka yang digunakan sebagai referensi pembuatan tugas akhir ini.

b. Studi literatur

Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan program yaitu mengenai MPEG-7, MPEG-7 *Low Level Audio Descriptors*, *Xquery*, *Discrete Wavelet Transform* (DWT), dan klasifikasi *deep learning*.

c. Analisis dan desain perangkat lunak

Tahap ini meliputi perancangan sistem berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini. Pada tahapan ini dibuat *prototype* sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Serta dilakukan desain suatu sistem dan desain proses-proses yang ada.

d. Implementasi perangkat lunak

Implementasi perangkat lunak ini dibangun dengan sistem perangkat bergerak yang berbasis bahasa pemrograman *Java* (Android Studio) dan *database* menggunakan MySQL. Sedangkan untuk *pre-processing* dan *processing* menggunakan bahasa Python.

e. Pengujian dan evaluasi

Pada tahapan ini dilakukan uji coba terhadap perangkat lunak yang telah dibuat. Pengujian yang dimaksud adalah pengujian fungsionalitas aplikasi yang dibangun. Pengujian dan evaluasi akan dilakukan dengan melihat kesesuaian dengan perencanaan. Tahap ini dimaksudkan juga untuk mengevaluasi jalannya aplikasi, mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

f. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
    a. Latar Belakang
    b. Rumusan Masalah
    c. Batasan Tugas Akhir
    d. Tujuan
    e. Metodologi
    f. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Pengujian dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka

1.6. Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini. Teori yang terkait adalah ekstraksi fitur berbasis MPEG-7, *Discrete Wavelet Transform* (DWT), *subgenre*, *sliding algorithm*, *Bhattacharyya Distance*, dan *deep learning*.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai perancangan perangkat lunak. Perancangan perangkat lunak meliputi perancangan data, arsitektur, proses dan perancangan antarmuka aplikasi.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan dan implementasi fitur-fitur penunjang aplikasi.

Bab V Pengujian dan Evaluasi

Membahas tentang lingkungan pengujian, skenario pengujian, dan evaluasi pengujian setelah aplikasi selesai dikembangkan.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

Lampiran

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

[Halaman ini sengaja dikosongkan]

BAB II
DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir.

2.1. *Music Information Retrieval*

Music Information Retrieval (MIR) sesuai dengan definisi Downie adalah usaha penelitian yang berusaha untuk mengembangkan inovatif musik berbasis konten, mencari skema, *interface* baru untuk membuat toko besar di dunia musik [1]. MIR bisa diartikan sebagai pemanggilan informasi suatu musik agar dapat memberikan informasi lagu yang kompleks.

Sudah banyak penelitian terkait mengenai audio yang mengimplementasikan MIR. Misalnya adalah *query by humming* yaitu memanggil lagu dengan bergumam [3], *cover song identification* yaitu mendeteksi lagu yang dinyanyikan kembali [4], *pathological voice detection* yaitu deteksi suara patologis [5], dan lain-lain. Berbagai macam metode ekstraksi fitur pada audio pun beraneka ragam. Namun pada pengerjaan Tugas Akhir ini, metode yang digunakan adalah ekstrasi fitur dengan basis MPEG-7 dan berfokus pada *song recognition* dan klasifikasi *subgenre* suatu lagu yang akan dijelaskan di Bab III.

2.2. MusicMoo

MusicMoo adalah aplikasi yang memungkinkan pengguna untuk mengidentifikasi judul lagu, tempo lagu, serta genre lagu dengan mendengarkan potongan lagu. Antarmuka aplikasi MusicMoo bisa dilihat pada Gambar 2.1.

Gambar 2.1 Aplikasi MusicMoo

Pada Gambar 2.1 bisa dilihat ada 3 menu dalam aplikasi MusicMoo. Menu pertama adalah *signature*, pada menu ini pengguna bisa mengidentifikasi judul lagu pada sebuah audio. Menu yang kedua adalah *cover song*, pada menu ini memungkinkan pengguna untuk mengetahui judul sebuah lagu walaupun lagu tersebut tidak dinyanyikan oleh penyanyi aslinya. Menu yang terakhir adalah detail lagu, dimana pada menu ini pengguna bisa mengetahui informasi tentang tempo, genre, dan mood dari suatu audio.

2.3. *WAV*

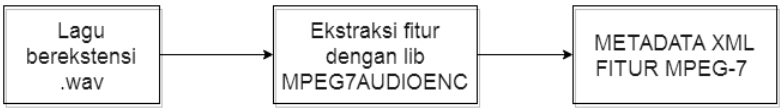
WAV adalah singkatan dari istilah dalam bahasa Inggris *Waveform Audio Format* merupakan standar format berkas audio yang dikembangkan oleh Microsoft dan IBM. Format WAV banyak digunakan oleh *handphone*, sehingga popularitas bisa menyamai file MP3 [24].

Di dalam tugas akhir ini *input* untuk aplikasi MusicMoo adalah sebuah audio. Audio dalam bentuk *file* ini harus berekstensi .wav.

2.4. MPEG-7

MPEG-7 adalah deskripsi standar konten multimedia dalam ISO/IEC 15938 [6]. Dari deskripsi yang terkait, konten tersebut dapat memungkinkan pencarian cepat dan effisien.

Ekstraksi fitur yang dilakukan ini menggunakan *library* Java yang bernama *MPEG7AudioEnc* yang bisa didapatkan pada *sourceforge*. *Input*nya adalah sebuah lagu bereksensi wav, dilakukan ekstraksi fitur. Hasil ekstraksi fitur pada MPEG-7 ini berupa metadata dengan format XML. Gambar 2.2 adalah tahapan untuk melakukan ekstraksi fitur berbasis MPEG-7.



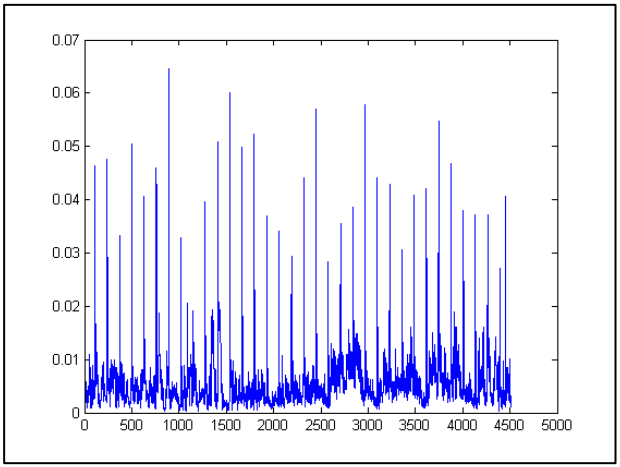
Gambar 2.2 Tahapan Ekstraksi Fitur MPEG-7

Hasil dari ekstraksi audio berbasis MPEG-7 ini adalah sebuah metadata XML yang berisi fitur-fitur berupa karakteristik sinyal. Sinyal yang dihasilkan berupa nilai digital yang panjang. Gambar 2.3 adalah contoh isi data dari XML yang dihasilkan oleh ekstraksi MPEG-7 yang merupakan contoh isi data pada fitur *Audio Power*. Data sinyal berupa angka-angka digital dalam

```
</Raw></SeriesOfScalar></UpperLimitOfHarmonicity></AudioDescriptor><AudioDescriptor xsi:type="AudioPowerType"><SeriesOfScalar hopSize="PT10N1000F"
totalNumOfSamples="4508"><Raw>0.0 0.0 6.757896E-11 4.5815136E-6 3.2904052E-5 3.4488403E-5 4.4055378E-5 3.1270883E-5 2.878371E-5 2.8276318E-5 3.7766105E-5
2.7014117E-5 3.9766284E-5 1.9448198E-5 1.9136802E-5 2.8626175E-5 2.011027E-5 2.2653594E-5 2.3187336E-5 1.2699299E-5 1.2039114E-5 4.4056733E-6 4.633146E-6
6.406585E-6 5.7510647E-6 9.815552E-6 3.875755E-6 8.5812735E-6 8.953873E-6 6.0477428E-6 3.953473E-6 7.506895E-6 3.8665407E-6 3.5911123E-6 5.778706E-5
0.0021328935 0.0069570723 0.007049585 0.0049764947 0.004210572 0.004178834 0.0063359877 0.0057332693 0.006155226 0.005608983 0.0062818555 0.0067735547
0.005084108 0.0034875455 0.0029903075 0.003428247 0.0045787916 0.0052722786 0.0051788352 0.0059483293 0.006681054 0.0067263194 0.0074646464 0.007397632
0.006502689 0.0057625044 0.0050914213 0.0050003855 0.0050545046 0.004969644 0.0048284847 0.0049023163 0.0048581953 0.004863815 0.004941797 0.004691671
0.004291982 0.0040212725 0.0036185256 0.0033670217 0.0031339915 0.0029430178 0.0027565483 0.0025371588 0.0023264084 0.0022042254 0.0019952073 0.0017434198
0.0015811616 0.0013872512 0.0012030818 0.0010563035 9.401313E-4 7.6175696E-4 6.170252E-4 5.196928E-4 4.6939E-4 3.9954833E-4 3.6172E-4 2.8449888E-4
2.4646585E-4 4.0583857E-4 0.012295584 0.07410384 0.08676415 0.13573341 0.14792123 0.11150838 0.08283244 0.06464404 0.043826148 0.024384199 0.013268818
0.009456868 0.00769835 0.007057071 0.006552733 0.0061454494 0.005219651 0.0047519 0.0052466546 0.004680657 0.0038680483 0.0032779046 0.0028865018
0.0022983756 0.0016499056 0.001051384 7.830468E-4 6.7472935E-4 5.615104E-4 6.525789E-4 7.332019E-4 7.239527E-4 7.35154E-4 5.01073E-4 3.6393775E-4
2.9672094E-4 2.5675393E-4 2.7132736E-4 0.006173165 0.017849037 0.014949953 0.021423021 0.009746517 0.0032799388 0.0031291589 8.2930323E-4 0.0010431454
9.0168265E-4 2.7827543E-4 3.432617E-4 2.606257E-4 7.186526E-5 9.2647555E-5 2.5626444E-4 1.9704555E-4 5.9139533E-5 1.5166146E-4 3.531694E-5 4.8983497E-5
8.3896826E-5 1.0435635E-4 2.262883E-4 0.0010240877 0.025180124 0.03927545 0.053139053 0.048805833 0.033853408 0.027622737 0.016899662 0.009504017 0.008066149
0.004858067 0.004493008 0.0038001598 0.0035502855 0.0034006871 0.0039526755 0.0031681082 0.00212173 0.0019057256 0.0013088818 0.001032002 0.0010910254
8.396533E-4 7.816088E-4 7.9826347E-4 6.024102E-4 4.7590592E-4 5.306261E-4 4.6406727E-4 4.934379E-4 4.3738922E-4 3.8011707E-4 3.1350303E-4 2.931304E-4
1.9978198E-4 2.1074855E-4 1.7610811E-4 4.2199838E-4 0.0077912044 0.017662242 0.012543642 0.017596325 0.005940225 0.0027218086 0.0021900716 5.661888E-4
```


Gambar 2.3 Contoh Isi Data Dari Metadata MPEG-7

jumlah yang sangat banyak. Ketika angka-angka ini di-*plot* maka terlihatlah fitur ini merupakan data dalam bentuk sinyal. Gambar 2.4 merupakan contoh *plotting* angka digital dari isi data pada Gambar 2.3.



Gambar 2.4 Contoh Plot Angka Digital Menjadi Sinyal

Ekstraksi fitur berbasis MPEG-7 ini mempunyai keunggulan yaitu:

1. Merupakan standar industri ISO/IEC 15938.

2. Data yang dihasilkan berupa metadata, sehingga lebih efisien.
3. Kaya akan fitur yang bisa digunakan untuk menggambarkan konten suatu multimedia. Sehingga fitur ini bisa digunakan untuk berbagai macam MIR.

2.5. MPEG-7 Low Level Audio Descriptors

MPEG-7 *Low Level Descriptors (*LLADs) adalah deskriptor yang disimpan menggambarkan variasi sifat frekuensi audio dari waktu ke waktu. Fungsinya adalah untuk mengidentifikasi identik suatu lagu apakah sama atau berbeda.

Pada metadata XML yang dihasilkan memiliki *syntax* seperti pada Gambar 2.5 di mana *name*=”*AudioLLDScalarType*” adalah nama fitur yang dihasilkan. Secara garis besar, kelas hirarki pada XML metadata yang dihasilkan oleh MPEG-7 adalah seperti pada Gambar 2.6 [7].

```
<!-- ################################################################################ -->
<!-- Definition of AudioLLDScalar datatype                                          -->
<!-- ################################################################################ -->
<complexType name="AudioLLDScalarType" abstract="true">
  <complexContent>
    <extension base="mpeg7:AudioDType">
      <choice>
        <element name="Scalar" type="float"/>
        <element name="SeriesOfScalar" minOccurs="1" maxOccurs="unbounded">
          <complexType>
            <complexContent>
              <extension base="mpeg7:SeriesOfScalarType">
                <attribute name="hopSize" type="mpeg7:mediaDurationType"
                           default="PT10N1000F"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </choice>
      <attribute name="confidence" type="mpeg7:zeroToOneType" use="optional"/>
    </extension>
  </complexContent>
</complexType>
```

Gambar 2.5 Syntax Metadata MPEG-7 LLADs

Gambar 2.6 Hirarki Kelas Pada MPEG-7 LLADs

Berikut adalah penjelasan setiap *Low Level Audio Descriptors* yang dihasilkan [6][2]. Pada Tugas Akhir yang lalu ada beberapa fitur yang digunakan, yaitu *Audio Power* dan *Audio Harmonicity* untuk klasifikasi tempo dan mood [28]. Selain itu fitur *Audio Spectrum projection* dan *Audio Spectrum Flatness* untuk *cover song recognition* [29]. Namun pada Tugas Akhir ini yang digunakan hanyalah *Audio Signature Type*, *Audio Specrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*. Untuk penjelasan akan dipaparkan pada Bab III.

2.5.1. *Audio Power*

Audio Power (AP) adalah fitur yang menggambarkan temporal daya sesaat dari sinyal audio. Koefisiennya dari kuadrat rata-rata nilai gelombang *s(n)* dalam *non-overlapping frame*. Tujuannya adalah untuk membandingkan suatu label sinyal. Persamaan 2.1 adalah cara mendapatkan nilai AP.

$$AP(l) = \frac{1}{N_{hop}} \sum_{n=0}^{N_{hop}-1} \left| s(n + lN_{hop}) \right|^2 \ (0 \ \leq l \leq L-1) \tag{2.1}$$

di mana:

L = total jumlah frame waktu.
s(n) = rata-rata *square waveform.*
l = indeks frame.
N_{hop} = sampel antara *successive non-overlapping.*

2.5.2. *Audio Waveform*

Audio Waveform adalah daya temporal yang diberikan oleh nilai *hopSize*, yang terdiri dalam menggambarkan setiap frame garis vertikal dari *min range* ke *max range*. Tujuannya untuk menampilkan ringkasan dari file audio. Dengan demikian, fitur ini dapat digunakan untuk perbandingan kecepatan gelombang.

2.5.3. Temporal Centroid

Temporal Centroid (TC) mendefinisikan sebagai rata-rata sampling sinyal *envelope.* Persamaan 2.2 adalah cara mendapatkan nilai TC.

$$TC = \frac{N_{hop}}{F_s} \frac{\sum_{l=0}^{L-1}(lEnv(l))}{\sum_{l=0}^{L-1} Env(l)} \tag{2.2}$$

di mana: *Env*(*l*) diperoleh dari Persamaan 2.3.

$$Env(l) = \sqrt{\frac{1}{N_w}\sum_{n=0}^{N_{w-1}} s^2\left(lN_{hop}+n\right)} (0 \le l \le L-1) \tag{2.3}$$

Keterangan:
Env(l) = sinyal envelope.
N_{hop} = sampel antara *successive non-overlapping*.
N_w = panjang dari *frame* dalam beberapa waktu.
n = indeks waktu.
s = spektrum yang diekstak dari *frame* ke-*l.*
L = jumlah total dari *frame.*

2.5.4. *Log Attack Time*

Log Attack Time (LAT) adalah waktu yang dibutuhkan untuk mencapai amplitudo maksimal dari waktu batas minimum. Tujuannya adalah sebagai deskripsi onset sampel suara tunggal dari alat musik yang berbeda. Nilainya didefinisikan sebagai log(basis desimal) dari durasi waktu T_{Start} ketika sinyal mulai dan T_{Stop} ketika mencapi nilai maksimum. Persamaan 2.4 adalah cara mendapatkan nilai LAT.

$$LAT = \log_{10}(T_{stop} - T_{start}) \tag{2.4}$$

2.5.5. *Audio Spectrum Centroid*

Audio Spectrum Centroid (ASC) merepresentasikan sebagai karakteristik dari sebuah spektrum. Bisa juga menunjukkan pusat dari sebuah spektrum. Secara perseptual, ASC memiliki hubungan kuat antara kejernihan suara. Persamaan 2.5 adalah cara mendapatkan nilai ASC.

$$ASC = \frac{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} \log_2\left(\frac{f'(k')}{1000}\right) P'(k')}{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} P'(k')} \tag{2.5}$$

di mana tiap koefisien yang dibutuhkan didapatkan pada Persamaan 2.6, Persamaan 2.7, Persamaan 2.8, dan Persamaan 2.9.

$$K_{low} = floor(\frac{6.25}{\Delta F}) \tag{2.6}$$

$$P'(k') = \left\{ \sum_{k=0}^{K_{low}} P(k), \qquad for\ k' = 0 \right\} \tag{2.7}$$

$$P'(k') = \begin{cases} \sum_{k=0}^{K_{low}} P(k), & for\ k' = 0 \\ P(k' + K_{low}), & for\ 1 \le k' \le N_{FT} - K_{low} \end{cases} \tag{2.8}$$

$$f'(k') = \begin{cases} 32.25 & for\ k' = 0 \\ f(k' + K_{low}) & for\ 1 \le k' \le \frac{N_{FT}}{2} - K_{low} \end{cases} \tag{2.9}$$

keterangan:
N_{FT} = ukuran dari *Fast Fourier Transform.*

P(k) = *power* spektrum yang diekstrak pada *frame* ke-*l*.
ΔF = interval frekuensi dari dua sinyal FFT.
k = frekuensi indeks.
f'(k') = frekuensi yang sesuai dengan indeks *k'*.

2.5.6. *Audio Harmonicity*

Audio Harmonicity adalah fitur yang mendeskripsikan 2 properti sinyal harmonik dari spektrum.Properti pertama *harmonic ratio*, yaitu rasio daya harmonik dari total daya dan yang kedua *upper limit harmonicity* yaitu frekuensi spektrum yang tidak dapat dianggap harmonis. Tujuannya untuk membedakan suara harmonik (alat musik contohnya) dan suara non-harmonik (*noise*, pidato tidak jelas, dsb).

2.5.7. *Harmonic Spectral Centroid*

Harmonic Spectral Centroid (HSC) adalah rata-rata dari durasi sinyal amplitudo *weighted mean* dari puncak harmonik spektrum. Nilainya diperoleh dari rata-rata jumlah total *frame centroid*. HSC bisa juga didefinisikan sebagai ukuran ketajaman sinyal timbral. Persamaan 2.10 adalah cara mendapatkan nilai HSC.

$$HSC = \frac{1}{L}\sum_{l=0}^{L-1} LHSC, \tag{2.10}$$

di mana nilai LHSC diperoleh dari Persamaan 2.11.

$$LHSC_l = \frac{\sum_{h=1}^{N_h}(f_{h,l}A_{h,l})}{\sum_{h=1}^{N_h} A_{h,l}} \tag{2.11}$$

Keterangan:
$f_{h,l}$ = frekuensi.

$A_{h,l}$ = amplitudo puncak harmonik.
N_h = jumlah harmonik yang diperhitungkan
L = jumlah frame dalam segmen suara.
$LHSC_l$ = lokal *Harmonic Spectral Centroid*.
h = frekuensi dasar.

2.5.8. *Harmonic Spectral Deviation*

Harmonic Spectral Deviation (HSD) adalah fitur yang mengukur deviasi dari puncak harmonik dari envelope dari spektrum lokal. Persamaan 2.12 adalah cara mendapatkan nilai HSD.

$$HSD = \frac{1}{L}\sum_{l=0}^{L-1} LHSD_l \tag{2.12}$$

di mana nilai $LHSD_l$ diperoleh dari Persamaan 2.13.

$$LHSD_l = \frac{\sum_{h=1}^{N_h}\left|\log_{10}(A_{h,l}) - \log_{10}(SE_{h,l})\right|}{\sum_{h=1}^{N_h}\log_{10}(A_{h,l})} \tag{2.13}$$

$SE_{h,l}$ adalah perkiraan interpolasi amplitudo puncak harmonik yang berdekatan $A_{h,l}$ seperti yang ditulis pada Persamaan 2.14.

$$SE_{h,l} = \begin{cases} \frac{1}{2}\left(A_{h,l} + A_{h+1,l}\right) \; if \; h = 1 \\ \frac{1}{3}\left(A_{h-1,l} + A_{h,l} + A_{h+1,l}\right) \; if \; 2 \le h \le N_H - 1. \\ \frac{1}{2}\left(A_{h-1,l} + A_{h,l}\right) \; if \; h = N_H \end{cases} \tag{2.14}$$

Keterangan:
$A_{h,l}$ = amplitudo puncak harmonik.
N_H = jumlah harmonik yang diperhitungkan.
L = jumlah frame dalam segmen suara.
$LHSC_l$ = lokal *Harmonic Spectral Centroid*.
h = indeks dari komponen harmonis.

2.5.9. *Harmonic Spectral Spread*

Harmonic Spectral Spread (HSS) merepresentasikan ukuran spektrum rata-rata menyebar dalam kaitan dengan HSC. Didefinisikan juga sebagai penyimpangan kekuatan *weighted-RMS* dari HSCL HSCl lokal. Persamaan 2.15 adalah cara mendapatkan nilai HSS.

$$HSS = \frac{1}{L}\sum_{l=0}^{L-1} LHSS_l \tag{2.15}$$

di mana nilai LHSS diperoleh dari Persamaan 2.16.

$$LHSS_l = \frac{1}{LHSS_l}\sqrt{\frac{\sum_{h=1}^{N_H}\left[\left(f_{h,l} - LHSC_l\right)^2 A_{h,l}^2\right]}{\sum_{h=1}^{N_H} A_{h,l}^2}} \tag{2.16}$$

Keterangan:
L = jumlah frame dari segment suara.
$LHSS_l$ = cerminan modulasi getaran yang kurang jelas dari $LHSD_l$.
$f_{h,l}$ = frekuensi.
$A_{h,l}$ = amplitudo puncak harmonik.
N_H = jumlah harmonik yang diperhitungkan
L= jumlah frame dalam segmen suara.
$LHSC_l$ = dijelaskan pada Persamaan 2.11.

2.5.10. *Harmonic Spectral Variation*

Harmonic Spectral Variation (HSV) mencerminkan variasi spektral antara frame yang berdekatan. HSV didefinisikan juga sebagai pelengkap untuk korelasi normalisasi antara amplitudo puncak harmonik yang diambil dari dua frame yang berdekatan. Persamaan 2.17 adalah cara mendapatkan nilai HSV.

$$HSV = \frac{1}{L}\sum_{l=0}^{L-1} LHSV_l \tag{2.17}$$

di mana nilai $LHSV_l$ diperoleh dari Persamaan 2.18.

$$LHSV_l = 1 - \frac{\sum_{h=1}^{N_H}(A_{h,l-1}\, A_{h,l})}{\sqrt{\sum_{h=1}^{N_H} A_{h,l-1}^2}\ \sqrt{\sum_{h=1}^{N_H} A_{h,l}^2}} \tag{2.18}$$

Keterangan:
L = jumlah frame dari segment suara.
$A_{h,l}$ = amplitudo puncak harmonik.
N_H = jumlah harmonik yang diperhitungkan.
h = indeks dari komponen harmonis.

2.5.11. *Audio Spectrum Spread Type*

Audio Spectrum Spread (ASS) didefinisikan sebagai momen sentral kedua spektrum log-frekuensi. Fitur ini diekstraksi dengan mengambil *Root Mean Square* (RMS) dari penyimpangan spektrum dari *Audio Spectrum Centroid*. Persamaan 2.19 adalah cara mendapatkan nilai ASS.

$$ASS = \sqrt{\frac{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} \left[\log_2\left(\frac{f'(k')}{1000}\right) - ASC\right]^2 P'(k')}{\sum_{k'=0}^{\left(\frac{N_{FT}}{2}\right)-K_{low}} P'(k')}} \quad (2.19)$$

di mana:
N_{FT} = ukuran dari *Fast Fourrier Transform.*
P'(k') = *power* spektrum yang diekstrak pada *frame* ke-*l.*
k = frekuensi indeks.
ASC = dijelaskan pada Persamaan 2.5.
K_{low} = dijelaskan pada Persamaan 2.6.
f '(k') = dijelaskan pada Persamaan 2.9.

2.5.12. *Audio Spectrum Projection*

Audio Spectrum Projection (ASP) merupakan fitur yang didapat dari teknik normalisasi *Singular Value Decomposition* (SVD) dan Independent Component Analysis (ICA). Teknik ini diterapkan untuk proses klasifikasi, identifikasi alat musik, dan lain-lain.

2.5.13. *Audio Spectrum Basis Type*

Audio Spectrum Basis (ASB) didefinisikan sebagai representasi proyeksi spektrum dimensi rendah, sistem klasifikasi. Ekstrasi fitur ini menggunakan teknik normalisasi standar *Singular Value Decomposition* (SVD) dan *Independent Component Analysis* (ICA).

2.5.14. *Audio Spectrum Flatness Type*

Audio Spectrum Flatness (ASF) didefinisikan sebagai cerminan kerataan properti suatu kekuatan spektrum. Persamaan 2.20 adalah cara mendapatkan nilai ASF.

$$ASF = \frac{\sqrt[hiK'_b - loK'_b + 1]{\prod_{k'=loK'_b}^{hiK'_b} P_g(k')}}{\frac{1}{hiK'_b - loK'_b + 1} \sum_{k'=loK'_b}^{hiK'_b} P_g(k')} \qquad (1 \leq b \leq B).$$

(2.20)

di mana tiap koefisien yang dibutuhkan didapatkan pada Persamaan 2.21, Persamaan 2.22, Persamaan 2.23, dan Persamaan 2.24.

$$loEdge = 2^{\frac{1}{4}N} \times 1\ kHz$$

(2.21)

$$hiEdge = 2^{\frac{1}{4}B} \times loEdge$$

(2.22)

$$loF_b = 0.95 \times loEdge \times 2^{\frac{1}{4}(b-1)}$$

(2.23)

$$hiF_b = 1.05 \times loEdge \times 2^{\frac{1}{4}b}$$

(2.24)

Keterangan:
B = nilai penentu batas atas *band.*
b = frekuensi dari indeks *band.*
$P_g(k')$ = *power* spektrum yang diekstrak pada *frame* ke-*l.*
hiK_b = frekuensi *integer* batas dari hiF_b.
loK_b = frekuensi *integer* batas dari loF_b.
band = 1 kHz.

2.5.15. *Audio Spectrum Envelope Type*

Audio Spectrum Envelope (ASE) adalah Fitur yang mendeskripsikan jumlah dari koefisien *power band* b. ASC diperoleh dengan menjumlahkan *power* spektrum dalam serangkaian frekuensi *bands*. Persamaan 2.25 adalah cara mendapatkan nilai ASE.

$$ASE(b) = \sum_{k=loK_b}^{hiK_b} P(k) \quad (1 \leq b \leq B_{in}),$$

(2.25)

di mana tiap koefisien yang dibutuhkan didapatkan pada Persamaan 2.26, Persamaan 2.27, dan Persamaan 2.28.

$$r = 2^j octaves \ (-4 \leq j \leq +3)$$

(2.26)

$$loF_b = loEdge \times 2^{(b-1)r} \ (1 \leq b \leq B_{in})$$

(2.27)

$$hiF_b = loEdge\ 2^{br} (1 \leq b \leq B_{in})$$

(2.28)

Keterangan:
loEdge = 62.5 Hz.
$B_{in} = \frac{8}{r}$.
b = frekuensi dari indeks *band.*
k = frekuensi indeks.
hiK_b = frekuensi *integer* batas dari $hiF_.$
loK_b = frekuensi *integer* batas dari $loF_{b.}$

2.5.16. *Audio Fundamental Frequency*

Audio Fundamental Frequency (AFF) merupakan fitur yang memberikan perkiraan mendasar frekuensi segmen f_0 di mana sinyal diasumsikan periodik.

Ada parameter yang ditambahkan dalam standarisasi MPEG-7 ini, yaitu:

- *loLimits*: batas bawah dari rentang frekuensi di mana f_0 telah dicari.
- *hiLimits*: batas atas dari rentang frekuensi di mana f_0 telah dicari.

Ukuran kepercayaan pada kehadiran periodisitas di bagian sinyal nilainya antara 0 dan 1.

2.5.17. *Audio Signature*

Audio Signature (AS) adalah suatu fitur yang menjadi mencerminkan identik suatu lagu. AS bisa juga disebut sebagai *fingerprint* dari sebuah lagu. Tujuannya untuk *matching* suatu sinyal identik dengan lagu apa.

2.5.18. *Sound Model*

Sound Model adalah Fitur yang digunakan untuk menghitung suatu *state sound model* histrogram.

2.6. *Fast Fourier Transform*

Fast Fourier Transform (FFT) adalah metode untuk mengubah sinyal dari *time domain* menjadi *frequency domain* [8]. Tujuannya adalah untuk mencari suatu informasi penting, seperti nilai maksimal, rata-rata atau standar deviasi, dalam *frequency domain* untuk tahap analisis [9][10].

2.7. *Discrete Wavelet Transform*

Discrete Wavelet Transform (DWT) adalah metode wavelet yang digunakan untuk melakukan dekomposisi pada *wavelet* sampai level N [8]. Tujuannya adalah untuk mengurangi *noise* pada sinyal dan memperkuat informasi di dalam sinyal tersebut dengan mempertahankan keutuhan informasi data [11] [12]. Ada banyak metode wavelet, namun dalam pengerjaan Tugas Akhir ini, metode wavelet yang digunakan adalah bior 2.8 dengan mengambil nilai *approximation coefficients*.

Untuk menentukan level dekomposisi ini tidak boleh sembarangan karena ketika level dekomposisi tinggi belum tentu yang baik, sebaliknya malah merusak sinyal sehingga menghilangkan informasi yang terkandung sinyal asli. Maka perlu dilakukan pemilihan level dekomposisi wavelet yang terbaik [13].

Langkah pertama adalah melakukan perhitungan pada Persamaan 2.29 yang dilakukan pada Matlab.

$$[\max value, index\ max] = max\left(abs\left(FFT(S - mean(S))\right)\right) \tag{2.29}$$

Maka akan didapatkan hasil *max value* dan *max index.* Kedua hasil ini digunakan untuk mencari *Fh*. *Fh* adalah *Frequency Range* pada Tabel 2.1. Untuk mendapatkan *Fh*, lakukan perhitungan pada Persamaan 2.30.

$$Fh = index \max * Fs/L \tag{2.30}$$

Di mana *Fs* adalah frekuensi sampling yaitu 1024, dan L adalah panjang sinyal. Hasil dari *Fh* dapat dilihat dari Tabel 2.1 sesuai dengan *Frequency Range.* Aturan untuk menentukan tingkat dekomposisi pada Tabel 2.1 dapat dinyatakan oleh Persamaan 2.31 [8].

$$\frac{f_q}{2^N + 1} \leq f_{char} \leq \frac{f_q}{2^N} \tag{2.31}$$

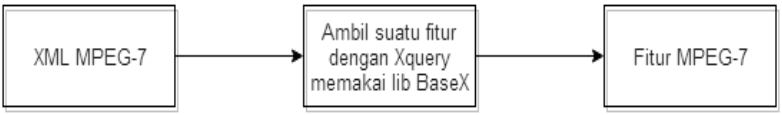
di mana f_q adalah *sampling frequency*, f_{char} adalah *dominant frequency*, dan *N* adalah level dekomposisi. Maka didapatlah level *decompose wavelet* terbaik.

Tabel 2.1 Penentuan Decompose Level Wavelet

***Decomposition Level* (L)**	***Frequency Range* (Hz)**
1	256-512
2	128-256
3	64-128
4	32-64
5	16-32
6	8-16
7	4-8
8	2-4
9	1-2
10	0.5-1
11	0.25-0.5
12	0.125-0.25
13	0.0625-0.125

2.8. *Xquery*

Xquery adalah bahasa untuk meng-*query* atau pemanggilan data di dalam suatu *database* dalam bentuk file XML [14]. Pada Tugas Akhir ini, *Xquery* digunakan untuk mengambil suatu fitur pada XML yang dihasilkan lewat ekstraksi fitur MPEG-7. Untuk mengimplementasikan *Xquery* menggunakan *library* Java yang bernama *BaseX* [15]. Gambar 2.7 adalah tahapan proses yang dilakukan menggunakan *Xquery*.



Gambar 2.7 Tahapan Proses Xquery

2.9. MIR (Music Information Retrieval)

MIR merupakan suatu istilah yang melakukan pengolahan sinyal dari suatu music untuk tingkat yang lebih lanjut. Dari MIR dapat dilakukan identifikasi *fingerprint*, *genre identification*, *cover identification*, dan sebagainya [16].

2.10. *Genre*

Genre musik adalah pengelompokan musik sesuai dengan kemiripan satu sama lain. Sebuah *genre* dapat juga didefinisikan oleh teknik musik, gaya, dan konteks musik. Dalam perkembangannya, di dunia music ada tingkatan yang lebih dalam dari *genre*, yaitu *subgenre*.

Untuk tugas akhir ini *genre* yang digunakan adalah dangdut. Musik dangdut digunakan di Tugas Akhir ini karena saat ini musik dangdut sedang naik daun kembali dan *subgenre* dari dangdut lebih jelas pembagiannya. Selain itu untuk *dataset* lagu dangdut sekarang sudah mudah didapatkan. Di dalam musik dangdut saat ini terdapat banyak macam variasi *subgenre* [17]. Dikarenakan sebuah *subgenre* pada *genre* dangdut terlalu banyak, maka pada percobaan ini dibatasi hanya sebatas *subgenre* dangdut, yaitu dangdut koplo, dangdut klasik, dan dangdut *house* / *remix* yang diperoleh dari *Youtube* dan sumber lain di internet.

Subgenre dangdut klasik adalah musik dangdut yang memiliki ciri nada yang cenderung lamban, mendayu-dayu dan masih ada nuansa india atau melayu [30]. Contoh dari penyanyi yang ada di *subgenre* dangdut klasik adalah Rhoma Irama, Meggy Z, dan A. Rafiq. *Subgenre* dangdut *remix* adalah musik yang menggunakan alat musik elektronik, seperti drum elektrik, dan teknologi musik untuk produksinya [32]. Contoh dari penyanyi yang ada di *subgenre* dangdut remix ini adalah Zaskia Gotik, Cita Citata, dan Melinda. *Subgenre* dangdut koplo adalah musik dangdut yang memiliki ciri tempo yang cenderung cepat dan alat music kendang menjadi bagian utama dalam musik ini [31]. Contoh dari penyanyi yang ada di *subgenre* dangdut koplo ini

adalah Via Vallen, Nella Kharisma, atau Orkes Musik seperti OM Sera atau OM Palapa.

2.11. 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* [18].

$$\begin{bmatrix} 0.78 & 0.71 & 0.69 & \dots & \dots & 0.72 \\ 0.72 & 0.58 & 0.53 & \dots & \dots & 0.57 \\ 0.71 & 0.71 & 0.60 & \dots & \dots & 0.53 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0.66 & 0.70 & 0.47 & \dots & \dots & 0.46 \end{bmatrix}_{n}$$

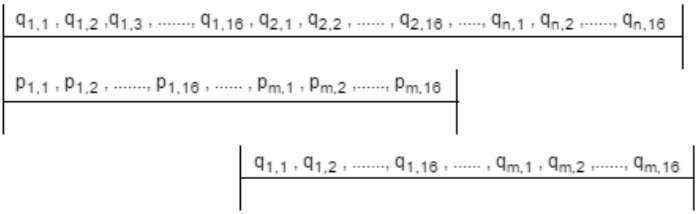
Gambar 2.8 Contoh Matriks Fitur Fingerprint

Gambar 2.8 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.

2.12. *Sliding Algorithm* dan *Bhattacharrya Distance*

Sliding Algorithm adalah 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 2.9 *Sliding Algorithm*

$$BC(p,q) = \sum_{x \in X} \sqrt{p(x)q(x)}$$

(2.32)

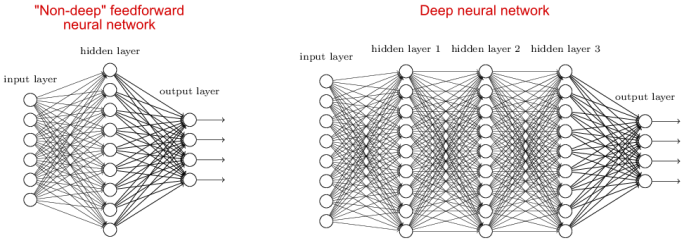
Gambar 2.9 menjelaskan bagaimana *Sliding Algorithm* bekerja [19]. *Sliding Algorithm* akan membandingkan setiap bagian dari matriks, hasil *distance* terkecil dari metode ini dianggap mewakili nilai *similarity* dari matriks tersebut. Pada Gambar 2.9 matriks q memiliki dimensi $n \times 16$ namun dijadikan 1 *array*, sedangkan pada matriks q memiliki dimensi $m \times 16$ dan juga dijadikan 1 *array* dengan keterangan tambahan, nilai n > m. Karena mememiliki perbedaan dimensi antara matriks q dan p maka *Sliding Algorithm* perlu diimplementasikan. Perhitungan distance pada *Sliding Algorithm* yang dipakai menggunakan Persamaan 2.32, yang merupakan rumus dari *Bhattacharrya Distance*[20]. *Bhattacharrya Distance* merupakan suatu algoritma yang menghitung kemiripan dengan cara menghitung selisih tiap fitur. Selisih tiap fitur tidak dihitung satu per satu, namun dengan mengambil rata-rata dari kelompok fitur lalu

dicari selisihnya. Pada penelitian ini fitur yang dilakukan perhitungan *Bhattacharrya Distance* adalah *subband* pada masing-masing sinyal.

2.13. *Deep Neural Network*

Deep Neural Network (DNN) adalah metode *machine learning* yang masuk kategori *deep learning*. DNN adalah NN dengan banyak *hidden layers* diantara input dan output. DNN tipikalnya adalah *feedforward network* yang saat ada aliran data dari input ke output tidak terdapat perulangan. Penggunaan jumlah *hidden layers dan* jumlah *neurons* fleksibel sesuai kebutuhan. Beda dari DNN dengan *neural network* biasa bisa dilihat pada Gambar 2.10.

Contoh pelatihan *training* data pada DNN menentukan parameter dari keputusan untuk mengklasifikasikan fungsi dari dua atau lebih kelas dan memaksimalkan penyaringan di setiap *layer*. Setelah tahap pembelajaran, mesin dapat memperkirakan pola yang tidak diketahui untuk diklasifikasikan.



Gambar 2.10 Perbedaan antara ANN dan DNN

Pada penelitian ini, dilakukan sesi *training* dan sesi *testing* untuk mencoba klasifikasi. Sesi *training* merupakan sesi melakukan pembelajaran terhadap variasi data-data yang dimiliki

oleh suatu label sedangkan sesi *testing* merupakan proses uji coba prediksi pada suatu data yang baru. Metode DNN dipilih karena percobaan sebelumnya ada yang menggunakan klasifkasi DNN [21]. Untuk melakukan implementasi DNN ini, digunakan *library sklflow* dari *tensorflow*. *Tensorflow* merupakan *open source library* yang dikembangkan oleh Google [25]. Dasar bahasa pemrograman dari *skflow* ini adalah bahasa pemrograman Python.

Pada DNN ini terdapat beberapa fungsi aktivasi yang bisa digunakan, seperti *relu*, *tanh*, *sigmoid*, *elu*, *softplus*, dan sebagainya. Pada Tugas Akhir ini fungsi aktivasi yang digunakan adalah *relu*.

$$f(x) = max(0, x) \tag{2.33}$$

Persamaan dari *relu* dapat dilihat pada Persamaan 2.33, dimana *x* merupakan masukan untuk *neuron*. Di dalam *neural network* terdapat sebuah masalah, yaitu hilangnya *gradient*. Fungsi aktivasi yang berdasar *gradient* adalah *sigmoid* dan *tanh*. Metode *gradient* ini memiliki pemahaman jika perubahan kecil pada parameter akan mempengaruhi keluaran daripada *neural network*. Jika perubahan pada parameter tidak begitu berpengaruh pada keluaran, maka *neural network* tidak akan mampu membaca parameter dengan efektif dan nilai dari *gradient* ini akan menjadi sangat kecil, dan bisa hilang. Fungsi aktivasi *sigmoid* dan *tanh* "memeras" masukan menjadi rentang keluaran yang sangat kecil. Akibatnya, masukan dengan jumlah yang besar akan dipetakan ke rentang yang sangat kecil. Dalam hal ini, perubahan besar terhadap masukan hanya akan berdampak kecil terhadap keluaran, dan nilai *gradient* menjadi kecil.

Hal ini akan menjadi lebih buruk jika menggunakan banyak *hidden layer*. Dimisalkan, lapisan pertama akan memetakan masukan yang besar ke keluaran yang lebih kecil, yang

akan dipetakan ke wilayah yang lebih kecil lagi oleh lapisan kedua, yang akan dipetakan ke wilayah yang lebih kecil lagi oleh lapisan ketiga dan seterusnya. Akibatnya, bahkan perubahan besar pada parameter lapisan pertama tidak banyak mengubah keluaran. Masalah ini bisa dihindari dengan menggunakan fungsi aktivasi yang tidak "memeras" masukan menjadi sesuatu yang lebih kecil, yaitu *relu*.

[Halaman ini sengaja dikosongkan]

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis dan perancangan sistem yang akan dibangun. Analisis membahas semua persiapan yang akan menjadi pokok pikiran pembuatan aplikasi ini. Mulai dari masalah yang melatarbelakangi, hingga analisis gambaran awal sistem yang akan dibuat. Perancangan sistem membahas hal-hal yang berkaitan dengan pondasi atau dasar pembuatan aplikasi, yang meliputi perancangan basis data, tampilan antar muka halaman aplikasi, hingga perancangan alur proses yang akan diimplementasikan di dalam aplikasi.

3.1. Analisis

Tahap analisis meliputi analisis masalah, analisis kebutuhan, deskripsi umum sistem, dan kasus penggunaan sistem yang dibuat.

3.1.1. Kontribusi Penelitian

Sebenarnya sudah ada penelitian-penelitian terkait mengenai MIR menggunakan MPEG-7 ini. Salah satunya adalah aplikasi *MusicMoo*. Namun untuk *song recognition* dan *subgenre recognition* dirasa adanya metode yang bisa membuat hasil dari aplikasi *MusicMoo* ini menjadi lebih baik.

Pada penelitian ini, MPEG-7 digunakan karena ekstraksi fitur yang dihasilkan berupa metadata. Di dalam metadata tersebut berisi angka-angka digital yang merupakan gambaran dari karakteristik-karakteristik sinyal suatu audio sesuai durasinya. Oleh karena itu, penelitian Tugas Akhir ini terkait dengan ilmu *Digital Signal Processing* (DSP).

3.1.2. Analisis Permasalahan

Di era *modern* ini, lagu dangdut berkembang dengan sangat pesat. Tidak hanya dalam hal jumlah produksi lagu-lagu

dangdut di pasaran, tetapi melainkan perkembangan musik dangdut itu sendiri. Musik dangdut mulai membentuk *sub-sub* dari music dangdut. Musik dangdut saat ini mulai bercampur dengan lagu-lagu rock, keroncong, bahkan sampai musik elektronik. Bahkan pada saat ini lagu-lagu dangdut mulai digemari kembali oleh masyarakat. Tetapi bukan lagu dangdut klasik yang seperti lagu dangdut pada tahun 1980-1990, melainkan lagu-lagu dangdut baru yang mulai laris di pasaran. Lagu-lagu dangdut yang laris saat ini dikenal dengan dangdut koplo dan dangdut *house* / *remix*. Setiap orang pasti mempunyai selera lagu dangdutnya sendiri-sendiri, namun tidak menutup kemungkinan juga jika seseorang yang sedang berjalan-jalan di suatu tempat umum seperti pasar lalu secara tidak sengaja mendengar lagu dangdut baru yang tidak pernah diketahuinya dan langsung menyukainya. Kejadian seperti ini sering dialami banyak orang. Biasanya, seseorang akan mencari lagu tersebut dengan cara mengetik lirik dan mencarinya dalam sebuah mesin pencarian. Namun bagaimana ketika sebuah lagu tersebut susah untuk dicari dalam mesin pencarian? Misalnya, bahasa yang susah (karena bahasa asing), lirik yang dinyanyikan terlalu cepat, dsb. Hal ini membuat seseorang gagal mendapatkan lagu yang baru diinginkan sehingga terkadang timbul perasaan kecewa.

Oleh karena itu, aplikasi MusicMoo ini dibuat untuk menangani permasalahan di atas. Ketika seseorang tertarik dengan lagu baru yang didengarnya, cukup membuka aplikasi perangkat bergerak MusicMoo dan merekam lagu setidaknya 45 detik. Maka aplikasi akan memberikan judul lagu yang dimaksud ataupun detail informasi lagu seperti judul, dan *subgenre*.

3.1.3. Analisis Kebutuhan

Kebutuhan utama dalam aplikasi ini difokuskan untuk mendeteksi judul dan *genre* suatu lagu. Contoh aplikasi yang sudah ada adalah MusicMoo. Dalam penelitian kali ini akan di uji cobakan metode baru untuk kedua modul tersebut. Oleh karena itu, aplikasi ini dirancang bertujuan untuk membuat hasil keluaran dari

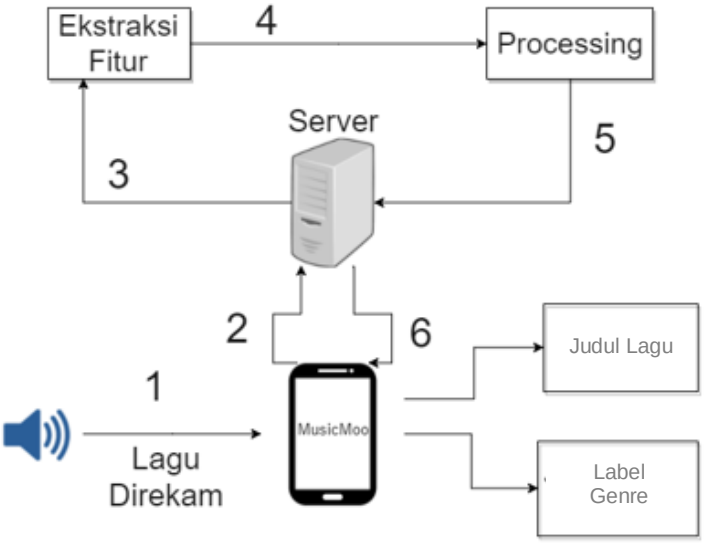
MusicMoo menjadi lebih baik. Tabel 3.1 merupakan daftar kebutuhan fungsional perangkat lunak yang dibangun sesuai dengan modul.

Tabel 3.1 Kebutuhan Fungsional Sistem

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-1	Melihat judul potongan lagu	Sistem dapat mengenali lagu dari *fingerprint* yang dimiliki dari potongan lagu
F-2	Mendeteksi *genre* lagu	Partisipan dapat mengenali sebuah lagu memiliki *genre* lagu apa.

3.1.4. Deskripsi Umum Sistem

Aplikasi yang akan dibuat pada Tugas Akhir ini adalah program aplikasi perangkat bergerak. Gambar 3.1 adalah gambaran alur jalan sistem.



Gambar 3.1 Deskripsi Umum Sistem

Keterangan pada penomoran Gambar 3.1:

1. Sebuah lagu direkam melalui aplikasi perangkat bergerak.
2. Lagu rekaman di-*upload* ke server.
3. Lagu rekaman dilakukan ekstraksi fitur dan pengambilan fitur oleh server Java.
4. Fitur yang diambil dilakukan *processing* pada server Python (melakukan DWT dan klasifikasi).
5. Hasil akan dikirim kembali pada server.
6. Server akan menampilkan hasil pada aplikasi perangkat bergerak.

Untuk penjelasan secara rinci akan dijelaskan di bawah mulai dari proses *input*, proses, hingga *output*.

3.1.4.1. *Input*

Pertama-tama, sebuah lagu direkam lewat aplikasi perangkat bergerak. Lakukan perekaman dengan menekan tombol rekam dengan durasi minimal 45 detik untuk *genre recognition* dan 25 detik untuk *song recognition*. Jika sudah, maka tekan tombol berhenti untuk mengakhiri rekaman. Maka hasil rekaman akan tersimpan dengan format .wav. Berikutnya adalah tekan tombol kirim. Maka lagu akan terunggah ke server dan siap diproses.

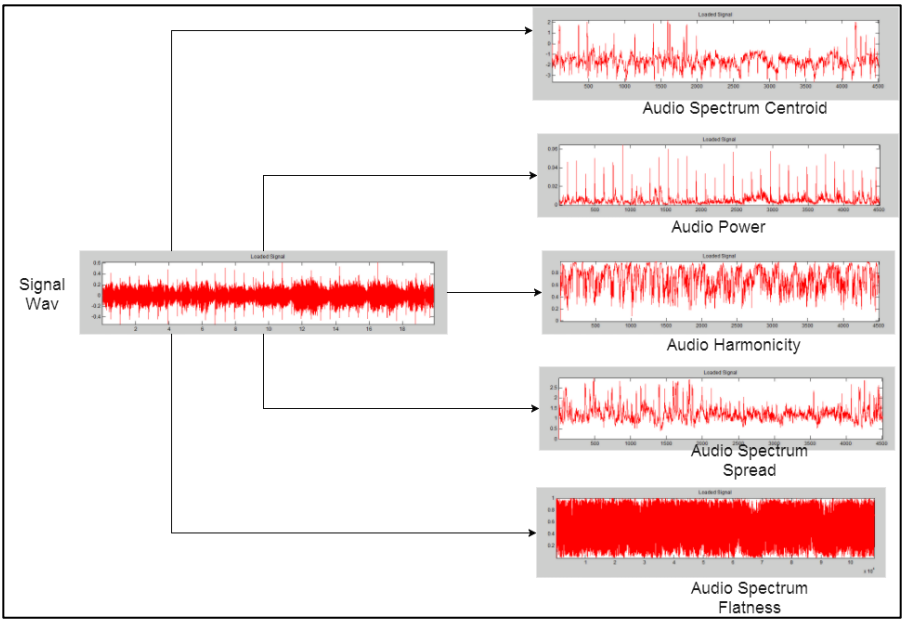
3.1.4.2. Proses

Pada tahap ini dijelaskan secara bertahap langkah-langkah yang dilakukan pada sistem dimulai dari proses ekstraksi fitur, pengolahan sinyal fitur, sampai menjadi hasil *output*.

3.1.4.2.1 Ekstraksi Fitur

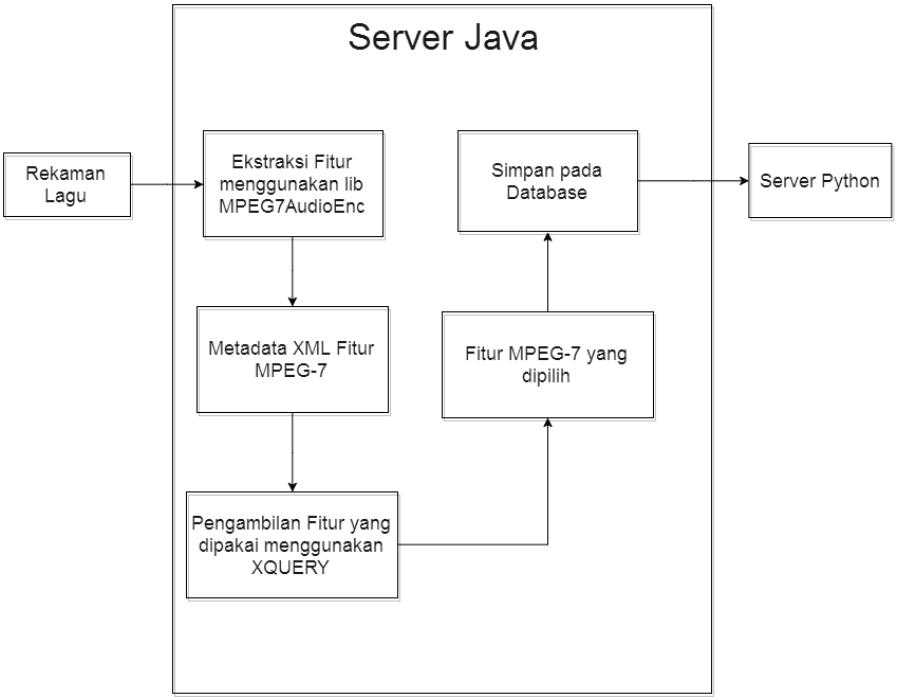
Setelah lagu ter*upload*, maka akan dilakukan ekstraksi fitur berbasis MPEG-7 yang dilakukan pada server Java. Pada tahap ini, tujuannya adalah untuk mendapatkan XML MPEG-7 dari sebuah rekaman lagu dan mengambil fitur yang digunakan untuk proses klasifikasi. Ektraksi fitur ini memakai *library* pada Java yang bernama *MPEG7AudioEnc* [26]. Untuk mengambil fitur dari XML yang dihasilkan menggunakan *Xquery* yang diimplementasikan oleh *library* Java yang bernama *BaseX* [27].

Gambar 3.2 adalah ilustrasi bagaimana dari wav utuh dilakukan ekstraksi fitur. Suatu sinyal wav dipecah-pecah menjadi karakteristik yang menggambarkan variasi-variasi dari sinyal penuh. Fitur-fitur inilah yang akan digunakan untuk proses klasifikasi. Pada proses klasifikasi, tidak semua fitur akan dipakai, hanya yang berpengaruh sajalah yang diambil dan dipakai untuk tahap *processing*.



Gambar 3.2 Ekstaksi Sinyal Wav

Untuk detail tahapan proses yang terjadi di server Java digambarkan pada Gambar 3.3 di mana suatu lagu akan dilakukan ekstraksi fitur sehingga menghasilkan metadata XML berisi fitur-fitur MPEG-7. Setelah metadata dihasilkan, maka dilakukan pengambilan fitur oleh *Xquery* untuk mengambil fitur-fitur yang akan dipakai untuk disimpan dalam *database.* Jika sudah maka tugas pada server Java selesai dan dilanjutkan tahap *processing* yang ditangani oleh server Python.



Gambar 3.3 Tahapan Ekstraksi Fitur Pada Server Java

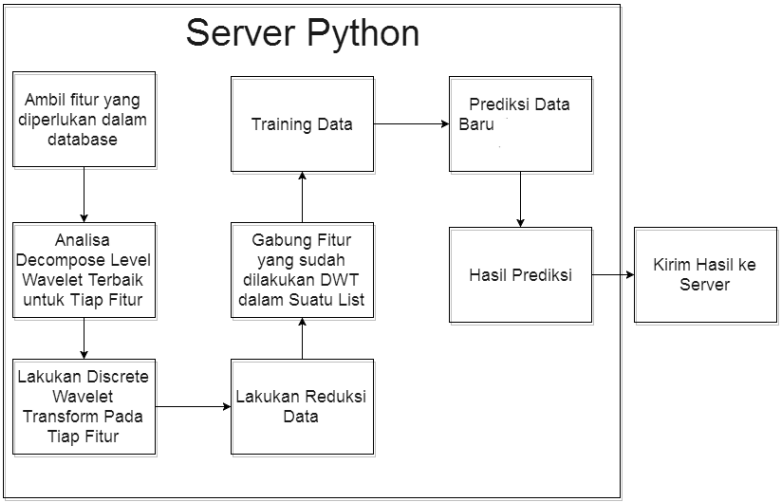
Untuk pemilihan fitur yang dipakai dan disimpan berdasarkan teori sebagai berikut:

1. Pada modul *song recognition* , audio fitur yang dipakai untuk proses adalah *Audio Signature*. Fitur ini dipakai karena mengandung keunikan dari sebuah audio.
2. Pada modul *subgenre*, audio fitur yang dipakai untuk proses adalah *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness* [22]. Fitur ini dipakai karena suatu lagu yang mempengaruhi *subgenre* adalah informasi kejernihan suara (ASC), penyimpangan spektrum dari sinyal asli (ASS), dan kerataan properti

suatu kekuatan spektrum (ASF). Maka untuk *data train* adalah gabungan ketiga sinyal yaitu ASC, ASS, dan ASF.

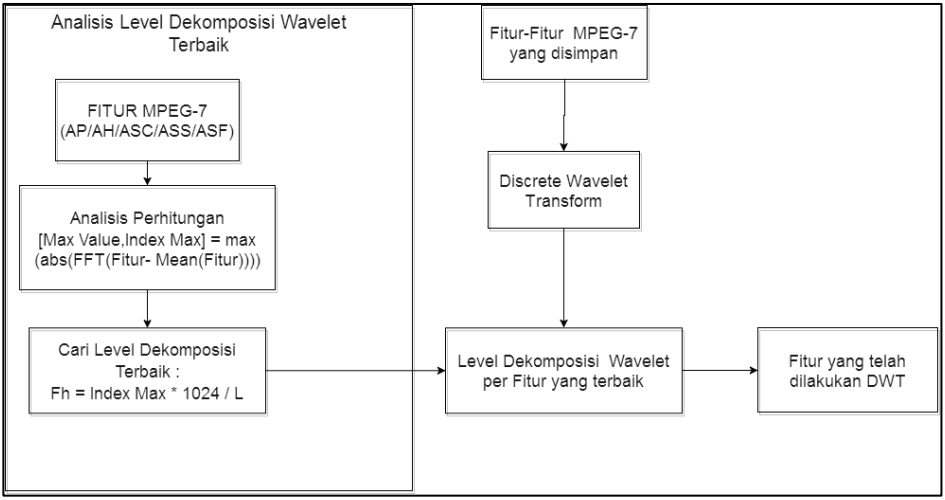
3.1.4.2.2 *Processing*

Pada tahap ini, fitur yang dipilih akan masuk ke tahap *processing*. Fitur yang dipilih ini akan dilakukan *Discrete Wavelet Transform* (DWT) dengan menggunakan tipe *wavelet bior* 2.8. Tujuannya adalah untuk menghilangkan *noise* yang terdapat pada sinyal. Implementasi yang dilakukan untuk melakukan DWT ini menggunakan *library* Python yang bernama *Pywt.* Untuk detail tahapan proses yang terjadi di server Python digambarkan pada Gambar 3.4.



Gambar 3.4 Tahapan Processing Pada Server Python

Langkah pertama yang dilakukan server Python adalah mengambil seluruh fitur-fitur yang sudah disimpan dalam *database*. Kemudian fitur-fitur inilah yang akan dilakukan proses DWT. Pada tahap melakukan DWT ini terdapat langkah-langkah yang dipaparkan pada Gambar 3.5.

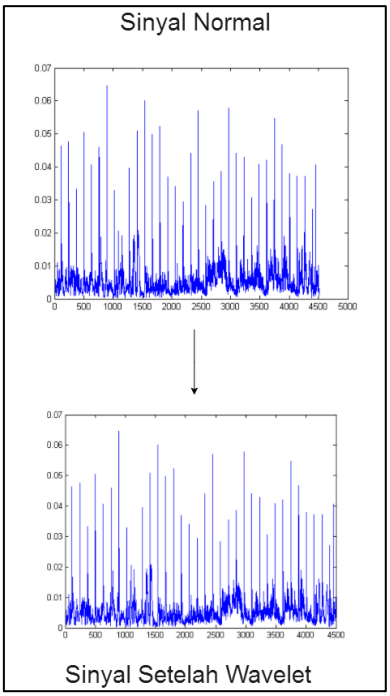


Gambar 3.5 Tahapan Melakukan DWT

Tahapan untuk melakukan DWT sebagai berikut:

1. Menentukan level dekomposisi fitur-fitur yang akan digunakan (*Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*).
2. Lakukan perhitungan pada step analisis perhitungan.
3. Cari level dekomposisi sesuai dengan rentang frekuensi yang terdapat pada Tabel 2.1.
4. Maka didapatkan level dekomposisi wavelet terbaik untuk tiap fitur.
5. Lakukan proses DWT untuk semua fitur yang akan dipakai sesuai dengan level dekomposisinya.

Contoh sinyal yang telah dilakukan wavelet dengan level dekomposisi terbaik pada Gambar 3.6. Gambar bagian atas adalah sinyal normal sedangkan bagian bawah adalah sinyal yang sudah terwavelet. Dari gambar tersebut menunjukan bahwa sinyal yang telah dilakukan DWT tidak merusak sinyal aslinya.



Gambar 3.6 Contoh Sinyal Setelah Terwavelet

Kemudian dilanjutkan tahapan pada server Python yaitu reduksi data. Reduksi data yang dimaksud adalah menyamakan panjang sinyal agar menjadi seragam untuk proses klasifikasi. Hal ini dilakukan karena ekstraksi fitur yang dihasilkan mempertimbangkan milidetik juga. Pada penelitian ini, telah dilakukan analisis pada sejumlah 296 lagu dari dataset dan menganalisis panjang data semua fitur yang sudah dilakukan DWT. Tujuannya adalah untuk mengambil panjang lagu minimal untuk dijadikan acuan penyamarataan panjang sinyal. Tabel 3.2 adalah hasil analisis panjang sinyal minimal untuk tiap fitur.

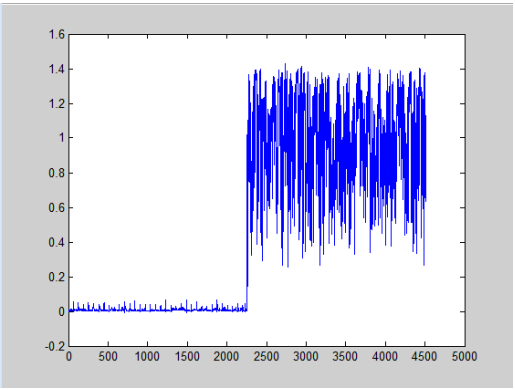
Tabel 3.2 Panjang Minimal Tiap Fitur

Fitur	Panjang minimal
Audio Spectrum Centroid	51
Audio Spectrum Spread	51
Audio Spectrum Flatness	107.904

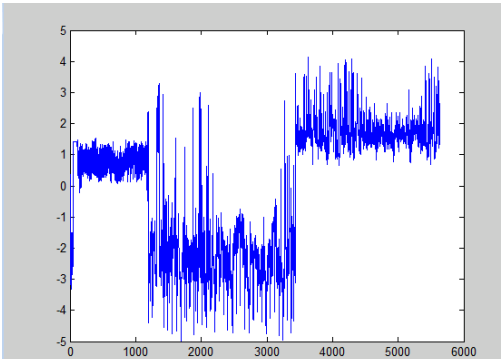
Maka panjang sinyal fitur setiap musik yang diambil sebanyak angka-angka tersebut, sisanya bisa diabaikan. Proses ini dijamin tidak akan merusak sinyal karena dari data awal panjang sinyal seragam semua yaitu 45 detik dan hanya membuang milidetik saja.

Setelah proses ini, akan masuk pada tahap berikutnya yaitu menggabungkan fitur dalam suatu list. Pada tahap ini, fitur-fitur yang sudah dilakukan DWT akan digabungkan menjadi satu *list* untuk menjadi data yamg siap diklasifikasikan.

Untuk modul *genre*, gabungan fitur yang dibentuk adalah kombinasi dari *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*. Sedangkan untuk modul *song recognition* hanya memerlukan fitur *Audio Signature*. Gambar 3.7 adalah contoh plot *list* sinyal dari *Audio Signature* dan Gambar 3.8 adalah contoh plot list sinyal dari kombinasi *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness*. Gabungan-gabungan fitur ini sudah dibagi sama panjang fitur per kolom untuk melakukan klasifikasi, sehingga keaslian informasi sinyal dijamin tidak akan rusak/berbeda.



Gambar 3.7 Fiture Audio Signature



Gambar 3.8 Gabungan Fitur ASC, ASS, dan ASF

Terakhir, adalah proses klasifikasi. Metode klasifikasi yang digunakan adalah *Deep Neural Network* (DNN). Sebelum tahap klasifikasi, ada proses yang namanya melatih data. Tujuannya adalah agar mesin dapat mengetahui beragam karakteristik sinyal yang dimaksud sesuai dengan label tertentu. Implementasi DNN dilakukan menggunakan *library* Python yang bernama *skflow*. Pada uji coba kali ini adalah melakukan *training* data untuk modul *genre* sebanyak 65 data per label. Tabel 3.3 adalah rincian penjelasan data *training*.

Tabel 3.3 Perincian Data Training

Perincian Data Training		
Modul	**Label**	**Jumlah Data Training**
Genre	Dangdut Klasik	65
	Dangdut Koplo	65
	Dangdut Remix	65

Setelah mesin membelajari karakteristik-karakteristik sinyal sesuai yang diberikan, maka prediksi data baru pun dapat dilakukan. Sehingga karakteristik lagu rekaman tadi dapat diprediksi memiliki *subgenre* apa sesuai dengan karakteristik sinyalnya. Hasil prediksi akan dibawa kembali ke server untuk ditampilkan pada aplikasi perangkat bergerak.

Pada uji coba modul *song recognition* akan menggunakan sebanyak 11 lagu dangdut. 11 lagu dangdut ini di dalamnya terdapat lagu-lagu yang ada di *subgenre* seperti data latih untuk modul *subgenre*.

3.1.4.3. *Output*

Masuk tahap terakhir yaitu adalah hasil. Lagu yang di-*upload* tadi, akan diprediksi oleh mesin yang sudah 'dilatih'lewat tampilan perangkat bergerak. Untuk *song recognitiom*, *output* suatu lagu dapat berupa judul. Pada modul *genre,* hasil berupa label dangdut klasik/dangdut koplo/dangdut remix. Untuk uji coba, akan diperinci lebih lagi pada Bab V.

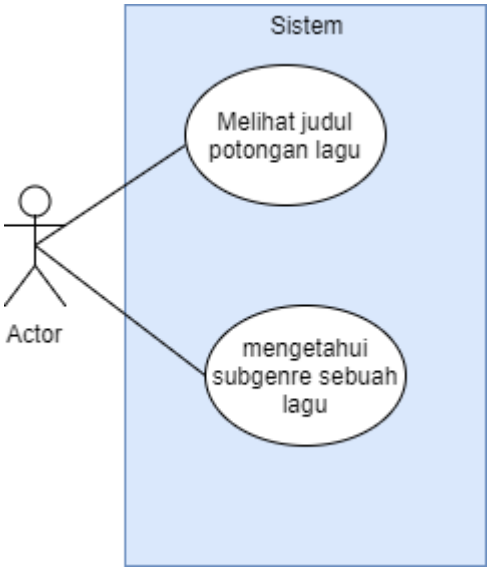
3.1.5. Kasus Penggunaan

Mengacu pada spesifikasi kebutuhan fungsional yang telah dipaparkan, dibuat kasus penggunaan yang selanjutnya akan disimpulkan dalam deskripsi umum sistem, yang diharapkan dapat memenuhi kebutuhan fungsional, berdasar pada kasus penggunaan

yang dibuat. Kasus penggunaan dijelaskan lebih lanjut pada Tabel 3.4 dan diagram kasus penggunaan ditunjukkan pada Gambar 3.9.

Tabel 3.4 Daftar Kode Kasus Penggunaan

Kode Kasus Penggunaan	Nama	Aktor
UC-1	Mengetahui judul sebuah lagu	Partisipan
UC-2	Mengetahui *subgenre* sebuah lagu	Partisipan



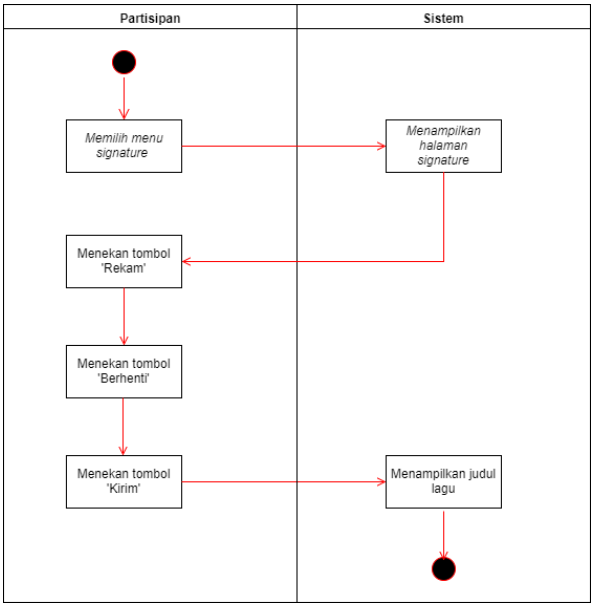
Gambar 3.9 Diagram Kasus Penggunaan

3.1.4.1. Melihat judul potongan lagu (UC-1)

Pada kasus penggunaan ini, partisipan dapat merekam suatu lagu untuk diketahui apa judul dari lagu tersebut. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.5.

Tabel 3.5 Melihat judul potongan lagu

Kode	UC-1
Nama	Melihat judul potongan lagu
Aktor	Partisipan
Deskripsi	Partisipan ingin mengetahui judul dari sebuah lagu
Kondisi Awal	Partisipan berada di menu *signature*
Kondisi akhir	Partisipan mengetahui judul lagu dari sebuah rekaman
Alur Kejadian Normal	
1. Partisipan memilih menu *signature* 2. Sistem menampilkan halaman dari menu *signature* 3. Partisipan menekan tombol 'Rekam', lalu merekam suara selama minimal 30 detik 4. Partisipan menekan tombol 'Berhenti' untuk menghentikan proses perekaman. 5. Partisipan menekan tombol 'kirim' 6. Sistem menampilkan judul lagu	
Alur Kejadian Alternatif	
-	
Eksepsi	
6.1. Jika tidak ada lagu yang ditemukan , maka sistem akan memberi pesan jika lagu tidak ditemukan	



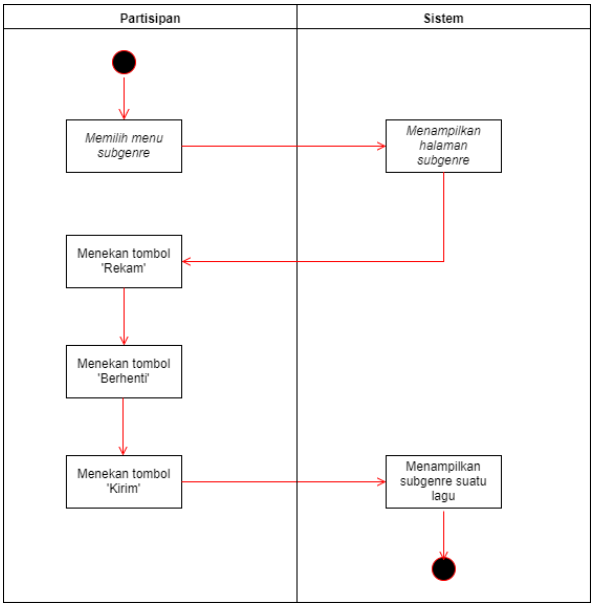
Gambar 3.10 Diagram Aktivitas Melihat Judul Potongan

3.1.4.2. Mengetahui *subgenre* sebuah lagu (UC-2)

Kasus penggunaan kode UC-2 diakses partisipan setelah berhasil merekam suatu lagu yang dimaksud, lalu diunggah ke server. Pada tahap ini partisipan akan mendapat hasil lagu yang dimaksud bertipe *subgenre* apa sesuai yang dinilai oleh sistem. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.6.

Tabel 3.6 mengetahui subgenre sebuah lagu

Kode	UC-2
Nama	Mengetahui *subgenre* potongan lagu
Aktor	Partisipan
Deskripsi	Partisipan ingin mengetahui *subgenre* dari sebuah lagu
Kondisi Awal	Partisipan berada di menu *subgenre*
Kondisi akhir	Partisipan mengetahui *subgenre* suatu lagu dari sebuah rekaman
Alur Kejadian Normal	
1. Partisipan memilih menu *subgenre* 2. Sistem menampilkan halaman dari menu *subgenre* 3. Partisipan menekan tombol 'Rekam', lalu merekam suara selama minimal 45 detik 4. Partisipan menekan tombol 'Berhenti' untuk menghentikan proses perekaman. 5. Partisipan menekan tombol 'kirim' 6. Sistem menampilkan *subgenre* dari lagu tersebut	
Alur Kejadian Alternatif	
-	



Gambar 3.11 Diagram Aktivitas Mengetahui Subgenre Lagu

3.2. Perancangan Sistem

Tahap ini meliputi perancangan basis data, tampilan antarmuka, dan perancangan alur proses penggunaan sistem yang diharapkan dapat memenuhi tujuan dari pengembangan aplikasi ini. Perlu diketahui bahwa aplikasi ini dibangun dalam kondisi lingkungan tertentu, dan dapat dioperasikan dalam lingkungan tertentu pula. Lingkungan pengembangan aplikasi adalah sebagai berikut.

Perangkat keras :

– Komputer : Prosesor Intel® Core™ i3-CPU (3.30GHz), RAM 4 GB, Graphic Intel ® Sandybridge Desktop

– Android : OS Android v5.1 Lollipop Chipset Mediatek MT6795 Helio X10, Octa-core 2.0 GHz & PowerVR G6200 GPU, RAM: 2 GB

Perangkat lunak :
Sistem operasi : Ubuntu 16.04 LTS, Android Lolipop
IDE : Android Studio 2.0, IntelliJ 2016 1.1
Basis Data : MySQL 5.6
SDK : SDK Android Lolipop level 23.

3.2.1. Perancangan Basis Data

Pada subbab ini dijelaskan mengenai perancangan basis data yang dalam hal ini digunakan untuk menyimpan data diri partisipan beserta rekam skor yang diperolehnya selama menggunakan aplikasi ini. Gambaran perancangan basis data dapat dilihat pada penjelasan di bawah. Saat ini, aplikasi masih menggunakan *database* relasional (MySQL) dikarenakan kekurangan sumber daya manusia, sedangkan seharusnya menggunakan *database* non relasional (NoSQL). Tabel 3.7, dan bel 3.8 adalah gambaran tabel atribut dan tipe data yang diterapkan pada MySQL.

Tabel 3.7 Perancangan Tabel Modul Song Recognition

Song Recognition	
Id	Int (pk,auto increment)
Judul	Varchar (250)
Signature	Text

Tabel 3.8 Perancangan Tabel Modul Genre

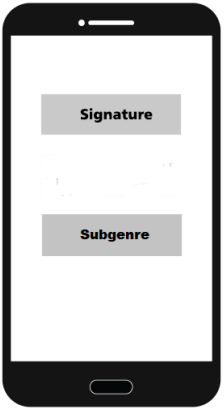
Genre	
id	Int (pk,auto increment)
judul	Varchar (250)
asc	Text
ass	Text
asf	MediumBlob

3.2.2. Perancangan Antar Muka

Subbab ini menjelaskan bagaimana rancangan antarmuka yang akan berinteraksi secara langsung dengan pengguna pada saat tahap implementasi.

3.2.2.1. Perancangan Tampilan Awal Aplikasi Dibuka

Pada tampilan awal ketika pengguna membuka aplikasi adalah terdapat 2 tombol yaitu *signature* dan *subgenre*. Tombol-tombol ini adalah fungsionalitas dari aplikasi MusicMoo. Pada pengerjaan Tugas Akhir ini, bagian yang dikerjakan adalah modul *song recognition* dan *subgenre* di mana fungsionalitas sistem diimplementasikan di dalam tombol detail lagu. Perancangan tampilan dapat dilihat pada Gambar 3.12.



Gambar 3.12 Perancangan Awal Tampilan Antarmuka

3.2.2.2. Perancangan Tampilan Menu Dalam Tombol

Pada tampilan menu ketika memencet salah satu tombol. Tampilan menu di dalam ketika tombol akan terlihat sama, hanya

yang membedakan adalah fungsionalitasnya. Perancangan tampilan menu digambarkan pada Gambar 3.13 di mana terdapat 3 tombol yaitu rekam, berhenti, dan kirim. Tombol rekam memiliki fungsi ketika ditekan akan melakukan perekaman lagu yang ingin diproses. Tombol berhenti memiliki fungsi ketika ditekan untuk menyelesaikan atau menghentikan proses perekaman. Lagu yang direkam tadi akan menjadi rekaman dengan format .wav. Terakhir tombol kirim memiliki fungsi melakukan *upload* lagu yang selesai direkam menuju server.

Gambar 3.13 Perancangan
Tampilan Menu Dalam Button

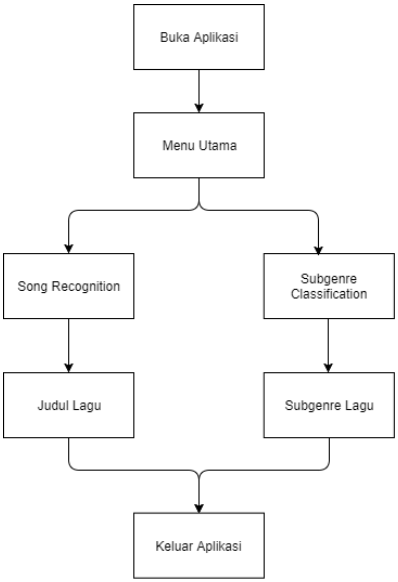
3.2.2.3. Perancangan Tampilan Hasil

Hasil yang ditampilkan pada aplikasi ini berbentuk *pop-up* notifikasi mengenai detail informasi lagu yang telah diproses. Perancangan tampilan hasil digambarkan pada Gambar 3.14. *Pop-up* dapat ditutup dengan cara menekan tulisan tutup pada pojok kanan bawah.

Gambar 3.14 Perancangan Tampilan Hasil

3.2.3. Perancangan Alur Proses Penggunaan Aplikasi

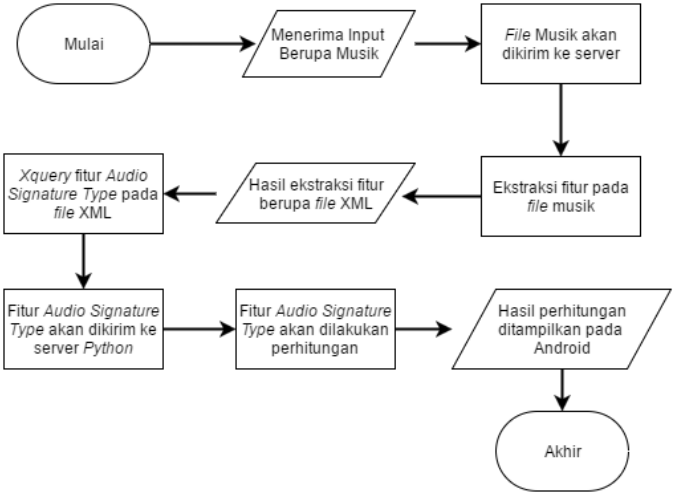
Pada tahap ini akan dijelaskan mengenai rancangan alur proses penggunaan aplikasi yang digunakan sebagai acuan dalam pembangunan aplikasi. Alur proses ini membantu memperlihatkan langkah demi langkah yang akan dilakukan pengguna sehingga terlihat jelas antara *input*, *proses*, dan *output.* Alur penggunaan aplikasi akan dijelaskan pada Gambar 3.15.



Gambar 3.15 Diagaram Alur Penggunaan Aplikasi

3.2.3.1. Modul *song recognition*

Proses ini 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.



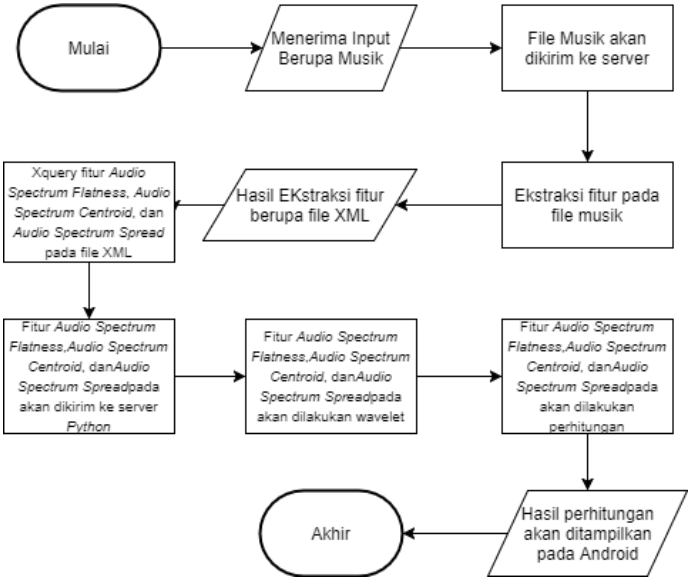
Gambar 3.16 Workflow Alur Modul *Song Recognition*

Pada Gambar 3.16 merupakan diagram alur dari modul *song recognition*. Pertama 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 *song recognition*. Fitur yang sesuai dengan modul *song recognition* 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 berupa judul dari lagu yang direkam.

3.2.3.2. Modul Klasifikasi *subgenre*

Proses pendeteksi *subgenre* memiliki kemiripan dengan modul *song recognition*. Potongan audio dari lagu akan didengarkan oleh aplikasi Android dan di-*upload* ke server.

Perbedaan hanya pada tahapan *pre-processing* pada server. Modul *subgenre* mengaplikasikan *Discrete Wavelet Transform* pada tahapan *pre-processing*. Setelah tahapan perhitungan dan klasifikasi selesai, server akan mengembalikan *string* yang berisi informasi tentang *subgenre* dari potongan lagu yang diperdengarkan.



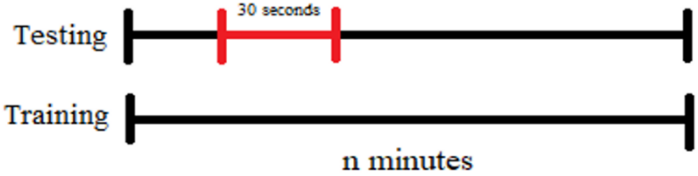
Gambar 3.17 Workflow Alur Modul klasifikasi *subgenre*

Pertama 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 *subgenre classification*. Fitur yang sesuai

dengan modul *subgenre classification* adalah fitur *Audio Spectrum Flatness*, *Audio Spectrum Centroid*, dan *Audio Spectrum Spread*. Ketiga fitur ini digunakan dalam perhitungan pada server Python untuk mencari kemiripan sebuah musik dengan data latih yang terdapat dalam *database*. Hasil dari perhitungan akan ditampilkan pada aplikasi Android berupa judul lagu asli dari lagu yang direkam. Diagram alur dari proses ini dapat dilihat pada Gambar 3.17.

3.2.4. Ketentuan Data Uji

Pada modul *song recognition*, sebuah lagu harus direkam dengan panjang minimal adalah 30 detik. Ilustrasi dari data uji modul *song recognition* bisa dilihat pada Gambar 3.18.



Gambar 3.18 Perancangan Data Uji Modul *song recognition*

Sementara itu untuk data uji modul *subgenre*, sebuah lagu harus direkam selama minimal 45 detik. Batasan dalam perekaman lagu ini bertujuan agar bisa mendapatkan *output* sesuai modul yang sedang diuji.

BAB IV
IMPLEMENTASI

Bab ini membahas implementasi dari analisis dan perancangan sistem yang telah dibahas pada Bab III. Namun dalam penerapannya, rancangan tersebut dapat mengalami perubahan minor sewaktu-waktu apabila dibutuhkan.

4.1. Lingkungan Implementasi

Dalam implementasinya, lingkungan yang digunakan sama seperti yang dituliskan pada rancangan, yakni menggunakan beberapa perangkat pendukung sebagai berikut.

4.1.1. Lingkungan Implementasi Perangkat Keras

Perangkat keras yang digunakan dalam implementasi pengembangan aplikasi ini adalah Komputer. Spesifikasi komputer yang digunakan adalah laptop dengan prosesor AMD E2-1800 APU *with* Radeon(tm) HD Graphics (2CPUs) dan RAM 4 GB.

4.1.2. Lingkungan Implementasi Perangkat Lunak

Penjelasan perangkat lunak yang digunakan dalam implementasi aplikasi ini adalah sebagai berikut:

- Ubuntu 16.04 sebagai sistem operasi pada *personal computer*.
- MySQL untuk mengimplementasikan rancangan basis data.

4.2. Implementasi Server

Subbab ini membahas tentang implementasi tampilan antarmuka yang telah dirancang dan dibahas pada Bab III. Selanjutnya akan dirinci dengan detail sebagai berikut.

4.2.1. Implementasi Konfigurasi Server dengan Flask

Kode Sumber 4.1 merupakan implementasi untuk melakukan konfigurasi server Flask. Dapat dilihat bahwa perlu dilakukan konfigurasi dulu yang terkait dengan nama *user*, *database*, *password,* serta *host.*

```
from flask import Flask, render_template, session, redirect, url_for
from flask.ext.mysql import MySQL
import re
import os

app = Flask(__name__)
mysql = MySQL()

app.config['MYSQL_DATABASE_USER'] = 'MusicMoo2'
app.config['MYSQL_DATABASE_PASSWORD'] = '12345'
app.config['MYSQL_DATABASE_DB'] = 'MusicMoo'
app.config['MYSQL_DATABASE_HOST'] = '10.151.64.84'

mysql.init_app(app)


if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5124, debug=True)
```

Kode Sumber 4.1 Konfigurasi Server Flask

4.2.2. Implementasi Mengubah Sinyal dari Time Domain menjadi Frekuensi Domain

Kode Sumber 4.2 adalah implementasi untuk mengubah sinyal dari *time domain* menjadi *frequency domain*. Tujuannya adalah untuk menganalisis sinyal mempunyai frekuensi maksimal berapa sebagai acuan level dekomposisi. *Input* yang diterima berupa sinyal dan *output*nya adalah hasil dari fungsi nilai dekomposisi level.

```
def frekuensi(arr):
    npArr = np.array(arr)
    mean = np.mean(npArr)
    fin = npArr - mean

    hasil = max(abs(np.fft.fft(fin.transpose(), axis = 0) ) )
    fft = np.array(abs(np.fft.fft(fin.transpose(), axis=0 )))
```

```
8.      maxIndex = np.where(fft == hasil)[0][0]
9.
10.     fre = float(maxIndex) * 1024 / len(arr)
11.
12.     return nilaiDekomposisi(fre)
```

Kode Sumber 4.2 Mengubah Sinyal dari Time Domain Menjadi Frequency Domain

4.2.3. Implementasi Mencari Nilai Dekomposisi Terbaik

Kode Sumber 4.3 adalah implementasi untuk mencari nilai dekomposisi *wavelet* terbaik. Fungsi ini digunakan setelah sinyal dari *time* domain diubah menjadi frekuensi domain. *Input* yang diterima adalah hasil perhitungan pada variabel fre pada fungsi yang dijelaskan pada Kode Sumber 4.2. *Outputnya* adalah nilai level dekomposisi.

```
1.  def nilaiDekomposisi(fre):
2.      if (fre > 512):
3.          return 0
4.      elif(256 <= fre and fre <= 512):
5.          return 1
6.      elif(128 <= fre and fre <= 256):
7.          return 2
8.      elif(64 <= fre and fre <= 128):
9.          return 3
10.     elif(32 <= fre and fre <= 64):
11.         return 4
12.     elif(16 <= fre and fre <= 32):
13.         return 5
14.     elif(8 <= fre and fre <= 16):
15.         return 6
16.     elif(4 <= fre and fre <= 8):
17.         return 7
18.     elif(2 <= fre and fre <= 4):
19.         return 8
20.     elif(1 <= fre and fre <= 2):
21.         return 9
22.     elif(0.5 <= fre and fre <= 1):
```

```
23.         return 10
24.     elif(0.25 <= fre and fre <= 0.5):
25.         return 11
26.     elif(0.125 <= fre and fre <= 0.25):
27.         return 12
28.     elif(0.0625 <= fre and fre <= 0.125):
29.         return 13
```

Kode Sumber 4.3 Mencari Nilai Dekomposisi Terbaik

4.2.4. Implementasi *Discrete Wavelet Transform*

Kode Sumber 4.4 berikut ini adalah implementasi untuk melakukan *Discrete Wavelet Transform* (DWT). Fungsi ini digunakan untuk menghilangkan *noise* dari audio asli. *Input* yang diterima berupa sinyal lagu dan n level dekomposisi. *Outputnya* berupa sinyal yang sudah terwavelet dengan level dekomposisi terbaik.

```
1.  import pywt
2.
3.  def waveletTransform(lagu, n):
4.      w = pywt.Wavelet('bior2.8')
5.      maks = pywt.dwt_max_level(data_len=len(lagu), f
    ilter_len=w.dec_len)
6.      if(n > maks):
7.          n = maks
8.      hasil = pywt.wavedec(lagu, "bior2.8", level = n
    )
9.
10.     return hasil[0]
```

Kode Sumber 4.4 Melakukan Wavelet

4.2.5. Implementasi Reduksi Data

Kode Sumber 4.5 berikut ini adalah implementasi untuk melakukan penyamaan panjang fitur pada yang didapatkan. Tujuannya adalah menyeragamkan data agar dapat diklasifikasn menggunakan DNN. Fungsi ini dilakukan bersamaan dengan

menjalankan fungsi implementasi *Discrete Wavelet Transform* (DWT).

```
1.  panjangminimASC = 51
2.  panjangminimASS = 51
3.  panjangminimASF = 107904
4.
5.  #Contoh fitur yang dilakukan reduksi panjang#
6.  wvlaguASC18=dc.waveletTransform(ASC18.ASC,levelASC)
    [:panjangminimASC]
7.  wvlaguASS18=dc.waveletTransform(ASS18.ASS,levelASS)
    [:panjangminimASS]
8.  wvlaguASF18=dc.waveletTransform(ASF18.ASF,levelASF)
    [:panjangminimASF]
```

Kode Sumber 4.5 Implementasi Reduksi Data

4.2.6. Implementasi Gabung Fitur

Kode Sumber 4.6 berikut adalah implementasi untuk melakukan penggabungan 2 atau 3 fitur ke dalam 1 *list*. *List* inilah yang nantinya digunakan untuk proses *training* yang digunakan untuk tahap klasifikasi menggunakan DNN. *Input* berupa fitur yang sudah dilakukan DWT beserta reduksi data. Untuk urutan tidak boleh terbalik, di mana *list* untuk modul *subgenre* urutannya sebagai berikut: 51 panjang pertama untuk fitur *Audio Spectrum Centroid*, 51 panjang berikutnya untuk fitur *Audio Spectrum Spread*, dan sisanya untuk fitur *Audio Spectrum Flatness.*

```
1.  gabunganfitur18=list(merge(wvlaguASC18,wvlaguASS18,
    wvlaguASF18)) #membuat gabungan list berisi ASC, AS
    S dan ASF untuk modul subgenre
```

Kode Sumber 4.6 Implementasi Gabung Fitur

4.2.7. Implementasi Deep Neural Network

Kode Sumber 4.7 merupakan implementasi untuk melakukan DNN pada modul *subgenre*. Pada implementasi ini

menggunakan 2 *hidden layer* dan 400 *neuron* di setiap *layer*. Implementasi ini menerima 2 *input* yaitu data latih dan data uji. X adalah parameter untuk data latih dan y adalah label dari masing-masing data uji. *Output* dari proses ini adalah prediksi *subgenre* dari data *input* baru suatu lagu.

```
1.  def Genre (inputsinyal):
2.      X = np.array([
3.          #Isi List Data Training sebanyak 65x per su
    bgenre lagu
4.      ])
5.
6.
7.      y = np.array([
8.          #Beri label pada masing masing list. Ada da
    ngdut klasik, dangdut koplo, dan dangdut remix
    sebanyak 65 per masing-masing label
9.          ])
10.
11.
12.     import tensorflow as tf
13.     import tensorflow.contrib.learn as skflow
14.
15.     feature_columns =
    [tf.contrib.layers.real_valued_column("",
    dimension=4)]
16.
17.
18.     classifier =
    skflow.DNNClassifier(feature_columns=feature_column
    s, hidden_units=[400, 400], n_classes=3)
19.
20.     classifier.fit(X,y)
21.
22.     from tensorflow.contrib.learn import SKCompat
23.
24.     prediksi =
    classifier.predict_classes([inputsinyal])
    //sinyal inputan
25.     if prediksi == [1] : print 'Dangdut Klasik'
26.     elif prediksi== [2] : print 'Dangdut Koplo'
```

```
27.    else:  print 'Dangdut Remix'
28.     return prediksi
29.
```

Kode Sumber 4.7 DNN Pada Modul Genre

4.2.8. Implementasi *Sliding Algorithm*

Implementasi ini menerima 2 *input* yaitu sinyal dari data latih dan data uji. *Output* dari proses ini adalah nilai terkecil dari *Sliding Algorithm*. Implementasi *Sliding Algorithm* terletak pada sisi server dan dijalankan ketika menerima *input* 2 sinyal yang akan dibandingkan. Perhitungan pada *Sliding Algorithm*, menggunakan perhitungan *Bhattacharyya Distance* pada setiap *subband* dari sinyal. *Output* dari fungsi *sliding* (Kode Sumber 4.8) adalah nilai kesamaan terkecil dari beberapa *subband* yang telah dibandingkan. Nilai kesamaan terkecil ini diasumsikan sebagai nilai kesamaan suatu sinyal terhadap sinyal lain.

```
1.  def mean(signal):
2.      mean= 0.0
3.      For i in signal
4.          mean += i
5.      mean/= len(signal)
6.      Return mean
7.
8.  def sliding(fulllagu, potongan):
9.      distance = []
10.     i = 0
11.     k = 0
12.     n = len(fulllagu) / len(potongan)
13.     full = mean(fulllagu);
14.     pot = mean(potongan);
15.     score = 0;
16.     while (akhir  < n):
17.        j = 0
```

```
18.         jumlah = 0.0
19.         score = math.sqrt( fulllagu[j] * potongan[j]
    );
20.         while(j < len(potongan)):
21.                 jumlah +=  math.sqrt( 1 - ( 1 /
    math.sqrt(full*pot*16*16) ) * score )
22.                 j = j+1
23.                 k = k+1
24.         distance.append(jumlah)
25.         akhir = akhir +16
26.         indexsek += 16
27.         k = indexsek
28.     return min(distance)
```

Kode Sumber 4.8 *Sliding Algorithm*

4.2.9. Implementasi Modul *Song Recognition*

Implementasi pada modul ini menerima *input* berupa suara dan menghasilkan *output* berapa judul asli dari potongan rekaman lagu. Implementasi modul *song recognition* terletak pada sisi komputer server. Modul *song recognition* akan dijalankan ketika *route* "/slidingSignature/" akan dijalankan (Kode sumber 4.9). Modul *song recognition* akan menerima *input* dari SQL *database* dan melakukan perhitungan. Perhitungan dilakukan dengan cara *Sliding Algorithm* dan menerima beberapa nilai. Nilai tersebut akan diurutkan dari yang terkecil hingga terbesar. Nilai terkecil dianggap sebagai lagu yang dicari.

```
1.  @app.route("/slidingSignature/")
2.  def slidingSignature():
3.      conn = MySQL.connect()
4.      cursor = conn.cursor()
5.
6.      cursor.execute('Select * from Lagu')
7.      conn.commit()
```

```
8.      data1 = cursor.fetchall() #isinya semua data
9.
10.     cursor.execute('Select * from Testing')
11.     conn.commit()
12.     testing = cursor.fetchall()
13.
14.     sinyalT = stringToList(testing[0][1])
15.
16.     sinyalTraining = []
17.     i = 0
18.     j = len(data1)
19.
20.     while(i < j):
21.         sinyal = stringToList(data1[i][2])
22.         #level = frekuensi(sinyal)
23.         hasil = (data1[i][1],sliding(sinyal, sinyal
    T))
24.         sinyalTraining.append(hasil)
25.         i = i+1
26.
27.     classifier = sorted(sinyalTraining, key=lambda
    x:x[1])
28.     i = 0
29.     ret = ""
30.     while(i < 5):
31.         ret += classifier[i][0] + " "
32.         #ret += classifier[i][0] + " "  +str(classi
    fier[i][1]) + "<br>"
33.         i = i+1
34.     return ret
```

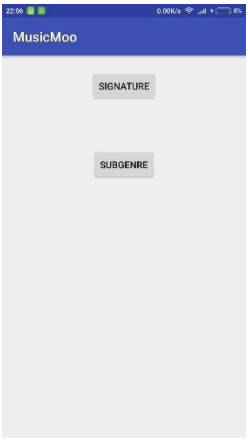
Kode Sumber 4.9 Implementasi Modul *Song Recognition*

4.3. Implementasi Perangkat Bergerak

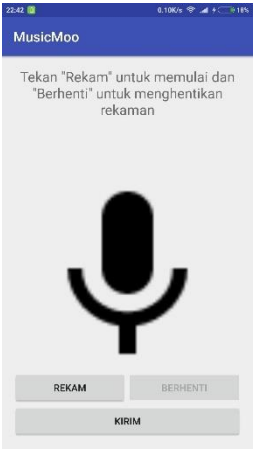
Pada subbab ini dijelaskan implementasi pada perangkat bergerak yang akan dibangun.

4.3.1. Implementasi Antar Muka

Berikut adalah tampilan implementasi antarmuka yang terlihat pada tampilan perangkat bergerak. Antarmuka dibangun sesuai yang dirancang pada subbab 3.2.2. Tampilan awal aplikasi dibuka dapat dilihat ada Gambar 4.1 sedangkan tampilan menu dalam suatu tombol dapat dilihat pada Gambar 4.2.



Gambar 4.1 Tampilan Awal Antarmuka Pada Perangkat Bergerak



Gambar 4.2 Tampilan Menu Di Dalam Tombol Pada Perangkat Bergerak

4.3.2. Implementasi Perekaman Audio

Kode Sumber 4.10 merupakan implementasi aplikasi Android untuk melakukan perekaman suara. Proses perekaman audio menggunakan *sample rate* 44100 Hz dan *encoding* PCM 16 Bit untuk mendapatkan hasil rekaman dengan kualitas terbaik.

```
1.  private void startRecording(){
2.      recorder = new AudioRecord(MediaRecorder.AudioS
    ource.MIC,
3.              RECORDER_SAMPLERATE, RECORDER_CHANNELS,
    RECORDER_AUDIO_ENCODING, bufferSize);
4.
5.      int i = recorder.getState();
6.      if(i==1)
7.          recorder.startRecording();
8.
9.      isRecording = true;
10.
11.     recordingThread = new Thread(new Runnable() {
12.
13.         @Override
```

```
14.         public void run() {
15.             writeAudioDataToFile();
16.         }
17.     },"AudioRecorder Thread");
18.
19.     recordingThread.start();
20. }
```

Kode Sumber 4.10 Implementasi Perekaman Audio

4.3.3. Implementasi Memberhentikan Perekaman Audio

Kode Sumber 4.11 merupakan implementasi aplikasi Android untuk memberhentikan proses perekaman suara.

```
1.  private void stopRecording(){
2.      if(null != recorder){
3.          isRecording = false;
4.
5.          int i = recorder.getState();
6.          if(i==1)
7.              recorder.stop();
8.          recorder.release();
9.
10.         recorder = null;
11.         recordingThread = null;
12.     }
13.
14.     copyWaveFile(getTempFilename(),getFilename());

15.     deleteTempFile();
16. }
```

Kode Sumber 4.11 Memberhentikan Perekaman Audio

4.3.4. Implementasi Penyimpanan Hasil Rekaman Audio

Kode Sumber 4.12 merupakan implementasi aplikasi Android untuk menyimpan hasil perekaman suara. Hasil yang disimpan adalah rekaman lagu dalam format wav.

```
1.  private String getFilename(){
2.      String filepath = Environment.getExternalStorag
    eDirectory().getPath();
3.      File file = new File(filepath,AUDIO_RECORDER_FO
    LDER);
4.
5.      if(!file.exists()){
6.          file.mkdirs();
7.      }
8.      selectedFilePath = file.getAbsolutePath() + "/"
     + "clip" + AUDIO_RECORDER_FILE_EXT_WAV;
9.  rn (file.getAbsolutePath() + "/" + "clip" + AUDIO_R
    ECORDER_FILE_EXT_WAV);
10. }
```

Kode Sumber 4.12 Penyimpanan Hasil Rekaman Audio

4.3.5. Implementasi Upload Lagu Ke Server

Kode Sumber ini merupakan implementasi aplikasi Android untuk meng-*upload* hasil perekaman suara ke server. Kode sumber dapat dilihat pada Lampiran Kode Sumber A.1 yang terletak pada lampiran.

4.4. Implementasi Ekstraktor MPEG-7

Pada subbab ini dijelaskan implementasi proses ekstraksi audio fitur berbasis MPEG-7.

4.4.1. Implementasi Mengimpor Library yang Digunakan

Kode Sumber 4.13 merupakan implementasi untuk mengimpor *library* yang digunakan yaitu *basex*, *basex-api*, *basex-xqj*-1.2.3, dan *MPEG7AudioEnc. Library* yang terdapat kata

basex digunakan untuk melakukan *Xquery*. Sedangkan *MPEG7AudioEnc* digunakan untuk membuat metadata XML hasil ekstraksi MPEG-7.

```
1.  import org.basex.core.*;
2.  import org.basex.core.cmd.*;
3.  import org.basex.io.serial.*;
4.  import org.basex.query.*;
5.  import org.basex.query.iter.*;
6.  import org.basex.query.value.*;
7.  import org.basex.query.value.item.*;
8.
9.  import java.io.IOException;
```

Kode Sumber 4.13 Mengimpor Library yang Digunakan

4.4.2. Implementasi Ekstraktor basis MPEG-7

Kode Sumber 4.14 merupakan implementasi untuk ekstrasi audio fitur metadata berbasis MPEG-7. Tujuannya adalah untuk mengelolah fitur yang dimiliki suatu audio untuk memperkaya informasi. *Input* berupa direktori *path file* wav dan *output* yg dihasilkan adalah metadata XML MPEG-7 yang berada di direktori *file* tempat audio disimpan.

```
1.  public class Descriptor{
2.      public static void main(String[] args){
3.          try {
4.              String sourceConfig = "C:\\Users\\mahar
    dika\\Documents\\NetBeansProjects\\MPEG-
    7_Extract\\config\\config.xml";
5.              InputStream inputStream = new FileInput
    Stream(sourceConfig);
6.              Reader reader = new InputStreamReader(i
    nputStream);
7.              Config config = ConfigXML.parse(reader)
    ;
8.              File folder = new File("D:\\Datasets TA
    \\");
9.              File[] listFile = folder.listFiles();
```

```
10.             for(int i = 0 ; i < listFile.length ; i
    ++) {
11.                 if(listFile[i].isFile()) {
12.                     AudioInputStream audioInputStre
    am = AudioSystem.getAudioInputStream(listFile[i]);

13.                     Document mpeg7 = MP7DocumentBui
    lder.encode(audioInputStream, config);
14.                     DOMSource domSource = new DOMSo
    urce(mpeg7);
15.                     StringWriter stringWriter = new
     StringWriter();
16.                     StreamResult result = new Strea
    mResult(stringWriter);
17.                     TransformerFactory transformerF
    actory = TransformerFactory.newInstance();
18.                     Transformer transformer = trans
    formerFactory.newTransformer();
19.                     transformer.transform(domSource
    , result);
20.                     String hasil = stringWriter.toS
    tring();
21.                     File file1 = new File("D:\\Data
    sets TA\\" + listFile[i].getName().split(".wav")[0]
     + ".xml");
22.                     FileWriter fileWriter = new Fil
    eWriter(file1, false);
23.                     fileWriter.write(hasil);
24.                     stringWriter.flush();
25.                     fileWriter.flush();
26.                     fileWriter.close();
27.                     System.out.println("i :" + i);

28.                 }
29.             }
30.         } catch (IOException e) {
31.             System.out.println(e);
32.         }catch (UnsupportedAudioFileException e){
33.             System.out.println(e);
34.         } catch (ParserConfigurationException e){
35.             System.out.println(e);
36.         } catch (TransformerException e){
37.             System.out.println("e");
```

```
38.         } catch (SAXException e){
39.             System.out.println("error baru");
40.         }
41.     }
42. }
```

Kode Sumber 4.14 Implementasi Ektraktor basis MPEG-7

4.5. Implementasi XQuery untuk Mengambil Fitur Audio

Pada subbab ini dijelaskan implementasi pengambilan fitur dari metadata XML yang dihasilkan oleh ekstraksi fitur berbasis MPEG-7.

4.5.1. Mengambil Nilai Audio Signature Type

Kode Sumber 4.15 merupakan kode untuk mengimplementasikan mengambil nilai *Audio Signature Type*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Signature Type.*

```
1.      private static String AudioSignatureType(String
    string) throws BaseXException{
2.
3.          String query =
4.                  "declare default element namespace
    \"urn:mpeg:mpeg7:schema:2001\";" +
5.                          "declare namespace mpeg7 =
    \"urn:mpeg:mpeg7:schema:2001\";" +
6.                          "declare namespace xsi = \"
    http://www.w3.org/2001/XMLSchema-instance\";" +
7.                          "for $x in doc(\"C:/Users/p
    onighzwa/Desktop/wav/Testing/XML/"+ string+".xml\")
    /Mpeg7/Description/MultimediaContent/Audio/" +
8.                          "AudioDescriptionScheme ret
    urn if($x/@xsi:type=\"AudioSignatureType\")then dat
    a($x/Flatness/SeriesOfVector/Mean) else \"\"";
9.          String hasil = new XQuery(query).execute(co
    ntext);
```

```
10.         return hasil;
11.         //System.out.println(hasil);
12.     }
```

Kode Sumber 4.15 Mengambil Nilai Audio Signature Type

4.5.2. Mengambil Nilai Audio Spectrum Centroid

Kode Sumber 4.16 merupakan implementasi untuk mengambil nilai *Audio Spectrum Centroid*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Spectrum Centroid*.

```
1.  public static String[] AudioSpectrumCentroidType(St
    ring path) throws BaseXException{
2.      String query =
3.              "declare default element namespace \"ur
    n:mpeg:mpeg7:schema:2001\";" +
4.                      "declare namespace mpeg7 = \"ur
    n:mpeg:mpeg7:schema:2001\";" +
5.                      "declare namespace xsi = \"http
    ://www.w3.org/2001/XMLSchema-instance\";" +
6.                      "for $x in doc(\""+path+"\")/Mp
    eg7/Description/MultimediaContent/Audio/AudioDescri
    ptor\n return if($x/@xsi:type=\"AudioSpectrumCentro
    idType\")then data($x/SeriesOfScalar/Raw) else \"\"
    ";
7.   String hasil = new XQuery(query).execute(context);

8.     String[] hasil1 = hasil.split("
9.  ");
10.     for(int i = 0 ; i<hasil1.length;i++){
11.         System.out.print(hasil1[i].trim().replace("
     ",","));
12.         if(i != hasil1.length-1)
13.            System.out.print(",");
14.     }
15.    return hasil1;
```

```
16.     //System.out.println(new XQuery(query).execute(
    context));
17. }
```

Kode Sumber 4.16 Mengambil Nilai Audio Spectrum Centroid

4.5.3. Mengambil Nilai Audio Spectrum Spread

Kode Sumber 4.17 merupakan implementasi untuk mengambil nilai *Audio Spectrum Spread*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Spectrum Spread.*

```
1.    public static String[] AudioSpectrumSpreadType(S
    tring path) throws BaseXException{
2.
3.        String query =
4.                "declare default element namespace \
    "urn:mpeg:mpeg7:schema:2001\";" +
5.                        "declare namespace mpeg7 = \
    "urn:mpeg:mpeg7:schema:2001\";" +
6.                        "declare namespace xsi = \"h
    ttp://www.w3.org/2001/XMLSchema-instance\";" +
7.                        "for $x in doc(\""+path+"\")
    /Mpeg7/Description/MultimediaContent/Audio/AudioDes
    criptor\n return if($x/@xsi:type=\"AudioSpectrumSpr
    eadType\")then data($x/SeriesOfScalar/Raw) else \"\
    "";
8.
9.  String hasil = new XQuery(query).execute(context);

10.       String[] hasil1 = hasil.split("
11. ");
12.        for(int i = 0 ; i<hasil1.length;i++){
13.            System.out.print(hasil1[i].trim().replac
    e(" ",","));
14.            if(i != hasil1.length-1)
15.               System.out.print(",");
16.        }
17.       return hasil1;
```

```
18.         //System.out.println(new XQuery(query).execu
    te(context));
19.     }
```

Kode Sumber 4.17 Mengambil Nilai Audio Spectrum Spread

4.5.4. Mengambil Nilai Audio Spectrum Flatness

Kode Sumber 4.18 merupakan implementasi untuk mengambil nilai *Audio Spectrum Flatness*. Kode sumber berikut adalah penerapan *query* pada data XML dengan metode *Xquery*. *Output* yang dihasilkan adalah nilai dari fitur *Audio Spectrum Flatness*.

```
1.     public  static String[] AudioSpectrumFlatnessTyp
    e(String path) throws BaseXException{
2.         String query =
3.                 "declare default element namespace \
    "urn:mpeg:mpeg7:schema:2001\";" +
4.                         "declare namespace mpeg7 = \
    "urn:mpeg:mpeg7:schema:2001\";" +
5.                         "declare namespace xsi = \"h
    ttp://www.w3.org/2001/XMLSchema-instance\";" +
6.                         "for $x in doc(\""+path+"\")
    /Mpeg7/Description/MultimediaContent/Audio/AudioDes
    criptor\n return if($x/@xsi:type=\"AudioSpectrumFla
    tnessType\")then data($x/SeriesOfVector/Raw) else \
    "\"";
7.  String hasil = new XQuery(query).execute(context);

8.        String[] hasil1 = hasil.split("
9.  ");
10.        for(int i = 0 ; i<hasil1.length;i++){
11.            System.out.print(hasil1[i].trim().replac
    e(" ",","));
12.            if(i != hasil1.length-1)
13.               System.out.print(",");
14.        }
15.       return hasil1;
```

```
16.     }
```

Kode Sumber 4.18 Mengambil Nilai Audio Spectrum Flatness

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada aplikasi yang dikembangkan. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsionalitas sistem yang telah dijabarkan pada Bab III dan terhadap tujuan dibuatnya aplikasi ini, yakni agar musik memiliki informasi yang lebih mendetail dan makin lengkap.

5.1. Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas sebagai berikut:

Prosesor : Prosesor Intel® Core™ i3-CPU
RAM : 4 GB
Jenis *Device* : Personal Computer
Sistem Operasi : Ubuntu 16.04 LTS

Sedangkan perangkat bergerak yang digunakan dalam pengujian adalah sebagi berikut:

Prosesor : Hexa-core Max1.8GHz
RAM : 3 GB
Jenis *Device* : Perangkat bergerak
Sistem Operasi : Android v5.1.1 Lollipop

5.2. Skenario Pengujian

Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan. Uji coba dilakukan untuk setiap modul yaitu *song recognition*, dan *subgenre*. Terdapat dua skenario saat ujicoba, dikarenakan hasil *output* yang berbeda di setiap modulnya:

1. Skenario 1: Melakukan uji coba pada modul *song recognition*.

- Sebuah lagu, direkam melalui aplikasi perangkat bergerak minimal 30 detik, jika sudah tekan tombol *stop*.
- *Upload* lagu dalam server.
- Server akan melakukan ekstraksi fitur dan mengambil fitur yang akan diproses dengan *Xquery.*
- Sistem menyimpan fitur yang dipakai dalam *database*.
- Melakukan *processing* pada server Python.
- Lakukan *training* data.
- Sistem akan memprediksi data baru (rekaman yang di-*upload* tadi).
- Hasil akan keluar sesuai dengan pembelajaran mesin.
- Hasil bisa berubah-rubah dikarenakan dipengaruhi oleh: *speaker* sumber suara/perekam suara dan *noise* suatu lingkungan.

2. Skenario 2: Melakukan uji coba lewat aplikasi perangkat bergerak.
    - Sebuah lagu, direkam melalui aplikasi perangkat bergerak minimal 45 detik, jika sudah tekan tombol *stop*.
    - *Upload* lagu dalam server.
    - Server akan melakukan ekstraksi fitur dan mengambil fitur yang akan diproses dengan *Xquery.*
    - Sistem menyimpan fitur yang dipakai dalam *database*.
    - Melakukan *processing* pada server Python.
    - Ambil data dari *database* lakukan DWT.
    - Lakukan *training* data.
    - Sistem akan memprediksi data baru (rekaman yang di-*upload* tadi).
    - Hasil akan keluar sesuai dengan pembelajaran mesin.
    - Hasil bisa berubah-rubah dikarenakan dipengaruhi oleh: *speaker* sumber suara/perekam suara dan *noise* suatu lingkungan.

Pada data akurasi Tugas Akhir ini, dilakukan uji coba menggunakan perangkat bergerak sesuai yang dijelaskan pada bab perancangan sistem. Tabel 5.1 memperlihatkan skenario pengujian pada tiap modul.

Tabel 5.1 Skenario Pengujian

Kode Pengujian	Skenario Pengujian
SP-UC1	Pengujian Modul *Song Recognition*
SP-UC2	Pengujian Modul *Subgenre*

5.2.1. Pengujian Fungsionalitas

Pengujian fungsionalitas aplikasi dilakukan secara mandiri dengan melakukan skenario yang sama dengan rancangan alur proses aplikasi sebagai tolok ukur keberhasilan pengujian, dan mengacu pada kasus penggunaan yang sebelumnya telah dijelaskan pada Bab III. Pengujian pada kebutuhan fungsionalitas dapat dijabarkan pada subbab berikut.

5.2.1.1. Pengujian Modul *Song Recognition*

Pengujian modul *song recognition* dimulai ketika pengguna melakukan perekaman pada suatu sumber suara. Perekaman dilakukan pada perangkat bergerak Android yang telah dilakukan instalasi aplikasi tugas akhir ini. Aplikasi akan mengeluarkan nilai *string* yang mengandung informasi dari judul lagu yang dicari. Rincian dari pengujian modul dapat dilihat pada Tabel 5.2. Hasil dari modul *song recognition* pada aplikasi jika lagu ditemukan dapat dilihat pada Gambar 5.1 dan jika tidak ada lagu yang ditemukan bisa dilihat pada Gambar 5.2.

Tabel 5.2 Pengujian Modul *Song Recognition*

Nomor	**SP-UC01**
Nama	Pengujian Modul *Song Recognition*
Tujuan	Mengecek apakah aplikasi dapat mendeteksi lagu dari potongan suara
Kondisi Awal	Pengguna membuka halaman modul *song recognition*
Skenario	Pengguna mendengarkan kepada aplikasi berupa potongan lagu
Langkah pengujian	1. Pengguna menekan tombol "rekam" untuk memulai proses perekaman 2. Pengguna menekan tombol "berhenti" untuk menghentikan proses perekaman 3. Pengguna menekan tombol "upload" untuk mendapatkan hasil sesuai dengan modul.
Masukan	Potongan lagu
Keluaran yang Diharapkan	Judul lagu asli dari potongan lagu yang didengarkan kepada aplikasi
Hasil Pengujian	Berhasil

Gambar 5.1 Hasil dari Modul song recognition

Gambar 5.2 Hasil Tidak Ada Lagu di Database

5.2.1.2. Pengujian Modul *Subgenre*

Pengujian pada modul *subgenre* ini dimulai dengan pengguna melakukan perekaman pada suatu lagu. Perekaman dilakukan melalui aplikasi perangkat bergerak. Rincian skenario pengujian pada kasus penggunaan dapat dilihat pada Tabel 5.3. Hasil dari modul *subgenre* pada aplikasi dapat dilihat pada Gambar 5.3.

Tabel 5.3 Pengujian Modul Subgenre

Nomor	**SP-UC02**
Nama	Pengujian Modul *Subgenre*
Tujuan	Mengecek apakah aplikasi dapat mendeteksi lagu dari potongan suara
Kondisi Awal	Pengguna membuka halaman modul *sungenre*
Skenario	Pengguna mendengarkan kepada aplikasi berupa potongan lagu
Langkah pengujian	1. Pengguna menekan tombol "rekam" untuk memulai proses perekaman 2. Pengguna menekan tombol "berhenti" untuk menghentikan proses perekaman 3. Pengguna menekan tombol "upload" untuk mendapatkan hasil sesuai dengan modul.
Masukan	Potongan lagu
Keluaran yang Diharapkan	*Subgenre* dari potongan lagu yang didengarkan kepada aplikasi
Hasil Pengujian	Berhasil

Gambar 5.3 Hasil dari Modul *subgenre*

5.3. Akurasi Pengujian Fungsionalitas

Pada subbab ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang

diberikan meliputi evaluasi pengujian kebutuhan fungsional sesuai dengan modul yang terdapat aplikasi.

Akurasi dihitung menggunakan Persamaan 5.1 yang merupakan perhitungan dengan cara *Positive Predictive Value* (PPV) sebagai berikut:

$$PPV = \frac{TP}{TP + FP} * 100\% \tag{5.1}$$

Di mana *TP* adalah *True Positif* yaitu jumlah data yang diprediksi benar sesuai dengan labelnya dan *FP* adalah *False Positif* yaitu jumlah data yang diprediksi dikatakan benar namun kenyataannya adalah salah. *TP* + *FP* bisa juga dibilang jumlah data keseluruhan yang dipakai dalam pengujian.

Untuk detail lagu pada data *testing* masing-masing modul dapat dilihat pada Tabel A.1, Tabel A.2, dan Tabel A.3 yang terletak pada lampiran. Berikut penjelasan uji coba yang dilakukan pada subbab di bawah.

5.3.1. Uji Coba Penentuan Threshold Modul *Song Recognition*

Penentuan *Threshold* ini diperlukan untuk penanganan kasus jika tidak ada lagu di dalam *database* Tugas Akhir kali ini. *Threshold* ditentukan setelah melakukan pengujian terhadap lagu yang terdapat di dalam *database*. Hasil dari pengujian ini berupa nilai yang merupakan jarak dari data latih dan data uji. Hasil dari pengujian bisa dilihat pada Tabel 5.4.

Tabel 5.4 Hasil *Threshold* Modul *Song recognition*

Judul Lagu	Nilai
Begadang	4.01
Judi	4.06
Benang Biru	4.25
Kereta Malam (Ori)	2.07
Jera	5.57
Jaran Goyang	3.43
Kimcil Kepolen	4.51
Kereta Malam (Koplo)	2.53
Janji Palsu	2.63
Goyang Dumang	3.11
Klepek-klepek	4.94

Tabel 5.5 Hasil *Threshold* lagu diluar *database*

Judul Lagu	Nilai
Bizarre Love Triangle	10.144
Asal Kau Tahu	11.21
Biar Ku Sendiri	9.01
Bagaimana Ku Tahu	11.15
Berdistraksi	9.1
Sayang	12.04
Berdua Saja	11.2
Rahasia	11.7
How Deep Is Your Love	7.03
Salamku Untuk Kekasihmu Yang Baru	7.65
Show Me the Meaning of Being Lonely	9.79

Dari Tabel 5.4 dapat dilihat jika nilai dari lagu-lagu yang ada di dalam *database* tidak ada yang melebihi nilai 6. Dari table 5.5 juga dapa dilihat jika nilai dari lagu-lagu yang tidak ada di *database* memiliki nilai tidak kurang dari 6. Maka dari itu dalam

Tugas Akhir kali ini, bisa diambil nilai 6 sebagai batas maksimal atau *threshold* dari nilai jarak antara data latih dan data uji.

5.3.2. Hasil Percobaan Modul *Song Recognition*

Tingkat akurasi modul *song recognition*, didapat ketika hasil judul lagu dari sistem sesuai dengan potongan judul yang diperdengarkan. Data *training* diambil dari internet dengan format .wav sebanyak 11 judul. Data untuk *testing* diacak dari 11 judul tersebut untuk dipotong 30 detik. Pada pengujian pertama data uji diambil secara acak dari bagian manapun musik pada data latih dan dilakukan perhitungan. Detail hasil dapat dilihat pada Tabel 5.6.

Tabel 5.6 Hasil Percobaan Modul *Song recognition*

Judul Lagu	Hasil
Begadang	V
Judi	V
Benang Biru	V
Kereta Malam (Ori)	V
Jera	V
Jaran Goyang	V
Kimcil Kepolen	V
Kereta Malam (Koplo)	V
Janji Palsu	X
Goyang Dumang	V
Klepek-klepek	V

Keterangan:
X: Potongan lagu yang tidak terdeksi.
V: Potongan lagu yang berhasil dideteksi.

Dari Tabel 5.4 dapat disimpulkan bahwa dari 11 percobaan terdapat 10 hasil yang benar dan 1 yang salah. Kesalahan 1 data *testing* tersebut dapat dihasilkan karena terjadi kemiripan yang

tinggi antar satu lagu dengan lagu lainnya. Sehingga untuk akurasi modul *song recognition* menurut Persamaan 5.1 adalah 90,9%.

5.3.3. Hasil Percobaan Modul Subgenre

Percobaan di modul ini telah dilakukan sebanyak 3 kali dengan jumlah *hidden layer* dan *neuron* yang berbeda di setiap pengujiannya. Penghitungan akurasi pada percobaan ini menggunakan Persamaan 5.1. Percobaan pertama menggunakan 3 *hidden layer* dengan 100 *neuron* di *layer* 1, 200 *neuron* di *layer* 1, dan 100 *neuron* di *layer* 3. Percobaan pertama menghasilkan akurasi sebesar 72.22% dengan rincian hasil percobaan bisa dilihat pada Lampiran B.1. Percobaan kedua menggunakan 4 *hidden layer* dengan 200 *neuron* di setiap *layer*. Percobaan kedua menghasilkan akurasi sebesar 61,11% dengan rincian hasil percobaan bisa dilihat pada Lampiran B.2. Percobaan ketiga menggunakan 2 *hidden layer* dengan 400 *neuron* di setiap *layer*. Percobaan ketiga menghasilkan akurasi sebesar 77,77%. Rincian hasil yang diperoleh dari percobaan ketiga dilampirkan pada Tabel 5.7 sebagai berikut.

Tabel 5.7 Hasil percobaan Modul Subgenre

Aktual	**Pengujian**		
	Dangdut Klasik	Dangdut Koplo	Dangdut Remix
Dangdut Klasik	6	0	0
Dangdut Koplo	0	3	3
Dangdut Remix	0	1	5

5.3.4. Uji Coba *Cross Validation* pada Modul Subgenre

Uji coba *cross validation* ini dilakukan dengan menggunakan metode *Stratified kFold*. Perbedaan dari *kFold* biasa ialah *Stratified kFold* akan memastikan setiap *fold* telah merepresentasikan data dari seluruh *dataset*. Dalam

penggunaannya, metode ini akan diterapkan menggunakan *model_selection* yang merupakan *library* dari bahasa pemrograman python.

Metode *kFold* ini bergantung pada jumlah *k* yang digunakan. Pada uji coba Tugas Akhir ini menggunakan *k* dengan jumlah 10. Untuk pembagian data latih dan data uji sebanyak, 10% untuk data uji dan 90% untuk data latih dari jumlah data yang sudah dijelaskan pada Bab III.

Tabel 5.8 Hasil Uji *Cross Validation*

***k* Ke-**	**Hasil**
1	0.815
2	0.868
3	0.833
4	0.833
5	0.833
6	0.852
7	0.852
8	0.852
9	0.875
10	0.812
Rata-rata	0.842

Hasil dari uji *Cross Validation* ini dapat dilihat pada Tabel 5.8. Hasil dari uji *Cross Validation* ini menghasilkan akurasi sebesar 0.842 atau 84.2%. Selain itu standar deviasi yang didapatkan sudah baik, yaitu sebesar 0.019.

5.4. Evaluasi Pengujian

Pada subbab ini membahas hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian kebutuhan fungsional beserta evaluasi berupa akurasi pada masing-masing modul.

5.4.1. Evaluasi Pengujian Fungsionalitas

Rangkuman mengenai hasil pengujian fungsionalitas dapat dilihat pada Tabel 5.9. Berdasarkan data pada tabel tersebut, semua skenario pengujian berhasil dan program berjalan dengan baik. Sehingga bisa ditarik kesimpulan bahwa fungsionalitas dari aplikasi telah dapat bekerja sesuai dengan yang diharapkan.

Tabel 5.9 Rangkuman Hasil Pengujian

Kode Pengujian	Skenario Pengujian	Hasil
SP-UC1	Pengujian Modul *Song Recognition*	Berhasil
SP-UC2	Pengujian Modul *Subgenre*	Berhasil

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan Tugas Akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian perangkat lunak yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Fitur audio yang diekstrak dengan menggunakan basis MPEG-7 menghasilkan 17 fitur MPEG-7 *Low Level Descriptors*.
2. *Fingerprint* dapat didapatkan dengan fitur *Audio Signature Type* pada fitur MPEG-7. Pendeteksian *fingerprint* dapat menggunakan algoritma *Bhattacharyya Distance* yang dimodifikasi.
3. Pada modul *subgenre,* audio fitur yang berpengaruh adalah fitur *Audio Spectrum Centroid*, *Audio Spectrum Spread*, dan *Audio Spectrum Flatness* pada fitur MPEG-7.
4. Tingkat akurasi untuk modul *song recognition* yaitu 90,90%
5. Tingkat akurasi untuk modul *subgenre* yaitu 77,77%.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi dan pengujian yang telah dilakukan:

1. Peningkatan akurasi untuk masing-masing modul agar hasil semakin membaik.
2. Jika ingin menggunakan *Deep Neural Network,* ada baiknya jika melakukan estimasi untuk mengetahui berapa

jumlah *hidden layer* dan *neuron* yang dibutuhkan untuk mendapat hasil akurasi yang maksimal.

3. Coba untuk mengaplikasikan metode *deep learning* lainnya seperti CNN (*Convolutional Neural Netwrok)* atau RNN (*Recurrent Neural Network)*.

DAFTAR PUSTAKA

[1] F. Wiering, "Can Humans Benefit from Music Information Retrieval?," in *Proceedings of the 4th International Conference on Adaptive Multimedia Retrieval: User, Context, and Feedback*, Berlin, Heidelberg, 2007, pp. 82–94.

[2] ISO/IEC (2001), "Information Technology — Multimedia Content Description Interface — Part 4: Audio," *FDIS 15938-4:2001*(*E*), June.

[3] M. Rocamora, P. Cancela, and A. Pardo, "Query by humming: Automatically building the database from music recordings," *Pattern Recognit. Lett.*, vol. 36, pp. 272–280, Jan. 2014.

[4] N. Chen, J. S. Downie, H. Xiao, and Y. Zhu, "Cochlear pitch class profile for cover song identification," *Appl. Acoust.*, vol. 99, pp. 92–96, Dec. 2015.

[5] G. Muhammad and M. Melhem, "Pathological voice detection and binary classification using MPEG-7 audio features," *Biomed. Signal Process. Control*, vol. 11, pp. 1–9, May 2014.

[6] H.-G. Kim, N. Moreau, and T. Sikora, *MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval*. John Wiley & Sons, 2005.

[7] "Audio | MPEG." [Online]. Available: http://mpeg.chiariglione.org/standards/mpeg-7/audio. [Accessed: 13-Jan-2017].

[8] R. X. Gao and R. Yan, *Wavelets: Theory and Applications for Manufacturing*, 2011 edition. New York; London: Springer, 2010.

[9] B. T. Nugraha, R. Sarno, D. A. Asfani, T. Igasaki, and M. N. Munawar, "Classification of driver fatigue state based on EEG using Emotiv EPOC+," *J. Theor. Appl. Inf. Technol.*, vol. 86, no. 3, pp. 347–359, Apr. 2016.

[10] R. Sarno, M. N. Munawar, B. T. Nugraha, R. Sarno, M. N. Munawar, and B. T. Nugraha, "Real-Time Electroencephalography-Based Emotion Recognition

System," *Int. Rev. Comput. Softw. IRECOS*, vol. 11, no. 5, pp. 456–465, May 2016.

[11] 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.

[12] 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.

[13] D. R. Wijaya, R. Sarno, and E. Zulaika, "Information Quality Ratio as a novel metric for mother wavelet selection," *Chemom. Intell. Lab. Syst.*, vol. 160, pp. 59–71, Jan. 2017.

[14] M. Gruhne, R. Tous, J. Delgado, M. Doeller, and H. Kosch, "Introduction of an Mpeg-7 Query Format."

[15] B. Team, "The XML Database," 27-May-2015. [Online]. Available: http://basex.org/home/. [Accessed: 15-Jan-2017].

[16] "why_mir." [Online]. Available: http://musicinformationretrieval.com/why_mir.html. [Accessed: 12-Dec-2017].

[17] "dangdut." [Online]. Available: https://id.wikipedia.org/wiki/Dangdut. [Accessed: 12-Dec-2017].

[18] H.-G. Kim, N. Moreau, and T. Sikora, "MPEG-7 Audio and Beyond Audio Content Indexing and Retrieval," p. 217.

[19] 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.

[20] M. M. Deza and E. Deza, *Encyclopedia of Distances*. Springer Science & Business Media, 2009.

[21] Arjun Raj Rajanna, Kamelia Aryafar, Ali Shokoufandeh, and Raymond Ptucha, "Deep Neural Networks: A Case Study for Music Genre Classification," in 2015 IEEE 14th International Conference on Machine Learning and Applications, 2012.

[23] S. Li, H. Li, and L. Ma, "Music Genre Classification Based on MPEG-7 Audio Features," in *Proceedings of the Second International Conference on Internet Multimedia Computing and Service*, New York, NY, USA, 2010, pp. 185–188.
[24] "WAV." [Online]. Available: https://en.wikipedia.org/wiki/WAV. [Accessed: 10-Jan-2018].
[25] "Introduction to Sckiti Flow." [Online]. Available: https://terrytangyuan.github.io/2016/03/14/scikit-flow-intro/. [Accessed: 10-Jan-2018].
[26] "Copyright." [Online]. Available: http://mpeg7audioenc.sourceforge.net/copyright.html. [Accessed: 10-Jan-2018].
[27] "XQuery Tutorial." [Online]. Available: http://www.w3schools.com/xml/xquery_intro.asp. [Accessed: 10-Jan-2018].
[28] Johanes A. R, Rancang Bangun Aplikasi MusicMoo dengan Metode MIR (Music Information Retrieval) pada Modul Mood,Genre Recognition,dan Tempo Estimation, Surabaya: Institut Teknologi Sepuluh Nopember, 2017.
[29] Mochammad Faris P. R., Rancang Bangun Aplikasi MusicMoo dengan Metode MIR (Music Information Retrieval) pada Modul Fingerprint, Cover Song Recognition, dan Song Recommendation, Surabaya: Institut Teknologi Sepuluh Nopember, 2017.
[30] A. N. Weintraub, Dangdut Stories: A Social and Musical History of Indonesia's Most Popular Music, Oxford: Oxford University Press, 2010.
[31] M. H. B. Raditya, Esensi Senggakan Pada Dangdut Koplo Sebagai Identitas Musikal, Yogyakarta: Universitas Gadjah Mada, 2013.
[32] P. D. Greene and T. Porcello, Wired for Sound: Engineering and Technologies in Sonic Cultures, Connecticut: Wesleyan University Press, 2005.

[Halaman ini sengaja dikosongkan]

LAMPIRAN A

LAMPIRAN KODE SUMBER

```
1.  private class PostDataTOServer extends AsyncTask<Vo
    id, Void, Void> {
2.          //Create hashmap Object to send parameters
    to web service
3.          HashMap<String, String> postDataParams;
4.          ProgressDialog progressDialog;
5.          TextView textView;
6.          @Override
7.          protected void onPreExecute() {
8.              super.onPreExecute();
9.
10.
11.             progressDialog = new ProgressDialog(Mai
    nActivity.this);
12.             progressDialog.setMessage("Please wait.
    ..");
13.             progressDialog.setCancelable(false);
14.             progressDialog.show();
15.
16.             textView = (TextView) findViewById(R.id
    .hasil);
17.         }
18.         @Override
19.         protected Void doInBackground(Void... arg0)
     {
20.             int serverResponseCode = 0;
21.
22.             HttpURLConnection connection;
23.             DataOutputStream dataOutputStream;
24.             String lineEnd = "\r\n";
25.             String twoHyphens = "--";
26.             String boundary = "*****";
27.
28.             int bytesRead,bytesAvailable,bufferSize
    ;
29.             byte[] buffer;
30.             int maxBufferSize = 1 * 1024 * 1024;
```

```
31.             File selectedFile = new File(selectedFi
    lePath);
32.
33.             String[] parts = selectedFilePath.split
    ("/");
34.             final String fileName = parts[parts.len
    gth-1];
35.
36.             try{
37.                 FileInputStream fileInputStream = n
    ew FileInputStream(selectedFile);
38.                 URL url = new URL(SERVER_URL);
39.                 connection = (HttpURLConnection) ur
    l.openConnection();
40.                 connection.setDoInput(true);//Allow
     Inputs
41.                 connection.setDoOutput(true);//Allo
    w Outputs
42.                 connection.setUseCaches(false);//Do
    n't use a cached Copy
43.                 connection.setRequestMethod("POST")
    ;
44.                 connection.setRequestProperty("Conn
    ection", "Keep-Alive");
45.                 connection.setRequestProperty("ENCT
    YPE", "multipart/form-data");
46.                 connection.setRequestProperty("Cont
    ent-Type", "multipart/form-
    data;boundary=" + boundary);
47.                 connection.setRequestProperty("uplo
    aded_file",selectedFilePath);
48.
49.                 //creating new dataoutputstream
50.                 dataOutputStream = new DataOutputSt
    ream(connection.getOutputStream());
51.
52.                 //writing bytes to data outputstrea
    m
53.                 dataOutputStream.writeBytes(twoHyph
    ens + boundary + lineEnd);
54.                 dataOutputStream.writeBytes("Conten
    t-Disposition: form-
    data; name=\"uploaded_file\";filename=\""
```

```
55.                         + selectedFilePath + "\"" +
     lineEnd);
56.
57.                 dataOutputStream.writeBytes(lineEnd
    );
58.
59.                 //returns no. of bytes present in f
    ileInputStream
60.                 bytesAvailable = fileInputStream.av
    ailable();
61.                 //selecting the buffer size as mini
    mum of available bytes or 1 MB
62.                 bufferSize = Math.min(bytesAvailabl
    e,maxBufferSize);
63.                 //setting the buffer as byte array
    of size of bufferSize
64.                 buffer = new byte[bufferSize];
65.
66.                 //reads bytes from FileInputStream(
    from 0th index of buffer to buffersize)
67.                 bytesRead = fileInputStream.read(bu
    ffer,0,bufferSize);
68.
69.                 //loop repeats till bytesRead = -
    1, i.e., no bytes are left to read
70.                 while (bytesRead > 0){
71.                     //write the bytes read from inp
    utstream
72.                     dataOutputStream.write(buffer,0
    ,bufferSize);
73.                     bytesAvailable = fileInputStrea
    m.available();
74.                     bufferSize = Math.min(bytesAvai
    lable,maxBufferSize);
75.                     bytesRead = fileInputStream.rea
    d(buffer,0,bufferSize);
76.                 }
77.
78.                 dataOutputStream.writeBytes(lineEnd
    );
79.                 dataOutputStream.writeBytes(twoHyph
    ens + boundary + twoHyphens + lineEnd);
80.
```

```
81.                 serverResponseCode = connection.get
    ResponseCode();
82.                 String serverResponseMessage = conn
    ection.getResponseMessage();
83.
84.                 Log.i(TAG, "Server Response is: " +
     serverResponseMessage + ": " + serverResponseCode)
    ;
85.
86.                 //response code of 200 indicates th
    e server status OK
87.
88.                 //closing the input and output stre
    ams
89.
90.                 String line;
91.                 BufferedReader br = new BufferedRea
    der(new InputStreamReader(connection.getInputStream
    ()));
92.                 //Log.d("Output",br.toString());
93.                 while ((line = br.readLine()) != nu
    ll) {
94.                     response += line;
95.                     Log.d("output lines", line);
96.                 }
97.
98.                 fileInputStream.close();
99.                 dataOutputStream.flush();
100.                        dataOutputStream.close();
101.
102.
103.
104.                } catch (FileNotFoundException e
    ) {
105.                    e.printStackTrace();
106.                    runOnUiThread(new Runnable()
     {
107.                        @Override
108.                        public void run() {
109.                            Toast.makeText(MainA
    ctivity.this,"File Not Found",Toast.LENGTH_SHORT).s
    how();
110.                        }
```

```
111.                        });
112.                    } catch (MalformedURLException e
    ) {
113.                        e.printStackTrace();
114.                        Toast.makeText(MainActivity.
    this, "URL error!", Toast.LENGTH_SHORT).show();
115.
116.                    } catch (IOException e) {
117.                        e.printStackTrace();
118.                        Toast.makeText(MainActivity.
    this, "Cannot Read/Write File!", Toast.LENGTH_SHORT
    ).show();
119.                    }
120.                    return null;
121.                }
122.                @Override
123.                protected void onPostExecute(Void re
    sult) {
124.                    super.onPostExecute(result);
125.                    progressDialog.dismiss();
126.                    //Toast.makeText(MainActivity.th
    is,response,Toast.LENGTH_LONG).show();
127.                    textView.setText(response);
128.                }
129.            }
130.        }
131.
```

Kode Sumber A.1 Upload Lagu ke Server

[Halaman ini sengaja dikosongkan]

LAMPIRAN B

LAMPIRAN TABEL

Tabel B.1 Hasil Percobaan Pertama Modul Subgenre

	Pengujian		
Aktual	Dangdut Klasik	Dangdut Koplo	Dangdut Remix
Dangdut Klasik	5	0	1
Dangdut Koplo	1	4	1
Dangdut Remix	0	2	4

Tabel B.2 Hasil Percobaan Kedua Modul Subgenre

	Pengujian		
Aktual	Dangdut Klasik	Dangdut Koplo	Dangdut Remix
Dangdut Klasik	5	1	0
Dangdut Koplo	0	3	3
Dangdut Remix	1	2	3

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Muhammad Nezar Mahardika, lahir pada tanggal 17 Agustus 1995 di Malang. Setelah lulus dari SMAN 4 Malang, penulis menempuh pendidikan perguruan tinggi di Institut Teknologi Sepuluh Nopember Surabaya di Jurusan Teknik Informatika Fakultas Teknologi Informasi pada tahun 2014. Penulis memiliki pengalaman menjadi asisten dosen pada mata kuliah Jaringan Komputer dan Sistem Basis Data. Penulis terlibat aktif dalam organisasi kemahasiswaan dan kegiatan kepanitiaan selama berkuliah, antara lain Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS pada tahun 2015-2017, staff Kementrian Komunikasi dan Informasi BEM ITS pada tahun 2015-2016, serta staff pada kegiatan Schematics tahun 2015 dan 2016. Dalam melakukan pengerjaan Tugas Akhir, penulis memiliki ketertarikan pada bidang Manajemen Informasi (MI). Untuk menghubungi penulis, dapat menghubungi melalui *email*: nezar.mahardika14@mhs.if.its.ac.id.

[Halaman ini sengaja dikosongkan]