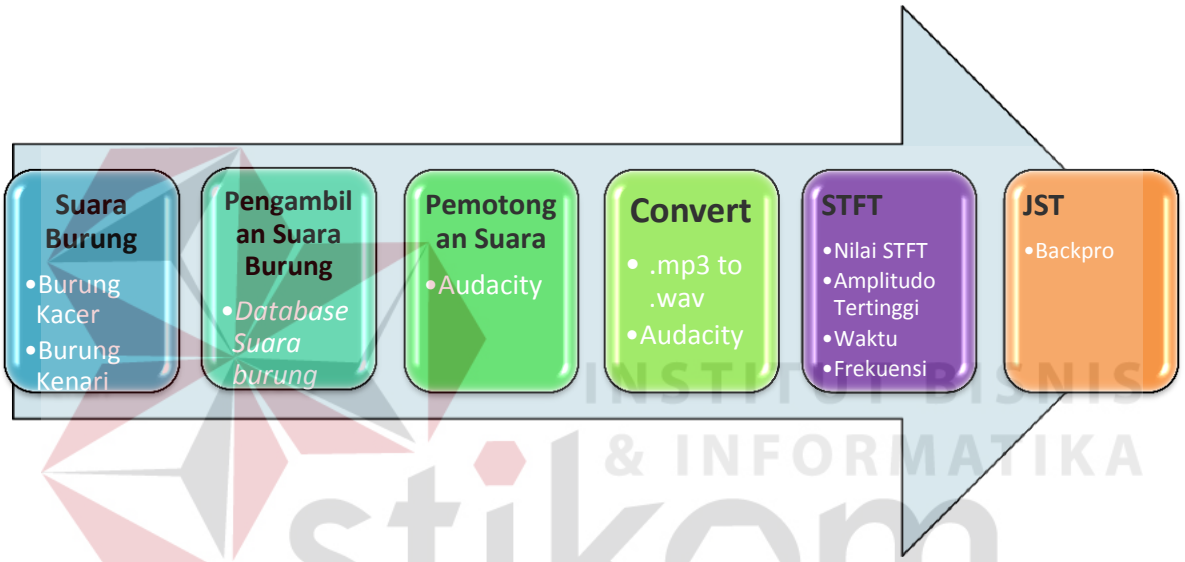
BAB III

METODE PENELITIAN

3.1 Model Penelitian

Penelitian yang dilakukan dapat dijelaskan melalui blok diagram seperti yang terlihat pada Gambar 3.1.



Gambar 3.1 Blok Diagram Penelitian

Berikut adalah keterangan setiap blok dari sistem blok diagram pada Gambar 3.1.

3.2 Suara Burung

Pada bagian ini, ada terdapat dua suara burung yang akan diteliti, yaitu burung Kacer dan burung Kenari. Suara burung masing–masing diambil dari *database* suara burung. Adapun, terdapat 3 pengambilan *database* dari setiap masing–masing jenis burung, dan masing–masing suara burung tesebut akan

dipotong masing–masing 10 detik sehingga menghasilkan 10 buah *sample* dari setiap data jenis burung. Sehingga, pada 1 jenis burung akan didapatkan *sample* sebanyak 30, yang akan digunakan sebagai data yang akan diolah pada penelitian ini. Karena tujuan akhir dari penelitian ini, nantinya adalah digunakan sebagai sistem penjurian, maka burung yang dipilih sebagai *sample* data adalah burung dengan kualitas suara yang merdu. Maka diambilah suara burung dari beberapa *database* dibawah ini.

1. Suara burung Kacer
    a) https://www.youtube.com/watch?v=iUpmJ6uA9CM
    b) http://www.suaraburungs.com/2014/10/koleksi-suara-kacer-untuk-memaster.html
    c) http://www.suaraburungs.com/2014/10/koleksi-suara-kacer-untuk-memaster.html
2. Suara burung kenari
    a) https://www.youtube.com/watch?v=gxWJQWFbPjM
    b) http://www.budidayakenari.com/2015/04/suara-kenari-juara-nasional.html
    c) http://www.budidayakenari.com/2015/04/suara-kenari-juara-nasional.html

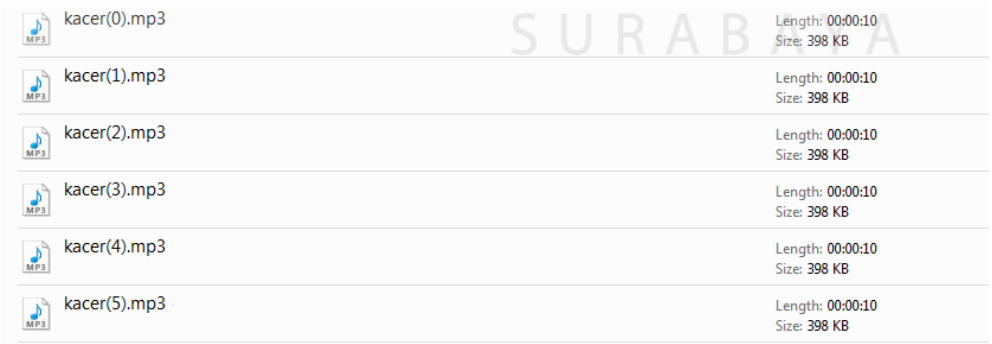
3.3 Pengambilan Suara Burung

Suara burung diambil dari beberapa *database* suara burung pemenang lomba yang ada pada internet. Hal ini dikarenakan, data yang diolah harus

merupakan suara kicau burung yang pernah menjuarai sebuah perlombaan. Karena, hasil akhir dari penelitian ini, nantinya diharapkan dapat dijadikan sebuah alat untuk dapat mengenali suara jenis burung dan dapat dijadikan tolak ukur *objective* dari sebuah penilaian suara burung. Ada beberapa web yang dapat dipertanggung jawabkan yang datanya diambil untuk dijadikan data *sample* pada penelitian ini.

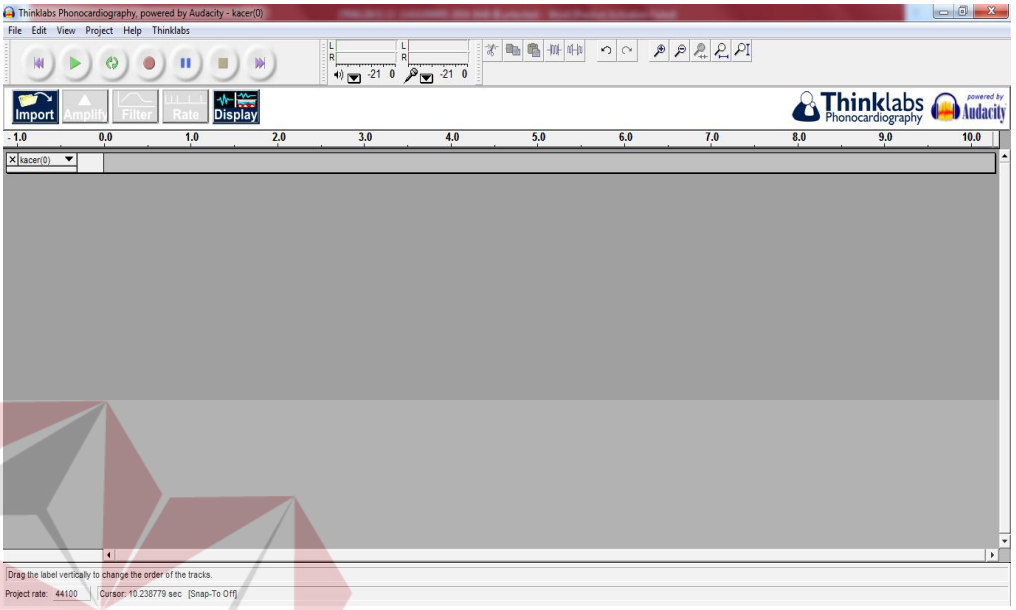
3.4 Hasil dan Pengolahan Data

File rekaman menggunakan *format* (*.mp3*) dan *sample rate* 44100 Hz. Masing-masing suara burung yang diambil dari *database* dipotong- potong sehingga masing – masing data menghasilkan data sebanyak 10 data suara burung. Hasil perekaman dipotong, dan diambil hanya 10 detik tiap *sample* nya. Pemotongan suara burung secara manual dilakukan dengan menggunakan *software Audacity.* Hasil pemotongan data suara burung dapat dilihat pada gambar 3.2.

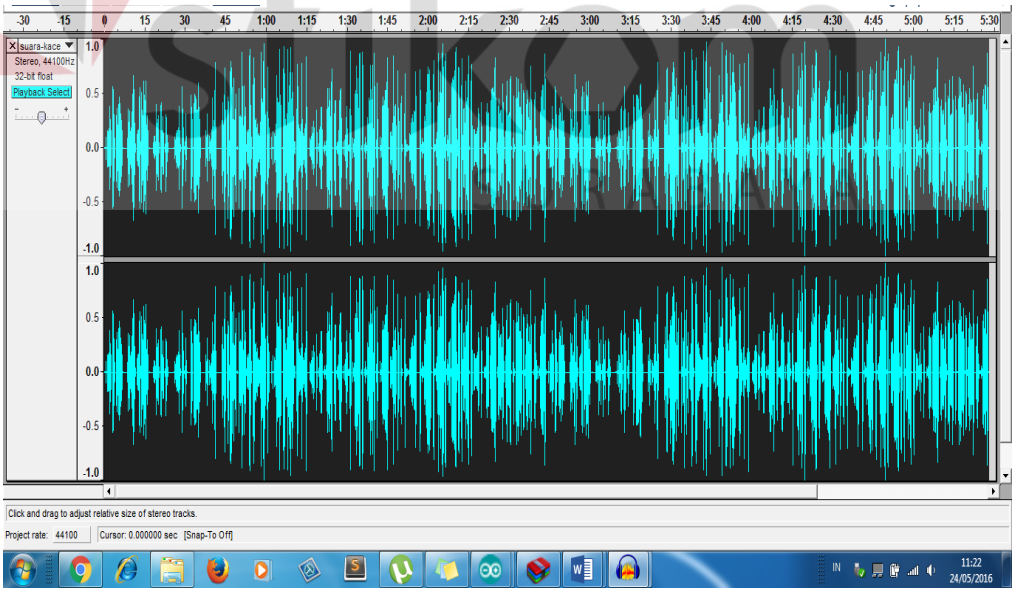


Gambar 3.2 Hasil Rekaman Berupa *File* Ber-*Format* (.Mp3)

File tersebut kemudian diubah atau dipotong secara manual dengan menggunakan perangkat lunak *Audacity* untuk menghilangkan rekaman kosong. Contoh tampilan *software Audacity* dapat dilihat pada gambar Gambar 3.3.



Gambar 3.3 Tampilan *Software Audacity*



Gambar 3.4 Tampilan Sinyal Suara Burung Kacer Dalam Domain Waktu

INSTITUT BISNIS
& INFORMATIKA
stikom
SURABAYA

INSTITUT BISNIS
& INFORMATIKA
stikom
SURABAYA

Seperti yang dijelaskan, bahwa setelah data dipotong, maka sinyal hasil *crop* tersebut disimpan menjadi *file* berformat (.wav) dengan *sample type* 44100 Hz Mono, 32-bit. Panjang waktu rekaman setelah *crop* hanya dibatasi sampai 10 detik. Sehingga untuk penelitian ini terdapat 60 *file* berformat (.wav). 30 *sample* untuk file burung Kacer dan 30 *sample* untuk burung Kenari.

3.6 Ekstraksi Ciri

Pada blok ini, *file* suara yang sudah diubah akan dianalisa menggunakan perangkat lunak *Matlab* untuk mendapatkan nilai STFT, nilai STFT adalah nilai perhitungan transformasi *fourier* yang di kalikan dengan fungsi window yang menunjukan nilai dalam domain waktu dan frekuensi. Adapun alur proses digambarkan seperti pada *flowchart* sederhana pada gambar 3.8 dibawah ini :

INSTITUT BISNIS
& INFORMATIKA
stikom
SURABAYA

nfft = 2*wlen; Banyaknya poin FFT.

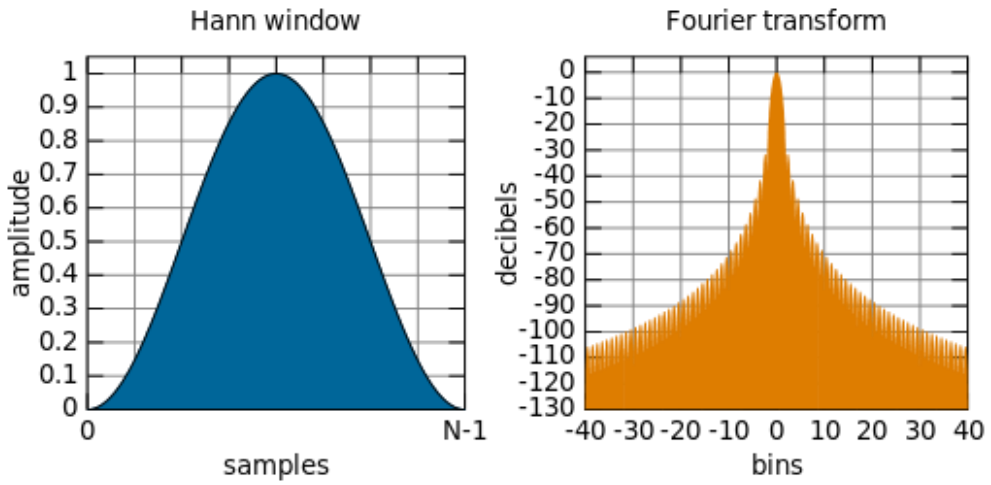
Adapun nilai wlen dan h adalah bilangan kelipatan 2 berpangkat (*power of 2*) untuk memudahkan perhitungan. Dalam hal ini, *window length* ditentukan sebesar 2048. Nilai tersebut dipilih untuk mendapatkan resolusi frekuensi dan resolusi waktu yang bagus. Resolusi frekuensi yang bagus adalah resolusi dengan jarak antar *window* yang lebar, artinya *range* frekuensi semakin kecil dan memberikan nilai yang semakin akurat. Namun dengan lebarnya *window*, maka resolusi waktu pun menjadi tidak bagus. Untuk lebih jelasnya dapat dilihat pada pembahasan di bab sebelumnya. Oleh karena itu diperlukan nilai dari *window length* yang dapat mengoptimalkan *range* dari kedua resolusi.

Adapun jenis *window* yang digunakan pada penelitian ini adalah *Hann* atau *Hanning window* dimana nilai koefisiennya bisa didapatkan dengan rumus berikut.

$$w(n) = \frac{1}{2}\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \quad (1).$$

atau

$$w(n) = sin^2\left(\frac{\pi n}{N-1}\right) \quad (2).$$

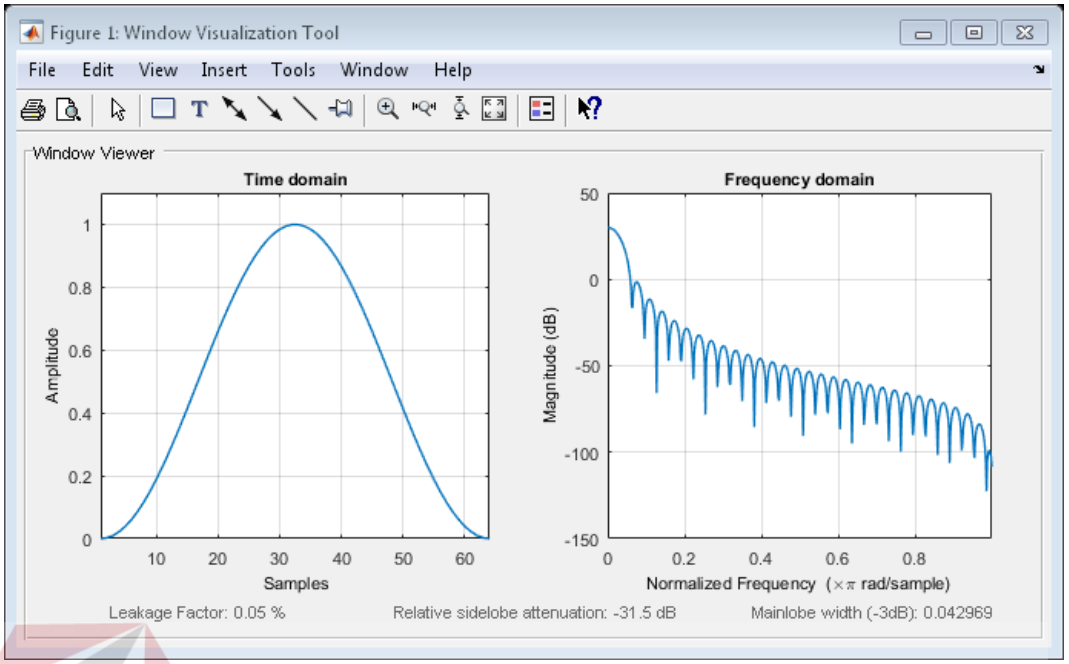


Gambar 3.9 Fungsi Hann (Kiri), dan Respon Frekuensinya (Kanan)

Hann window atau juga disebut jendela kosinus yang ditinggikan biasanya dipakai sebagai fungsi *window* dalam pemprosesan sinyal digital untuk menjalankan transformasi *fourier* dimana ujung dari kosinus menyentuh nilai nol, sehingga *side-lobe* berada pada 18 dB per oktaf.. Adapun keunggulan dari *hann window* adalah sangat rendahnya artifak distorsi atau *aliasing* dan lebarnya *main-lobe* (*lobe* dimana di terdapat energi maksimal).

Pada aplikasi *Matlab*, fungsi *hann window* dapat ditulis dengan sintaks w = hann(L), dimana L adalah panjang dari *window tersebut*

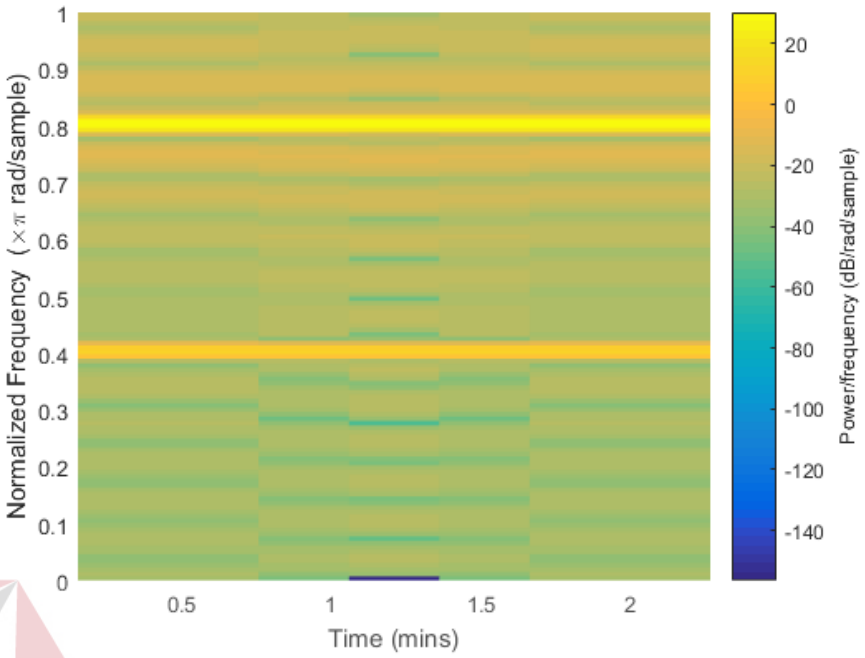
Berikut adalah contoh visualisasi 64-point *hann window* pada *Matlab* dengan menggunakan *Window Visualisation Tool.* Dengan sintaks sebagai berikut: L = 64; wvtool(hann(L))



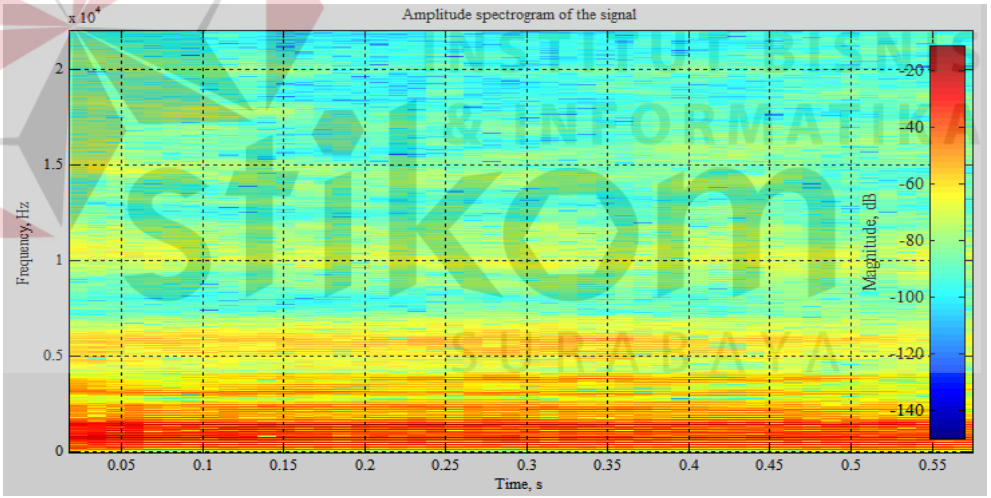
Gambar 3.10 Visualisasi 64-Point *Hann Window* Pada *Matlab*

Kemudian setelah menentukan *window*, maka dapat dilakukan perhitungan STFT dimana pada *Matlab* dapat dibuat sebuah fungsi perhitungan dengan sintaks [stft, f, t] = stft(x, wlen, h, nfft, fs), dimana matriks STFT akan merepresentasikan waktu pada kolom (sumbu x), dan frekuensi pada baris (sumbu y) dalam sebuah spektrogram atau sebuah grafik yang memberikan informasi tentang perubahan gelombang dalam rentang waktu, frekuensi, dan intensitas amplitudo. Intensitas amplitudo pada suatu frekuensi dan pada suatu waktu (waktu, frekuensi) di dalam spektrogram dinyatakan dengan nilai warna tertentu (*grayscale* atau RGB).

Spektrogram pada *Matlab* pada umumnya dapat dibuat dengan sintaks s=spectrogram(x,window,noverlap,nfft).



Gambar 3.11 Contoh Grafik Spektrogram Pada *Matlab*



Gambar 3.12 Contoh Spektrogram Pada Penelitian Ini

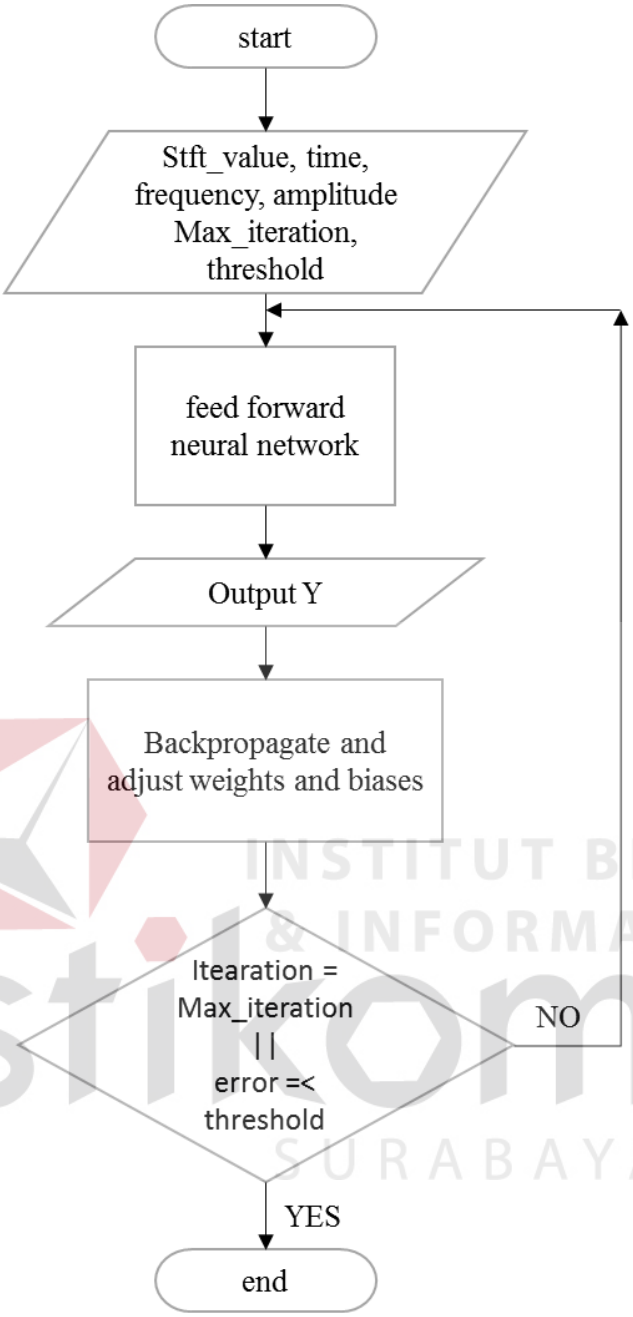
Gambar diatas mencontohkan bentuk spektrogram pada umumnya dimana sumbu x merepresentasikan waktu, sumbu y merepresentasikan frekuensi, dan intensitas amplitudo atau energi (umumnya dalam dB) dinyatakan dalam spektrum

warna tertentu. Pada spektrogram, panjang *window* dapat mempengaruhi intensitas warna dan resolusi dari waktu dan frekuensi.

3.7 Pengujian

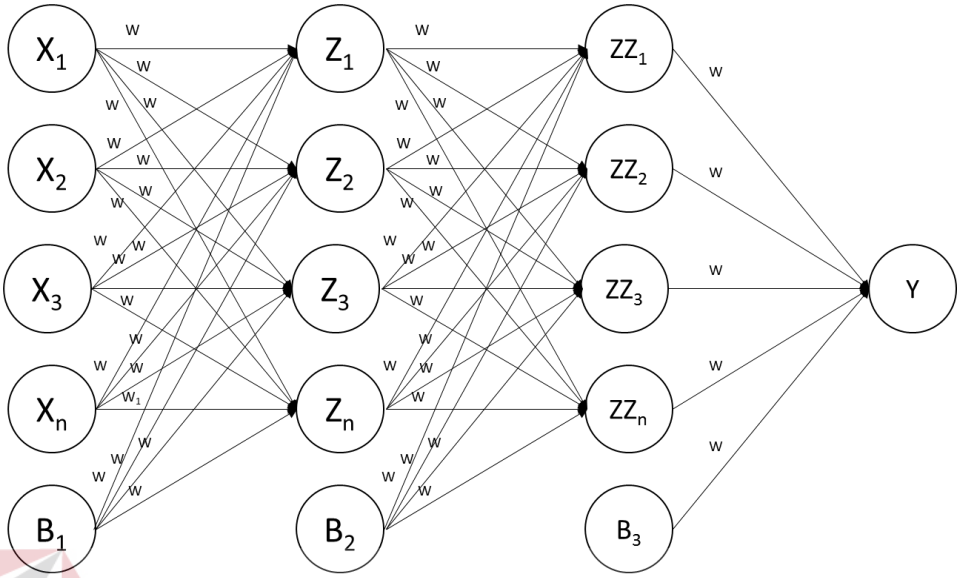
Setelah dilakukannya pembuatan *coding* STFT dengan menggunakan program MATLAB 2013, maka masing–masing data yang telah dikonversi menjadi file ber-ekstensi .wav dimasukkan kedalam *coding* STFT tersebut untuk mendapatkan karakteristik dari suara burung yang berupa nilai STFT, amplitudo tertinggi, waktu saat amplitudo tertinggi, dan frekuensi saat amplitudo tertinggi. Pengujian yang dilakukan selanjutnya adalah pengujian dengan metode jaringan syaraf tiruan.

Pengujian yang dilakukan adalah menggunakan metode jaringan saraf tiruan *backpropagation* dengan input nilai STFT, nilai frekuensi dan waktu saat amplitude tertinggi, dan nilai rata-rata amplitudo pada sebuah sinyal suara. Nilai tersebut berjumlah sebanyak 2 untuk suara burung kacer, dan banyaknya rekaman adalah sebanyak 60 (kacer 30 dan kenari 30). Seluruh hasil keluaran dari ekstraksi ciri akan di normalisasi terlebih dahulu. Agar tidak terdapat data yang nilainya terlalu besar dari data lainnya. Adapun setiap masukan dilatih terlebih dahulu dengan nilai target yang sudah ditentukan untuk mendapatkan keluaran yang diinginkan yaitu status burung kacer atau burung kenari. Seperti pada gambar alur di bawah.



Gambar 3.13 Diagram Alur Proses *Backpropagation*

Arsitektur *backpropagation* yang akan dibuat adalah sebagai berikut:



Gambar 3.14 Arsitektur *Backpropagation* Pada Penelitian Ini

Banyaknya *neuron* pada kedua *hidden layer* didapatkan dari hasil *trial and error*. Hasil *trial and error* ini adalah untuk mendapatkan bobot yang tepat dan dapat memilah antara suara burung kacer atau suara burung kenari.

Nilai awal bobot B1, B2, B3, dan nilai W pada masing masing *layer* adalah nilai random dari -1 sampai 1. Nilai target adalah nilai 1 sebanyak 20 (untuk input kacer), dan nilai 0 sebanyak 20 (untuk input kenari).

Pada gambar di atas, *input* dari ekstraksi ciri diwakili oleh x_{1i}, x_{2i}, x_{3i}, ..., x_{ni} dimana n berjumlah sebanyak 20, Sedangkan x_i berjumlah sebanyak 30 sesuai dengan banyaknya responden. Adapun urutan data responden yang diinputkan adalah 20 burung kacer, kemudian 20 burung kenari. Sedangkan *output* dari bagian ini adalah status mengenali burung kacer ataupun burung kenari pada setiap data rekaman yang diinputkan. Setiap input Xni dan bias B1 akan dikalikan

dengan Wi pada layer input, dan kemudian hasilnya akan menjadi input bagi masing-masing neuron Zi, begitu juga dengan bias dan diulang sebanyak (15). Kemudian hasil inputan pada Zi akan diaktivasi dengan fungsi sigmoid biner untuk menghasilkan nilai antara 0 sampai 1.

Setelah masing-masing neuron diaktivasi, maka output dari Zi akan menjadi input ZZi. Untuk melanjutkan perhitungan ke *hidden layer 2*, dan output, maka output dari Zi dan juga bias B2 dikalikan pada W pada *hidden layer* 1 untuk dimasukkan pada *hidden layer 2* (ZZi). Masukkan dari *Hidden layer* 1 (Zi) nantinya akan diaktivasi dahulu sebelum menjadi input ZZi. Input pada ZZi nantinya akan digunakan untuk menghitung nilai keluaran pada alur maju dengan cara mengalikan ZZi dan bias B3 dengan bobot Wi pada *hidden layer 2*. Hasil dari perkalian ini akan menghasilkan nilai output Y yang nilainya akan diaktivasi untuk perhitungan alur mundur.

Setelah menghasilkan keluaran Y yang telah diaktivasi, maka akan dicari selisih (*error*) dari target awal dengan Y, dan kemudian menghitung koreksi bobot dan bobot bias dan mengubah bobot garis yang berhubungan langsung dengan unit keluaran. Dengan cara yang sama, dihitung faktor δ di setiap unit di *hidden layer* sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di layar di bawahnya. Demikian seterusnya hingga semua faktor δ di unit *hidden* yang berhubungan langsung dengan unit masukan dihitung. Umumnya kondisi penghentian yang dipakai adalah jumlah iterasi atau kesalahan.

Nilai akhir keluaran pada setiap data inputan suara nantinya akan di saring atau di beri *threshold*, dimana nantinya nilai sama dengan atau lebih besar dari 0.5 akan dianggap 1, dan nilai lebih kecil dari 0.5 akan dianggap 0.