# RANCANG BANGUN APLIKASI PRESENSI DENGAN MEDIA SUARA MENGGUNAKAN MFCC DAN ANN BERBASIS ANDROID

**Asterik Rafael Winoto**

*Teknik Informatika Universitas Ma Chung, Villa Puncak Tidar N-1 Malang email:*
*311610002@student.machung,ac.id*

***Abstrak***

*Suara manusia memiliki keragaman masing-masing di tiap individu sehingga dapat digunakan sebagai media presensi. Dengan teknologi Mel-Frequency Cepstral Coefficient dan Artificial Neural Network berbasis Android diharapkan dapat membuat sebuah presensi suara yang mudah, aman, dan murah. Dataset dikumpulkan dari 6 pegawai toko Jersey Bike Malang dengan 50 data suara per pegawai sepanjang 3 detik. Data suara diproses oleh MFCC untuk memperoleh nilai koefisien Cepstral dan digunakan sebagai Input ANN. Uji coba arsitektur model di lakukan beberapa kali hingga didapatkan 5 model terbaik yang selanjutnya digunakan untuk uji coba akurasi terhadap tambahan suara Noise. Hasil dari uji coba dipilih 1 model arsitektur yang memiliki akurasi Testing tertinggi yaitu dengan 5 Hidden Layer dengan jumlah Node sebanyak 1000 dan 180 sesuai dengan urutan Layer dengan 10 data Noise yang menghasilkan persentase akurasi Train sebesar 91.6 % dan persentase akurasi*
*Test sebesar 91.6%*

***Kata kunci:***
*Presensi, Mel-Frequency Cepstral Coefficient, Artificial Neural Network, Android.*

***Abstract***

*Human voice has a diversity of each in each individual so that it can be used as a presence media. With Android-Based MelFrequency Cepstral Coefficient technology and Artificial Neural Networks, it is expected to produce easy, safe, and inexpensive sounds. The dataset was collected from 6 Jersey Bike Malang shop employees with 50 voice data per employee for 3 seconds. Voice data is processed by the MFCC to obtain the Cepstral coefficient and is used as ANN Input. Model architectural trials were carried out several times to obtain the 5 best models which will then be used for additional sound verification verification trials. The results of testing select 1 architectural model that has the highest accuracy of Testing, namely 5 Hidden Layers with the number of Nodes as many as 1000 and 180 in accordance with the order of the Layer with 10 Noise data, which results in a Train accuracy percentage of 91.6% and can be tested correctly amounted to 91.6%.*

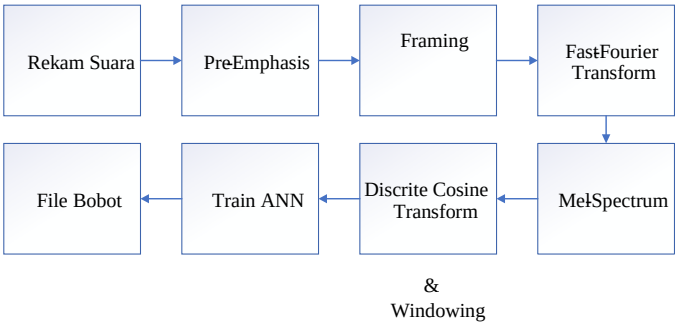
***Keywords:***
*Presence, Mel-Frequency Cepstral Coefficient, Artificial Neural Networks, Android.*

## 1. PENDAHULUAN

Presensi adalah sebuah dokumen kehadiran yang sering sekali digunakan di banyak kegiatan seperti sekolah dan tempat kerja. Mesin presensi dibuat untuk memudahkan perhitungan presensi karyawan dibanding metode penulisan tangan, memantau kedatangan dan kepulangan karyawan, serta menorganisir semua catatan data dan detailnya dalam satu sistem yang terintegrasi (Megawangi, 2019). Proses pengambilan data presensi harus mudah dan juga memiliki penyimpanan yang aman. Namun mesin presensi memiliki harga yang cukup tinggi untuk pemilik usaha tingkat menengah kebawah sehingga perlu sebuah teknologi yang berbiaya terjangkau sehingga dapat memenuhi kebutuhan metode presensi yang baik. Suara manusia yang berbeda-beda di tiap individu dapat dijadikan sebagai sebuah media untuk kegiatan presensi. Teknologi yang dapat digunakan yaitu teknologi kecerdasan buatan. Teknologi kecerdasan buatan adalah ilmu komputer yang mempelajari cara bagaimana perangkat mesin atau komputer bisa bekerja seperti manusia (Kusumadewi & Purnomo, 2010). *Machine Learning* adalah ilmu untuk membuat komputer belajar dan bertindak seperti manusia, dan meningkatkan pembelajaran mereka dari waktu ke waktu secara otonom, dengan memberi mereka data dan informasi dalam bentuk pengamatan dan interaksi dunia nyata. (Faggella, 2019). Dengan teknologi *Mel-Frequency Cepstral Coefficient* dan *Artificial Neural Network* berbasis *Android* diharapkan dapat membuat sebuah presensi suara yang mudah penggunaanya dan aman penyimpanannya dengan biaya yang relatif murah.

## 2. METODE PENELITIAN

Pada penelitian ini dilakukan pengambilan data suara terlebih dahulu dengan mengumpulkan 50 data suara per pegawai dari toko Jersey Bike Malang yang sejumlah 6 orang pegawai. Panjang tiap rekaman suara adalah sepanjang 3 detik dengan format *.wav* menggunakan *Smartphone Xiaomi Redmi Note 5*. Data suara lalu diubah menjadi bentuk koefisien Cepstral dengan melewati proses MFCC yang hasilnya digunakan sebagai input ANN. Bagan dari proses rekam suara hingga menghasilkan *File* Bobot *.h5* terdapat pada gambar 1 berikut ini.




Gambar 1 Proses Pengenalan Suara Pegawai

**2.1 Pengertian *Mel-Frequency Cepstral Coefficient***
MFCC bertugas untuk mengubah sinyal suara menjadi sebuah vektor koefisien cepstral dengan melalui prosesproses filtering yang memperhatikan sensitivitas pendengaran manusia (Bäckström, 2019). MFCC sering digunakan sebagai ekstraksi ciri dari sinyal suara dalam penelitian bidang *Speech Recognition*. Berikut merupakan proses dalam MFCC.

**2.1.1 *Pre-Emphasis***
Pre-Emphasis adalah proses untuk mempertahankan atau menguatkan sinyal frekuensi yang lebih tinggi sehingga mengurangi sinyal yang lebih lemah atau sering disebut *Noise*. Pre-Emphasis menggunakan persamaan sebagai berikut:

$$y(n) = x(n) - ax(n-1), 0.9 \leq \alpha \leq 2.0$$

$y(n)$ = Hasil *Filtering*
$x(n)$ = Sinyal input
$a$ = Konstanta filter Pre-Emphasis
$n$ = Waktu

**2.1.2 *Framing***
*Framing* merupakan proses pembagian sinyal suara menjadi beberapa *Frame* pendek. Sinyal suara selalu berubah seiring dengan waktu sehingga mustahil untuk mengekstrak fitur dengan sekaligus dan perlu adanya batasan-batasan agar tidak terjadi distorsi.

**2.1.3 *Windowing***
*Frame* yang telah dibuat diletakkan dengan posisi tumpang tindih dan selang waktu yang harus lebih kecil dari panjang Frame itu sendiri sesuai dengan fungsi Window – nya. Tujuan dari proses ini agar sinyal suara yang ada di Frame – Frame tersebut tetap kontinyu dan tidak ada informasi yang hilang dari awal sampai akhir. Windowing menggunakan persmaan sebagai berikut.

$$Y(n) = X(n) * W(n)$$

$Y$ = Hasil *Windowing*
$X$ = Sinyal *Input*
$W$ = Fungsi *Window*

Fungsi *Window* yang digunakan adalah fungsi *Hamming*.

$$W(n) = 0.54 - 0.46cos(2\pi nM - 1),\ 0 \leq n \leq M - 1$$

$n$ = Indeks frame, rentang 0 sampai $M$ -1
$M$ = Jumlah sampel *Frame*
$W$ = Hasil fungsi *Window*

**2.1.4 Fast Fourier Transform**
*Fast Fourier Transform* adalah sebuah metode *Fourier Transform* yang digunakan untuk memperoleh nilai frekuensi yang terkandung pada sebuah sinyal suara. FFT sendiri merupakan pengembangan dari *Discrite Fourier Transform*. FFT menggunakan persamaan sebagai berikut:

$$X(k) = \sum_{n=0}^{N-1} X(n)\ e^{-2\pi jkn/N}$$

$X(k)$ = Nilai frekuensi
$X(n)$ = Nilai sinyal input
$N$ = Jumlah sampel sinyal
$n$ = Variabel sinyal input
$k$ = Variabel frekuensi

Sinyal input dibagi menjadi 2 bagian yaitu urutan ganjil dan genap dengan rumus seperti berikut:

$$X(k) = Xganjil(k) + e^{-\frac{2\pi jkn}{N}} Xgenap(k)$$

***2.1.5 Mel-Spectrum***

Tiap *Pitch* dari frekuensi yang dihasilkan oleh sinyal suara akan diukur ke dalam bentuk skala *Mel* dengan rumus berikut:

$$fmel = 2595\ log_{10}\left(1 + \frac{f}{700}\right)$$

$fmel$ = Sinyal Skala Mel
$f$ = Nilai sinyal Frekuensi

Untuk mengembalikan nilai skala Mel ke dalam nilai frekuensi, bisa digunakan rumus berikut ini:

$$f(m) = 700\ (\ 10\mathrm{m} / 2595 - 1\ )$$

$f(m)$ = Nilai Sinyal Frekuensi
m = Nilai skala Mel

Frekuensi yang telah disesuaikan dengan skala Mel selanjutnya dipakai dalam mencari nilai Spektrum. Spektrum berupa angka bin FFT sehingga persamaan dari *Filterbank* adalah seperti pada rumus berikut:

$$s(i) = floor[(nfft + 1)\frac{f(i)}{fs}]$$

$s(i)$ = Nilai Spektrum
$f(i)$ = Nilai sinyal Frekuensi
$fs$ = *Sample Rate*
$nfft$ = Jumlah filter pada *Filterbank*

**2.1.6 *Discrite Cosine Transform***
Nilai Spektrum yang dihasilkan selanjutnya diubah kedalam bentuk Cepstrum untuk diambil nilai koefisien Cepstral dengan persamaan berikut:

$$c(n) = \sum_{n=0}^{N-1} log_{10} s(m)\cos(\frac{\pi n(m-0.5)}{M});\ \mathrm{n}=0,1,2,\ldots.\mathrm{C}\text{-}1$$

$s(m)$ = Nilai Spektrum

$M$ = Jumlah filter *Filterbank*
$C$ = Indeks jumlah MFCC
$m$ = Indeks jumlah *Filterbank* sebanyak *M*
$c$(n) = Nilai koefisien Cepstral

**2.2 Pengertian *Artificial Neural Network*** *Artificial Neural Network* (ANN) adalah sebuah arsitektur komputer yang menyerupai kerja otak. Terdiri dari Node sebagai model yang disederhanakan yang nantinya memberi sinyal input kepada *Node* lain. Sinyal tiap *Node*nya akan berbeda-beda sehingga menjadikan nya sebuah mesin pintar dibandingkan dengan komputer traditional biasa. Proses ANN terbagi menjadi dua yaitu *Forward Pass* dan *Backpropagation.* Forward pass adalah proses mengolah sinyal input dengan bobot yang tersedia pada saat melewati *Hidden Layer* hingga sampai ke *Output Layer*. *Forward Pass* menggunakan persamaan berikut ini:

$$V(n) = \sum( W(n) \times X(n)) + (1 * b)$$

$V(n)$ = Hasil nilai sinyal $W$
$(n)$ = Nilai bobot
$X(n)$ = Nilai sinyal input
$b$ = Nilai bias

Setiap *Layer* memiliki fungsi aktivasi yang berfungsi untuk mengaktifkan atau tidak suatu sinyal. Fungsi aktivasi yang biasa digunakan adalah ReLu seperti pada persamaan berikut ini:

$$Y(x) = max(0, V(n))$$

Pada *Output Layer* digunakan fungsi aktivasi *Softmax* untuk menghitung probabilitas dari kelas target dengan menggunakan persamaan berikut:

$$Softmax(x_i) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

Kemudian hasil *Output* akan di evaluasi dengan metode *Backpropagation* untuk memperbaiki nilai bobot sehingga hasil *Output* dapat mengarah kelas yang tepat. Langkah awal adalah dengan menghitung *Error Rate* dengan persamaan berikut:

$e = d - Y$

$e$ = *Error rate*
$d$ = Hasil yang diinginkan
$Y$ = Hasil yang didapat

Setelah itu dilakukan proses *Backward Pass* yaitu dengan menghitung nilai-nilai *Local Gradient* yang terdapat pada *Hidden Layer*. Proses dimulai dari *Hidden Layer* terluar dengan rumus sebagai berikut:

$$\delta o = \varphi'(x) * (d - Y(x))$$
$$\varphi'(x) = \varphi(x) * (1 - \varphi(x))$$

$\delta o$ = *Local Gradient Output Layer*
$\varphi'(x)$ = Fungsi Aktivasi Invers
$x$ = Indeks Perulangan
$d$ = Hasil yang diinginkan
$Y(x)$ = Fungsi aktivasi

Proses dilanjutkan dengan mencari *Local Gradient* pada *Node Hidden Layer* berikutnya namun dengan rumus yang berbeda karena terdapat bobot yang terletak di antara *Node* berikutnya.

$$\delta h = \varphi'(x) * (\delta o \times W(n))$$

$\delta h$ = *Local Gradient Hidden Layer*
$\varphi'(x)$= Fungsi Aktivasi Invers $W(n)$
= Bobot ke $n$

Perhitungan *Local Gradient* terus berlanjut sampai *Node Hidden Layer* terakhir. Setelah itu dilakukan penyesuaian bobot di tiap jaringan nya dengan rumus berikut:

$$W(n+1) = \mathrm{W}(n) + \alpha * W(n-1) + \eta * \delta(n) * x$$

$W(n+1)$ = Nilai bobot baru
$\mathrm{W}(n)$ = Nilai bobot lama $\alpha$ = *Learning Rate*
$\delta$ = *Local Gradient Node*
$x$ = Jumlah bobot

**2.3 Metode Optimasi *Adaptive Moment Estimation***
*Adaptive Moment Estimation* atau yang biasa disingkat Adam adalah sebuah metode optimasi yang menggabungkan 2 algoritma yaitu *Adaptive Gradient* dan RMSprop. Adam melakukan penyimpanan nilai *GradientGradient* di perhitungan sebelumnya untuk menghitung rata-rata nilai *Gradient* pada momentum pertama maupun momentum kedua. Dari kedua rata-rata tersebut di adaptasi parameter -nya untuk digunakan sebagai pembaharuan nilai bobot.

**3. HASIL DAN PEMBAHASAN**
Dalam penelitian ini, IDLE digunakan sebagai IDE untuk melakukan pemograman bahasa Python3.7 dan dalam pembuatan aplikasi *Mobile* menggunakan *Android Studio*. Proses rekaman suara untuk mendapatkan dataset dilakukan di aplikasi *Recording Smartphone Xiaomi Redmi Note 5*. Rekaman suara akan di *Load* dengan *Sampling Rate* sebesar 44100 Khz dan berduarasi 3 detik sehingga saat melalui proses MFCC dihasilkan matriks berukuan 20 x 259 yang selanjutnya diubah ke dalam bentuk matriks 1x5180 dan dimasukkan kedalam *File* berformat *.csv*. Sebelum proses *Training* ANN dataset di bagi dengan perbandingan 80:20 untuk data *Train* dan data *Test*. Selanjutnya dilakukan *Training* ANN dengan 4 jenis percobaan yaitu dengan data *Train* tanpa tambahan suara *Noise*, dataset dengan tambahan dataset sejumlah 10 *Noise*, dataset dengan tambahan dataset sejumlah 20 *Noise*, dan dataset dengan tambhan dataset sejumlah 30. *Noise* ditambahkan pada saat proses MFCC dengan pengaturan jumlah sesuai dengan uji coba yang dilakukan. Khusus untuk data *Test* di tiap kelas memiliki 2 macam data yaitu 5 data tanpa noise dan 5 data dengan tambahan *Noise* untuk melihat tren *Error* saat proses prediksi setelah proses *Train* selesai. Dalam proses *Training* ANN *Epoch* yang digunakan sebanyak 8000 untuk mendapatkan hasil akurasi yang lebih tinggi. Output Layer akan berisi 6 *Class* atau

pegawai. Untuk menentukan model yang paling disarankan di tiap uji coba maka dilakukan uji signifikansi dengan uji *One-Way* ANOVA, homogenitas varian, serta Post-Hoc. Setelah mendapatkan model yang paling disarankan di tiap uji coba maka akan dilakukan kembali pencarian model terbaik dari keempat model yang telah dipilih dengan menggunakan uji signifikansi yang sama. Bila terdapat lebih dari 1 model memiliki hasil rata-rata tertinggi dibanding model-model yang lain serta tidak terdapat perbedaan yang signifikan pada model tersebut maka model terbaik akan dipilih dengan melihat ukuran *File h5* yang terkecil karena aplikasi ini menggunakan Android dan *Web Service* sehingga perlu sebuah sistem yang tidak berukuran besar agar proses presensi bisa lebih optimal. Sebelumnya dibuat 5 grup pada tiap model yang berisi data Noise dan data suara tanpa Noise yang dibagi secara seimbang di tiap grup nya yang kemudian dilakukan klasifikasi suara terhadap data yanng ada pada tiap grup tersebut. Jumlah prediksi benar di tiap grup kemudian di catat untuk selanjutnya digunakan sebagai bahan uji coba signifikansi. 2 pernyataan yang mengawali uji signifikansi tersebut sebagai berikut.

$H_0$ : Akurasi Model 1 =Akurasi Model 2 = Akurasi Model 3 = Akurasi Model 4 = Akurasi Model 5 $H_1$ : Setidaknya satu model yang memiliki perbedaan signifikan

**3.1 Hasil Uji Coba dengan Data Tanpa Noise**

Percobaan pertama dilakukan dengan menggunakan data *Train* tanpa tambahan suara *Noise.* Dalam percobaan ini dilakukan repetisi berulang kali sehingga ditemukan 5 model arsitektur ANN dengan akurasi *Train* dan *Test* terbaik. Hasil akurasi Train dari percobaan pertama ini menghasilkan persentase yang cukup baik terutama pada 5
model arsitektur terbaik yang telah dipilih yaitu dapat mencapai lebih dari 90 % dengan nilai tertinggi nya yaitu 94.58%. Namun pada akurasi Test terlihat bahwa persentase tertinggi hanya berkisar 83.3%. Hal ini diakibakan dari tidak adanya data suara *Noise* pada data *Train* sehingga sering terjadi salah prediksi pada saat memprediksi suara yang telah ditambah *Noise*. Nilai Recall di beberapa Class pada 5 model tersebut memiliki persentase yang cukup rendah seperti contoh pada Class 0 yang hanya mencapai persentase tertinggi sebanyak 50%. Sama halnya dengan nilai *Recall*, nilai *Precision* juga rendah di beberapa *Class* yang menandakan hasil prediksi masih cukup kurang presisi. Hasil dari uji coba dapat dilihat pada tabel 1.1 dibawah ini.

Tabel 1.1 Hasil Pengujian Data Suara Tanpa Noise

| Model | Jumlah *Node* | Akurasi *Train* | Akurasi *Test* | *Recall* (*Class* 0, 1, 2, 3, 4, 5) | *Precision* (*Class* 0, 1, 2, 3, 4, 5) |
|---|---|---|---|---|---|
| 1 | 1000, 180 | 91.66 % | 80 % | 50 %, 90 %, 70 %, 90 %, 80 %,100 % | 83 %, 69 %, 100 %, 60 %, 89 %, 100 % |
| 2 | 1000, 180, 50 | 92.08 % | 83.3 % | 40 %, 80 %, 80 %, 100 %, 100 %, 100 % | 67 %, 67 %, 100 %, 77 %, 100 %, 91 % |
| 3 | 1000, 180, 90 | 94.58 % | 80 % | 40 %, 100 %, 60 %, 100 %, 80 %, 100 % | 80 %, 62 %, 86 %, 83 %, 100 %, 83 % |
| 4 | 1000, 180, 90, 50 | 94.58 % | 76.67 % | 50 %, 80 %, 90 %, 70 %, 70 %, 100 % | 71 %, 73 %, 69 %, 78 %, 88 %, 83 % |
| 5 | 1000, 180, 90, 50, 20 | 91.25 % | 76.66 % | 40 %, 100 %, 70 %, 80 %, 80 %, 90 % | 50 %, 77 %, 88 %, 89 %, 100 %, 64 % |

Pada grafik gambar 2 terlihat bahwa jumlah terprediksi dari data suara dengan *Noise* cukup banyak pada proses *Training* data Train tidak *Noise* apapun. Namun jumlah bisa dikatakan sedikit

karena dari kelima model ratarata kesalahan prediksi pada data dengan tambahan *Noise* sekitar 9 data. Dapat disimpulkan bahwa dengan metode Pre-Emphasis dalam mengingat bahwa mereduksi suara frekeunsi rendah saja belum cukup sehingga perlu bantuan penambahan suara *Noise* pada data *Train*.


Gambar 2 Grafik Jumlah Prediksi Benar di 2 Tipe Suara pada Uji Coba Training Tanpa Noise

Untuk hasil dari uji coba signifikansi, ditemukan hasil nilai terkecil sehingga didapat model 1 sebagai model yang signifikansi *One-Way* ANOVA lebih dari 0.05 yaitu 0.492 paling disarankan yaitu dengan 20.962 Kb. Perbandingan sehingga pernyataan $H_0$ diterima maka pemilihan model besar ukuran terdapat pada tabel 1.2 berikut ini.
yang paling disarankan menggunakan ukuran *File h5*

Tabel 1.2 Perbandingan Besar File h5 pada 5 Model di Uji Coba Tanpa Data Noise

| Model 1 | Model 2 | Model 3 | Model 4 | Model 5 |
|---|---|---|---|---|
| 20.962 Kb | 20.997 Kb | 21.026 Kb | 21.046 Kb | 21.050 Kb |

**3.2 Hasil Uji Coba dengan Tambahan Data Noise Sebanyak 10**

Percobaan kedua kali ini menggunakan penambahan 10 suara Noise pada data *Train* agar diharapkan dapat mengurangi kesalahan prediksi seperti pada percobaan pertama. Hasilnya adalah akurasi Train mengalami sedikit penurunan namun masih dapat melebihi persentase 90 % dengan nilai tertinggi adalah sebesar 92.5%. Penambahan suara Noise terbukti dapat mengurangi kesalahan prediksi yang tercermin pada persentase akurasi Test yang meningkat cukup besar yaitu dengan persentase tertinggi 91.6%. Hanya di beberapa *Class* di model tertentu seperti pada model ke 4 yang pada *Class* ke 2 memiliki nilai *Recall* yang rendah yaitu sekitar 60 % namun nilai *Precision* nya sudah hampir mencapai persentase 90 % sehingga dalam uji coba kali ini prediksi yang dilakukan jauh lebih baik dari sebelumnya karena tingkat *Precision* dari kelima model sangat tinggi.

Tabel 2.1 Hasil Pengujian Data Suara Tanpa Noise

| Model | Jumlah *Node* | Akurasi *Train* | Akurasi *Test* | *Recall* (*Class* 0, 1, 2, 3, 4, 5) | *Precision* (*Class* 0, 1, 2, 3, 4, 5) |
|---|---|---|---|---|---|
| 1 | 1000, 180 | 92.5 % | 90 % | 90 %, 100 %, 70 %, 100 %, 80 %, 100 % | 90 %, 91 %, % , 83 %, % , 100 % |
| 2 | 1000, 180, 50 | 91.6 % | 91.6 % | 80 %, 100 %, 100 %, 90 %, 90 %, 90 % | 100 %, 91 %, 100 %, 90 %, 75 %, 100 % |
| 3 | 1000, 180, 90 | 92.5 % | 91.6 % | 90 %, 90 %, 80 %, 90 %, 100 %, 100 % | 100 %, 90 %, 89 %, 82 %, 91 %, 100 % |
| 4 | 1000, 180, 90, 50 | 91.66 % | 83.33 % | 90 %, 100 %, 60 %, 90 %, 60 %, 100 % | 75 %, 83 %, 86 %, 75 %, 100 %, 91 % |
| 5 | 1000, 180, 90, 50, 20 | 91.6 % | 91.6 % | 100 %, 90 %, 80 %, 90 %, 100 %, 90 % | 91 %, 82 %, 89 %, 90 %, 100 %, 100 % |

Pada grafik gambar 3 terlihat bahwa jumlah terprediksi menurunkan jumlah terprediksi benar dari data tanpa benar dari kedua macam data tidak terpaut jauh seperti di *Noise*. Hanya pada model ke 4 yang mengalami penurunan uji coba sebelumnya. Ini menandakan penambahan 10 data jumlah terprediksi benar pada data suara tanpa *Noise* yang *Noise* pada data Train berhasil

mengenali data suara diperkirakan disebabkan oleh ketidakcocokan arsitektur dengan *Noise* dengan lebih baik dan juga tidak model sehingga akurasinya agak menurun.


Gambar 3 Grafik Jumlah Prediksi Benar di 2 Tipe Suara pada Uji Coba Training dengan Jumlah Noise Sebanyak 10

Untuk hasil dari uji coba signifikansi, ditemukan hasil nilai signifikansi kurang dari 0.05 yaitu 0.000038 sehingga pernyataan $H_1$ diterima namun 2 model yaitu model 1 dan 4 yang memiliki perbedaan signifikan emiliki rata-rata jumlah prediksi benar yang lebih rendah dari 3 model lainnya sehingga pemilihan model yang paling disarankan menggunakan ukuran File h5 terkecil dari 3 model tersebut sehingga didapat model 2 sebagai model yang paling disarankan yaitu dengan 20.997 Kb. Perbandingan besar ukuran terdapat pada tabel 2.2 berikut ini.

Tabel 2.2 Perbandingan Besar File h5 pada 3 Model di Uji Coba Dengan 20 Data Noise

| Model 2 | Model 3 | Model5 |
|---|---|---|
| 20.997 Kb | 21.026 Kb | 21.050 Kb |

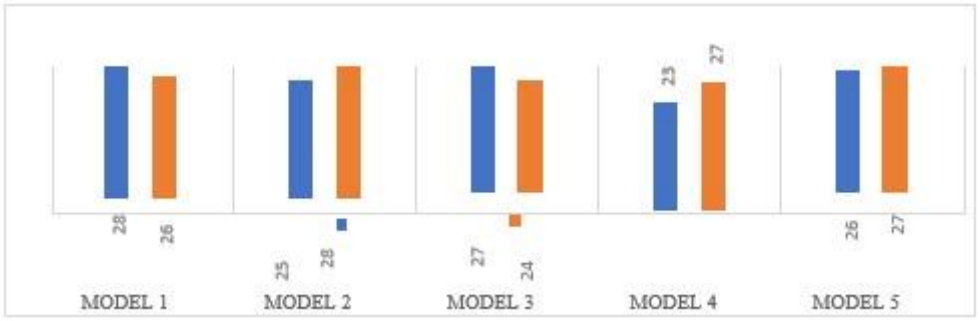
**3.3 Hasil Uji Coba dengan Tambahan Data Noise Sebanyak 20**

Dari Tabel 3.1 terlihat bahwa penambahan 20 data *Noise* ke dalam data Train mempengaruhi persentase *Recall* di beberapa model seperti contoh pada model ke 4 yang mengalami penurunan di hampir semua *Class*. Penambahan tersebut mempengaruhi akurasi prediksi pada data tanpa *Noise* yang semula hampir tanpa kesalahan prediksi namun kali ini terjadi peningkatan kesalahan prediksi. Namun walaupun hampir seluruh model mengalami penurunan prediksi namun penurunan tersebut tidak signifikan dan masih tergolong cukup baik.

Tabel 3.1 Hasil Pengujian Data Suara Tanpa Noise

| Model | Jumlah *Node* | Akurasi *Train* | Akurasi *Test* | *Recall* (*Class* 0, 1, 2, 3, 4, 5) | *Precision* (*Class* 0, 1, 2, 3, 4, 5) |
|---|---|---|---|---|---|
| 1 | 1000, 180 | 89.58 % | 90 % | 100 %, 90 %, 90 %, 100 %, 60 %,100 % | 71 %, 100 %, 100 %, 83 %, 100 %, 100 % |
| 2 | 1000, 180, 50 | 89.9 % | 88.3 % | 100 %, 80 %, 70 %, 90 %, 90 %, 100 % | 77 %, 89 %, 88 %, 82 %, 100 %, 100 % |
| 3 | 1000, 180, 90 | 91.25 % | 85 % | 100 %, 70 %, 80 %, 100 %, 70 %, 90 % | 67 %, 78 %, 100 %, 83 %, 100 %, 100 % |
| 4 | 1000, 180, 90, 50 | 92.08 % | 83.33 % | 70 %, 80 %, 80 %, 80 %, 90 %, 100 % | 70 %, 89 %, 89 %, 80 %, 75 %, 100 % |
| 5 | 1000, 180, 90, 50, 20 | 92.08 % | 90 % | 100 %, 100 %, 70 %, 80 %, 90 %, 100 % | 83 %, 83 %, 100 %, 89 %, 100 %, 91 % |

Pada grafik gambar 4 terlihat bahwa di beberapa model terjadi penurunan jumlah terprediksi benar pada data tanpa *Noise* namun sebaliknya pada data dengan tambahan *Noise* jumlahnya cenderung tetap dan sama di tiap modelnya. Hal ini menyebabkan penurunan akurasi *Test* di hampir semua model.




Gambar 4 Grafik Jumlah Prediksi Benar di 2 Tipe Suara pada Uji Coba Training dengan Jumlah Noise Sebanyak 20

Untuk hasil dari uji coba signifikansi, ditemukan hasil nilai terkecil sehingga didapat model 1 sebagai model yang signifikansi *One-Way* ANOVA lebih dari 0.05 yaitu 0.482 paling disarankan yaitu dengan 20.962 Kb. Perbandingan sehingga pernyataan $H_0$ diterima maka pemilihan model besar ukuran terdapat pada tabel 3.2 berikut ini.
yang paling disarankan menggunakan ukuran *File h5*

Tabel 3.2 Perbandingan Besar File h5 pada 5 Model di Uji Coba Tanpa Data Noise

| Model 1 | Model 2 | Model 3 | Model 4 | Model 5 |
|---|---|---|---|---|
| 20.962 Kb | 20.997 Kb | 21.026 Kb | 21.046 Kb | 21.050 Kb |

**3.4 Hasil Uji Coba dengan Tambahan Data Noise Sebanyak 30**

Pada Tabel 4.1 di atas terlihat adanya kenaikan dan penurunan akurasi *Test* dari kelima model seperti contoh pada model ke 1 dan ke 2 mengalami penurunan dan model ke 3, model ke 4, dan model ke 5 mengalami kenaikan akurasi bahkan model ke 5 memiliki akurasi *Test* tertinggi dari seluruh uji coba yang telah dilakukan yaitu sebesar 93.3 %. Persentase *Recall* juga sedikit lebih baik dari uji coba sebelumnya dengan banyak nya persentase 100 % dan 90 % di tiap model. Namun ada penurunan persentase pada *Precision* di beberapa model bahkan pada model ke 2 *Class* ke 3 memiliki persentas *Precision* sebesar 64 %.

Tabel 4.1 Hasil Pengujian Data Suara Tanpa Noise

| Model | Jumlah *Node* | Akurasi *Train* | Akurasi *Test* | *Recall* (*Class* 0, 1, 2, 3, 4, 5) | *Precision* (*Class* 0, 1, 2, 3, 4, 5) |
|---|---|---|---|---|---|
| 1 | 1000, 180 | 92.08 % | 85 % | 100 %, 90 %, 80 %, 80 %, 60 %, 100 % | 83 %, 90 %, 80 %, 73 %, 86%, 100 % |
| 2 | 1000, 180, 50 | 91.25 % | 86.67 % | 100 %, 90 %, 90 %, 70 %, 70 %, 100 % | 91 %, 100 %, 100 %, 64 %, 78 %, 91 % |
| 3 | 1000, 180, 90 | 90.41 % | 90 % | 90 %, 100 %, 100 %, 90 %, 70 %, 90 % | 90 %, 91 %, 100 %, 82 %, 78 %, 100 % |
| 4 | 1000, 180, 90, 50 | 88.75 % | 86.67 % | 100 %, 90 %, 70 %, 80 %, 80 %, 100 % | 83 %, 100 %, 88 %, 73 %, 80 %, 100 % |
| 5 | 1000, 180, 90, 50, 20 | 95.42 % | 93.3 % | 100 %, 90 %, 80 %, 90 %, 100 %, 100 % | 83 %, 90 %, 100 %, 100 %, 100 %, 91 % |

Pada grafik pada gambar 5 terlihat bahwa penurunan jumlah terprediksi benar pada data tanpa *Noise* makin meningkat hampir di semua model. Kenaikan akurasi *Test* di beberapa model juga diakibatkan kenaikan jumlah tersebut adalah yang terbanyak dari semua uji coba yang dilakukan sebelumnya. Hal ini diakibatkan dari penambahan data *Noise* pada data Train yang jumlahnya lebih banyak dari data tanpa *Noise*. Hasilnya hampir

terprediksi benar dari data dengan *Noise* seperti pada model 3 dan 5 yang dapat mencapai angka 29. Jumlah semua model jumlah terprediksi benar dari data tanpa *Noise* lebih sedikit dari data dengan *Noise*.


Gambar 5 Grafik Jumlah Prediksi Benar di 2 Tipe Suara pada Uji Coba Training dengan Jumlah Noise Sebanyak 30

Untuk hasil dari uji coba signifikansi, ditemukan hasil nilai signifikansi *One-Way* ANOVA kurang dari 0.05 yaitu 0.018 sehingga pernyataan $H_1$ diterima dan model 5 memiliki nilai perbedaan signifikan yang paling tinggi sehingga pada uji coba ini disarankan menggunakan model 5.

### 3.5 Hasil Pengujian Sistem Penentuan Model Arsitektur Terbaik

Setelah melakukan uji coba dengan 4 macam jumlah data *Noise* dan menemukan model-model terbaik di tiap uji coba dengan menggunakan uji signifikansi, maka selanjutnya model-model tersebut akan dipilih satu untuk diimplementasikan ke aplikasi presensi suara. Metode yang digunakan adalah dengan uji signifikansi seperti pada uji coba sebelumnya. Model-model yang terpilih adalah model 1 pada uji coba tanpa data *Noise*, model 2 pada uji coba dengan 10 data *Noise*, model 1 pada uji coba tanpa data *Noise*, dan model 5 pada uji coba dengan 30 data *Noise*. Hasil dari uji coba signifikansi, ditemukan hasil nilai signifikansi *One-Way* ANOVA kurang dari 0.05 yaitu 0.0112 sehingga pernyataan $H_1$ diterima dan pada model 2 dari uji coba dengan 10 data *Noise* didapat nilai ANOVA kurang dari 0.05 yaitu 0.0489 sehingga model inilah yang akan diimplementasi ke aplikasi presensi suara berbasis Android.

## 4. KESIMPULAN

### 4.1 Kesimpulan

Kesimpulan dari hasil yang didapat bedasarkan uji coba terhadap metode MFCC dan ANN pada aplikasi presensi menggunakan suara adalah sebagai berikut:

1. Metode *Mel-Frequency Cepstral Coefficient* dan *Artificial Neural Network* dapat mengenali suara pegawai dengan baik yaitu dapat memprediksi suara pegawai dengan akurasi *Training* sebesar 91.6 % dan akurasi *Testing* sebesar 91.6 % pada arsitektur model dengan jumlah *Hidden Layer* sebanyak 5 yaitu dengan Layer pertama berisi 1000 *Node*, dan *Layer* kedua berisi 180 *Node* dengan menggunakan data Noise pada data Train sebanyak 10 data sehingga dapat digunakan untuk proses presensi pegawai.

2. Penambahan data *Noise* pada data *Train* sangat mempengaruhi peningkatan hasil akurasi pada data Test dibandingkan dengan tanpa menggunakan suara *Noise*.

3. Semakin banyak jumlah *Noise* yang digabungkan kedalam data *Train* juga meningkatkan akurasi prediksi pada data *Test* terutama pada data *Test* yang telah ditambahkan data *Noise*.

4. Aplikasi *Mobile* dibantu dengan *Web Service* dapat menjalankan aplikasi pengenalan suara yang terdapat proses MFCC dan ANN di dalamnya dengan cepat yaitu dengan waktu 0.4 sampai 1 detik.

5. *Firebase* dapat mengurangi kecurangan manipulasi data presensi dan memudahkan penyimpanan data presensi.

### 4.2 Saran

Penelitian mengenai pengenalan suara sangat diminati pada saat ini dengan berbagai kebutuhan untuk memudahkan pekerjaan manusia sehingga saran dapat membantu perkembangan penelitian mengenai pengenalan suara. Pada penelitian ini ada beberapa perbaikan dan peningkatan sebagai berikut:

1. Perlu adanya penambahan data suara juga suara *Noise* untuk menghindari terjadinya *Overfitting* akibat kurangnya variasi pada data *Test*.

2. Perlu adanya suatu metode *Filter* yang lebih baik sehingga dapat meningkatkan akurasi dalam pengenalan suara.

3. Perlu adanya uji coba pengenalan suara terhadap pengaruh jumlah *Class* yang banyak dengan akurasi yang didapat.

4. Perlu adanya penambahan status sakit atau ijin pada daftar presensi.

**5. REFERENSI**

1. Bäckström T 2019, *Cepstrum and MFCC*. Diakses pada 24 Februari 2020, <https://wiki.aalto.fi/display/ITSP/Cepstrum+and+MFCC>.

2. Faggella D 2019, *What is Machine Learning?* Diakses pada 27 Februari 2020, <https://emerj.com/ai-glossary-terms/what-is-machinelearning>.

3. Kusumadewi S dan Purnomo H 2010, *Aplikasi Logika Fuzzy untuk Pendukung Keputusan*, Graha Ilmu, Yogyakarta.

4. Megawangi S R 2019, *Mesin Absensi Karyawan: Sejarah, Tujuan, Manfaat, Jenis*. Diakses pada 6 Maret 2020, https://jojonomic.com/blog/mesin-absensikaryawan/>.