IDENTIFIKASI EMOSI MELALUI SUARA MENGGUNAKAN *SUPPORT VECTOR MACHINE* DAN *CONVOLUTIONAL NEURAL NETWORK*

Disusun Oleh:

N a m a : Rio Galang Jati Respati

NIM : 17523118

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2021

HALAMAN PENGESAHAN DOSEN PEMBIMBING

IDENTIFIKASI EMOSI MELALUI SUARA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN *SUPPORT VECTOR MACHINE*

TUGAS AKHIR

Disusun Oleh:

N a m a : Rio Galang Jati Respati

NIM : 17523118

Yogyakarta, 10 Maret 2021

Pembimbing,

(Annie Kurniawardhani S.Si. M.Kom.)

HALAMAN PENGESAHAN DOSEN PENGUJI

IDENTIFIKASI EMOSI MELALUI SUARA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN *SUPPORT VECTOR MACHINE*

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 4 Agustus 2021

Tim Penguji

Arrie Kurniawardhani S.Si. M.Kom. _____

Anggota 1
Dr. RM Sisdarmanto Adinandra, S.T., M.Sc. _____

Anggota 2
Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Rio Galang Jati Respati

NIM : 17523118

Tugas akhir dengan judul:

IDENTIFIKASI EMOSI MELALUI SUARA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN *SUPPORT VECTOR MACHINE*

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 10 Maret 2021

(Rio Galang Jati Respati)

HALAMAN PERSEMBAHAN

Assalamualaikum Wr. Wb.

Puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis diberi kemudahan dalam menyelesaikan Laporan Tugas Akhir dengan baik.

Terimakasih atas motivasi, dukungan, dan do'a dari semua pihak yang telah ikut serta dalam penyelesaian pembuatan laporan Tugas Akhir. Penulis mengucapkan banyak terima kasih kepada:

1. Allah SWT, yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan laporan ini dengan baik
2. Keluarga, yang telah memberikan bantuan secara material maupun mental dan do'a yang selalu diucapkan untuk selesainya laporan yang dibuat.
3. Ibu Arrie Kurniawardhani, sebagai dosen pembimbing yang sudah bersusah payah untuk memberikan saran agar laporan yang dihasilkan menjadi lebih baik lagi.
4. Rahma Pangastuti, yang telah membantu saya dalam penulisan agar dapat dimengerti banyak orang.
5. Teman-teman yang telah memberikan waktu luangnya untuk menemani penulis ketika suntuk mengerjakan laporan.

Semoga laporan Tugas Akhir ini dapat bermanfaat dan digunakan sebagai mana mestinya untuk semua bidang.

HALAMAN MOTO

“Sesungguhnya sesudah kesulitan itu ada kemudahan. Maka apabila kamu telah selesai (dari sesuatu urusan), kerjakanlah dengan sungguh-sungguh (urusan) yang lain, dan hanya kepada Tuhanmulah hendaknya kamu berharap.”

(Q.S Al-Insyirah : 5-8)

“Dan Dia menundukkan apa yang ada di langit dan apa yang ada di bumi untukmu semuanya (sebagai rahmat) dari-Nya. Sungguh, dalam hal yang demikian itu benar-benar terdapat tanda-tanda (kebesaran Allah) bagi orang-orang yang berpikir.”

(Q.S Al-Jasiyah : 13)

KATA PENGANTAR

Assalamu'alaikum Wr.Wb.

Alhamdulillahirabbil'aalamiin, segala puji dan syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat, hidayah, berkah dan karunia-Nya yang takkan terhitung. Sholawat serta salam senantiasa tercurahkan kepada Nabi Muhammad SAW yang senantiasa mendoakan keselamatan umatnya dan telah membawa umatnya dari zaman jahilliyah menuju zaman islam yang terang benderang dan penuh pengetahuan serta teknologi seperti sekarang, sehingga penulis dapat meyelesaikan tugas akhir dengan Judul "IDENTIFIKASI EMOSI MELALUI SUARA MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK DAN SUPPORT VECTOR MACHINE".

Penulisan tugas akhir ini merupakan salah satu syarat untuk memperoleh gelar Sarjana Komputer (S.Kom) pada Program Studi Informatika, Universitas Islam Indonesia. Penulis menyadari bahwa dalam penulisan skripsi ini tidak lepas dari bantuan, arahan, bimbingan, kritik, dan saran dari berbagai pihak sehingga penelitian ini dapat terselesaikan dengan baik. Oleh karena itu, pada kesempatan ini penulis ingin mengucapkan terimakasih kepada:

1. Allah SWT yang telah memberikan semua nikmat dan karunia-Nya sehingga penulis berhasil menyelesaikan penelitian skripsi ini.
2. Prof. Fathul Wahid, S.T., M.Sc., Ph.D. selaku Rektor Universitas Islam Indonesia
3. Hendrik, S.T., M.Eng. selaku Ketua Jurusan Informatika.
4. Dr. Raden Teduh Dirgahayu, S.T., M.Sc. selaku Ketua Program Studi Informatika – Program Sarjana
5. Arrie Kurniawardhani, S.Si., M.Kom. selaku dosen pembimbing yang telah memberikan banyak waktunya untuk membimbing, memberi motivasi serta mengarahkan dalam penyusunan skripsi ini dengan penuh semangat dan kesabaran.
6. Seluruh dosen Program Studi Informatika Universitas Islam Indonesia yang telah banyak memberikan ilmu, motivasi dan semangatnya dalam proses beajar di perkuliahan.
7. Kedua orang tua yang saya cintai yang selalu memberikan bantuan motivasi dan doa yang terus mengalir hingga saat ini.
8. Saudara serta kerabat yang selalu mendukung dan memberi nasihat dalam penyusunan skripsi.

9. Rahma Pangastuti serta sahabat-sahabatku yang telah memberi banyak dukungan dalam proses penulisan skripsi ini.
10. Seluruh pihak yang tidak dapat saya sebutkan satu persatu yang telah banyak membantu saya dalam penyusunan skripsi.

Semoga segala bantuan dan dukungan yang telah diberikan selama ini mendapatkan balasan yang lebih baik dari Allah SWT. Penulis menyadari masih terdapat banyak kekurangan dalam pengerjaan skripsi ini sehingga penulis sangat mengharapkan adanya masukan dan kritikan yang membangun. Semoga skripsi ini dapat bermanfaat untuk pengembangan ilmu pengetahuan khususnya dalam bidang farmasi, baik sedikit maupun banyak.

Wassalamu'alaikum Wr.Wb.

Yogyakarta, 5 Juli 2021

(Rio Galang Jati Respati)

INTISARI

Penggunaan suara dalam *data science* telah banyak dilakukan pada beberapa penelitian seperti mendeteksi suara hewan, ataupun objek tertentu. Penelitian tentang emosi manusia sedang gencar sekali, salah satunya adalah mendeteksi emosi manusia melalui suara. Dewasa ini pendeteksian emosi tidak hanya digunakan untuk riset di bidang akademik seperti psikologi, *neuroscience*, psikiater, ilmu kognitif dan lainya. Tetapi juga diaplikasikan secara praktis seperti *call centre*, *gaming industry*, bidang medis, dan lainnya. Deteksi emosi bisa dilakukan melalui dua cara yaitu, melalui wajah serta melalui suara. Dalam mendeteksi emosi menggunakan wajah, *Face Landmark* yang dibutuhkan berpengaruh untuk menghasilkan akurasi yang bagus. Namun, ketika menggunakan suara, amplitudo dari suara tersebut yang berpengaruh dalam menghasilan akurasi yang bagus. Pada penelitian kali ini, peneliti menggunakan suara sebagai media dalam mendeteksi suara. JL Corpus *dataset* beserta RAVDESS *dataset* digunakan sebagai *dataset* dari program yang dibuat untuk mendeteksi emosi melalui suara. *Dataset* hanya berisi suara yang sudah dilabeli, label yang diberikan menunjukkan emosi dari suara tersebut. Emosi yang menjadi label adalah netral, tenang, sedih, marah, senang, takut, jijik, dan terkejut. Untuk mengekstraksi fitur dari suara tersebut perlu bantuan dari *Mel Frequency Cepstrum Coefficents* (MFCCs). Metode klasifikasi yang digunakan adalah dua metode yang ada dalam *Machine Learning*, yaitu *Convolutional Neural Network* (CNN) dan *Support Vector Machine* (SVM). Metode klasifikasi CNN memiliki kinerja yang fleksibel. Metode CNN dapat dikatakan fleksibel karena metode CNN memiliki kemampuan menggeneralisir yang lebih baik saat menggunakan jumlah data yang sama. Waktu yang dibutuhkan untuk mendeteksi satu suara saat menggunakan metode SVM selalu lebih sedikit ketika dibandingkan dengan metode CNN.

Kata kunci: *Convolutional Neural-Network; Emotional Detection; Machine Learning; Mel Frequency Cepstrum Coefficents*; *Support Vector Machine.*

GLOSARIUM

Amplitudo	pengukuran skalar yang nonnegatif dari besar osilasi suatu gelombang.
CNN	salah satu jenis neural network yang biasa digunakan pada data image dan suara
Dataset	Data yang sudah terlabeli
Frekuensi	banyaknya getaran yang terjadi dalam kurun satu detik
Flowchart	rangkaian berupa bagan atau alur dengan simbol tertentu yang menggambarkan proses atau urutan proses dengan spesifik beserta hubungannya dengan instruksi dan proses lainnya
MFCCs	koefisien ciri yang berisi nilai-nilai yang mewakili sinyal ucapan.
Optimizer	algoritma atau metode yang digunakan untuk mengubah atribut jaringan saraf seperti bobot dan kecepatan pembelajaran (learning rate) untuk mengurangi *loss*
SVM	suatu teknik untuk menemukan hyperplane yang bisa memisahkan dua set data dari dua kelas yang berbeda

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penggunaan suara dalam *data science* telah banyak dilakukan pada beberapa penelitian seperti mendeteksi suara hewan, ataupun objek tertentu. Penelitian tentang emosi manusia sedang gencar sekali, salah satunya adalah mendeteksi emosi manusia melalui suara (Jain dkk., 2018). Dewasa ini pendeteksian emosi tidak hanya digunakan untuk riset di bidang akademik seperti psikologi, *neuroscience*, psikiater, ilmu kognitif dan lainya. Tetapi juga diaplikasikan secara praktis seperti *call centre*, *gaming industry*, bidang medis dan lainya (Jain dkk., 2018). Deteksi emosi bisa dilakukan melalui dua cara yaitu, melalui wajah serta melalui suara. Dalam mendeteksi emosi menggunakan wajah, *Face Landmark* yang dibutuhkan berpengaruh untuk menghasilkan akurasi yang bagus. Namun, ketika menggunakan suara, amplitudo dari suara tersebut yang berpengaruh dalam menghasilan akurasi yang bagus. Pada penelitian kali ini, peneliti menggunakan suara sebagai media dalam mendeteksi emosi. Pendeteksian emosi menggunakan suara dipilih karena dapat diaplikasikan ke berbagai hal yang tidak membutuhkan wajah, seperti mendeteksi emosi melalui suara yang digunakan di dalam *cockpit* pesawat untuk mengetahui kondisi psikologis dari pilot untuk menghindari kecelakaan. Pendeteksian emosi melalui suara juga dapat digunakan pada percakapan *Call Centre* untuk menganalisis kebiasaan pada pelanggan untuk membantu mengembangkan kualitas dari pelayanan ketika menghubungi lewat telepon (Prakash dkk., 2015).

Penelitian sebelumnya telah menggunakan algoritma Pembelajaran Mesin seperti *Convolutional Nerual Network* (CNN) dan *Support Vector Machine* (SVM) untuk klasifikasi dan deteksi suara (Jain dkk., 2017; Zhao dkk., 2019; Jain dkk., 2018; Bertero dkk., 2018; Semwal dkk., 2017; Tripathi dkk., 2019). Salah satu penelitian menyebutkan bahwa penggunaan metode CNN menghasilkan akurasi yang lebih tinggi dibandingkan penggunaan metode lainnya (Zhao dkk., 2019). Di sisi lain SVM banyak digunakan karena dianggap sebagai metode yang mudah digunakan oleh banyak orang (Jain dkk., 2017). Metode CNN dan SVM merupakan metode pembelajaran yang menggunakan data latih berlabel (*supervised learning*), sehingga dilakukan perbandingan mana yang lebih cocok untuk mendeteksi emosi pada suara.

Meskipun algoritma Pembelajaran Mesin telah banyak digunakan dalam klasifikasi suara, satu metode tidak selalu cocok untuk setiap tipe data atau tipe suara. Berdasarkan uraian di atas, penelitian ini memiliki tujuan untuk mengimplementasikan dan membandingkan metode CNN dan SVM dalam proses deteksi emosi pada suara. Dengan penerapan metode ini diharapkan proses pendeteksian emosi dapat dilakukan dengan akurat, dan dengan membandingkan kedua metode tersebut maka dapat diketahui metode mana yang lebih cocok atau sesuai untuk data yang digunakan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka dapat dirumuskan rumusan masalah sebagai berikut:

a. Bagaimana mendeteksi emosi melalui suara menggunakan CNN dan SVM?
b. Manakah yang lebih akurat dan lebih cepat untuk mendeteksi emosi melalui suara di antara CNN dan SVM?

1.3 Batasan Masalah

Penelitian ini memiliki beberapa batasan agar tidak menyimpang dari permasalahan di atas. Batasan masalah pada penelitian ini sebagai berikut:

a. Suara yang digunakan masih berbahasa Inggris.
b. Dalam satu suara pada dataset hanya terdapat satu emosi.
c. Emosi yang dapat dikenali oleh model yang dibuat ada delapan yaitu, netral, tenang, senang, sedih, marah, takut, jijik dan terkejut.

1.4 Tujuan Penelitian

Tujuan dilakukan penelitian ini sebagai berikut:

a. Untuk mengetahui proses mendeteksi emosi melalui suara menggunakan CNN dan SVM.
b. Untuk menentukan metode mana yang memiliki akurasi lebih tinggi antara CNN dan SVM untuk mendeteksi emosi melalui suara.

1.5 Manfaat Penelitian

1.5.1 Bagi penulis

Mendapatkan wawasan dalam bidang deteksi emosi melalui suara pada manusia. Penulis dapat mengetahui teori, metode, dan langkah penyelesaian deteksi emosi melalui suara pada manusia. Penelitian ini melatih kemampuan penulis dalam menganalisis

1.5.2 Bagi Peneliti Lain

Penelitian ini dapat dijadikan rujukan dalam penelitian lanjutan yang lebih mendalam dan kompleks. Penelitian ini dapat dijadikan referensi dalam bidang informatika.

1.5.3 Bagi Instansi Terkait

Pihak yang berkepentingan dalam bidang psikologi dapat menjadikan penelitian ini sebagai rujukan bahwa CNN dan SVM dapat diterapkan untuk mendeteksi emosi melalui suara, sehingga psikiater dapat melakukan pekerjaan lebih efektif dan efisien.

1.6 Metodologi Penelitian

Langkah -langkah yang diterapkan dalam penelitian ini agar mencapai tujuan yang diinginkan adalah sebagai berikut:

a. Tahap Pengumpulan Data

Pengumpulan data diperlukan untuk mendukung pelaksanaan penelitian. Data ini yang nantinya akan dijadikan sebagai bahan untuk pengembangan program pendeteksian emosi. Data tersebut akan digunakan untuk pelatihan, validasi serta pengujian.

b. Perancangan Sistem

Perancangan merupakan penggambaran perencanaan sistem agar lebih terstruktur dan memudahkan peneliti dalam mengimplementasikan sistem. Perancangan yang dibuat meliputi diagram alir proses sistem dengan menggunakan *flowchart*.

c. Implementasi Sistem

Implementasi adalah menerapkan hasil penelitian kepustakaan sebagai langkah untuk memecahkan masalah yang ditemukan. Langkah ini meliputi peng*input*an data suara manusia, pengolahan data, dan keluaran berupa emosi dari suara manusia tersebut.

d. Pengujian Sistem

Pengujian dilakukan untuk memastikan bahwa sistem memenuhi tujuan penelitian, yaitu mendeteksi emosi pada suara manusia, dan membandingkan kedua metode tersebut dengan jumlah data masukan yang sama.

1.7 Sistematika Laporan

Dalam penyusunan tugas akhir ini, sistematika penulisan dibagi menjadi beberapa bab sebagai berikut:

BAB I Pendahuluan

Berisi latar belakang masalah praktis yang menjadi dasar penelitian, pentingnya deteksi emosi manusia, dan solusi dari permasalahan yang ada. Berdasarkan latar belakang yang ada,

rumusan masalah tersebut kemudian dijadikan dasar perencanaan pemecahan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian dan sistematika penulisan.

Bab II Landasan Teori

Termasuk di dalamnya adalah uraian pustaka sesuai dengan topik penelitian sebagai dasar penelitian dan beberapa penelitian sejenis yang telah dilakukan. Teori-teori dalam bab ini menggunakan jurnal, buku, dan artikel sebagai bahan referensi dengan topik mulai dari masalah yang berkaitan dengan deteksi suara, emosi manusia, pembelajaran mesin, jaringan saraf konvolusional, mesin vektor pendukung, dan beberapa alat yang digunakan.

BAB III Metodologi Penelitian

Berisi berbagai tahapan dan kebutuhan penelitian, sebagai acuan pemecahan masalah dalam penelitian ini. Bab ini meliputi pengumpulan data dan perancangan sistem menggunakan diagram alir.

BAB IV Hasil dan Pembahasan

Berisi hasil dan pembahasan setiap proses dalam sistem, pengujian kinerja sistem, kekuatan sistem, dan kelemahan. Pengujian kinerja sistem menggunakan uji pada data tes.

BAB V Kesimpulan dan Saran

Berisi kesimpulan tentang hasil penelitian, kesimpulan tersebut memenuhi tujuan penelitian atau tidak memenuhi rekomendasi penelitian, sehingga peneliti lain dapat melanjutkan penelitian melalui keterbatasan dan kekurangan penelitian.

BAB II
LANDASAN TEORI

2.1 Emosi

Emosi manusia adalah suatu perasaan yang muncul di dalam diri seseorang sebagai akibat dari adanya rangsangan, baik dari dalam diri sendiri maupun dari luar (Anggraini, 2018). Emosi memiliki reaksi yang kompleks mangandung aktivitas dengan derajat yang tinggi sehingga terjadi perubahan perilaku yang akan menimbulkan kegoncangan yang kadang-kadang terjadi ketegangan dalam hubungannya dengan lingkungan (Manizar, 2016). Emosi seseorang dapat diketahui dari dua hal yaitu, berdasarkan *Face Landmark* pada wajah dan berdasarkan suara yang dikeluarkan (Anjaini, 2019). Jenis emosi manusia yang dikeluarkan beragam, diantaranya adalah marah, jijik, takut, senang, netral, sedih dan terkejut dapat dilihat pada Gambar 2.1.

(a) (b) (c) (d)

(e) (f) (g)

Gambar 2.1 Contoh beragam dari emosi yaitu, (a) marah, (b) jijik, (c) takut, (d) senang, (e) netral, (f) sedih, dan (g) terkejut

Sumber : Kwong dkk., (2018)

2.2 Suara

Suara dapat juga diartikan sebagai audio. Gelombang suara adalah gelombang yang dihasilkan dari suatu benda yang bergetar pada frekuensi audio tertentu. Telinga manusia hanya bisa mendengar suara antara 20Hz hingga 20KHz. Angka di 20Hz menunjukkan frekuensi

suara terendah yang dapat didengar oleh telinga manusia, sedangkan 20KHz menunjukkan frekuensi suara tertinggi yang dapat didengar oleh telinga manusia. Bentuk dari gelombang suara sangat beragam, bentuk tersebut tergantung dari tekanan media perantara, salah satunya adalah udara. Suara diciptakan dari getaran yang ditimbulkan dari sebuah objek, dan hal tersebut yang menyebabkan udara disekitarnya pun ikut bergetar. Getaran dari udara inilah yang menyebabkan gendang telinga manusia bergetar, lalu otak yang akan menginterpretasikan sebagai suara. Gelombang umumnya memiliki tiga sifat penting untuk kerja audio yaitu, periode gelombang, amplitudo, dan frekuensi. Periode gelombang adalah jarak antar titik gelombang dan titik ekuivalen pada fase berikutnya. Amplitudo adalah kekuatan atau daya gelombang sinyal. Frekuensi adalah jumlah getaran yang terjadi di tiap detiknya. (Purnama, 2012).

2.3 Deteksi Suara

Deteksi suara adalah pengembangan sistem dimana komputer memungkinkan menerima masukan berupa kata yang diucapkan. Kata-kata yang diucapkan bentuknya akan diubah menjadi sinyal digital lau sinyal tersebut akan disesuaikan dengan kode tertentu untuk mengidentifikasi kata-kata tersebut. Untuk mengenal suara membutuhkan sampel kata sebenarnya yang diucapkan dari speaker. Sampel data tersebut akan disimpan dalam komputer, dan kemudian digunakan sebagai *dataset* dalam mencocokkan kata yang diucapkan selanjutnya. Dikarenakan sample kata tersebut diucapkan oleh speaker, maka kualitas dari sample kata tersebut tergantung pula dari yang dihasilkan oleh speaker (Rusdi dkk., 2018).

2.4 Ekstraksi Fitur

Ekstraksi fitur ditujukan untuk menajamkan perbedaan pola sehingga dapat memudahkan dalam pemisahan emosi pada proses klasifikasi. Metode ekstraksi fitur yang peneliti gunakan kali ini adalah Mel-Frequency Cepstral Coefficients (MFCC). MFCC dianggap sebagai metode standar untuk seleksi ekstraksi fitur dalam suara karena dapat merepresentasikan sinyal suara dengan baik. Langkah-langkah untuk menghitung MFCC adalah sebagai berikut (Widodo dkk., 2019):

a. Pre-emphasis Filtering

Filter ini mempertahankan frekuensi-frekuensi tinggi pada spektrum yang tereliminasi saat proses produksi suara. Filter pre-emphasis dapat dihitung dengan Persamaan 2.1:

$$s^1(\mathrm{n}) = \mathrm{s}(n) - \propto * s(n-1)$$

(2.1)

Keterangan:

s^1 : Hasil audio setelah dilakukan pre-emphasis

s(n) : Sinyal audio sebelum dilakukan pre-emphasis

α : 0.9375

b. *Frame* Blocking

Frame blocking digunakan untuk memotong-motong sinyal suara menjadi beberapa *frame* agar dapat diproses secara short-time untuk memperoleh karakter frekuensi yang relatif stabil

c. Windowing

Windowing dilakukan untuk mengurangi efek aliasing atau sinyal tak kontinyu pada awal dan akhir masing-masing *frame* yang dapat terjadi akibat proses *frame* blocking.

d. FFT

FFT digunakan untuk mengkonversi masing-masing *frame* sinyal suara dari domain waktu ke domain frekuensi. Dalam sinyal bicara, sistem pendengaran sangat sensitif terhadap karakteristik frekuensi sehingga sinyal bicara lebih mudah dianalisis pada domain frekuensi.

e. Mel Frequency Wrapping

Sinyal bicara terdiri dari nada dengan frekuensi yang berbeda-beda. Untuk masing-masing nada dengan frekuensi aktual, f, diukur dalam Hz, pitch subjektif diukur dengan skala ‘mel’. Skala mel-frequency bersifat linier untuk frekuensi di bawah 1000 Hz dan logaritmik untuk frekuensi di atas 1000 Hz.

f. Discrete Cosine Transform

Pada langkah terakhir, spectrum log mel harus dikonversikan kembali menjadi domain waktu menggunakan Discrete Cosine Transform, hasilnya disebut mel frequency cepstral coefficients (MFCCs).

2.5 *Support Vector Machine*

Support Vector Machine (SVM) merupakan salah satu model klasifikasi yang umum digunakan dalam mendeteksi emosi melalui suara (Jain dkk., 2018). Konsep dasar SVM

merupakan kombinasi harmonis dari teori-teori komputasi sebelumnya, seperti margin hyperplane dan kernel. Berbeda dengan strategi neural network yang mencari hyperplane pemisah antar kelas, SVM berusaha untuk menemukan hyperplane yang terbaik pada *input* space. Awalnya SVM hanya dapat melakukan linear classifier, namun dengan berkembangnya waktu SVM dapat melakukan non-linear classifier dengan bantuan kernel (Azhar dkk., 2015).

Hyperplane disini berfungsi untuk memisahkan data yang dimasukkan tersebut untuk ke beberapa kategori. Untuk mencari Hyperplane yang terbaik, maka Hyperplane akan dilatih menggunakan beberapa algoritma untuk menyesuaikan dengan data training. Selama melatih Hyperplane tersebut, nilai fitur dari sinyal suara tersebut akan diekstraksi untuk menentukan kelas dari suara tersebut. (Jain dkk., 2018)

2.6 *Convolutional Neural Network*

Jaringan konvolusional atau biasa disebut *Convolutional Neural Network* (CNN) adalah jenis jaringan saraf khusus yang digunakan untuk mengolah data dengan topologi mirip grid. Nama jaringan saraf konvolusional menunjukkan bahwa jaringan tersebut menggunakan operasi matematika yang disebut konvolusi. Konvolusi sendiri adalah operasi linier. Oleh karena itu, jaringan konvolusional adalah jaringan saraf yang menggunakan konvolusi minimal di setiap lapisannya (Zhao dkk., 2019). *Convolutional Neural Network*s (ConvNets) adalah kasus khusus Jaringan Syaraf Tiruan (JST) dan saat ini dianggap sebagai model terbaik untuk memecahkan masalah pengenalan dan deteksi objek. (LeCun dkk., 2015)

2.6.1 Konvolusi

Konvolusi merupakan *layer* pertama yang menerima *input* suara langsung pada arsitektur. Operasi pada layer ini sama dengan operasi konvolusi yaitu melakukan operasi kombinasi linier filter terhadap daerah lokal. Filter merupakan representasi bidang reseptif dari neuron yang terhubung ke dalam daerah lokal pada *input* gambar. Lapisan Konvolusi melakukan operasi konvolusi pada *output* dari layer sebelumnya. Layer tersebut adalah proses utama yang mendasari sebuah CNN. Tujuan dilakukannya konvolusi pada data suara adalah untuk mengekstraksi fitur dari suara *input*. Secara umum operasi konvolusi dapat dilihat pada Persamaan 2.2:

$$s(t) = (x * w)(t) \qquad (2.2)$$

Keterangan:

s(t) : Fungsi hasil operasi konvolusi

x : *Input*

w : Bobot (kernel)

2.6.2 Fungsi Aktivasi

Fungsi aktivasi adalah node yang ditambahkan di akhir setiap keluaran jaringan saraf. Fungsi aktivasi disebut juga fungsi transfer dan digunakan untuk menentukan keluaran dari jaringan saraf. Ada dua jenis fungsi aktivasi yaitu linier dan non linier (Sharma dkk., 2017). Dalam arsitektur CNN, fungsi aktivasi adalah kalkulasi akhir dari keluaran peta fitur, atau pembangkitan pola fitur setelah proses kalkulasi konvolusi atau penggabungan. Beberapa jenis fungsi aktivasi yang sering digunakan dalam penelitian antara lain fungsi sigmoid, tanh, Rectified Linear Unit (ReLU), Leaky ReLU (LReLU) dan Parametric ReLU.

2.6.2.1 Rectified Linear Unit

ReLU adalah fungsi linier yang akan mengeluarkan *input* secara langsung jika nilai yang dihasilkan positif, namun jika hasilnya negatif, itu akan menghasilkan nol. ReLU banyak digunakan karena dianggap sebagai fungsi aktivasi yang lebih mudah dilatih dan menghasilkan performa yang lebih baik (Brownlee dkk., 2019).

2.6.3 *Dropout*

Lapisan Dropout secara acak menetapkan unit input ke 0 dengan frekuensi laju pada setiap langkah selama waktu pelatihan, yang membantu mencegah overfitting. Input yang tidak disetel ke 0 ditingkatkan sebesar 1/(1 - rate) sehingga jumlah dari semua input tidak berubah (Chollet dkk., 2015).

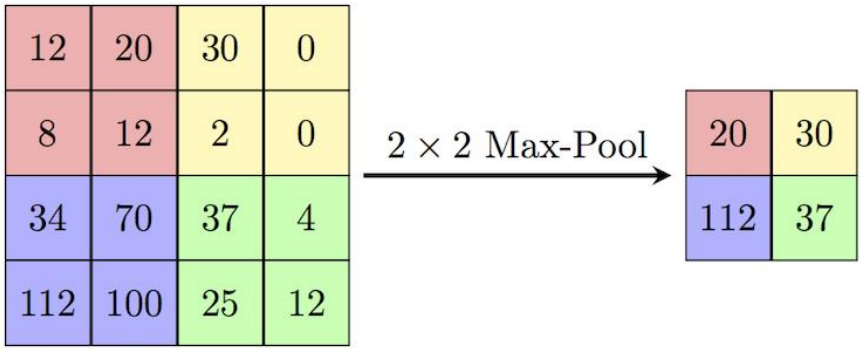
2.6.4 *Fully Connected Layer*

Feature map yang dihasilkan dari *feature extraction layer* masih berbentuk *multidimensional array*, sehingga harus melakukan "*flatten*" atau reshape *feature map* menjadi sebuah vector agar bisa digunakan sebagai *input* dari *fully-connected layer*. *Fully Connected Layer* yang dimaksud disini adalah *Multi Layer Perceptron* yang sudah pernah dipelajari. Lapisan *Fully Connected* adalah lapisan di mana semua neuron aktivasi dari lapisan sebelumnya terhubung semua dengan neuron di lapisan selanjutnya seperti halnya jaringan saraf tiruan biasa. Setiap aktivasi dari lapisan sebelumnya perlu diubah menjadi data satu dimensi sebelum dapat dihubungkan ke semua neuron di lapisan. Lapisan *Fully Connected*

biasanya digunakan pada metode *Multi Layer Perceptron* untuk mengolah data sehingga bisa diklasifikasikan. Perbedaan antara lapisan *Fully Connected* dan lapisan konvolusi biasa adalah neuron di lapisan konvolusi terhubung hanya ke daerah tertentu pada *input*, sementara lapisan *Fully Connected* memiliki neuron yang secara keseluruhan terhubung. Namun, kedua lapisan tersebut masih mengoperasikan produk dot, sehingga fungsinya tidak begitu berbeda (Danukusumo dkk., 2017).

2.6.5 *Max Pooling*

Fungsi *Pooling* dapat secara bertahap mengurangi jumlah parameter dan kalkulasi dalam jaringan. Metode yang paling umum digunakan dalam penggabungan adalah Max Pooling. Misalnya, jika kita menggunakan pool terbesar dengan ukuran kernel pool 2x2, maka untuk setiap pergeseran filter, nilai terbesar di area piksel 2x2 akan dipilih, seperti yang ditunjukkan pada Gambar 2.2 (Sena dkk., 2017).



Gambar 2.2 Contoh terjadinya *Max-pooling* 2x2

Sumber: computersciencewiki.org (2018)

2.7 Confusion Matrix

Penentuan kinerja suatu model klasifikasi dapat dilihat dari parameter pengukuran kinerjanya yaitu tingkat akurasi, recall, dan presisi. Untuk menghitung faktor-faktor tersebut, kita membutuhkan matriks yang disebut *Confusion Matrix*. Salah satu Confusion-matrix yang sering digunakan dalam pengukuran dapat dilihat pada Gambar 2.3 (Fawcet dkk., 2006).

		Actual Labels	
		YES	NO
Predicted Labels	YES	True Positive (TP)	False Positive (FP)
	NO	False Negative (FN)	True Negative (TN)

Gambar 2.3 Tabel Confusion-Matrix yang sering digunakan

Sumber: (Hamid, 2012)

Berdasarkan gambar di atas, terdapat beberapa nilai dalam matriks tersebut, yaitu "True Positive" (TP), "True Negative" (TN), "False Positive" (FP), dan "False Negative" (FN), semua kemungkinan kejadian sebenarnya positif (P) dan semua kemungkinan kejadian sebenarnya negatif (N). Nilai ini dapat digunakan untuk menghitung akurasi dengan Persamaan (2.4)

$$\text{Akurasi} = \frac{TP + TN}{P + N} * 100\% \quad (2.4)$$

Akurasi digunakan sebagai parameter seberapa akurat model melakukan klasifikasi. Sedangkan untuk menghitung tingkat ketelitian dalam prediksi kejadian dapat digunakan Persamaan (2.5).

$$\text{Presisi Prediksi} = \frac{TP}{TP + FP} * 100\% \quad (2.5)$$

Presisi menjelaskan seberapa akurat model memprediksi peristiwa positif dalam serangkaian aktivitas prediktif. Perhitungan yang tepat biasanya berguna dalam mengembangkan model prediksi curah hujan di suatu daerah. Selain presisi dan akurasi, untuk dapat melihat lebih detail performansi suatu sistem, recall atau kepekaan sistem terhadap suatu kelas juga dapat dilihat. Recall dapat dihitung menggunakan Persamaan (2.6).

$$Recall = \frac{TP}{TP + FN} * 100\% \quad (2.6)$$

2.8 Penelitian Sebelumnya

Berdasarkan hasil dari *dataset* yang telah digunakan, akurasi yang tinggi tidak terpengaruh dari jumlah data yang telah digunakan di dalam *dataset*, dikarenakan ada banyak

faktor lainnya yaitu, kejernihan sebuah suara dan pembagian *dataset* yang merata. Hal ini didukung oleh Bertero yang melakukan percobaan menggunakan data yang cukup banyak dengan jumlah lebih dari 5000 data, namun akurasi yang dihasilkan sebesar 66.1% belum cukup akurat. Hal ini dikarenakan terdapat salah satu emosi yang terlalu banyak datanya berupa emosi senang, sehingga data tersebut dicurigai berisi emosi "netral" dikarenakan suara yang dikeluarkan menyerupai emosi senang dan netral, serta *dataset* tersebut memiliki frekuensi yang cukup signifikan (Bertero dkk., 2017).

Penelitian lain yang dilakukan oleh Semwal dkk. (2017) menunjukkan percobaan yang menggunakan 535 file dapat menghasilkan akurasi yang lebih tinggi sebesar 80% dikarenakan file emosi dalam *dataset* tersebut dibagi secara rata. Hal tersebut dibuktikan oleh percobaan menggunakan *dataset* yang sama yaitu EmoDB dapat menghasilkan akurasi dengan persentase 95.33% (Zhao dkk., 2019).

Percobaan tersebut dapat dilihat dari metode fitur ekstraksi yang digunakan seperti MFCCs (Jain dkk., 2017; Jain dkk., 2018; Pinto dkk., 2020). Penelitian yang dilakukan oleh Jain dkk. (2017) membandingkan metode fitur ekstraksi yang menggunakan MFCCs dalam percobaan mendapatkan akurasi hingga 98% di salah satu emosinya sedangkan metode fitur ekstraksi menggunakan LPCC menghasilkan akurasi yang menurun hingga 70%. Penelitian lain yang dilakukan oleh Jain dkk. (2018) membandingkan LPCC memiliki akurasi yang lebih baik sebesar 28%, sedangkan MFCCs mendapatkan 24% dan menggabungkan kedua metode tersebut sehingga akurasi yang dihasilkan cukup tinggi sebesar 56%.

Ketika metode klasifikasi yang dibandingkan maka *Support Vector Machine* (SVM) beserta *Convolutional Neural Network* (CNN) masih menjadi metode klasifikasi yang sering muncul. SVM banyak digunakan karena dianggap bahwa metode tersebut sangat *simple* dan efisien karena terdapat library yang membantu yaitu LIBSVM (Jain dkk., 2017), hal tersebut berbeda dengan CNN yang harus membuat model jaringannya terlebih dahulu (Bertero dkk., 2017). CNN diharuskan membangun jaringan dalam beberapa lapis, seperti layer konvolusi, *max-pooling*, dan fully connected layer. (Bertero dkk., 2017).

Tabel 2.1 merupakan perbandingan dari beberapa penelitian yang sudah pernah dilakukan sebelumnya, berdasarkan tabel tersebut bisa dilihat walaupun menggunakan *dataset* yang berbeda beda, jika membandingkan metode SVM dan CNN maka metode CNN terlihat memiliki akurasi yang lebih tinggi. Namun, selain metode yang digunakan, faktor yang lainnya juga berpengaruh terhadap akurasi yang dihasilkan seperti fitur ekstraksi yang digunakan.

Tabel 2.1 Beberapa penelitian dengan menggunakan metode CNN dan SVM

No	*Dataset*	Ekstraksi Fitur	Metode	Akurasi (%)
1	TEDLIUM v2	Konvolusi	CNN	66.1 (Bertero dkk., 2017)
2	EmoDB	LLDs	SVM	80 (Semwal dkk., 2017)
3	Linguistic Data Consortium	MFCCs	SVM	79.22 (Jain dkk., 2017)
4	Youtube	MFCCs	SVM	81 (Jain dkk., 2018)
5	EmoDB	Konvolusi	CNN	95 (Zhao dkk., 2019)
6	RAVDESS	MFCCs	CNN	91 (Pinto dkk., 2020)
7	RAVDESS	LPCC	SVM	51 (Shegokar dkk., 2016)

BAB III
METODELOGI PENELITIAN

3.1 Data

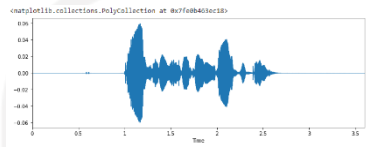
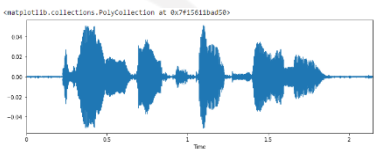
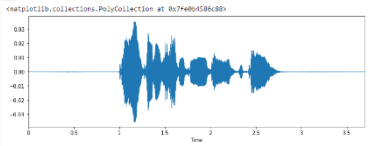
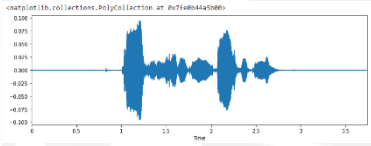
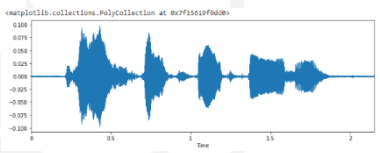
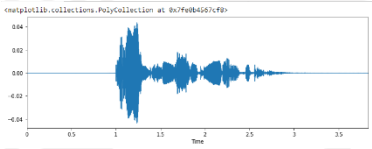
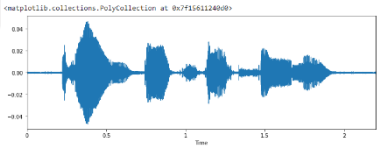
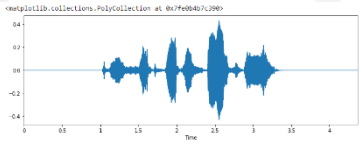
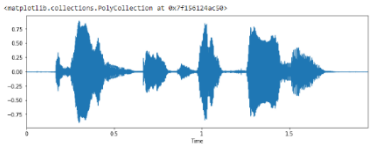
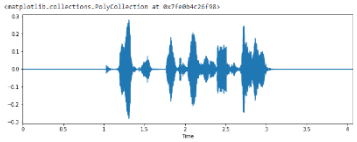
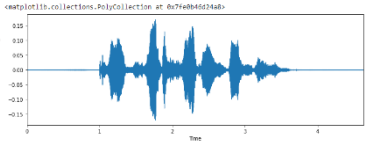
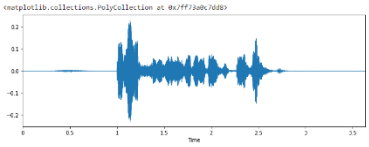
Data yang digunakan dalam penelitian ini merupakan suara manusia dengan berbahasa Inggris dan sudah dilabeli sebelumnya. Peneliti menggunakan dua *dataset* yang berbeda yaitu, JL Corpus (James dkk., 2018) dan *The Ryerson Audio-Visual Dataset of Emotional Speech and song* (RAVDESS) (Livingstone dkk., 2018). Di kedua *dataset* tersebut, satu suara hanya menghasilkan satu emosi saja.

RAVDESS *dataset* dikumpulkan dan data tersebut diunggah melalui *Kaggle* pada tautan (https://www.kaggle.com/uwrfkaggler/RAVDESS-emotional-speech-audio). Aksen yang digunakan pada *dataset* ini berasal dari Amerika Utara. *Dataset* RAVDESS memiliki dua jenis file suara, yaitu file yang berasal dari percakapan dan file yang berasal dari lagu. Jumlah suara yang berasal dari percakapan terdapat 1440 suara, sedangkan suara yang berasal dari lagu terdapat 1012 suara. Setiap suara memiliki durasi yang beragam dari 3,5 detik hingga empat detik. Pembagian suara perempuan dan laki laki dibagi rata karena di dalam *dataset* tersebut terdapat 12 orang laki laki dan 12 orang perempuan. Delapan emosi yang terdapat dalam *dataset* tersebut yaitu netral, tenang, senang, sedih, marah, takut, jijik, dan terkejut. Jumlah data per emosi yang berasal dari percakapan adalah sebagai berikut, kelas emosi netral berjumlah 96 suara, sedangkan emosi yang lainnya berjumlah sama yaitu 192 suara. Jumlah data yang dimiliki pada lagu berbeda dengan yang berasal dari percakapan, kelas emosi jijik dan terkejut tidak terdapat satu suara di kelas tersebut. Namun, jumlah dari kelas lainnya sama ketika berasal dari percakapan (Livingstone dkk., 2018). Format suara yang digunakan adalah 16bit, 48kHz, dan file tersebut bertipe .wav.

JL Corpus *dataset* dikumpulkan dan data tersebut diunggah melalui platform *kaggle* pada tautan (https://www.kaggle.com/tli725/jl-corpus). Aksen yang digunakan pada *dataset* ini berasal dari New Zealand. Data di dalam *dataset* berisikan suara dari percakapan normal manusia, baik laki-laki maupun perempuan, dengan durasi singkat sekitar dua detik setiap emosi. Emosi yang terdapat dalam *dataset* tersebut ada sepuluh, yaitu netral, senang, sedih, marah, gelisah, meminta maaf, tegas, prihatin, mendorong, dan bergairah. Namun, percobaan kali ini digunakan empat emosi, yaitu netral, senang, sedih, dan marah. Hal tersebut dikarenakan emosi yang sama dengan RAVDESS *dataset* hanyalah empat kelas tersebut saja. Jumlah suara per emosi sebanyak 120 suara, baik laki laki maupun perempuan jumlah per

emosi sama. Format suara yang digunakan sama pada RAVDESS *dataset* yaitu 16bit, 48kHz, dan file tersebut bertipe .wav. (James dkk., 2018). Tabel 3.1 adalah contoh amplitudo yang dihasilkan dari berbagai emosi pada RAVDESS *dataset* beserta Jl Corpus *dataset*.

Tabel 3.1 Amplitudo dari suara wanita pada dua *dataset* yang berbeda

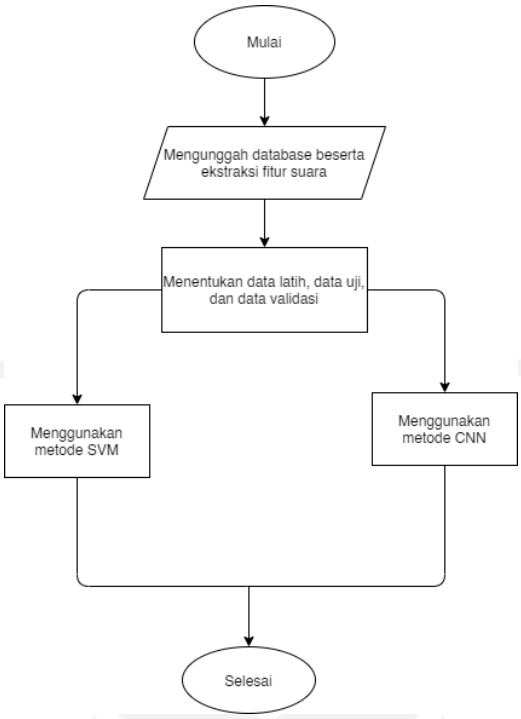
Emosi	***Dataset***	
	RAVDESS	**Jl Corpus**
Netral		
Tenang		-
Senang		
Sedih		
Marah		
Takut		-
Jijik		-
Terkejut		-

Amplitudo dari suara perempuan yang digunakan dapat dilihat pada Tabel 3.1. Amplitudo yang dihasilkan di tiap emosi berbeda semua, untuk emosi yang menyentuh amplitudo di atas 0.1 yaitu, emosi senang, marah, jijik, takut, dan terkejut. Sedangkan amplitudo yang di bawah 0.1 yaitu, tenang, netral, dan sedih. Pada RAVDESS dataset terlihat pola pada setiap suara ada yang serupa, seperti pada emosi Netral dan emosi Senang. Kedua emosi tersebut memiliki pola, pada detik pertama amplitudo suara naik sesaat, lalu turun dan kemudian pada detik kedua amplitudo suara naik kembali sesaat lalu turun lagi. Berbeda dengan RAVDESS yang memiliki pola di dua kelas saja, Jl Corpus dataset memiliki pola pada di tiap kelas suara yang dimiliki. Pola yang dimiliki adalah amplitudo naik pada detik 0.2 lalu turun sesaat hingga 0.7. Pada detik 0.7 naik kembali, lalu turun lagi hingga detik satu. Hal tersebut terjadi berulang hingga detik 1.8.

3.2 Perancangan

Perancangan di sini menjelaskan bagaimana alur dari proses terjadinya bagaimana alur dari program yang akan dibuat. Perancangan tersebut dibuat dalam bentuk *flowchart* seperti pada Gambar 3.1. Pada Gambar 3.1 terlihat perancangan program dilakukan melalui beberapa tahapan, yaitu:

1. Menggunggah dataset beserta ekstraksi fitur suara
   Dataset yang digunakan akan diunggah satu persatu untuk dimasukkan ke dalam program, selagi dataset diunggah terjadi proses ekstraksi fitur suara yang dilakukan oleh MFCCs.
2. Menentukan data latih, data uji, dan data validasi
   Setelah seluruh suara pada dataset diekstraksi fiturnya, maka dibagi data tersebut menjadi data latih, data uji, dan data validasi.
3. Menggunakan metode klasifikasi untuk mendeteksi suara
   Data yang telah dibagi akan diolah oleh dua metode klasifikasi yang digunakan yaitu, Support Vector Machine (SVM0 dan Convolutional Neural Network (CNN).



Gambar 3.1 *Flowchart* program perancangan program.

3.2.1 Mengunggah *dataset* beserta ekstraksi fitur

Awal yang dilakukan adalah mengunggah *dataset* yang digunakan beserta mengekstraksi fitur pada suara tersebut. Pada saat proses ekstraksi fitur suara, peneliti menggunakan metode MFCCs. MFCCs berfungsi untuk mengesktraksi fitur suara.

Ketika ingin menggunakan MFCCs dibutuhkan beberapa parameter seperti nilai frekuensi dan data audio dari setiap amplitudo pada suara. Pada penelitian kali ini jumlah nilai MFCCs yang diambil dari proses yang telah dilalui sebanyak 40, jumlah tersebut adalah jumlah default yang diberikan oleh Librosa. Fitur suara yang didapatkan lalu dikumpulkan ke dalam sebuah *array* bernama ‘fitur’, agar lebih mudah untuk diolah kembali.

3.2.2 Pembagian *dataset*

Pembagian dataset ditentukan setelah melalui percobaan untuk menentukan pembagian data yang pas untuk mendapatkan akurasi yang baik. Setelah ditemukan pembagian data yang pas, pembagian tersebut digunakan di ketiga percobaan yang dilakukan. Percobaan yang telah dilakukan adalah membagi data latih dan data validasi dalam rasio:

a. Pembagian saat menggunakan metode SVM
    1. 60% data latih dan 40% data tes (Ghazvini, 2020).
    2. 70% data latih dan 30% data tes (Ahmad, 2020).

3. 80% data latih dan 20% data tes (Mustaqeem, 2018).

b. Pembagian saat menggunakan metode CNN

1. 60% data latih, 20% data validasi, dan 20% data tes (Bonafilia, 2020).
2. 70% data latih, 15% data validasi, dan 15% data tes (Pandey, 2019).
3. 80% data latih, 10% data validasi, dan 10% data tes (Elsai, 2021).

3.2.3 Support Vector Machine

Perancangan disini menjelaskan bagaimana alur dari proses terjadinya metode *Support Vector Machine* (SVM). SVM digunakan karena metode klasifikasi tersebut masih popular di gunakan banyak orang. Hal tersebut dikarenakan metode tersebut memiliki proses latih yang sangat cepat serta mudah digunakan (Bertero, 2018). Pada perancangan ini menjelaskan bagaimana alur dari pelatihan model beserta pengujian model yang telah dibuat. Perancangan tersebut dibuat dalam bentuk *flowchart* seperti pada Gambar 3.2.

Gambar 3.2 *Flowchart* program saat melakukan metode pelatihan SVM

Data yang telah dibagi menjadi data latih dan data validasi akan diolah menggunakan metode klasifikasi SVM. Data latih digunakan untuk melatih model yang akan dibuat, sedangkan data validasi akan digunakan untuk menghitung akurasi dari model yang telah

dibuat. Percobaan ini dilakukan sebanyak tiga kali, karena menggunakan tiga *dataset* yang berbeda, dan akan menghasilkan tiga model yang berbeda.

Pada saat melatih model klasifikasi, dibutuhkan beberapa paramater yang akan membantu saat melatih model tersebut. Parameter yang dibutuhkan ada beberapa yaitu, kernel, C, gamma, degree dll. Pada penelitian kali ini menggunakan kernel *Radial Basis Function* (RBF) dan C bernilai 1000. Parameter C ditentukan untuk mengetahui seberapa banyak yang ingin dihindari saat misklasifikasi pada setiap data latih. Jenis kernel dan nilai C ditentukan setelah melalui beberapa percobaan seperti mencoba dua kernel yang berbeda yaitu Linear dan RBF. Jumlah C yang dicoba beberapa macam yaitu, 10, 100, dan 1000. Jumlah C yang semakin besar, maka hyperplane yang dipilih akan memiliki margin yang semakin kecil, jika hyperplane tersebut memberikan kinerja yang lebih baik saat melakukan pelatihan. Hal ini menyebabkan jika jumlah C semakin besar, maka semakin spesifik model yang kita buat untuk dataset yang digunakan (Liu, 2014).

Model yang telah dilatih akan diuji untuk mengetahui kinerja dari model yang telah dibuat. Kinerja yang ingin diketahui ada tiga yaitu, Akurasi, Presisi, Recall. Ketiga nilai kinerja tersebut dihitung menggunakan Persamaan 2.4 hingga Persamaan 2.6.

3.2.4 *Convolutional Neural Network*

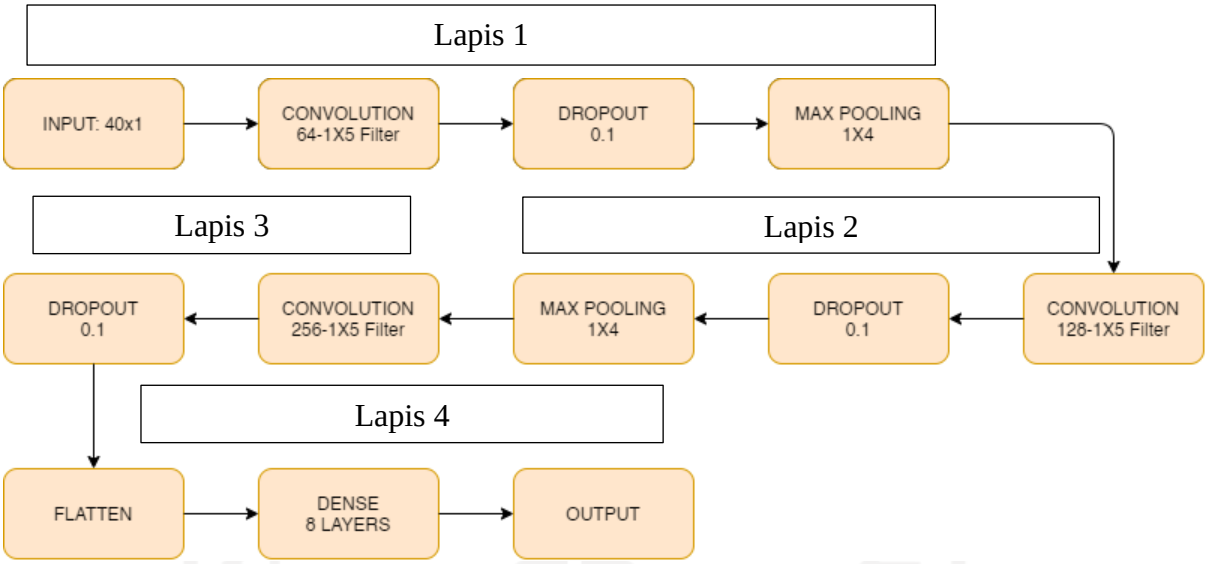
Perancangan disini menjelaskan bagaimana alur dari proses terjadinya metode *Convolutional Neural Network* (CNN). Metode klasifikasi CNN digunakan karena metode klasifikasi tersebut memiliki akurasi yang sangat bagus (Zhao, 2019). Pada perancangan ini menjelaskan bagaimana alur dari pelatihan model beserta pengujian model yang telah dibuat.

Ketika ingin menggunakan metode klasifikasi CNN ada beberapa hal yang perlu dipersiapkan, seperti mengubah ukuran dataset, membangun arsitektur, dll. Alur dari tersebut dibuat dalam bentuk *flowchart* seperti pada Gambar 3.3.

Gambar 3.3 *Flowchart* program dengan metode klasifikasi CNN

Sebelum melatih model menggunakan data latih kita harus membangun arsitektur jaringan terlebih dahulu. Arsitektur yang digunakan mengadopsi dari arsitektur yang dibuat oleh salah satu pengguna (Gupta, 2020) dari Github (https://github.com/), lalu arsitektur diubah untuk menyesuaikan data yang digunakan.

Arsitektur jaringan pada sumber ditunjukkan pada Gambar 3.4. Jaringan ini terdiri dari beberapa layer di tiap lapisan nya yaitu, Konvolusi, fungsi aktivasi ReLU, *Max-Pooling*, dan *Dropout*. Komponen tersebut terdapat pada dua lapisan setelahnya lalu diakhiri dengan lapisan *Fully Connected* yang berisikan, *Flatten*, *Dense*, dan *Softmax*.



Gambar 3.4 Arsitektur dari model CNN pada referensi.

Arsitektur pada referensi, dicoba menggunakan dataset pada percobaan kali ini. Setelah dilakukan percobaan, struktur pada referensi tidak memiliki akurasi yang cukup bagus pada dataset yang digunakan. Oleh karena itu, arsitektur diubah untuk menyesuaikan dataset yang digunakan.

Setelah arsitektur jaringan telah dibuat, nilai ukur yang digunakan akan ditentukan. Nilai ukur dari model yang kita buat adalah Loss dan Akurasi dari tiap epoch-nya Setelah semua diatur kita bisa melatih model yang ingin kita buat. Penelitian ini menggunakan jumlah epoch sebanyak 200 kali (Han, 2018). Jumlah batch size yang dicoba pada penelitian kali ini ada tiga yaitu, 32, 64, dan 120. Model yang telah dilatih akan diuji untuk mengetahui kinerja dari model yang telah dibuat. Kinerja yang ingin diketahui ada tiga yaitu, Akurasi, Presisi, Recall. Ketiga nilai kinerja tersebut dihitung menggunakan Persamaan 2.4 hingga Persamaan 2.6.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi

Semua tahapan perancangan menggunakan *flowchart* pada Gambar 3.1 diimplementasikan ke dalam kode program dengan bahasa pemrograman *Python* versi 3.6.9. Beberapa paket dari *library* yang digunakan adalah sebagai berikut:

a. Ekstraksi Fitur

Paket yang dibutuhkan untuk melakukan ekstraksi fitur pada suara adalah **librosa** (https://librosa.org/). Paket dari python ini digunakan untuk analisis musik dan audio. Paket **librosa** menyediakan kode yang diperlukan untuk membuat sistem pencarian yang ada pada musik. Di dalam paket tersebut tidak hanya untuk mengekstraksi fitur suara saja, tetapi bisa untuk melihat amplitudo dari suara yang digunakan dengan bantuan fungsi (`display`) (McFee dkk., 2015). Untuk menggunakan paket tersebut harus di import terlebih dahulu seperti pada Gambar 4.1.

```
import librosa
from librosa import display
```

Gambar 4.1 Kode saat ingin mengimpor paket dari **librosa**

b. Support Vector Machine

Paket yang dibutuhkan ketika ingin menggunakan metode SVM adalah **sklearn** (scikit-learn) (https://scikit-learn.org/). Paket tersebut memiliki beberapa alat untuk Pembelajaran Mesin yang efisien seperti, Klasifikasi, Regresi, Klastering, dll. Di dalam paket tersebut tidak hanya digunakan untuk melatih model SVM, namun juga bisa digunakan untuk membagi data menjadi data latih beserta data validasi menggunakan fungsi (`train_test_split`). Paket **sklearn** juga bisa untuk menghitung performa model dengan jelas menggunakan fungsi (`classification_report`) (Pedregosa dkk., 2011). Untuk mengimpor paket **sklearn** bisa dilihat pada Gambar 4.2.

```
Import sklearn

from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix
```

Gambar 4.2 Kode saat ingin mengimpor paket dari **sklearn**

c. Convolutional Neural Network

Paket yang dibutuhkan ketika ingin menggunakan metode CNN adalah **keras** (https://keras.io/). Paket ini adalah sebuah deep learning API (Application Programming Interface) yang tertulis pada Python, dan berjalan diatas *platform* Pembelajaran Mesin TensorFlow. Fokus dari paket ini untuk memungkinkan eksperimen yang cepat. Paket tersebut di dalamnya terdapat bermacam macam fungsi yang bisa digunakan, seperti (`sequential`) untuk menumpuk layer yang dibutuhkan saat membangun arsitektur jaringan yang diinginkan. Fungsi yang lainnya adalah (`layers`), fungsi tersebut digunakan untuk memanggil layer yang diinginkan seperti *Dropout, Flatten, Activation,* dll. Tak lupa pula fungsi (`Model`) dibutuhkan karena fungsi tersebut yang akan melatih model yang kita inginkan (Chollet dkk., 2015). Untuk mengimpor yang telah dijelaskan diatas dapat dilihat pada Gambar 4.3

```
import keras

from keras.models import Sequential
from keras.layers import Dense, Embedding
from keras.layers import Input, Flatten, Dropout, Activation
from keras.layers import Conv1D, MaxPooling1D
from keras.models import Model
```

Gambar 4.3 Kode saat ingin mengimpor paket dari **keras**

4.1.1 Mengunggah *dataset* beserta ekstraksi fitur

Dataset akan diunggah dan fitur dari setiap suara yang diunggah akan diekstraksi oleh bantuan MFCCs. Pada Gambar 4.4 adalah kode ketika ingin menentukan *dataset* mana yang ingin digunakan.

```
path = 'drive/My Drive/Dataset FIX/Jl Corpus'
```

(a)

```
path = 'drive/My Drive/Dataset FIX/RAVDESS'
```

(b)

```
path = 'drive/My Drive/Dataset FIX/'
```

(c)

Gambar 4.4 Kode untuk menentukan lokasi dari *dataset* yang digunakan; (a) Jl Corpus, (b) RAVDESS, (c) gabungan

```
path = 'drive/My Drive/Dataset FIX/Jl corpus'
fitur = []

for subdir, dirs, files in os.walk(path):
  for file in files:
      try:
        #Load librosa array, obtain mfcss, store the file and the mfcss
information in a new array
        X, sample_rate = librosa.load(os.path.join(subdir,file),
res_type='kaiser_fast')
        mfccs = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate,
n_mfcc=40).T,axis=0)
        # The instruction below converts the labels (from 1 to 8) to a
series from 0 to 7
        file = int(file[7:8]) - 1
        arr = mfccs, file
        fitur.append(arr)
      except ValueError:
        continue
```

Gambar 4.5 Kode untuk mengupload *dataset* yang digunakan beserta kode untuk melakukan ekstraksi fitur dari suara yang telah diupload

Gambar 4.5 merupakan suatu proses mengunggah *dataset* yang digunakan dan ekstraksi fitur menggunakan Mel Frequency Cepstral Coefficients (MFCCs). Pada Gambar 4.4 tertulis *dataset* mana yang ingin digunakan. Awalnya setiap suara dari *folder* tersebut akan dimasukkan satu per satu dan setiap suara akan diambil ekstraksi ciri menggunakan MFCCs.

Proses yang dilakukan saat mengambil nilai karakteristik dari suara tersebut dimulai ketika setiap suara yang dimasukkan, jumlah frekuensi di setiap detik beserta nilai pada setiap Amplitudo yang ada pada suara keseluruhan diambil. Lalu kedua nilai tersebut diolah menggunakan kode (`librosa.feature.mfcc`), di dalamnya terdapat beberapa parameter yang dibutuhkan, seperti parameter (`y`) yang berisikan nilai pada setiap amplitudo dan (`sr`) adalah nilai frekuensi. Kedua parameter tersebut adalah nilai yang diambil pada proses sebelumnya. Jumlah nilai MFCCs yang ingin kita gunakan sebanyak 40, jumlah tersebut adalah jumlah *default* yang diberikan oleh librosa. Maka, setiap suara yang telah diunggah akan memiliki 40 fitur suara. Label dari data yang digunakan adalah 1 – 8, kemudian label tersebuut diubah menjadi 0 – 7 karena angka awal pada *array* adalah angka 0. Nilai MFCCs yang didapatkan beserta nilai label dimasukkan ke dalam *array* bernama ‘fitur’ agar dapat diolah kembali. Proses ini dilakukan berulang kali hingga semua suara yang berada di *dataset* berhasil diekstraksi.

4.1.2 Pembagian *dataset*

Pada Gambar 4.6 terlihat pembagian *dataset* dilakukan menggunakan kode (`train_test_split`). Kode tersebut akan membagi data yang telah dikumpulkan menjadi data train dan data tes. Namun, ketika ingin menggunakan metode CNN dibutuhkan nilai validasi. Oleh karena itu, pada Gambar 4.7 terlihat kode (`train_test _split`) dilakukan sebanyak dua kali untuk membagi data latih, data validasi, dan data tes.

Pembagian rasio saat melakukan metode pelatihan SVM sebanyak 80% untuk data latih dan 20% untuk data tes. Pembagian tersebut berbeda ketika menggunakan CNN karena penggunaan CNN membutuhkan data validasi. Oleh karena itu, pembagian rasio ketika menggunakan metode CNN menjadi 80% data latih, 10% data validasi, dan 10% data tes.

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2)
```

Gambar 4.6 Kode ketika ingin membagi *dataset* tersebut dalam rasio yang diinginkan pada pelatihan SVM

```
X_train, X_val, y_train, y_val = train_test_split(X,y, test_size=0.20,
random_state=50)
X_val, X_test, y_val, y_test = train_test_split(X_val,y_val, test_size=
0.50, random_state= 50 )
```

Gambar 4.7 Kode ketika ingin membagi *dataset* tersebut dalam rasio yang diinginkan pada pelatihan CNN

Pembagian data sesuai dengan skenario pada Subbab 3.2.2. Percobaan dilakukan untuk mengetahui rasio pembagian terbaik di ketiga *dataset*. Hasil dari kinerja percobaan tersebut dapat dilihat pada Tabel 4.1 ketika menggunakan metode SVM, dan Tabel 4.2 ketika menggunakan metode CNN.

Pada Tabel 4.1 terlihat bahwa pembagian data latih sebanyak 80% dan data tes sebanyak 20% memiliki akurasi, presisi, f1 score yang lebih baik dibandingkan dua pembagian lainnya pada saat menggunakan metode SVM. Oleh karena itu, pada saat menggunakan metode SVM pembagian data latih sebanyak 80% dan data tes sebanyak 20% akan digunakan pada penelitian kali ini.

Pada Tabel 4.2 terlihat bahwa pembagian data latih sebanyak 80%, data validasi sebanyak 10%, dan data tes sebanyak 10% memiliki akurasi, presisi, f1 score yang lebih baik dibandingkan dua pembagian lainnya pada saat menggunakan metode CNN. Oleh karena itu, pada saat menggunakan metode klasifikasi CNN pembagian data latih sebanyak 80%, data validasi sebanyak 10%, dan data tes sebanyak 10% akan digunakan pada penelitian kali ini.

Tabel 4.1 Perbandingan akurasi ketika menggunakan jumlah pembagian data latih dan data validasi yang berbeda ketika menggunakan metode SVM

Dataset	**Data Latih**	**Data Validasi**	**Akurasi (%)**	**Presisi (%)**
Jl Corpus	**80**	**20**	**96**	**96**
Jl Corpus	70	30	94	95
Jl Corpus	60	40	93	94
RAVDESS	**80**	**20**	**63**	**61**
RAVDESS	70	30	60	59
RAVDESS	60	40	59	57
Gabungan	**80**	**20**	**69**	**65**
Gabungan	70	30	67	64
Gabungan	60	40	66	62

Tabel 4.2 Perbandingan akurasi ketika menggunakan jumlah pembagian data latih dan data validasi yang berbeda ketika menggunakan metode CNN

Dataset	**Data Latih**	**Data Validasi**	**Data Tes (%)**	**Akurasi (%)**	**Presisi (%)**
Jl Corpus	**80**	**10**	**10**	**95**	**95**
Jl Corpus	70	15	15	92	91
Jl Corpus	60	20	20	91	91
RAVDESS	**80**	**10**	**10**	**73**	**72**
RAVDESS	70	15	15	73	72
RAVDESS	60	20	20	70	71
Gabungan	**80**	**10**	**10**	**85**	**81**
Gabungan	70	15	15	76	77
Gabungan	60	20	20	78	76

Jumlah data yang digunakan pada tiap metode klasifikasi yang digunakan terlihat pada Tabel 4.3 untuk metode klasifikasi SVM dan Tabel 4.4 untuk metode klasifikasi CNN. Jumlah data tersebut didapatkan setelah melalui proses pembagian yang terjadi pada Gambar 4.6 dan Gambar 4.7.

Tabel 4.3 Jumlah suara yang digunakan saat melakukan metode SVM di ketiga *dataset*

Emosi	***Dataset***					
	Jl Corpus		**RAVDESS**		**Gabungan (RAVDESS dan Jl Corpus)**	
	Data Latih	**Data Validasi**	**Data Latih**	**Data Validasi**	**Data Latih**	**Data Validasi**
Netral	184	56	154	34	339	89
Tenang	-	-	302	74	296	80
Senang	201	39	304	72	500	116
Sedih	191	49	296	80	487	129
Marah	192	48	303	73	500	116
Takut	-	-	297	79	294	82
Jijik	-	-	150	42	151	41
Terkejut	-	-	155	37	162	30

Tabel 4.4 Jumlah suara yang digunakan saat melakukan metode CNN di ketiga *dataset*

Emosi	***Dataset***								
	Jl Corpus			**RAVDESS**			**Gabungan**		
	Data Latih	**Data Validasi**	**Data Tes**	**Data Latih**	**Data Validasi**	**Data Tes**	**Data Latih**	**Data Validasi**	**Data Tes**
Netral	184	28	28	154	15	19	339	44	45
Tenang	-	-	-	302	31	43	296	40	40
Senang	201	19	20	304	45	27	500	62	54
Sedih	191	26	23	296	39	41	487	71	58
Marah	192	23	25	303	32	41	500	58	58
Takut	-	-	-	297	44	35	294	33	49
Jijik	-	-	-	150	17	25	151	17	24
Terkejut	-	-	-	155	22	15	162	16	14

4.1.3 *Support Vector Machine* (SVM)

Dari percobaan yang telah dilakukan, semua tahapan penerapan model klasifikasi SVM dalam program yang telah dirancang dengan *flowchart* seperti pada Gambar 3.2 akan diimplementasikan sebagai berikut:

1. Melatih model SVM

   Penggunaan fungsi SVC dari sckit *package*s untuk melatih model yang dibuat.

```
#Kernel yang digunakan linear
start_time = time.time()
svclassifier = SVC(kernel='rbf', C = 1000)
svclassifier.fit(X_train, y_train)
print("--- Latih Data selesai, Latih data selama : %s detik ---" %
(time.time() - start_time))
```

Gambar 4.8 Kode ketika ingin melatih data yang sudah di split sebelumnya dengan metode SVM

Pada Gambar 4.8 terlihat beberapa parameter yang digunakan saat melatih model SVM. Kernel yang digunakan yaitu kernel RBF (Radial Basis Function) dan parameter C yang digunakan sebanyak 1000. Metode (`fit`) digunakan untuk melatih data latih yang ditentukan pada saat pembagian data latih dan data validasi. Kernel RBF dan parameter C ditentukan setelah melalui beberapa perbandingan menggunakan kernel beserta parameter C yang berbeda. Perbedaan tersebut dapat terlihat pada Tabel 4.5.

Tabel 4.5 Perbandingan menggunakan metode SVM di berbagai *dataset*

Dataset	**C**	**Waktu Latih Data (detik)**		**Akurasi (%)**	
		Linear	**RBF**	**Linear**	**RBF**
Jl Corpus	1	0.70	0.07	95	63
	100	7.79	0.04	94	88
	1000	44.00	**0.04**	93	**96**
RAVDESS	1	54.91	0.56	57	30
	100	438.65	0.47	54	58
	1000	793.00	**0.50**	55	**63**
Gabungan	1	82.88	1.00	64	37
	100	984.61	0.80	62	61
	1000	1394.44	**0.90**	58	**69**

Pada Tabel 4.5 terlihat bahwa kernel RBF dan jumlah C sebanyak 1000 menggungguli kernel linier beserta jumlah C yang lainnya. Kernel linear bahkan membutuhkan waktu yang sangat lama hingga menembus 1000 detik, berbeda dengan kernel RBF yang membutuhkan tidak lebih dari satu detik.

Pada saat menggunakan kernel linear, walaupun nilai C diperbesar tetapi tidak memberikan akurasi yang lebih baik, bahkan akurasi yang dihasilkan menurun. Sedangkan saat menggunakan kernel RBF, semakin banyak nilai C yang digunakan akurasi menjadi semakin baik. Kernel Linear mempunyai kinerja buruk disebabkan oleh data yang digunakan tidak bersifat linear. Dalam segi waktu yang dibutuhkan saat melatih kernel linear sangat banyak, bahkan ketika nilai C ditambahkan waktu yang dibutuhkan bertambah hingga dua kali lipat. Hal tersebut berbeda dengan kernel RBF, ketika nilai C ditambahkan waktu yang dibutuhkan berkurang.

2. Menguji model yang telah dibuat

Langkah terakhir setelah melatih model SVM adalah menggunakan model tersebut untuk mendeteksi emosi dari suara yang telah diunggah dari *dataset*. Model yang diuji menggunakan kernel RBF dan nilai C sebanyak 1000.

```
#(0 = neutral, 01 = calm, 02 = happy, 03 = sad, 04 = angry, 05 =
fearful, 06 = disgust, 07 = surprised).
y_pred = svclassifier.predict(X_test)
print(classification_report(y_test, y_pred))

matrixSVM = confusion_matrix(y_test, y_pred)
print (matrixSVM)
```

Gambar 4.9 Kode yang dibutuhkan ketika ingin menggunakan model tersebut untuk memprediksi yang kita inginkan

Prediksi kita lakukan menggunakan data uji yang telah dibagi sebelumnya, lalu kita bisa melihat kinerja dari model yang telah kita buat menggunakan fungsi (`classification_report`) yang terlihat pada Gambar 4.9.

Tabel 4.6 Perbandingan hasil akurasi, presisi, recall, dan f1-score pada tiap *dataset*

Dataset	**Waktu saat mendeteksi**		**Akurasi (%)**	**Presisi (%)**	**Recall (%)**
	Semua suara	**Satu suara**			
RAVDESS	0.05	0.0008	63	61	61
Jl Corpus	0.007	0.002	96	96	96
Gabungan	0.09	0.0025	69	65	65

Pada Tabel 4.6 menunjukkan bahwa model SVM yang telah dibuat, sangat cocok jika digunakan pada Jl Corpus *dataset*, karena kinerja yang dihasilkan diatas 90%. Namun nilai tersebut jatuh ketika menggunakan RAVDESS *dataset* hingga menyentuh 63% pada akurasi. Hal tersebut terjadi karena suara yang berasal dari RAVDESS *dataset* memiliki amplitudo yang beralun, hal itu menyebabkan amplitudo yang dihasilkan berbeda dibandingkan pada saat percakapan biasa. Ketika kedua *dataset* tersebut digabungkan nilai kinerja terlihat lebih baik hampir menyentuh angka 70% pada akurasinya. Meskipun nilai kinerja hampir menyentuh 70%, waktu untuk mendeteksi suara baik keseluruhan maupun hanya satu suara tidak sampai satu detik.

Tabel 4.7 Perbandingan nilai Presisi, *Recall*, *F1-score* di tiap kelas pada ketiga *dataset*

Kelas	**Presisi (%)**			**Recall (%)**		
	RAVDESS	**Jl Corpus**	**Gabungan**	**RAVDESS**	**Jl Corpus**	**Gabungan**
Netral	56	95	77	68	98	79
Tenang	64	-	56	82	-	71
Senang	64	93	80	58	97	81
Sedih	74	98	67	62	94	59
Marah	74	100	80	70	96	82
Takut	61	-	62	63	-	56
Jijik	61	-	56	48	-	34
Terkejut	38	-	41	38	-	57
Rata Rata	61.5	96.5	65.875	61.125	96.25	64.875

Pada Tabel 4.7 dapat dilihat nilai Presisi, *Recall*, *F1-score* yang dihasilkan oleh model yang telah dilatih pada Jl corpus *dataset* terlihat bagus bahkan ada yang menyentuh angka 100% pada presisi untuk kelas Sedih. Namun, hasil yang bagus tersebut tidak terlihat pada RAVDESS *dataset*. Angka yang dihasilkan ada yang menyentuh 38% pada presisi kelas Terkejut. Ketika digabungkan dengan Jl corpus *dataset* angka yang dihasilkan terlihat sedikit lebih baik karena pada Presisi, Recall, dan F1-score ada yang menyentuh angka 80% yaitu pada kelas Senang dan kelas Marah. Namun, untuk kelas Terkejut masih menjadi kelas yang memiliki nilai Presisi, Recall, dan F1-score yang terendah jika dibandingkan dengan kelas lainnya.

Tabel 4.8 Jumlah emosi yang benar dan yang tertebak ke kelas lain pada RAVDESS *dataset*

Kelas	Emosi Benar	Emosi yang tertebak ke kelas lain								Jumlah suara
		Netral	Tenang	Senang	Sedih	Marah	Takut	Jijik	Terkejut	
Netral	23	-	4	0	4	0	1	5	4	41
Tenang	61	4	-	5	14	2	1	6	2	95
Senang	42	0	2	-	1	6	7	6	2	66
Sedih	50	2	2	2	-	2	10	0	0	68
Marah	51	0	0	4	1	-	6	3	4	69
Takut	50	2	1	10	6	7	-	0	6	82
Jijik	20	0	4	1	1	1	1	-	5	33
Terkejut	14	3	0	8	3	4	3	2	-	37

Tabel 4.9 Jumlah emosi yang benar dan yang tertebak ke kelas lain pada Jl Corpus *dataset*

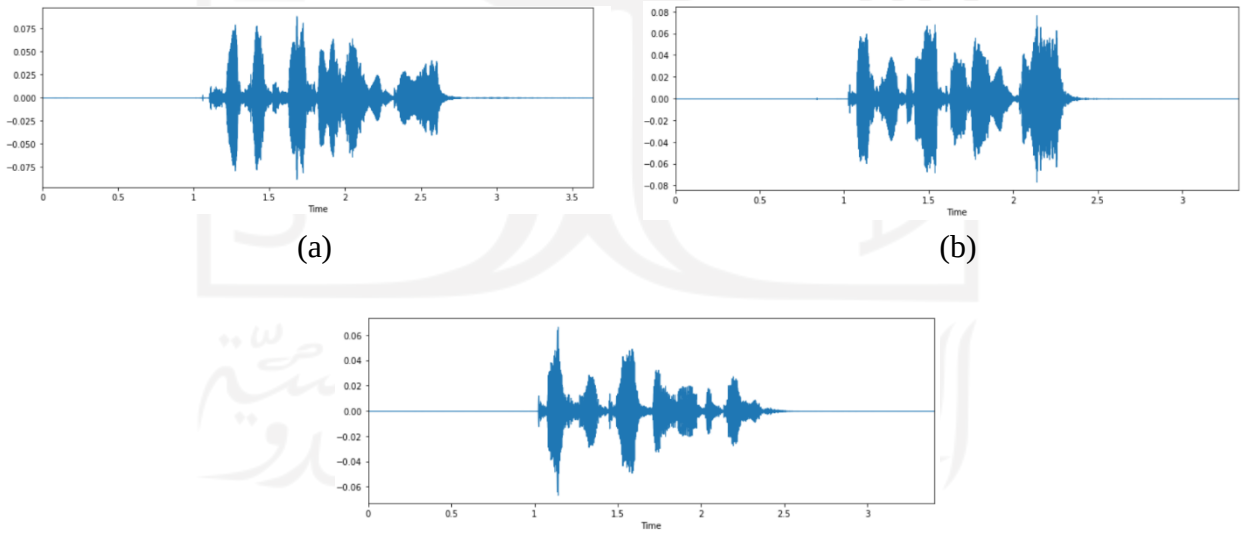
Kelas	Emosi Benar	Emosi yang tertebak ke kelas lain				Jumlah Suara
		Netral	Senang	Sedih	Marah	
Netral	55	-	1	2	0	58
Senang	38	0	-	1	2	41
Sedih	46	1	0	-	0	47
Marah	46	0	0	0	-	46

Tabel 4.10 Jumlah emosi yang benar dan yang tertebak ke kelas lain pada *dataset* gabungan (RAVDESS dan Jl Corpus)

Kelas	Emosi Benar	Emosi yang tertebak ke kelas lain								Jumlah Suara
		Netral	Tenang	Senang	Sedih	Marah	Takut	Jijik	Terkejut	
Netral	70	-	6	0	13	0	1	1	0	91
Tenang	57	6	-	5	14	1	6	10	2	101
Senang	94	1	3	-	3	7	5	2	3	118
Sedih	76	9	8	3	-	2	11	2	3	114
Marah	95	1	0	8	2	-	8	3	2	119
Takut	46	0	1	2	15	8	-	1	1	73
Jijik	14	2	4	1	1	1	0	-	2	25
Terkejut	17	0	1	3	5	2	5	8	-	41

Pada Tabel 4.8 RAVDESS *dataset* terlihat memiliki karakter suara yang menyerupai di beberapa kelas, seperti kelas tenang menyerupai dengan kelas sedih, kelas sedih menyerupai dengan kelas takut, dan kelas marah menyerupai dengan kelas senang. Hal tersebut berdasarkan jumlah suara terbanyak yang tertebak salah ke kelas lainnya. Pada Tabel 4.9 terlihat kelas Marah pada Jl Corpus dataset tidak memiliki suara yang tertebak ke kelas lain, sedangkan kelas Netral dan Senang memiliki suara yang tertebak ke kelas lain. Pada Tabel 4.10 terlihat jumlah emosi yang tertebak oleh model, walaupun RAVDESS dataset sudah digabungkan oleh Jl Corpus dataset masih memiliki banyak sekali emosi yang ditebak tidak benar. Emosi yang ditebak tidak benar diantaranya seperti kelas Netral, Tenang, dan Takut. Ketiga kelas emosi tersebut banyak sekali yang tertebak ke kelas emosi Sedih.

Dari ketiga percobaan di atas kelas emosi Marah mengungguli kelas emosi lainnya. Bahkan saat menggunakan Jl corpus *dataset* dapat menebak seluruh emosi dengan benar. Kelas emosi Sedih menjadi terbaik kedua setelah emosi Marah pada RAVDESS *dataset* dan Jl corpus *dataset*. Namun, jumlah emosi yang tertebak benar berkurang saat kedua *dataset* digabungkan. Kelas emosi Terkejut memiliki jumlah emosi yang tertebak sangat sedikit jika dibandingkan dengan kelas lainnya pada RAVDESS *dataset*.



(a) (b)

(c)

Gambar 4.10 Perbendingan amplitudo tiga suara pada RAVDESS *dataset*;
(a) data uji terkejut, (b) data latih senang, (c) data latih terkejut

Pada Gambar 4.10 terlihat data latih sudah menyerupai data tes dari kelas Terkejut, karena nilai amplitudo dan pola yang dimiliki sudah menyerupai. Namun, jika dibandingkan

dengan data latih pada kelas Senang, data latih tersebut memiliki amplitudo yang lebih mirip jika dibandingkan dengan data latih kelas Terkejut. Hal tersebut dibuktikan dengan nilai maksimal amplitudo pada data uji kelas Terkejut menyentuh angka 0.075. Angka tersebut lebih menyerupai data latih kelas Senang karena kelas tersebut memiliki angka amplitudo maksimal hingga 0.8, sedangkan pada data latih kelas Terkejut memiliki angka maksimal amplitudo hingga 0.6. Oleh karena itu, data uji terkejut masuk ke kelas Senang bukan ke kelas Terkejut.

4.1.4 *Convolutional Neural Network* (CNN)

Dari tiap percobaan yang dilakukan peneliti, semua tahapan dalam penerapan model klasifikasi CNN pada program yang telah dirancang dengan *flowchart* seperti pada Gambar 3.3 akan diimplementasikan sebagai berikut:

1. *Preprocessing*

Preprocessing dilakukan dengan menambahkan dimensi pada data yang digunakan. *Preprocessing* dilakukan agar CNN dapat berjalan dengan sempurna, karena format dari keluaran CNN ada tiga dimensi yaitu panjang *batch*, dimensi fitur, dan 1. Untuk melakukan *Preprocessing* dapat dilihat pada Gambar 4.11. Pada Gambar 4.12 terlihat belum ada angka satu sebelum di lakukan *Preprocessing*, namun setelah dilakukan *Preprocessing* terdapat angka satu seperti pada Gambar 4.13.

```
x_traincnn = np.expand_dims(X_train, axis=2)
x_valcnn = np.expand_dims(X_val, axis=2)
x_testcnn = np.expand_dims(X_test, axis=2)
```

Gambar 4.11 Kode ketika ingin mengekspansi data yang digunakan

```
((2729, 40), (341, 40), (342, 40))
```

Gambar 4.12 Jumlah data sebelum dilakukan ekspansi dimensi

```
((2729, 40, 1), (341, 40, 1), (342, 40, 1))
```

Gambar 4.13 Jumlah data setelah dilakukan ekspansi dimensi

2. Membangun arsitektur jaringan

Arsitektur jaringan yang dibuat sesuai dengan *flowchart* pada Gambar 3.4, terdapat lima lapis jaringan yang terdapat pada model yang peneliti buat.

Gambar 4.14 berisi kode dari arsitektur CNN yang ingin dibuat. Arsitektur yang digunakan mengadopsi dari arsitektur yang dibuat oleh salah satu pengguna (Gupta, 2020) dari Github (https://github.com/), lalu arsitektur tersebut diubah untuk menyesuaikan data yang digunakan. Pada lapis pertama terdapat beberapa layer yaitu, Konvolusi, fungsi aktivasi ReLU, *Dropout*, dan *Max Pooling*. Lapis kedua, ketiga, dan keempat memiliki jumlah layer yang sama dengan lapis pertama, sedangkan lapis kelima adalah lapis *fully connected* yang berisikan *Flatten*, *Dense*, dan fungsi aktivasi *Softmax*.

```
model = Sequential()

model.add(Conv1D(64, 7,padding='same',
                 input_shape=(40,1)))
model.add(Activation('relu'))
model.add(Dropout(0.2))
model.add(MaxPooling1D(pool_size=(2)))
model.add(Conv1D(128, 7,padding='same',))
model.add(Activation('relu'))
model.add(Dropout(0.2))
model.add(MaxPooling1D(pool_size=(2)))
model.add(Conv1D(256, 7,padding='same',))
model.add(Activation('relu'))
model.add(Dropout(0.6))
model.add(MaxPooling1D(pool_size=(2)))
model.add(Conv1D(256, 7,padding='same',))
model.add(Activation('relu'))
model.add(Dropout(0.6))
model.add(MaxPooling1D(pool_size=(2)))
model.add(Flatten())
model.add(Dense(256))
model.add(Activation('softmax'))
model.summary()
```

Gambar 4.14 Kode lima lapis jaringan yang dibuat oleh penelitan.

Pada percobaan kali ini, lapisan pertama menggunakan filter sebanyak 64, dan ukuran kernel di tiap filter ada tujuh. Padding yang digunakan adalah “same”, arti dari hal tersebut adalah kita dapat mengatur dimensi *output* agar tetap sama seperti dimensi input. Pada bagian (`input_shape`) sesuai dari data yang telah dimiliki, karena jumlah nilai MFCCs yang digunakan sebanyak 40 maka (`input_shape`) yang digunakan juga 40. Layer aktivasi terdapat banyak jenisnya, dan kali ini menggunakan ReLU sebagai layer aktivasinya. Layer selanjutnya adalah *Dropout*, layer tersebut akan mengatasi *overfitting*. Layer terakhir pada lapis pertama adalah *Max Pooling*.

Pada lapisan kedua, ketiga, dan keempat hampir sama dengan lapisan pertama, yang membedakan hanyalah pada jumlah *Filter* dan jumlah *Dropout* yang digunakan. Jumlah *Filter* yang digunakan adalah 128 pada lapisan kedua, 256 pada lapisan ketiga dan keempat. Nilai *Dropout* yang digunakan pada *layer* pertama dan kedua sebanyak 0.2, sedangkan pada *layer* ketiga dan keempat sebanyak 0.6.

Pada lapisan kelima atau bisa disebut *fully connected layer*. Lapisan *Flatten* berfungsi untuk merubah hasil yang telah dilalui dua layer sebelumnya menjadi satu kolom. Hasil dari lapisan *Flatten* itu akan dipindahkan ke layer selanjutnya, yaitu lapisan *Dense*. Lapisan *Dense* ini yang akan menghasilkan nilai *output*. Selanjutnya nilai tersebut akan diproses oleh lapisan *Softmax*, lapisan tersebut berfungsi untuk menentukan kelas dari tiap suara. Hasil dari arsitektur jaringan yang dibuat dapat dilihat pada Gambar 4.15. Arsitektur jaringan ini akan digunakan pada ketiga *dataset* yang telah ditentukan sebelumnya.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 40, 64)            512
_________________________________________________________________
activation (Activation)      (None, 40, 64)            0
_________________________________________________________________
dropout (Dropout)            (None, 40, 64)            0
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 20, 64)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 20, 128)           57472
_________________________________________________________________
activation_1 (Activation)    (None, 20, 128)           0
_________________________________________________________________
dropout_1 (Dropout)          (None, 20, 128)           0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 10, 128)           0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 10, 256)           229632
_________________________________________________________________
activation_2 (Activation)    (None, 10, 256)           0
_________________________________________________________________
dropout_2 (Dropout)          (None, 10, 256)           0
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 5, 256)            0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 5, 256)            459008
_________________________________________________________________
activation_3 (Activation)    (None, 5, 256)            0
_________________________________________________________________
dropout_3 (Dropout)          (None, 5, 256)            0
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 2, 256)            0
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 256)               131328
_________________________________________________________________
activation_4 (Activation)    (None, 256)               0
=================================================================
Total params: 877,952
Trainable params: 877,952
Non-trainable params: 0
```

Gambar 4.15 Alur dari arsitektur CNN yang telah dibuat.

Arsitektur yang telah dibuat akan dibandingkan dengan arsitektur sumber. Perbandingan arsitektur pada penelitian kali ini dengan arsitektur sumber terletak pada ukuran kernel, nilai dropout, optimizer yang digunakan dan jumlah lapisan dari arsitektur. Perbedaan dapat dilihat pada Tabel 4.11.

Tabel 4.11 Perbedaan arsitektur jaringan sumber dan setelah diubah.

Perbedaan	**Arsitektur Jaringan**	
	Sumber	**Pada penelitian kali ini**
Lapis jaringan	4	5
Optimizer yang digunakan	RMSprop	Adam
Ukuran Kernel Konvolusi	5	7
Nilai Dropout	0.1	0.2 dan 0.6
Not pada Layer Dense	8	256

Nilai yang ditentukan pada arsitektur pada penelitian kali ini didapatkan setelah melalui beberapa kali percobaan. Percobaan yang dilalui ketika menggunakan dataset Gabungan dapat dilihat pada Tabel 4.12. Pada Tabel 4.12 terlihat bahwa nilai perbedaan pada arsitketur penelitian kali ini terbukti memiliki nilai yang terbaik.

Tabel 4.12 Hasil percobaan menggunakan arsitektur yang berbeda pada dataset Gabungan.

Lapis Jaringan	**Optimizer**	**Ukuran Kernel Konvolusi**	**Nilai Dropout**	**Not pada Layer Dense**	**Akurasi (%)**
4	RMSprop	5	0.1	8	77
4	RMSprop	7	0.1	8	82
4	RMSprop	7	0.2 dan 0.6	256	80
4	Adam	5	0.2 dan 0.6	256	82
4	Adam	7	0.1	8	81
4	Adam	7	0.2 dan 0.6	256	80
5	RMSprop	5	0.1	8	78
5	RMSprop	7	0.1	8	81
5	RMSprop	7	0.2 dan 0.6	256	81
5	Adam	5	0.2 dan 0.6	256	79
5	Adam	7	0.1	8	80
5	**Adam**	**7**	**0.2 dan 0.6**	**256**	**85**

Perbandingan hasil ketiga dataset dari kedua arsitektur dapat dilihat pada Tabel 4.13. Pada Tabel 4.13 terlihat arsitektur sumber memiliki nilai kinerja yang buruk ketika menggunakan RAVDESS dataset. Selisih nilai akurasi yang dihasilkan oleh kedua arsitektur pada RAVDESS dataset menyentuh 15%. Selisih yang cukup jauh tersebut menunjukkan arsitektur yang telah diubah memiliki kinerja yang lebih baik jika dibandingkan arsitektur sumber.

Tabel 4.13 Perbedaan hasil dari arsitektur sumber dan arsitektur pada penelitian kali ini.

Dataset	**Arsitektur sumber**			**Arsitektur pada penelitian kali ini**		
	Akurasi (%)	**Presisi (%)**	**Recall (%)**	**Akurasi (%)**	**Presisi (%)**	**Recall (%)**
RAVDESS	58	58	55	73	72	74
Jl Corpus	83	88	83	95	95	94
Gabungan	72	70	71	85	83	83

3. Pelatihan arsitektur yang telah dibuat

Arsitektur yang telah dibuat akan dilatih sebanyak 200 *epoch* (Han, 2018). Kode untuk melakukan pelatihan pada model bisa dilihat pada Gambar 4.16.

```
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',
              metrics=['accuracy'])
cnnhistory=model.fit(x_traincnn, y_train, batch_size=64, epochs=200,
                     validation_data=(x_testcnn, y_test))
```

Gambar 4.16 Kode ketika ingin melatih data tersebut menggunakan model yang telah dibuat sebelumnya

Pada Gambar 4.16 terlihat kode (`model.compile`), kode tersebut berfungsi untuk mengkonfigurasi dari model pada saat melakukan latih pada data. Dari kode tersebut kita bisa melihat bahwa untuk bagian *loss* pada model kita menggunakan fungsi (`sparse_categorical_crossentropy`). Fungsi ini digunakan karena label dari data kita adalah kategorikal. Parameter lainnya adalah *optimizer* dan *metrics*. *Optimizer* diperlukan karena di dalamnya ada yang dibutuhkan saat melakukan compile pada model. *Optimizer* yang digunakan adalah 'adam', *optimizer* ini dipercaya sebagai salah satu yang terbaik jika dibandingkan yang lainnya. Hal tersebut disebabkan *learning rate* yang digunakan selalu berubah menyesuaikan data latih yang digunakan di setiap *epoch*nya, sedangkan optimizer yang lain tidak melakukan hal itu. Pada parameter *metrics* tertulis *accuracy*, karena ingin

melihat performa model yang digunakan berdasarkan akurasi yang dihasilkan dari model yang telah dibuat. Selanjutnya terdapat kode (`model.fit`), kode tersebut berfungsi untuk melatih model setiap *epoch* nya. Di dalamnya terdapat beberapa parameter, jumlah *epoch* yang ingin digunakan sebanyak 200 (Han, 2018), dan jumlah (`batch_size`) yang kita gunakan sebanyak 64. Penentuan jumlah *batch size* yang digunakan berdasarkan perbandingan dengan beberapa jumlah batch size lainnya. Hal tersebut dapat dilihat pada Tabel 4.12.

Tabel 4.14 Perbandingan ketika menggunakan tiga batch size yang berbeda di tiap *dataset*

Dataset	**Batch size**	**Waktu Latih Data (detik)**	**Akurasi (%)**
Jl Corpus	32	314.64	97
	64	**272.96**	**95**
	128	252.51	93
RAVDESS	32	803.37	70
	64	**727.38**	**73**
	128	683.33	72
Gabungan	32	1114.22	79
	64	**983.31**	**85**
	128	865.62	83

Pada Tabel 4.12 terlihat *batch size* berjumlah 64 mengungguli di Akurasi dan *F1-score* hampir di tiap *dataset*, hanya saat menggunakan Jl corpus *batch size* berjumlah 32 lebih baik jika dibandingkan *batch size* berjumlah 64. *Batch size* sebanyak 128 lebih baik hanya dalam waktu latih data yang dilakukan oleh model yang telah dibuat, namun selisih yang diperoleh tidak berbeda jauh dengan batch 64. Oleh karena itu, *batch size* berjumlah 64 akan digunakan selanjutnya.

Nilai *Loss* dan Akurasi pada training dan validasi sebanyak 200 *epoch* dapat dilihat pada Tabel 4.13. Pada Tabel 4.13 bisa dilihat *epoch* ke-200 pada RAVDESS *dataset*, nilai Akurasi dan Loss pada data latih maupun data validasi memiliki nilai yang terbaik jika dibandingkan dengan *epoch* lainnya. Hal tersebut berlaku ketika menggunakan *dataset* gabungan. Berbeda ketika menggunakan Jl Corpus *dataset*, nilai dari Akurasi dan Loss yang terbaik terjadi di *epoch* ke-140 dan ke-160, selebihnya angka yang dihasilkan bahkan menurun. Namun, model pada *epoch* ke-200 tetap digunakan, karena penurunan yang dialami tidaklah banyak.

Tabel 4.15 Tabel perbedaan akurasi dan *loss* pada data tes maupun data validasi per *epoch*; a. RAVDESS, b. Jl Corpus, dan c. Gabungan (RAVDESS dan Jl Corpus)

Epoch	**Akurasi (Latih)**			***Loss* (Latih)**			**Akurasi (Validasi)**			***Loss* (Validasi)**		
	a	**b**	**c**	**a**	**B**	**c**	**a**	**b**	**c**	**a**	**b**	**c**
1	0.10	0.19	0.12	8.94	12.03	16.44	0.13	0.23	0.21	3.38	2.61	3.93
20	0.40	0.66	0.42	1.49	0.72	1.43	0.49	0.68	0.46	1.70	0.97	1.51
40	0.56	0.81	0.55	1.14	0.42	1.12	0.54	0.81	0.61	1.27	0.57	1.15
60	0.67	0.87	0.61	0.87	0.32	1.01	0.68	0.85	0.66	1.02	0.46	0.99
80	0.74	0.94	0.64	0.66	0.15	0.87	0.71	0.89	0.72	0.90	0.30	0.84
100	0.81	0.95	0.73	0.50	0.10	0.71	0.70	0.90	0.71	0.83	0.22	0.78
120	0.86	0.96	0.75	0.39	0.10	0.64	0.71	0.87	0.77	0.79	0.25	0.68
140	0.90	0.96	0.79	0.28	0.10	0.55	0.72	**0.95**	0.78	0.70	**0.19**	0.65
160	0.89	**0.98**	0.79	0.28	**0.03**	0.51	0.74	0.91	0.80	0.68	0.25	0.58
180	0.90	0.97	0.84	0.27	0.04	0.43	0.73	0.93	0.78	0.71	0.21	0.59
200	**0.91**	0.97	**0.84**	**0.22**	0.05	**0.39**	**0.76**	0.92	**0.84**	**0.66**	0.22	**0.55**

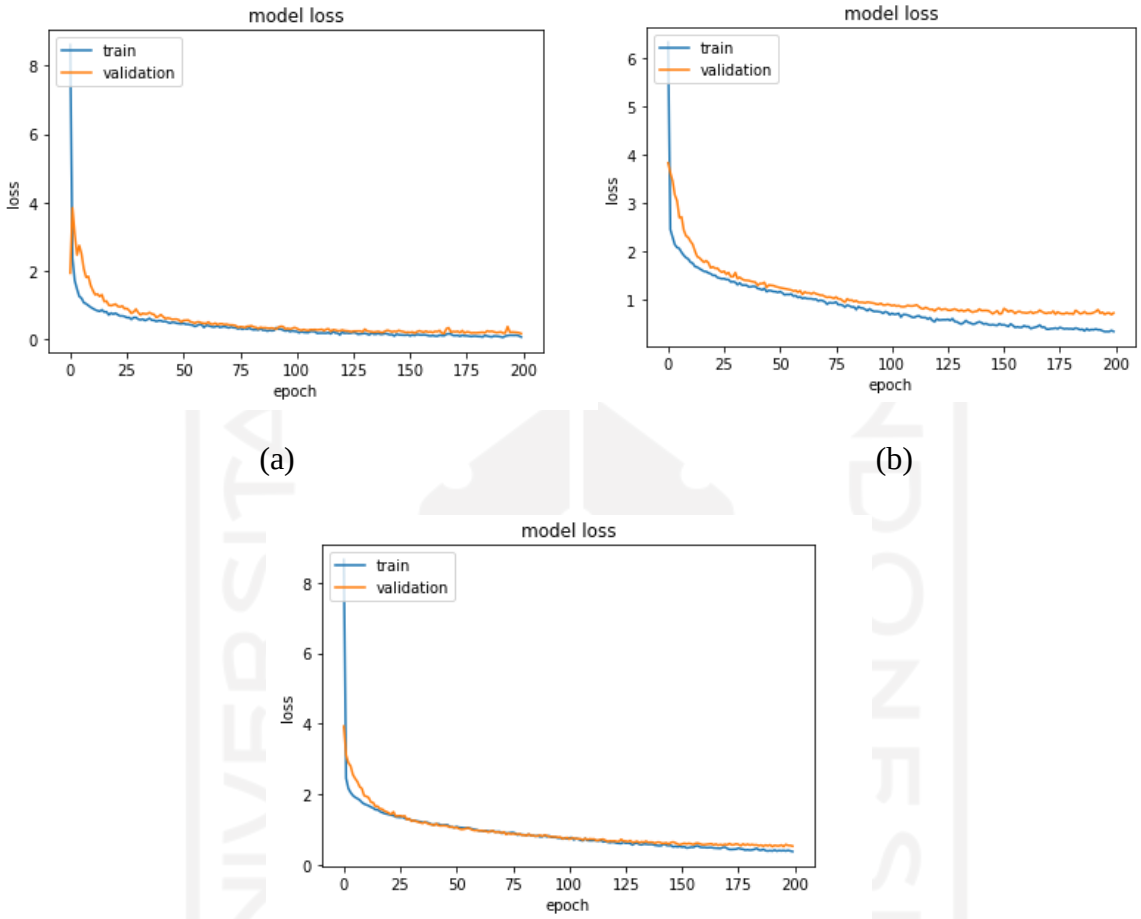
Susunan kode untuk menampilkan grafik perkembangan nilai loss di setiap *epoch* saat pelatihan CNN ditunjukkan pada Gambar 4.17. Sedangkan perkembangan nilai akurasinya ditunjukkan pada Gambar 4.19. Grafik tersebut ditampilkan untuk mengetahui apakah model yang dibuat mengalami overfitting atau tidak.

```
plt.plot(cnnhistory.history['loss'])
plt.plot(cnnhistory.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
```

Gambar 4.17 Kode untuk melihat perkembangan *loss* di tiap *epoch* saat melatih model CNN

Saat ingin melihat perkembangan dari Loss pada model yang telah dibuat bisa menggunakan kode dari Gambar 4.17. Fungsi (`plt`)digunakan untuk menampilkan hal tersebut di tiap *epoch*-nya. Fungsi (`plt.plot`) digunakan untuk menampilkan apa yang ingin dilihat, pada kali ini kita ingin melihat Loss dari data latih dan data validasi. Judul dari plot tersebut juga bisa diganti sesuai dengan apa yang kita inginkan dengan fungsi (`plt.title`).

Fungsi (`plt.xlabel`) dan (`plt.ylabel`) digunakan untuk memberi nama pada sumbu x dan y pada plot kita. Fungsi (`plt.legend`) untuk memberikan tanda pada plot yang telah dibuat.



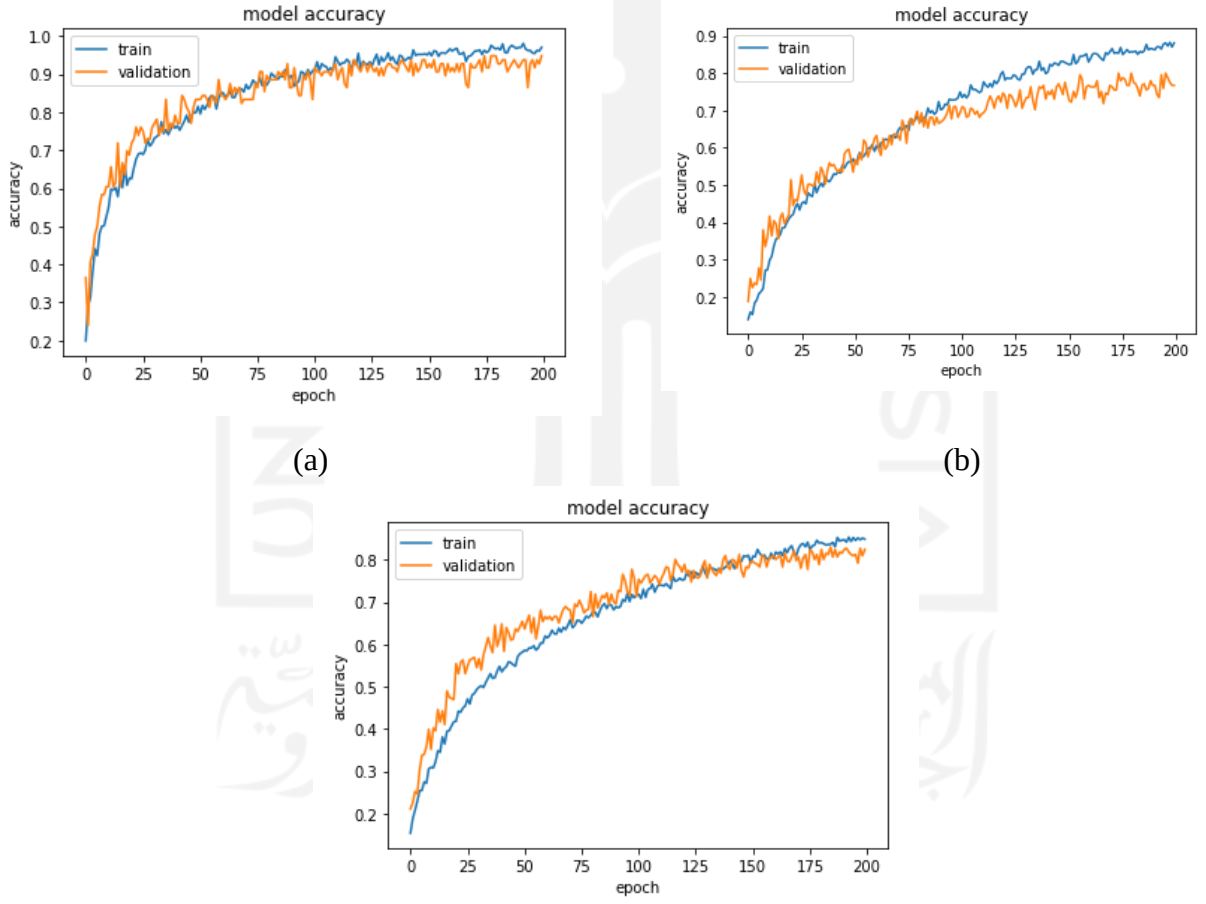
Gambar 4.18 Grafik *loss* dari model CNN ketika melatih data;
(a) Jl Corpus, (b) RAVDESS, dan (c) gabungan
(b)

Loss yang sedikit akan membuat model tersebut terlihat lebih baik, karena tujuan dari fungsi tersebut adalah untuk menghitung kuantitas yang harus diminimalkan oleh model saat selama pelatihan (Chollet, 2015). Pada Gambar 4. 18 bisa dilihat bahwa *loss* yang dialami oleh model yang telah kita buat sudah cukup rendah hampir menyentuh angka 0. Angka tersebut cukup rendah baik dari data latih maupun data validasi. Pada RAVDESS *dataset* terjadi overfitting pada *epoch* 125 hingga 200, tetapi overfitting yang terjadi tidak terlalu besar sehingga grafik yang terjadi masih cukup bagus.

```
plt.plot(cnnhistory.history['accuracy'])
plt.plot(cnnhistory.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'validation'], loc='upper left')
plt.show()
```

Gambar 4.19 Kode untuk melihat perkembangan akurasi di tiap *epoch* saat melatih model CNN

Perkembangan dari akurasi pada model yang telah dibuat bisa menggunakan kode yang dapat dilihat pada Gambar 4.15. Kode yang tertulis hampir sama ketika kita ingin melihat perkembangan dari *loss*, yang berbeda hanya pada fungsi (`plt.plot`). Pada fungsi tersebut diganti dengan (`accuracy`) untuk data latih dan (`val_accuracy`) untuk data validasi.



Gambar 4.20 Grafik akurasi dari model CNN ketika melatih data; (a) Jl Corpus, (b) RAVDESS, dan (c) gabungan.

Semakin tinggi accuracy maka semakin bagus pula model yang telah dibuat, karena accuracy ini menghitung seberapa sering prediksi kita sesuai dengan label (Chollet, 2015). Pada Gambar 4.20 bisa kita lihat bahwa model yang kita buat memiliki akurasi yang bagus pada Jl corpus *dataset*, tetapi pada RAVDESS *dataset* terjadi *overfitting* yang cukup besar pada data validasi. Tetapi, setelah digabungkan dengan Jl corpus *dataset*, *overfitting* tidak terjadi lagi.

4. Pengujian model yang telah dibuat

Model yang telah dilatih akan diuji kinerjanya terhadap data tes yang telah ditentukan sebelumnya. Pada Gambar 4.21 dapat dilihat kita menggunakan kode (`model.predict`) untuk memprediksi data tes menggunakan model yang telah kita buat. Lalu hasil kinerja dari model dapat dilihat dengan bantuan fungsi (`classification_report`). Perbandingan hasil Akurasi, Presisi, *Recall*, dan *F1-score* di tiap *dataset* bisa dilihat pada Tabel 4.14.

```
#(0 = neutral, 01 = calm, 02 = happy, 03 = sad, 04 = angry, 05 = fearfu
l, 06 = disgust, 07 = surprised).

predictionCNN = model.predict_classes(x_testcnn)
report = classification_report(y_test, predictionCNN)
print(report)

matrix = confusion_matrix(y_test, predictionCNN)
print (matrix)
```

Gambar 4.21 Grafik akurasi dari model CNN ketika melatih data; (a) Jl Corpus, (b) RAVDESS, dan (c) gabungan

Tabel 4.16 Perbandingan hasil Akurasi, Presisi, *Recall*, dan *F1-score* pada tiap *dataset* menggunakan metode CNN

Dataset	**Waktu saat mendeteksi (detik)**		**Akurasi (%)**	**Presisi (%)**	**Recall (%)**
	Semua suara	**Satu suara**			
RAVDESS	0.46	0.04	73	72	74
Jl Corpus	0.41	0.44	95	95	94
Gabungan	0.66	0.04	85	83	83

Pada Tabel 4.14 dapat dilihat model CNN yang telah dibuat memiliki nilai yang cukup bagus pada Akurasi, Presisi, Recall, maupun F1-score. RAVDESS *dataset* memiliki nilai kinerja yang masih dapat diterima karena nilai kinerja masih di atas 70%. Waktu untuk

mendeteksi suara sudah cukup baik di semua *dataset*, tidak ada yang menyentuh satu detik. Nilai Presisi, *Recall*, dan *F1-score* yang dihasilkan tiap kelas pada ketiga *dataset* bisa dilihat pada Tabel 4.15.

Tabel 4.17 Perbandingan nilai *Presisi*, *Recall*, dan *F1-score* di tiap kelas pada ketiga *dataset*

Kelas	**Presisi (%)**			**Recall (%)**		
	RAVDESS	Jl Corpus	Gabungan	RAVDESS	Jl Corpus	Gabungan
Netral	65	93	91	58	100	89
Tenang	85	-	93	77	-	93
Senang	76	90	84	81	90	94
Sedih	74	100	90	63	96	79
Marah	76	96	84	76	92	93
Takut	76	-	80	71	-	73
Jijik	79	-	84	76	-	67
Terkejut	46	-	61	87	-	89
Rata Rata	72	95	83	74	94	83

Dari Tabel 4.15 dapat dilihat Presisi, *Recall*, dan *F1-score* pada ketiga *dataset* sudah cukup baik, tetapi kelas Terkejut pada RAVDESS *dataset* memiliki presisi yang cukup buruk hingga menyentuh 46%. Tetapi hal tersebut menjadi lebih baik ketika digabungkan dengan Jl corpus *dataset* hingga menyentuh 61%. Tidak hanya kelas Terkejut yang menjadi lebih baik, tetapi kelas lainnya pun banyak yang menjadi lebih baik seperti kelas Netral yang memiliki presisi 65% naik menjadi 91%. Jumlah emosi yang ditebak dapat diketahui dengan bantuan kode (`confession_matrix`). Hasil dari (`confession_matrix`) tersebut dipaparkan pada Tabel 4.18, 4.19, dan 4.20.

Tabel 4.18 Jumlah emosi yang benar dan yang tertebak ke kelas lain pada RAVDESS *dataset*

Kelas	Emosi Benar	Emosi yang tertebak ke kelas lain								Jumlah Suara (Prediksi)
		Netral	Tenang	Senang	Sedih	Marah	Takut	Jijik	Terkejut	
Netral	11	-	5	0	1	0	0	0	0	17
Tenang	33	2	-	0	4	0	0	0	0	39
Senang	22	0	2	-	1	2	2	0	0	29
Sedih	26	1	2	0	-	4	1	1	0	35
Marah	31	1	0	2	2	-	4	0	1	41
Takut	25	1	0	0	4	2	-	0	1	33
Jijik	19	0	1	0	2	1	1	-	0	24
Terkejut	13	3	0	3	1	1	2	5	-	28

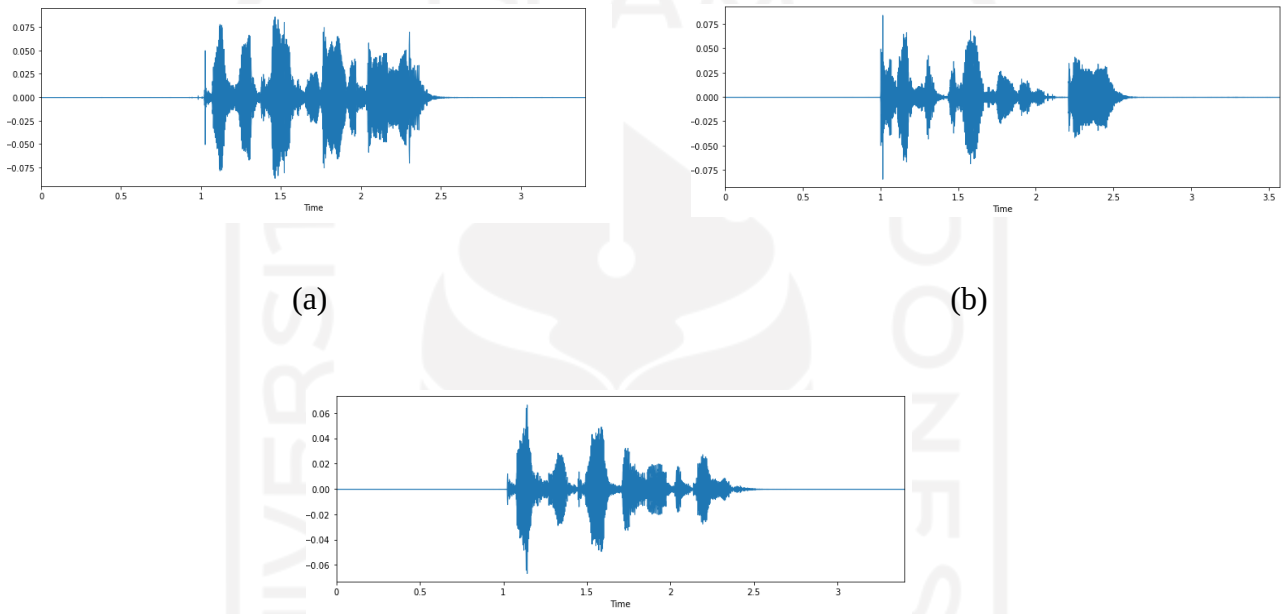
Tabel 4.19 Jumlah emosi yang benar dan yang tertebak ke kelas lain pada Jl Corpus *dataset*

Kelas	Emosi Benar	Emosi yang tertebak ke kelas lain				Jumlah Suara (Prediksi)
		Netral	Senang	Sedih	Marah	
Netral	28	-	1	1	0	30
Senang	18	0	-	0	2	20
Sedih	22	0	0	-	0	22
Marah	23	0	1	0	-	24

Tabel 4.20 Jumlah emosi yang benar dan yang tertebak ke kelas lain pada *dataset* gabungan (RAVDESS dan Jl Corpus)

Kelas	Emosi Benar	Emosi yang tertebak ke kelas lain								Jumlah Suara (Prediksi)
		Netral	Tenang	Senang	Sedih	Marah	Takut	Jijik	Terkejut	
Netral	40		1	0	2	0	0	0	1	44
Tenang	37	2	-	0	1	0	0	0	0	40
Senang	51	0	2		1	1	5	1	0	61
Sedih	46	1	0	0		0	4	0	0	51
Marah	54	0	0	1	0		2	5	2	64
Takut	36	0	0	1	5	2		1	0	45
Jijik	16	2	0	0	1	0	0		0	19
Terkejut	11	0	0	1	2	1	2	1	-	18

Pada Tabel 4.18 terlihat masih banyak kelas yang tertebak salah., seperti kelas netral banyak yang tertebak ke kelas tenang, kelas tenang banyak yang tertebak ke kelas sedih, hingga kelas terkejut banyak yang tertebak ke kelas jijik. Namun, Jl corpus *dataset* sangat minim sekali salah tertebak ke kelas lain, hal tersebut terbukti dari Tabel 4.19. Jumlah emosi yang salah berkurang cukup banyak setelah digabungkan dengan Jl corpus, hal tersebut berdasarkan Tabel 4.20. Banyaknya emosi yang salah ke kelas tertentu disebabkan karakter suara pada suatu kelas menyerupai kelas lainnya, seperti kelas netral dan sedih, lalu kelas terkejut dan senang.



(a) (b)

(c)

Gambar 4.22 Perbendingan amplitudo tiga suara pada RAVDESS *dataset*; (b) data uji terkejut, (b) data latih jijik, (c) data latih terkejut

Pada Gambar 4.22 terlihat data latih sudah menyerupai data tes dari kelas Terkejut, karena nilai amplitudo dan pola yang dimiliki sudah menyerupai. Namun, jika dibandingkan dengan data latih pada kelas Jijik, data latih tersebut memiliki amplitudo yang lebih mirip jika dibandingkan dengan data latih kelas Terkejut. Hal tersebut dibuktikan dengan nilai maksimal amplitudo pada data uji kelas Terkejut menyentuh angka 0.075. Angka tersebut lebih menyerupai data latih kelas Jijik karena kelas tersebut memiliki angka amplitudo maksimal hingga 0.75, sedangkan pada data latih kelas Terkejut memiliki angka maksimal amplitudo hingga 0.6. Oleh karena itu, data uji terkejut masuk ke kelas Jijik bukan ke kelas Terkejut.

4.2 Perbedaan hasil implementasi

Tahap perbandingan ini dilakukan untuk membandingkan hasil dari ketiga percobaan yang telah dilakukan. Namun perbandingan terkesan tidak adil mengingat metode SVM menggunakan data tes sebanyak 20% sedangkan metode CNN hanya menggunakan data tes sebanyak 10%. Oleh karena itu perbandingan yang dilakukan menggunakan data tes sebanyak 10% di kedua metode. Tabel 4.19 menunjukkan hasil dari tiap percobaan yang telah dilakukan.

Tabel 4.21 Perbandingan Tiap Percobaan

Model	***Dataset***	**Akurasi (%)**	**Presisi (%)**	**Recall (%)**	**Waktu Deteksi Satu Suara (Detik)**
SVM	RAVDESS	63	61	61	0.0011
	Jl Corpus	96	96	96	0.0009
	Gabungan	68	62	62	0.0022
Rata rata		**75,6**	**73**	**73**	**0.0017**
CNN	RAVDESS	73	72	74	0.04
	JL Corpus	95	95	94	0.44
	Gabungan	85	83	83	0.04
Rata rata		**84,3**	**83,3**	**83,6**	**0.17**

Berdasarkan perbandingan dari Tabel 4.21, dapat diketahui bahwa metode SVM dan CNN memiliki kelebihan dan kekurangan. Dari kedua metode yang digunakan memiliki satu kesamaan yaitu, kelas Terkejut memiliki nilai Presisi yang tidak baik di ketiga database. Kelas Terkejut memiliki nilai Presisi yang tidak baik dikarenakan amplitudo yang dihasilkan menyerupai kelas emosi yang lainnya. Hal tersebut dikarenakan emosi terkejut bisa bersifat positif, negatif, atau netral (Cherry, 2021).

Metode CNN memiliki kinerja yang baik saat menggunakan RAVDESS dataset. Akurasi yang didapatkan sebesar 73%, Nilai akurasi yang baik ketika menggunakan metode CNN masih jauh jika dibandingkan dengan penelitian sebelumnya (Pinto dkk., 2020) yang mencapai nilai 91% pada Akurasi. Hal tersebut dikarenakan jumlah data yang digunakan berbeda pada penelitian kali ini. Penelitian kali ini hanya menggunakan 2452 file dan hanya data suara yang digunakan. Sedangkan, penelitian yang dilakukan oleh Pinto (2020) menggunakan semua data RAVDESS, yaitu suara dan video dengan total sebanyak 7000 lebih.

Hal tersebut juga berlaku saat menggunakan metode SVM. Walau kinerja pada metode SVM yang digunakan pada penelitian kali ini tidak begitu bagus, yaitu 63%, tetapi akurasi yang dihasilkan tersebut lebih baik dari penelitian sebelumnya, yaitu 51% (Shegokar dkk., 2016). Hal tersebut dikarenakan kernel yang digunakan berbeda saat melakukan percobaan. Shegokar (2016) menggunakan kernel Linear sedangkan penelitian kali ini menggunakan kernel RBF. Jumlah data yang digunakan saat penelitian kali ini beserta penelitian yang dilakukan oleh Shegokar (2016) sama. Hal tersebut membuktikan bahwa kernel RBF lebih baik jika dibandingkan kernel Linear ketika data yang digunakan tidak linear.

Pada saat digabungkan dengan Jl Corpus dataset, nilai Akurasi beserta nilai rata rata dari Presisi, Recall, dan F1-score terlihat lebih baik ketika menggunakan metode CNN dan SVM. Pada Tabel 4.14 dapat dilihat ketika menggunakan CNN nilai Presisi dan Recall dari hampir tiap emosi naik. Namun, hal tersebut berbeda ketika menggunakan SVM, pada Tabel 4.7 terlihat nilai Presisi dan Recall masih banyak yang turun. Oleh karena itu, CNN memiliki kemampuan generalisir yang lebih baik jika dibandingkan dengan SVM ketika data yang digunakan cukup banyak.

Berdasarkan hal di atas, metode SVM memiliki kinerja yang lebih buruk jika dibandingkan dengan metode CNN. Namun, metode SVM memiliki kelebihan dalam kecepatan saat mendeteksi satu suara. SVM lebih cepat 0.0392 detik jika dibandingkan dengan CNN. Walaupun metode SVM memiliki kelebihan pada kecepatan saat mendeteksi satu suara, metode tersebut tidak mengahasilkan kinerja yang baik (73%).

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pada penelitian ini dapat diperoleh kesimpulan sebagai berikut:

a. Implementasi *Convolutional Neural Network* (CNN) dan *Support Vector Machine* (SVM) untuk mendeteksi emosi emosi melalui suara dilakukan dengan dua tahapan utama yaitu ekstraksi fitur suara yang dilakukan oleh MFCC (Mel Frequency Cepstrum Coefficients). Tahapan dilanjutkan dengan membuat model yang cocok di tiap *dataset* yang digunakan (SVM atau CNN).
b. Metode klasifikasi CNN memiliki akurasi yang lebih baik jika dibandingkan dengan motode klasifikasi SVM, hal tersebut berlaku di ketiga dataset yang telah diuji. Pada RAVDESS dataset, CNN mendapatkan akurasi sebesar 73% sedangkan SVM hanya menyentuh 63%. Jika menggunakan Jl Corpus dataset kedua metode klasifikasi memiliki akurasi yang akurat yaitu, 96% untuk metode klasifikasi SVM dan 95% untuk metode klasifikasi CNN. Ketika dua dataset tersebut digabungkan, metode klasifikasi SVM memiliki akurasi yang lebih baik sebesar 68%. Namun, metode klasifikasi CNN memiliki akurasi yang jauh lebih baik sebesar 85%.
c. Waktu yang dibutuhkan untuk mendeteksi satu suara saat menggunakan metode SVM selalu lebih sedikit dibandingkan dengan metode CNN.

5.2 Saran

Masih banyak kekurangan dalam penelitian ini, sehingga perlu dibuat beberapa saran dalam penelitian selanjutnya. Saran berikut dapat dipertimbangkan untuk penelitian selanjutnya:

1. Data *input* (suara) dari penelitian ini masih menggunakan bahasa Inggris sehingga perlu dilakukan penelitian selanjutnya terkait data *input* (suara) yang menggunakan bahasa indonesia.
2. Pengembangan deteksi emosi diharapkan dapat menggunakan lebih banyak emosi yang beragam pada penelitian selanjutnya.
3. Penelitian selanjutnya diharapkan proses deteksi dapat ditemukan algoritma yang lebih efisien sehingga latih data yang dilakukan tidak cukup lama.

DAFTAR PUSTAKA

Ahmad, M., Aftab, S. (2017). Analyzing the Performance of SVM for Polarity Detection with Different Datasets. *International Journal of Modern Education and Computer Science*, (pp. 29-36).

Anggraini, N.A., & Fadillah, N. (2018). Analisis Deteksi Emosi Manusia dari Suara Percakapan Menggunakan Matlab dengan Metode KNN. *Jurnal Nasional Informatika dan Teknologi Jaringan*, 3(2), 176-179.

Anjaini, A.S., Gautama, A.P., Anggis, N.S. (2019). Implementasi dan Analisis Simulasi Deteksi Emosi Melalui Pengenalan Suara Menggunakan Mel-Frequency Cepstrum Coefficient dan Hidden Markov Model Berbasis IOT. *e-Proceeding of Engineering*, (pp. 2100-2107).

Azhar, A., Jondri, & Wisety, U.N. (2015). Prediksi Perkembangan Kondisi Pasien Terapi HIV dengan Menggunakan Representasi ALE-index sebagai Invariant Nucleotida sequence dan Suport Vector Machine. *e-Proceeding of Engineering*, 2(1), 1305.

Bertero, D., & Fung, P. (2017). A first look into a *Convolutional Neural Network* for speech emotion detection. *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, (pp.5115-5119). Clear Water Bay.

Bonafilia, D., Tellman, B., Anderson, T. (2020). Sen1Floods11: A georeferenced dataset to train and test deep learning flood algorithms for sentinel-1. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, (pp. 835-845).

Brownlee, J. (2019). A Gentle Introduction to the Rectified Linear Unit (ReLU). Diakses pada 20 Maret 2021 dari https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/.

Cherry, Kendra. (2021). The 6 Types of Basic Emotions and Their Effect on Human Behavior. Diakses pada 3 Juli 2021 dari https://www.verywellmind.com/an-overview-of-the-types-of-emotions-4163976.

Chollet, F., & dkk. (2015). Keras. GitHub. Dikutip dari https://github.com/fchollet/keras.

Danukusumo, Kefin Pudi. (2017). Implementasi Deep Learning Menggunakan *Convolutional Neural Network* Untuk Klasifikasi Citra Candi Berbasis GPU. *Skripsi*. Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Atma Jaya.

Dewi, S.R. (2018). Deep Learning Object Detection pada Video Menggunakan Tensorflow dan *Convolutional Neural Network*. *Skripsi*. Yogyakarta: Program Studi Statistika Fakultas Matematika Dan Ilmu Pengetahuan Alam Universitas Islam Indonesia.

Elsai, T.P. (2021). Developing Ethiopian Paper Currency Recognition Model Using Convolutional Neural Network (CNN). *Computational and Mathematical Methods in Medicine*.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters 27*, (pp. 861-874).

Ghazvini, M., Dehghani, M.M., Ahmadi, M. (2020). Technological assessment and modeling of energy-related CO2 emissions for the G8 countries by using hybrid IWO algorithm based on SVM. *Energy Science and Engineering*, (pp. 1285-1308).

Ghubta, Abhay. (2020). Speech-Emotion-Recognition-using-ML-and-DL. Diakses pada 20 Februari 2021 dari https://github.com/abhay8463/Speech-Emotion-Recognition-using-ML-and-DL/blob/master/Deep%20Learning/SER(Deep_Learning).ipynb.

Hamid, B., Aleissa, E., Rauf, A. (2012). Anticipating software fault proneness using classifier ensemble: An optimize approach. *Proceedings of the IASTED International Conference on Software Engineering*, (pp.1-6).

Han, B., Yao, Q., Yu, X. (2018). Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in Neural Information Processing Systems*, (pp. 8527-8537).

Jain, M., Narayan, S., Balaji, P., Bharath, K.P., Bhowmick, A., Khartik R., & Muthu, R.K. (2017). Speech Emotion Recognition using *Support Vector Machine*. *International Journal of Smart Home*, 6(2), 101-108.

Jain, U., Nathani, K., Ruban, N., Raj, A.N.J., Zhuang, Z., & Mahesh, V.G.V. (2018). Cubic SVM Classifier Based Feature Extraction and Emotion Detection from Speech. *Proceedings of International Conference on Sensor Networks and Signal Processing*, (pp.386-391). Xi'an.

James, J., Tian, L., & Watson, C. (2018). An Open Source Emotional Speech Corpus for Human Robot Interaction Aplications, *Proceedings of Interspeech*.

Kwong, J.C.T., Garcia, F.C.C., Abu, P.A.R., & Reyes, R.S.J. (2018). Emotion Recognition via Facial Expression: Utilization of Numerous Feature Descriptors in Different Machine Learning Algorithms. *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, (pp. 2045-2049). Jeju.

Lecun, Y., Bengio, & Y., Hinton, G. (2015). Deep Learning. *Nature* 521, 436-444.

Liu, Y., Lian, J., Bartolacci, M. (2014). Density-based penalty parameter optimization on C-SVM. *Scientific World Journal*.

Livingstone, S.R., & Russo, F.A. (2018). The Ryerson Audio-Visual *Dataset* of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. *PloS ONE*, 13(5): e0196391

Manizar HM, E. (2017). MENGELOLA KECERDASAN EMOSI. Tadrib, 2(2), (pp. 198-213).

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In Proceedings of the 14th python in science conference (Vol. 8).

Mustaqeem, A., Anwar, S., Majid, M. (2018). *Multiclass Classification of Cardiac Arrhythmia Using Improved Feature Selection and SVM Invariants*, (pp. 10).

Nayoan, R.A.N. (2019). Analisis Sentimen Berbasis Fitur Pada Ulasan Tempat Wisata Menggunakan Metode *Convolutional Neural Network* (CNN). *Skripsi*. Yogyakarta: Fakultas Program Studi Informatika Tekonologi Industri Universitas Islam Indonesia.

Nguyen, M. (2018). Illustrated Guide to LSTM's and GRU's: A step by step explanation. Diakses pada 15 Maret 2021 dari https://towardsdatascience.com/illustrated-guide-tolstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Nurhikmat, T. (2018). Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma Convolutional Network (CNN) Pada Citra Wayang Golek. *Skripsi*. Yogyakarta: Program Studi Statistika Fakultas Matematika dan Ilmu Pengetahuan Alam

Pandey, S., Janghel, R. (2019). Automatic detection of arrhythmia from imbalanced ECG database using CNN model with SMOTE. *Australasian Physical and Engineering Sciences in Medicine*, (pp. 1129-1139).

Pinto, M., Polignano, M., Lops, P., & Semamaro, G. (2020). Emotions Understanding Model from Spoken Language using Deep Neural Networks and Mel-Frequency Cepstral Coefficients. *IEEE Conference on Evolving and Adaptive Intelligent Systems*.

Praksah, C., Gaikwad, V. (2015). Analysis of Emotion Recognition System through Speech Signal Using KNN & GMM Classifier. *IOSR Journal of Electronics and Communication Engineering*, (pp.2278-2834).

Purnama, Agus. (2012). Sinyal Audio (Gelombang Suara). Diakses pada 16 Maret 2021 dari http://elektronikadasar.web.id/sinyal-audio-gelombang-suara/.

Rismiyati (2016). Implementasi *Convolutional Neural Network* Untuk Sortasi Mutu Salak Ekspor Berbasis Citra Digital. *Tesis*. Program Studi S2 Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Gadjah Mada.

Rusdi, M., & Yani, A. (2018). Sistem Kendali Peralatan Elektronik Melalui Media Bluetooth Menggunakan Voice Recognition. *Journal of Electrical Technology*, 3(1), 27-33.

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

Semwal, N., Kumar, A., & Narayanan, S. (2017). Automatic Speech Emotion Detection System using Multi-domain Acoustic Feature Selection and Classification Models. *International Conference on Identity, Security, and Behavior Analysis*, (pp.1-6). Visakhapatnam.

Sena, Samuel. (2017). *Neural Network*. Diakses pada 18 Maret 2021 dari https://medium.com/@samuelsena/pengenalan-deep-learning-8fbb7d8028ac.

Shegokar, P. & Sircar, P. (2016). Continous Wavelet Transform based Speech Emotion Recognition. *International Conference on Signal Processing and Communication Systems (ICSPCS)*, (pp 1-8). Surfer's Paradise

Tripathi, S., Kumar, A., Ramesh, A., Singh, C., & Yenigalla, P. (2019). Deep Learning based Emotion Recognition System Using Speech Features and Transcriptions. *Proceedings of International Conference on Computational Linguistics and Intelligent Text Processing*, (pp. 1-12). La Rochelle.

Widodo, F.Y., Sunardi, & Fadlil, A. (2019). Identifikasi Suara Pada Sistem Presensi Karyawan Dengan Metode Ekstraksi MFCC. *Jurnal Sains Komputer & Informatika (J-SAKTI)*. 3(1), (pp. 115-125).

Zhao, J., Mao, X., & Chen, L. (2019). Speech emotion recognition using deep 1D & 2D CNN LSTM networks. *Biomedical Signal Processing and Control*, 47, 312-323.

LAMPIRAN

	precision	recall	f1-score	support
0	0.56	0.68	0.61	34
1	0.64	0.82	0.72	74
2	0.64	0.58	0.61	72
3	0.74	0.62	0.68	80
4	0.74	0.70	0.72	73
5	0.61	0.63	0.62	79
6	0.61	0.48	0.53	42
7	0.38	0.38	0.38	37
accuracy			0.63	491
macro avg	0.61	0.61	0.61	491
weighted avg	0.64	0.63	0.63	491

(a)

	precision	recall	f1-score	support
0	0.95	0.98	0.96	56
2	0.93	0.97	0.95	39
3	0.98	0.94	0.96	49
4	1.00	0.96	0.98	48
accuracy			0.96	192
macro avg	0.96	0.96	0.96	192
weighted avg	0.96	0.96	0.96	192

(b)

	precision	recall	f1-score	support
0	0.77	0.79	0.78	89
1	0.56	0.71	0.63	80
2	0.80	0.81	0.80	116
3	0.67	0.59	0.63	129
4	0.80	0.82	0.81	116
5	0.62	0.56	0.59	82
6	0.56	0.34	0.42	41
7	0.41	0.57	0.48	30
accuracy			0.69	683
macro avg	0.65	0.65	0.64	683
weighted avg	0.69	0.69	0.68	683

(c)

Lampiran 1. Nilai classification report dari model SVM yang telah dibuat ketika menggunakan; (a) RAVDESS dataset, (b) Jl Corpus dataset, (c) Dataset gabungan (RAVDESS dan Jl Corpus)

	precision	recall	f1-score	support
0	0.65	0.58	0.61	19
1	0.85	0.77	0.80	43
2	0.76	0.81	0.79	27
3	0.74	0.63	0.68	41
4	0.76	0.76	0.76	41
5	0.76	0.71	0.74	35
6	0.79	0.76	0.78	25
7	0.46	0.87	0.60	15
accuracy			0.73	246
macro avg	0.72	0.74	0.72	246
weighted avg	0.75	0.73	0.73	246

(a)

	precision	recall	f1-score	support
0	0.93	1.00	0.97	28
2	0.90	0.90	0.90	20
3	1.00	0.96	0.98	23
4	0.96	0.92	0.94	25
accuracy			0.95	96
macro avg	0.95	0.94	0.95	96
weighted avg	0.95	0.95	0.95	96

(b)

	precision	recall	f1-score	support
0	0.95	0.87	0.91	45
1	0.95	0.93	0.94	40
2	0.78	0.93	0.85	54
3	0.82	0.81	0.82	58
4	0.90	0.90	0.90	58
5	0.80	0.65	0.72	49
6	0.78	0.75	0.77	24
7	0.60	0.86	0.71	14
accuracy			0.84	342
macro avg	0.82	0.84	0.82	342
weighted avg	0.85	0.84	0.84	342

(c)

Lampiran 2. Nilai classification report dari model CNN yang telah dibuat ketika menggunakan; (a) RAVDESS dataset, (b) Jl Corpus dataset, (c) Dataset gabungan (RAVDESS dan Jl Corpus)

```
    0  2  3  4
0  [[76  1  6  0]
2   [ 3 66  0  1]
3   [ 5  3 74  0]
4   [ 0  6  0 76]]
```

(a)

```
    0  1  2  3  4  5  6  7
0 [[23  4  0  2  0  2  0  3]
1  [ 4 61  2  2  0  1  4  0]
2  [ 0  5 42  2  4 10  1  8]
3  [ 4 14  1 50  1  6  1  3]
4  [ 0  2  6  2 51  7  1  4]
5  [ 1  1  7 10  6 50  1  3]
6  [ 5  6  6  0  3  0 20  2]
7  [ 4  2  2  0  4  6  5 14]]
```

(b)

```
    0  1  2  3  4  5  6  7
0 [[70  6  1  9  1  0  2  0]
1  [ 6 57  3  8  0  1  4  1]
2  [ 0  5 94  3  8  2  1  3]
3  [13 14  3 76  2 15  1  5]
4  [ 0  1  7  2 95  8  1  2]
5  [ 1  6  5 11  8 46  0  5]
6  [ 1 10  2  2  3  1 14  8]
7  [ 0  2  3  3  2  1  2 17]]
```

(c)

Lampiran 3. Nilai Confession Matrix dari model SVM yang telah dibuat ketika menggunakan; (a) RAVDESS dataset, (b) Jl Corpus dataset, (c) Dataset gabungan (RAVDESS dan Jl Corpus

```
     0  2  3  4
0 [[68  0  1  1]
2  [ 1 65  0 11]
3  [ 4  0 81  0]
4  [ 0  3  0 82]]
```

(a)

```
     0  1  2  3  4  5  6  7
0 [[11  2  0  1  1  1  0  3]
1  [ 5 33  2  2  0  0  1  0]
2  [ 0  0 22  0  2  0  0  3]
3  [ 1  4  1 26  2  4  2  1]
4  [ 0  0  2  4 31  2  1  1]
5  [ 0  0  2  1  4 25  1  2]
6  [ 0  0  0  1  0  0 19  5]
7  [ 0  0  0  0  1  1  0 13]]
```

(b)

```
     0  1  2  3  4  5  6  7
0 [[39  1  0  3  0  0  2  0]
1  [ 1 37  1  0  1  0  0  0]
2  [ 0  0 50  0  2  0  0  2]
3  [ 1  1  2 47  0  5  2  0]
4  [ 0  0  3  0 52  2  0  1]
5  [ 0  0  7  6  1 32  0  3]
6  [ 0  0  1  0  2  1 18  2]
7  [ 0  0  0  1  0  0  1 12]]
```

(c)

Lampiran 4. Nilai Confession Matrix dari model CNN yang telah dibuat ketika menggunakan; (a) RAVDESS dataset, (b) Jl Corpus dataset, (c) Dataset gabungan (RAVDESS dan Jl Corpus