TUGAS AKHIR - IF184802

PENGENALAN BAHASA ISYARAT INDONESIA DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) MENGGUNAKAN KINECT 2.0

MUHAMMAD FAHMI ABDURRAHMAN
NRP 05111440000028

Dosen Pembimbing
Wijayanti Nurul Khotimah, S.Kom., M.Sc.
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR - IF184802

PENGENALAN BAHASA ISYARAT INDONESIA DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) MENGGUNAKAN KINECT 2.0

MUHAMMAD FAHMI ABDURRAHMAN
NRP 05111440000028

Dosen Pembimbing
Wijayanti Nurul Khotimah, S.Kom., M.Sc.
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – IF184802

INDONESIAN SIGN LANGUAGE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK (CNN) ALGORITHM AND KINECT 2.0

MUHAMMAD FAHMI ABDURRAHMAN
NRP 05111440000028

Advisor
Wijayanti Nurul Khotimah, S.Kom., M.Sc.
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

INFORMATICS DEPARTMENT
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PENGENALAN BAHASA ISYARAT INDONESIA DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) DENGAN KINECT 2.0

TUGAS AKHIR

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

Oleh :
Muhammad Fahmi Abdurrahman
NRP : 05111440000028

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Wijayanti Nurul Khotimah, S.Kom., M.Sc.
NIP: 198603122012122004 ..... (pembimbing 1)

Dr. Eng. Nanik Suciati, S.Kom., M.Kom.
NIP: 197104281994122001 ..... (pembimbing 2)

SURABAYA
JANUARI 2019

[Halaman ini sengaja dikosongkan]

PENGENALAN BAHASA ISYARAT INDONESIA DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) MENGGUNAKAN KINECT 2.0

Nama Mahasiswa : Muhammad Fahmi Abdurrahman
NRP : 05111440000028
Jurusan : Departemen Informatika FTIK-ITS
Dosen Pembimbing I : Wijayanti Nurul Khotimah, S.Kom., M.Sc.
Dosen Pembimbing II : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

ABSTRAK

Bahasa isyarat memungkinkan tuna rungu dan tuna wicara untuk saling berkomunikasi. Namun, tidak semua masyarakat memahami bahasa isyarat yang saat ini dipakai. Dengan perkembangan teknologi, penerjemahan bahasa isyarat dapat dilakukan secara real-time, yang memudahkan komunitas tuna rungu dan tuna wicara berkomunikasi dengan masyarakat umum.

Pada penelitian sebelumnya, penggunaan teknologi terkini guna mendukung pembelajaran pemakaian bahasa isyarat Indonesia sudah dilakukan, namun hanya sebatas bahasa isyarat dengan gerakan statis saja. Pada Tugas Akhir ini, penulis mengusulkan sebuah metode yang mampu mengenali secara real-time bahasa isyarat Indonesia dari 20 kata dan menerjemahkannya ke dalam gambar dan teks menggunakan Kinect 2.0. dan Algoritma Convolutional Neural Network (CNN) untuk mengenali gerakan tangan.

Hasil pengujian dalam Tugas Akhir ini menunjukkan bahwa algoritma CNN yang digunakan untuk mendeteksi gerakan isyarat dengan akurasi mencapai 78%. Hasil tersebut dapat ditingkatkan dengan menambahkan data training maupun menggunakan atau menggabungkan metode dan perangkat lain dalam mengenali bahasa isyarat.

Kata kunci – Bahasa isyarat Indonesia, Convolutional Neural Network, Kinect 2.0.

[Halaman ini sengaja dikosongkan]

INDONESIAN SIGN LANGUAGE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK (CNN) ALGORITHM AND KINECT 2.0

Name	**: Muhammad Fahmi Abdurrahman**
NRP	**: 05111440000028**
Major	**: Informatics Department, FTIK-ITS**
Advisor I	**: Wijayanti Nurul Khotimah, S.Kom., M.Sc.**
Advisor II	**: Dr.Eng. Nanik Suciati, S.Kom., M.Kom.**

ABSTRACT

Sign languages allows hearing and speech-impaired to interact with each other. However, not all people understand sign language that is currently used. With development in technology, sign language translation can be done in real time, make hearing and speech-imparired community interact with most common people much easier.

In previous research, the usage of latest technology to support learning Indonesian sign language usage has been done, but it is limited to static sign language only. In this final project, writer suggest a method to recognize Indonesian sign language in real-time and translate it to image and text using Kinect 2.0 and Convolutional Neural Network (CNN) Algorithm for recognizing the hand gesture.

This final project's test result shows that CNN algorithm which used in the sign language detection achieves an accuracy up to 78%. The result can be impoved by increasing the data training size and using or combining several methods or devices in recognizing sign language.

Keywords – Indonesian sign language, Convolutional Neural Network, Kinect 2.0.

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمنِ الرَّحِيمِ-

Puji syukur penulis panjatkan kehadirat Allah SWT karena berkat rahmat dan karunia-Nya penulis dapat menyelesaikan tugas akhir yang berjudul "Pengenalan Bahasa Isyarat Indonesia dengan Algoritma Convolutional Neural Netwrok (CNN) menggunakan Kinect 2.0". Penyusunan tugas akhir ini diajukan untuk memenuhi salah satu syarat memperoleh gelar Sarjana di Departemen Informatika Fakultas Teknologi Informasi dan Komunikasi Institut Teknologi Sepuluh Nopember Surabaya.

Dalam penyusunan dan penulisan tugas akhir ini tidak terlepas dari bantuan, bimbingan, serta dukungan dari berbagai pihak. Oleh karena itu, dalam kesempatan ini penulis menyampaikan terima kasih sebesar-besarnya kepada:

1. Orangtua penulis yang selalu memberi dukungan penuh dan semangat untuk menyelesaikan tugas akhir ini.
2. Ibu Wijayanti Nurul Khotimah, S.Kom., M.Sc. dan Ibu Dr.Eng. Nanik Suciati, S.Kom., M.Kom. yang telah bersedia untuk menjadi dosen pembimbing tugas akhir sehingga penulis dapat mengerjakan tugas akhir dengan arahan dan bimbingan yang baik dan jelas.
3. Bapak dan ibu dosen Departemen Informatika ITS yang banyak memberikan ilmu serta bimbingan selama masa perkuliahan.
4. Rekan kerja, Faradina Surya Eka Putri, yang selalu mendukung dan memotivasi penulis.
5. Teman-teman kontrakan E-103; Widhi, Nobby, Fintanto, Dharmawan, Hanendyo yang telah berbagi suka dan duka dalam menjalani perkuliahan selama ini.
6. Seluruh teman-teman TC14 yang sering membantu saya dalam perkuliahan.

7. Pihak-pihak lain yang turut membantu selama berkuliah yang tidak dapat disebutkan satu-persatu.

Penulis menyadari bahwa tugas akhir ini masih jauh dari kesempurnaan. Kritik dan saran yang membangun akan penulis terima dengan senang hati agar tugas akhir ini dapat bermanfaat khususnya bagi penulis dan umumnya bagi kita semua.

Surabaya, Januari 2019

Penulis

DAFTAR ISI

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

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

1.1 Latar Belakang

Kekurangan pada pendengaran sering berdampak pada kemampuan verbal pada orang dengan gangguan pendengaran, sehingga mereka menggunakan bahasa isyarat dan bahasa tubuh untuk berkomunikasi. Bahasa isyarat merupakan media bagi para penderita tuna rungu dan tuna wicara untuk berkomunikasi dengan sekitarnya. Penderita tuna rungu wicara mengalami kesulitan dalam berkomunikasi dengan orang normal. Hal ini dikarenakan ketidakpahaman orang normal dengan system isyarat ini [1]. Dengan demikian, proses pertukaran informasi sulit terjadi. Dimana kita tahu bahwa informasi tidak dapat terlepas dari diri kita. Untuk itu dibutuhkan sistem untuk menerjemahkan bahasa isyarat ke dalam Bahasa Indonesia agar dapat tercipta komunikasi yang lebih baik.

Bahasa isyarat adalah bahasa yang mengutamakan komunikasi manual, bahasa tubuh, dan gerak bibir untuk berkomunikasi. Penyandang tunarungu adalah kelompok utama yang menggunakan bahasa ini, biasanya mengkombinasikan bentuk tangan, orientasi dan gerak tangan, lengan dan tubuh, serta ekspresi wajah untuk mengungkapkan pikiran mereka [2]. Bahasa Isyarat sangat dipengaruhi oleh latar belakang budaya dan kebiasaan dimana orang tersebut tinggal dan berasal. Di Indonesia, Bahasa Isyarat diterapkan dalam dua bentuk, yaitu Sistem Isyarat Bahasa Indonesia (SIBI) dan Bahasa Isyarat Indonesia (BISINDO). Bahasa Isyarat Indonesia adalah sistem komunikasi yang praktis dan efektif untuk penyandang tunarungu Indonesia yang telah dikembangkan oleh kaum tunarungu, sedangkan Sistem Bahasa Isyarat Indonesia (SIBI) adalah sistem hasil rekayasa dan ciptaan

dari orang normal untuk berkomunikasi dengan penyandang difabel tunarungu dan bukan berasal dari penyandang difabel tunarungu. Dalam kehidupan sehari-hari, penderita tuna rungu dan wicara berkomunikasi dengan Bahasa Isyarat yang mengacu pada SIBI [1]. Contoh bahasa isyarat SIBI dapat dilihat pada gambar 1.1.

Gambar 1.1 Contoh gerakan bahasa isyarat

Bahasa Isyarat tidak memiliki lingkup pengguna yang besar seperti bahasa lisan, sehingga tidak banyak orang yang dapat mengenali atau mengerti Bahasa Isyarat yang disampaikan lawan bicaranya. Hal ini menimbulkan kebutuhan alat bantu untuk mempelajari Bahasa Isyarat dengan mudah.

Pada tahun 2010, Microsoft meluncurkan teknologi baru berupa perangkat keras sensor Kinect. Pada awalnya, teknologi tersebut ditujukan sebagai konsol XBox 360 sehingga pemain dapat menggunakan gerakan tubuhnya sebagai pengendali permainan. Teknologi sensor Kinect ini dapat dikembangkan lebih lanjut agar dapat digunakan untuk mengenali Bahasa Isyarat [3].

Penelitian mengenai Pengenalan Bahasa Isyarat Indonesia masih terbatas dan masih membutuhkan pengembangan.

Sebelumnya, sudah ada penelitian yang dibuat oleh Wijayanti Nurul Khotimah tentang pengenalan Bahasa Isyarat Indonesia menggunakan Kinect 1.0. Dalam penelitian tersebut, bahasa yang dapat dideteksi, yaitu Alquran, Bentuk, Gang, Hai, Hamba, Hormat, Ketua, dan Wadah. Bahasa yang dapat dikenali dalam penelitian ini hanya berupa gerakan statis dan kosa katanya pun masih terbatas, sehingga perlu ditelusuri lebih lanjut [4].

Dalam Tugas Akhir ini, akan dikembangkan suatu program aplikasi untuk pembelajaran Bahasa Isyarat dengan menggunakan sensor Kinect untuk mendeteksi gerakan tangan dan kemudian membandingkan pola gerakan tangan tersebut dengan data-data referensi menggunakan algoritma Convolutional Neural Network (CNN).

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana menentukan titik *skeleton* yang representative dari keluaran *skeleton* Kinect 2.0 untuk bahasan isyarat?
2. Bagaimana mengimplementasikan algoritm *Convolutional Neural Network* (CNN) berdasarkan titik skeleton yang telah ditentukan?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Teknologi yang dipakai adalah Kinect 2.0.
2. Gerakan tangan yang digunakan tidak melibatkan pergerakan jari tangan.

1.4 Tujuan

Tujuan pembuatan dari Tugas Akhir ini, yaitu untuk mengimplementasikan algoritma *Convolutional Neural Network* (CNN) untuk mengenali Bahasa Isyarat menggunakan Kinect 2.0.

1.5 Manfaat

Manfaat yang diperoleh dari pembuatan Tugas Akhir ini adalah untuk mengenali Bahasa Isyarat yang diharapkan dapat membantu orang berkebutuhan khusus, yaitu tuna rungu dan tuna wicara, dalam berkomunikasi dengan orang yang tidak memiliki kebutuhan khusus.

1.6 Metodologi

Pembuatan Tugas Akhir ini dilakukan menggunakan metodologi sebagai berikut:

A. Studi literatur
   Tugas Akhir ini akan menggunakan literatur paper yang berasal dari jurnal internasional bereputasi, yaitu IEEE. Selain itu, akan digunakan sejumlah referensi yang diperlukan dalam pembuatan aplikasi, yaitu mengenai Convolutional Neural Network (CNN) yang berasal dari buku, internet, ataupun materi dalam suatu mata kuliah yang berhubungan dengan metode yang akan digunakan.

B. Perancangan perangkat lunak
   Fitur yang terdapat pada aplikasi pengenalan Bahasa Isyarat ini adalah:
   1. Menerima masukan dari *user* berupa gerakan tangan dan posisi gerakan tangan tersebut ke alat Kinect 2.0.
   2. Memproses masukan tersebut dengan *classifier* yang telah ditentukan yaitu *Convolutional Neural Network* (CNN).
   3. Menampilkan hasil berupa gambar dan teks yang merupakan terjemahan dari gerakan tangan tersebut di atas.

C. Implementasi dan pembuatan sistem
Aplikasi ini dibangun menggunakan perangkat lunak Bahasa Pemrograman C# dan Python, Integrated Development Environment (IDE) Microsoft Visual Studio, Library Keras dan Kinect 2.0.

D. Uji coba dan evaluasi
Pengujian akan dilakukan oleh tiga orang pengguna. Pengguna tersebut akan diminta untuk melakukan gerakan yang telah ditentukan kemudian dihitung akurasi gerakan tersebut dari aplikasi yang telah dibuat.

E. Penyusunan laporan tugas akhir
Pada tahap ini dilakukan penyusunan laporan yang berisi dasar teori, dokumentasi dari perangkat lunak, dan hasil-hasil yang diperoleh selama pengerjaan tugas akhir.

1.7 Sistematika Penulisan

Buku tugas akhir ini terdiri dari beberapa bab, yang dijelaskan sebagai berikut:

BAB I PENDAHULUAN
Bab ini berisi latar belakang masalah, rumusan dan batasan masalah, tujuan dan manfaat pembuatan tugas akhir, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

BAB II TINJAUAN PUSTAKA
Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir ini.

BAB III ANALISIS DAN PERANCANGAN
Bab ini membahas analisis dari sistem yang dibuat meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaan, arsitektur, data, dan antarmuka.

BAB IV IMPLEMENTASI

Bab ini membahas implementasi dari rancxangan sistem yang dilakukan pada tahap perancangan. Penjelasan implementasi meliputi implementasi antarmuka aplikasi dan pembuatan kebutuhan fungsional aplikasi.

BAB V PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi untuk mengetahui kemampuan aplikasi.

BAB VI PENUTUP

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi selanjutnya.

BAB II
DASAR TEORI

Pada bab ini akan dibahas mengenai dasar teori yang menjadi dasar pembuatan Tugas Akhir ini. Pokok permasalahan yang akan di bahas mengenai teknologi yang mendukung dalam pembuatan tugas akhir seperti Kinect 2.0, Kinect SDK, *Convolutional Neural Network* (CNN), dan pengetahuan umum mengenai bahasa isyarat.

2.1 Tuna rungu

Tuna rungu dapat diartikan sebagai keterbatasan yang dimiliki seseorang dalam mendengar sesuatu karena tidak berfungsinya organ pendengaran yang dimilikinya. Ketunarunguan dapat dibedakan menjadi dua kategori, yaitu tuli (*deaf*) dan kurang dapat mendengar (*low hearing*) [5]. Tuli adalah keadaan dimana organ pendengaran telah mengalami kerusakan yang sangat parah dan mengakibatkan tidak berfungsinya pendengaran. Sedangkan kurang dapat mendengar adalah keadaan dimana organ pendengaran mengalami kerusakan tetapi masih dapat berfungsi untuk mendengar.

2.2 Bahasa Isyarat

Bahasa isyarat adalah sarana berkomunikasi bagi penderita tuna rungu. Bahasa isyarat dikembangkan dan memiliki karakteristik sendiri di berbagai negara. Di Indonesia, bahasa isyarat yang digunakan berdasarkan pada SIBI. Ada 4 jenis bahasa isyarat dalam SIBI [6], yaitu:

1. Isyarat Pokok: melambangkan sebuah kata atau konsep.
2. Isyarat Tambahan: melambangkan awalan, akhiran, dan partikel (imbuhan).
3. Isyarat Bentukan: dibentuk dengan menggabungkan isyarat pokok dan isyarat tambahan.
4. Abjad Jari: dibentuk dengan jari-jari untuk mengeja huruf.

Pada Tugas Akhir ini terdapat 20 bahasa isyarat yang digunakan seperti yang dapat dilihat pada Gambar 2.1. Bahasa isyarat tersebut kemudian dikelompokkan berdasarkan 6 semantik kategori:

1. Makna kata ganti orang.
2. Makna kata tanya.
3. Makna bangunan.
4. Makna keluarga.
5. Makna sapaan.
6. Makna lain-lain.

Gambar 2.1 Gerakan bahasa isyarat yang digunakan

2.3 Kinect 2.0

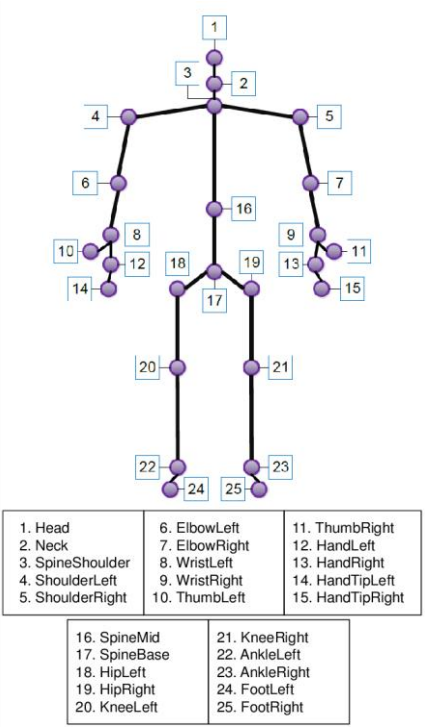
Generasi kedua dari Kinect yang dirilis oleh Microsoft pada tahun 2014 adalah versi terbaru Kinect dari yang pertama kali dikeluarkan pada tahun 2010. Perangkat Kinect 2.0 seperti yang terlihat pada Gambar 2.2, terdapat tiga lensa yaitu kamera RGB yang digunakan untuk menangkap spektrum warna, *infrared emitters* yang memproyeksikan spektrum inframerah dan sensor kedalaman yang menghasilkan gambar mendalam dari seseorang atau objek dengan menganalisis informasi inframerah. Dan sebuah *microphone* array yang dapat menemukan lokasi timbulnya suara. Alhasil, ada enam sumber data yang dihasilkan, termasuk warna, inframerah, kedalaman, indeks tubuh, tubuh, dan suara [7].

Gambar 2.2 Kinect 2.0

2.4 Kinect SDK

Kinect SDK adalah pustaka yang dibuat oleh Microsoft untuk pengembangan aplikasi perangkat lunak yang menggunakan Kinect sebagai alat input utama. Kinect SDK dapat diimplementasikan dengan bahasa pemrograman C#, C++, dan JavaScript. Pustaka ini memiliki beberapa fitur diantaranya *skeleton tracking*, *thumb tracking*, *end of hand tracking*, *open/close hand gesture* dan lainnya [8].

Pada Tugas Akhir ini, Kinect SDK digunakan untuk menentukan koordinat 3D dari 15 posisi titik skeleton.



Gambar 2.3 Titik skeleton yang ditangkap oleh Kinect

2.5 Python

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Python juga didukung oleh komunitas yang besar.

Python mendukung multi paradigma pemrograman, utamanya; namun tidak dibatasi; pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional.

Salah satu fitur yang tersedia pada python adalah sebagai bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa skrip meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa skrip. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi [9].

2.6 Keras

Keras adalah *library neural network* berbasis sumber terbuka. Keras dapat dijalankan diatas *Tensorflow, Microsoft Cognitive Toolkit,* atau *Theano.* Didesain untuk dapat menjalankan eksperimentasi yang cepat dari *deep neural network,* Keras difokuskan agar mudah digunakan, *modular,* dan dapat disambung. Keras dikembangkan sebagai bagian dari penelitian dari proyek ONEIROS (*Open-ended Neuro-Elecrtonic Intellegent Robot Operating System)* dan pengarang sekaligus pemeliharanya adalah *Francois Chollet.*

Pada tahun 2017, TensorFlow milik Google memutuskan untuk mendukung Keras dengan *library core*-nya. Chollet memaparkan bahwa Keras lebih dipahami sebagai antar muka daripada sebagai *machine-learning framework* yang berdiri sendiri. Keras menyediakan abstraksi yang intuitif, membuat Keras mudah digunakan untuk mengembangkan model *deep learning* dengan apapun *computational backend*-nya [10].

2.7 Microsoft Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap yang dapat digunakan untuk melakukan pengembangan perangkat lunak, baik itu perangkat lunak bisnis, perangkat lunak pribadi, ataupun komponen perangkat lunaknya

dalam bentuk perangkat lunak berbasis *console*, Windows, ataupun berbasis *website* [11].

2.8 Convolution Neural Network (CNN)

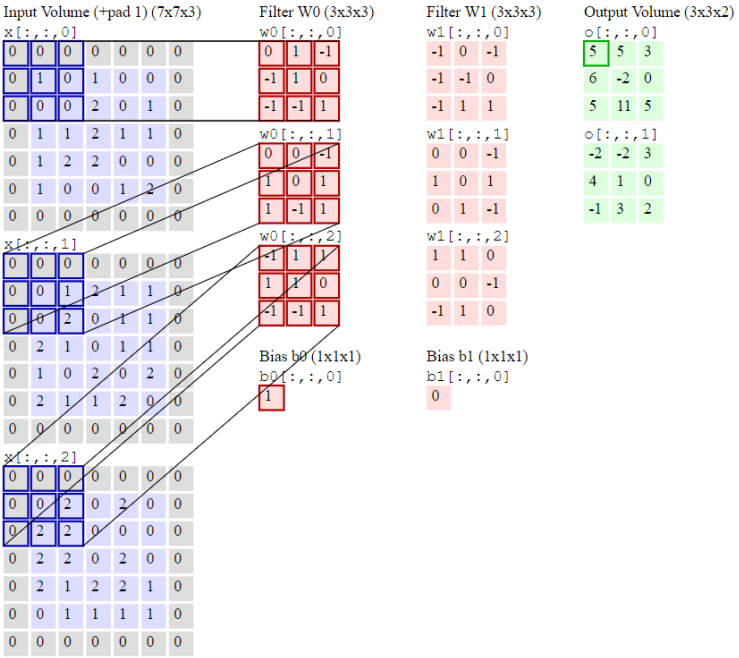
Di dalam machine learning, *Convolutional Neural Network* adalah salah satu algoritma yang dalam, menggunakan *feed-forward* ANN yang telah sukses digunakan untuk menganalisa gambar (visual). CNN terinspirasi oleh proses biologis dimana pola konektivitas antara neuron terinspirasi oleh organisasi korteks visual dari hewan. Neuron korteks perorangan merespons rangsangan hanya di wilayah terbatas bidang visual yang dikenal sebagai bidang reseptif. Bidang reseptif neuron yang berbeda sebagian tumpang tindih sehingga mencakup keseluruhan bidang visual [12].

CNN menggunakan variasi dari multilayer perceptrons agar meminimalisir preprocessing. Ini berarti CNN mempelajari filter yang biasanya dalam algoritma pembelajaran biasa dibuat secara manual. Independensi ini membuat algoritma CNN unggul diantara algoritma lainnya [12].

Dalam penerapan CNN, terdiri dari 4 layer, yaitu:

a. ***Convolutional Layer***

Convolution adalah operasi untuk mengekstraksi fitur-fitur yang menonjol yang didapat dari sebuah input. Dalam proses 3D *convolution*, input yang berukuran N x N x D akan dibandingkan dengan kernel/filter H, yang berukuran H x H x D. Kernel akan dibandingkan dengan input dengan cara menggesernya satu persatu sebanyak *Stride* S dari paling ujung kiri atas, hingga sampai ke paling ujung kanan bawah. Setiap input yang dibandingkan dengan kernel H akan menghasilkan sebuah H fitur [13].



Gambar 2.3 Contoh Proses Konvolusi

b. ***Non-Linear / Activation Layer***

Pada layer ini adalah layer yang penting, berisi fungsi trigger untuk memberi tanda jika menemukan kemiripan dari fitur pada setiap hidden layer [13]. Pada CNN ada beberapa fungsi yang biasa digunakan, seperti:

1. *Linear / Identitiy Activation*

$$f(x) = x$$

2. *Logistic / Sigmoid Activation*

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

3. *TanH Activation*

$$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$$

4. Softmax Activation

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{i=1}^{J} e^{x_J}} \text{ for } i = 1, \dots, j$$

5. *Rectified Linear Unit (RELu) Activation*

$$f(x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$$

c. ***Pooling / Sub-sampilng Layer***

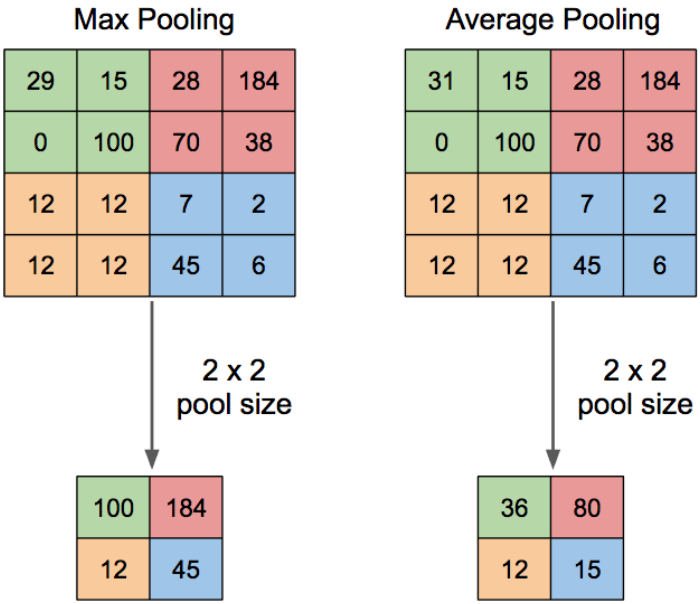
Pooling adalah operasi untuk mengurangi resolusi dari fitur yang didapat. Tahap ini merupakan tahap untuk mengatasi distorsi dan noise data [13]. Metode yang paling sering digunakan untuk *pooling* adalah sebagai berikut yang dicontohkan seperti pada Gambar 2.3:

1. *Max Pooling*

   Max Pooling adalah cara merepresentasikan suatu area yang dipilih dengan salah satu nilai yang terbesar.

2. *Average Pooling*

   Average Pooling adalah cara merepresentasikan suatu area dengan rata-rata dari area tersebut.



Gambar 2.4 Contoh Proses Max Pooling dan Average Pooling

d. ***Fully Connected Layer***

Layer ini selalu digunakan sebagai layer terakhir. Layer ini menggabungkan dan menghitung jumlah bobot dari fitur-fitur sebelumnya, lalu dikalkulasi sebagai "bahan" untuk menetukan hasil dari klasifikasi [13].

Output fitur dari layer sebelumnya berbentuk *array* multidimensi, maka dilakukan proses *flatten*, atau reshape fitur dari *array* multidimensi menjadi sebuah vektor agar dapat digunakan untuk input di *Fully Connected Layer*.

[Halaman ini sengaja dikosongkan]

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis permasalahan dan perancangan dari sistem yang dibangun. Analisis permasalahan membahas permasalahan yang diangkat dalam pengerjaan tugas akhir. Analisis kebutuhan mencantumkan kebutuhan-kebutuhan yang diperlukan oleh sistem. Selanjutnya dibahas mengenai perancangan sistem yang dibuat.

3.1 Analisis Perangkat Lunak

Bahasa isyarat merupakan sarana komunikasi untuk penderita tunarungu. Walaupun lumayan sulit untuk mengartikan isyarat yang diberikan, hal tersebut telah membantu penderita tunarungu untuk berkomunikasi dengan sekitar. Namun masih banyak yang belum mengerti apa arti isyarat bagi yang diberi atau memberi isyarat.

Perangkat lunak ini bertujuan untuk membantu pengguna dalam mempelajari bahasa isyarat Indonesia yang sesuai dengan SIBI. Data isyarat tersebut didapatkan dari *training* data yang dilakukan oleh pengguna. Dengan menggunakan Microsoft Visual Studio dan Kinect 2.0, penulis mengekstraksi koordinat *skeleton joints* yang ditangkap oleh Kinect 2.0 kemudian disimpan ke dalam sebuah file ekstensi .csv untuk dijadikan *training* data.

Untuk tahap *testing* data, setelah perangkat lunak mengekstraksi koordinat *skeleton* pengguna, data tersebut akan dicocokkan dengan training data yang telah dibuat sebelumnya menggunakan metode *Convolution Neural Network* (CNN). Setelah seluruh tahap sudah selesai dilakukan, perangkat lunak akan memberikan keluaran berupa arti bahasa isyarat Indonesia yang dimaksud oleh pengguna.

3.1.1 Deskripsi Umum Perangkat Lunak

Tugas Akhir yang dibangun ini adalah sebuah modul pengenalan bahasa isyarat Indonesia dengan menggunakan teknologi Kinect 2.0. Pengguna utama dalam perangkat lunak ini adalah semua orang yang ingin mempelajari bahasa isyarat Indonesia. Pengguna dapat menggunakan isyarat yang sudah ada di dalam perangkat lunak ataupun memberikan bahasa isyarat baru dengan mengacu pada SIBI.

3.1.2 Spesifikasi Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang akan dibuat ini melibatkan dua hal, yakni kebutuhan fungsional maupun kebutuhan non-fungsional. Dimana masing-masing berhubungan dengan keberhasilan dalam pembuatan tugas akhir ini.

3.1.2.1 Kebutuhan Fungsional Perangkat Lunak

Dalam pengembangan perangkat lunak, terdapat beberapa kebutuhan fungsional yang mendukung jalannya perangkat lunak. Kebutuhan fungsional tersebut adalah sebagai berikut:

a) Mendeteksi *skeleton* pengguna
   Perangkat lunak dapat mendeteksi pengguna yang sedang berada di depan Kinect 2.0.
b) Mengekstraksi koordinat *skeleton*
   Perangkat lunak dapat mendeteksi bagian-bagian dari *skeleton* pengguna yang akan diekstraksi guna melakukan proses klasifikasi gerakan.
c) Menerjemahkan Bahasa Isyarat
   Perangkat lunak dapat menerjemahkan bahasa isyarat Indonesia yang dihasilkan melalui proses ekstraksi data dinamis *skeleton* pengguna.

3.1.2.2 Kebutuhan Non-Fungsional

Disamping kebutuhan fungsional, terdapat juga beberapa kebutuhan non-fungsional dalam mendukung dan menambah performa perangkat lunak. Kebutuhan non-fungsional tersebut adalah sebagai berikut:

a) Penyesuaian intensitas cahaya
   Intensitas cahaya merupakan salah satu unsur penting dalam penggunaan sensor di Kinect 2.0. Jika intensitas cahaya rendah, data *skeleton* yang diambil oleh Kinect 2.0 akan menjadi tidak stabil. Oleh karena itu, dalam menggunakan perangkat lunak ini, sebaiknya dilakukan di ruangan yang mempunyai intensitas cahaya yang cukup.
b) Posisi Kinect dengan pengguna
   Dalam mencapai hasil oleh data perangkat lunak (baik itu pengambilan data mapun uji coba data) yang sempurna, jarak optimal Kinect 2.0 dengan pengguna adalah antara 150 sampai 200 cm, dengan tinggi meja 75 cm.

3.1.3 Identifikasi Pengguna

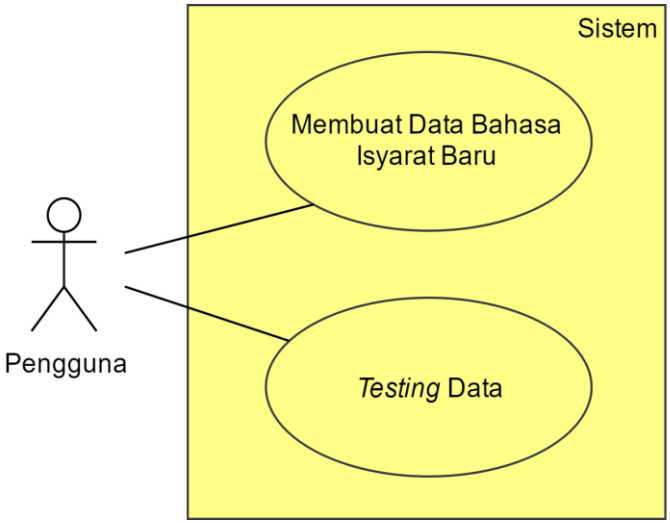
Dalam tugas akhir yang dibangun ini, pengguna yang akan terlibat dalam menjalankan perangkat lunak hanya satu orang saja, yaitu orang yang akan melakukan pengenalan bahasa isyarat Indonesia.

3.2 Perancangan Perangkat Lunak

Subbab ini membahas bagaimana rancangan dari tugas akhir ini. Hal yang dibahas meliputi model kasus penggunaan, definisi aktor, definisi kasus penggunaan, arsitektur umum sistem, rancangan antarmuka perangkat lunak, dan rancangan proses perangkat lunak.

3.2.1 Model Kasus Penggunaan

Dari hasil analisa deskripsi umum perangkat lunak dan spesifikasi kebutuhan perangkat lunak yang telah dijelaskan, maka model kasus penggunaan untuk perangkat lunak pengenalan bahasa isyarat dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram Kasus Penggunaan Perangkat Lunak

3.2.2 Definisi Aktor

Aktor yang terdapat dalam sistem aplikasi ini terlihat pada Tabel 3.1.

Tabel 3.1 Definisi Kasus Penggunaan

No	Nama	Deskripsi
1	Pengguna	Merupakan aktor yang bertugas untuk menambahkan data *training* dan melakukan *testing* gerakan isyarat, seluruh fungsionalitas yang ada di

		dalam sistem dapat digunakan oleh pengguna.

3.2.3 Definisi Kasus Penggunaan

Pada Gambar 3.1 telah dijelaskan bahwa aktor yang dalam hal ini disebut pengguna mempunyai dua kasus penggunaan, yakni membuat data bahasa isyarat Indonesia yang baru dan melakukan *testing* data. Rincian mengenai kasus penggunaan tersebut dapat dilihat pada Tabel 3.2.

Tabel 3.2 Definisi Kasus Penggunaan

No	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
1	UC-01	Membuat Data Bahasa Isyarat Baru	Pengguna membuat data bahasa isyarat Indonesia yang baru
2	UC-02	*Testing* Data	Pengguna melakukan *testing* data dengan melakukan gerakan bahasa isyarat Indonesia yang tersedia.

3.2.3.1 Kasus Penggunaan Membuat Data Bahasa Isyarat Baru

Spesifikasi kasus penggunaan membuat data bahasa isyarat baru dapat dilihat pada Tabel 3.3.

Tabel 3.3 Spesifikasi Kasus Penggunaan Membuat Data Bahasa Isyarat Baru

Nama Kasus Penggunaan	Membuat Data Bahasa Isyarat Baru
Nomor	UC-01
Deskripsi	Kasus penggunaan aktor untuk membuat data bahasa isyarat baru

Aktor	Pengguna
Kondisi Awal	Pengguna sudah menjalankan aplikasi dan perangkat Kinect 2.0 telah tersambung
Alur Normal	1. Pengguna memasukan nama bahasa isyarat yang akan dibuat di dalam *textbox* perangkat lunak 2. Pengguna menekan tombol "*Train*" di dalam perangkat lunak 3. Perangkat lunak menerima inputan dari Kinect 2.0 dan ketika *skeleton* pengguna ditemukan 4. Pengguna melakukan gerakan bahasa isyarat yang akan dibuat 5. Perangkat lunak mengekstrak data *skeleton* pengguna sebanyak 40 *frame* 6. Perangkat lunak menyimpan hasil ekstraksi ke dalam sebuah berkas berekstensi .csv
Alur Alternatif	A1. Kinect 2.0 tidak menemukan *skeleton* pengguna
Kondisi Akhir	Perangkat lunak membuat data gerakan yang baru

3.2.3.2 Kasus Penggunaan *Testing* Data

Spesifikasi kasus penggunaan *Testing* Data dapat dilihat pada Tabel 3.4.

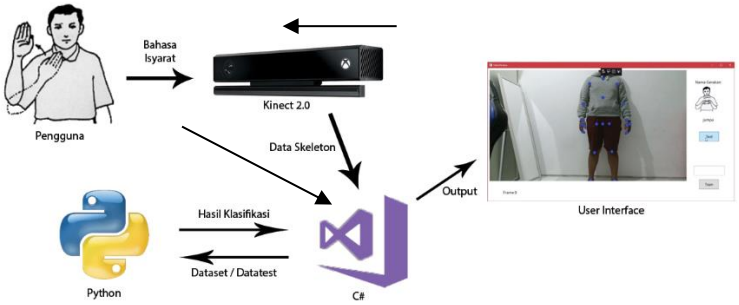
Tabel 3.4 Spesifikasi Kasus Penggunaan *Training Dataset*

Nama Kasus Penggunaan	***Testing* Data**
Nomor	UC-02
Deskripsi	Kasus penggunaan aktor untuk melakukan *testing* data isyarat bahasa Indonesia yang telah dibuat
Aktor	Pengguna

Kondisi Awal	Pengguna dalam keadaan menjalankan perangkat lunak dan sudah ada *classifier* data isyarat bahasa Indonesia di dalam perangkat lunak
Alur Normal	1. Pengguna menekan tombol "Start Testing" di dalam perangkat lunak 2. Perangkat lunak menerima inputan dari Kinect 2.0 dan ketika *skeleton* pengguna ditemukan A1. Kinect 2.0 tidak menemukan *skeleton* pengguna 3. Pengguna melakukan gerakan bahasa isyarat 4. Perangkat lunak mengekstrak data *skeleton* pengguna sebanyak 40 *frame* untuk dikalkulasi 5. Perangkat lunak menyimpan hasil ekstraksi ke dalam sebuah berkas berekstensi .csv 6. Hasil ekstraksi akan diambil oleh program python untuk kalkulasi dan klasifikasi dengan CNN 7. Program python akan memberi keluaran hasil klasifikasi ke dalam berkas berekstensi .txt 8. Hasil klasifikasi akan dibaca oleh perangkat lunak 9. Perangkat lunak menampilkan isyarat bahasa Indonesia hasil klasifikasi
Alur Alternatif	A1. Kinect 2.0 tidak menemukan *skeleton* pengguna
Kondisi Akhir	Perangkat lunak memberikan keluaran berupa bahasa isyarat yang dimaksud oleh pengguna baik itu dalam bentuk tulisan maupun dalam bentuk gambar

3.2.4 Arsitektur Umum Sistem

Arsitektur umum pada perangkat lunak ini memiliki perangkat tambahan Kinect 2.0 sebagai perangkat masukan. Implementasi aplikasi dibuat menggunakan Microsoft Visual Studio. Arsitektur umum perangkat lunak yang akan dibuat dapat dilihat pada Gambar 3.2.

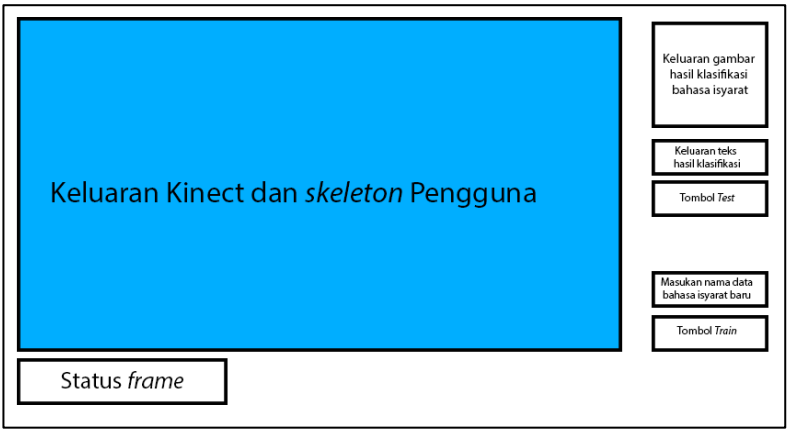


Gambar 3.2 Arsitektur Umum Sistem

3.2.5 Rancangan Antarmuka Perangkat Lunak

Rancangan antarmuka perangkat lunak diperlukan untuk memberikan gambaran umum kepada pengguna bagaimana sistem yang ada dalam perangkat lunak ini berinteraksi dengan pengguna. Selain itu, rancangan ini juga memberikan gambaran bagi pengguna apakah tampilan yang sudah disediakan oleh perangkat lunak mudah untuk dipahami dan digunakan, sehingga akan muncul kesan pengalaman pengguna yang baik dan mudah.

Rancangan antarmuka perangkat lunak ini hanya memiliki satu Windows dan memiliki beberapa kontrol yang sekiranya dapat dipahami oleh pengguna. Rancangan antarmuka perangkat lunak dapat dilihat pada Gambar 3.3.



Gambar 3.3 Rancangan Antarmuka Perangkat Lunak

3.2.6 Rancangan Proses Perangkat Lunak

Pada rancangan proses perangkat lunak akan dijelaskan mengenai proses yang terjadi dalam sistem untuk memenuhi fungsionalitas yang ada pada perangkat lunak. Proses ini penting agar perangkat lunak dapat berjalan secara baik dan benar.

3.2.6.1 Rancangan Proses Menentukan Data

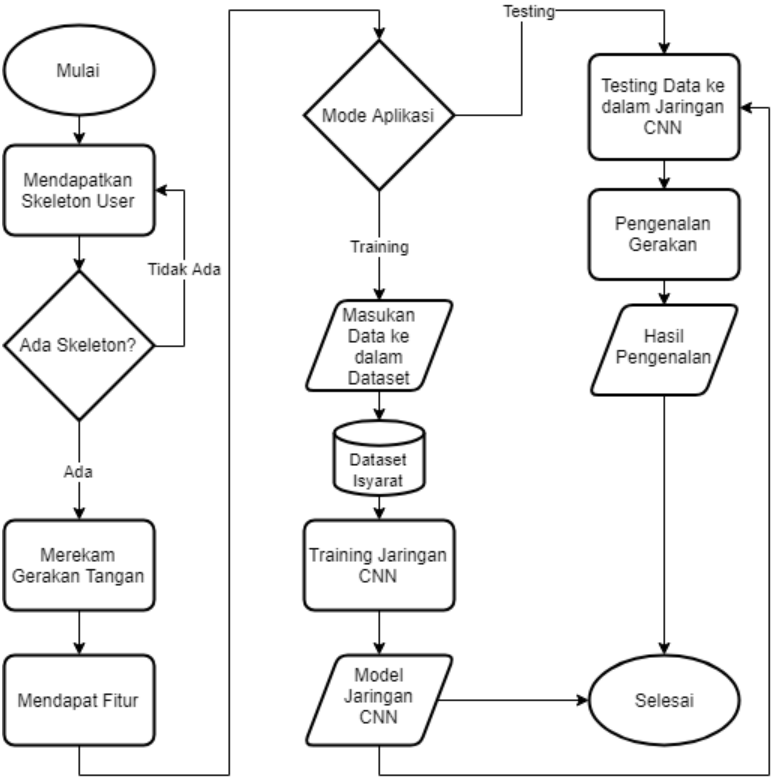
Proses penentuan data dilakukan dengan mengambil titik *skeleton* di badan bagian atas, karena bahasa isyarat, terutama SIBI, tidak banyak menggunakan *full body gesture* (gerakan semua badan). Titik yang akan diambil yaitu Kepala (H), Leher (N), Tulang Belakang Leher (SS), Bahu Kanan (SR), Bahu Kiri (SL), Siku Kanan (ER), Siku Kiri (EL), Pergelangan Tangan Kanan (WR), Pergelangan Tangan Kiri (WL), Tangan Kanan (HR), Tangan Kiri (HL), Ujung Tangan Kanan (HTR), Ujung Tangan Kiri (HTL), Jempol Tangan Kanan (TR), dan Jempol Tangan Kiri (TL).

3.2.6.2 Rancangan Proses Ekstrasi Data

Proses menentukan data akhir dilakukan dengan mengambil data *skeleton* dari pengguna, yang akan diambil 15 titik dari data *skeleton* tersebut. Dari setiap titik pada *skeleton*, akan memiliki variabel posisi X, Y, dan Z, maka akan didapatkan 45 data. Fitur ini akan diambil sejumlah 40 *frame* sehingga akan menjadi data berjumlah 1800 untuk tiap masukan dari bahasa isyarat.

3.2.6.3 Rancangan Proses Implementasi CNN untuk Menentukan Pengenalan Gerakan Tangan

Proses implementasi algoritma CNN dilakukan dengan cara membandingkan fitur data *training* dengan data uji, dengan cara memasukkan fitur data uji ke dalam jaringan CNN yang telah sebelumnya diberi data *training.* Jaringan akan memberikan keluaran berupa kelas dari gerakan yang diinputkan dari data uji tersebut. Rancangan proses implementasi CNN untuk menentukan pengenalan gerakan tangan dapat dilihat pada Gambar 3.4.



Gambar 3.4 Diagram Alir Implementasi CNN untuk Menentukan Pengenalan Gerakan Tangan

1. Perekaman gerakan oleh Kinect 2.0

Dalam perekaman gerakan yang dilakukan oleh Kinect 2.0 gerakan yang terekam kemudian diolah menjadi fitur data. Perangkat lunak mengambil total 40 *frame* untuk dikalkulasi dan dilakukan implementasi CNN untuk mengidentifikasi posisi gerakan yang dilakukan.

2. Penentuan *skeleton joints* yang digunakan

Terdapat 15 *skeleton joints* yang diolah dalam Tugas Akhir ini. Masing-masing *skeleton joints* diinterpretasikan dalam bentuk

koordinat X, Y dan Z seperti yang sudah dijelaskan pada Subbab 2.6. Lima belas *skeleton joints* tersebut adalah:

1. Kepala (H)
2. Leher (N)
3. Tulang Belakang Leher (SS)
4. Bahu Kanan (SR)
5. Bahu Kiri (SL)
6. Siku Kanan (ER)
7. Siku Kiri (EL)
8. Pergelangan Tangan Kanan (WR)
9. Pergelangan Tangan Kiri (WL)
10. Tangan Kanan (HR)
11. Tangan Kiri (HL)
12. Ujung Tangan Kanan (HTR)
13. Ujung Tangan Kiri (HTL)
14. Jempol Tangan Kanan (TR)
15. Jempol Tangan Kiri (TL)

3. Penentuan data

Untuk menentukan fitur dilakukan pengambilan data *skeleton* dari pengguna, yang akan diambil 15 titik dari *skeleton,* tiap titiknya berisi 3 posisi, yaitu posisi X, Y, dan Z.

4. Mendapatkan data akhir

Setelah mendapatkan data awal, selanjutnya data tersebut akan diambil sebanyak 40 *frame*, sehingga akan didapat 1800 data dalam setiap data Bahasa isyarat

5. Implementasi CNN dan pengenalan gerakan tangan

Dalam proses pengenalan gerakan tangan, perangkat terlebih dahulu melakukan proses *training* jaringan CNN tersebut. *Training* ini dilakukan agar *weight* dari setiap *layer* berubah menjadi lebih optimal.

Di aplikasi ini menggunakan jaringan CNN yang terdiri dari 4 tahap, terdiri dari 3 tahap ekstraksi fitur dan 1 tahap klasifikasi,

dengan total 11 *layer*. Tahap ekstraksi terdiri dari 3 bagian. 2 bagian awal disebut juga tahap *convolution*, masing-masing terdiri *convolutional layer, activation layer*, dan *pooling layer*. Bagian terakhir dari ekstraksi terdiri dari *fully-connected layer* dan *activation layer*. Sedangkan tahap klasifikasi terdiri dari *fully-connected layer* dan *activation layer*. Yang membedakan *layer* terakhir tahap ekstraksi dan *layer* tahap klasifikasi adalah ukuran dari *fully-connected layer* dan fungsi aktivasi yang digunakan.

Tahap *convolution* pertama memasukkan dataset yang berukuran 1800 data ke dalam martiks berukuran 40 (jumlah frame) x 15 (jumlah skeleton) x 3 (jumlah sumbu) kedalam input *convolutional layer*. Akan dilakukan komputasi *convolution* dengan jumlah filter *K*, ukuran filter *f x f*, diberikan *zero padding*, dan memiliki *stride* dengan ukuran *S x S*. Setelah itu hasil *convolution* akan melewati *activation layer* yang menggunakan fungsi ReLU. Setelah itu dilanjut dengan *pooling layer* menggunakan metode *Max Pooling* dengan ukuran pool *p x p*, dengan stride *s x s*.

Tahap *convolution* kedua memasukkan output dari tahap *convolution* pertama beukuran *w x h x* d sebagai input *convolutional layer*. Akan dilakukan komputasi konvolusi dengan jumlah filter K, ukuran filter *f x f*, diberikan *zero padding*, dan memiliki *stride* dengan ukuran *S x S*. Setelah itu hasil *convolution* akan melewati *activation layer* yang menggunakan fungsi ReLU. Setelah itu dilanjut dengan *pooling layer* menggunakan metode *Max Pooling* dengan ukuran *pool p x p*, dengan *stride s x s*.

Tahap ketiga adalah *fully-connected layer* yang pertama, memberi fungsi *flatten* kepada hasil dari tahap *convolution* kedua yang berbentuk matriks *w x h x d* menjadi matriks 1 dimensi berukuran *1 x 1 x (w x h x d*). Lalu hasil dari *flatten layer* ini akan dihubungkan dengan *dense layer*. *Dense layer* ini adalah *layer* yang mengolah input menjadi *layer* yang dipengaruhi *weight* dari setiap neuronnya. Lalu melewati *activation layer* yang menggunakan fungsi TanH.

Tahap terakhir adalah tahap klasifikasi yang diawali dengan *fully-connected layer*, dengan memberi *dense layer* sebesar jumlah kelas yang ada yaitu 20. Lalu melewati *activation layer* yang menggunakan fungsi aktivasi *softmax*, yang akan memberikan bobot dari setiap kelas yang ada. Kelas yang memiliki bobot yang paling besar adalah kelas hasil dari klasifikasinya.

3.2.6.4 Rancangan Proses *Training* Data

Proses *training* data dilakukan dengan menyimpan fitur data yang didapatkan pada Subbab 3.2.1.1 ke dalam sebuah berkas berekstensi .csv. Data training yang disimpan ada sebanyak 1300 gerakan dengan masing-masing kata pada Gambar 2.1 sebanyak 65 gerakan.

3.2.6.5 Rancangan Proses *Testing* Data

Proses testing data dilakukan dengan mendapatkan fitur baru yang dilakukan secara real-time. Data yang didapat pada Subbab 3.2.1.4 akan diteruskan ke dalam jaringan CNN. Keluaran hasil klasifikasi yang dilakukan berupa prediksi gerakan bahasa isyarat yang dimaksud oleh pengguna dalam bentuk gambar dan juga tulisan bahasa isyarat.

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi dari perancangan yang sudah dilakukan pada bab sebelumnya. Implementasi berupa kode sumber dengan bahasa pemrograman C# dan Python untuk membangun program. Sebelum masuk ke penjelasan implementasi, akan ditunjukkan terlebih dahulu lingkungan untuk melakukan implementasi.

4.1 Lingkungan Pembangunan

Dalam membangun aplikasi ini digunakan beberapa perangkat pendukung baik perangkat keras maupun perangkat lunak. Lingkungan pembangunan dijelaskan sebagai berikut.

4.1.1 Lingkungan Pembangunan Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan aplikasi ini adalah sebuah perangkat laptop dengan spesifikasi sebagai berikut:

- Prosesor Intel(R) Core(TM) i5-7600U CPU @ 2.60GHz
- Memori (RAM) 8,00 GB
- GPU Nvidia GT 940M
- Kinect Sensor 2.0

4.1.2 Lingkungan Pembangunan Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan untuk membuat aplikasi ini sebaga berikut.

- Microsoft Visual Studio 2017
- Python versi 3.6.6
- Keras versi 2.2.0
- TensorFlow versi 1.9.0
- Kinect SDK 2.0

- CUDA versi 9.0
- CuDNN versi 7.1.4
- Atom sebagai *Integrated Development Environment* (IDE)
- Windows 10 Home 64 bit sebagai sistem operasi

4.2 Implementasi Antarmuka

Modul pengenalan bahasa isyarat yang akan dibuat hanya akan memiliki satu window utama yang sudah mencakup semua fungsionalitas perangkat lunak yang dibutuhkan. Tampilan antarmuka perangkat dapat dilihat pada Gambar 3.3. Sedangkan kode sumber untuk antarmuka perangkat lunak dapat dilihat pada Kode Sumber 4.1.

```
1   <Window x:Class="CNNkinect.MainWindow"
2      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5      xmlns:mc="http://schemas.openxmlformats.org/markup-
       compatibility/2006"
6      xmlns:local="clr-namespace:CNNkinect"
7      mc:Ignorable="d"
8      Title="MainWindow" Height="720" Width="1280"
9      Loaded="Window_Loaded" Closed="Window_Closed"
    WindowStyle="SingleBorderWindow">
10     <Grid>
11       <Viewbox Width="1008" Height="593" Margin="10,10,0,0"
         HorizontalAlignment="Left" VerticalAlignment="Top">
12         <Grid>
13           <Image Name="camera" Width="1920" Height="1080" />
14           <Canvas Name="canvas" ClipToBounds="True" />
15         </Grid>
16       </Viewbox>
17       <Label Name="output" Content="Nama Gerakan:" FontSize="20"
         HorizontalContentAlignment="Center"
         HorizontalAlignment="Center" VerticalAlignment="Top"
         Width="155" Height="50" Margin="1057,52,61.6,0"></Label>
18       <Label Name="NamaGerakan" FontSize="20"
         HorizontalContentAlignment="Center"
         HorizontalAlignment="Center" VerticalAlignment="Top"
         Width="155" Height="46" Margin="1057,244,61.6,0"></Label>
19       <Label Name="framerate" Content="" FontSize="20"
         HorizontalAlignment="Left" VerticalAlignment="Top" Width="155"
         Height="50" Margin="132,616,0,0"></Label>
20       <TextBox Name="InputTest" HorizontalAlignment="Left"
         VerticalAlignment="Top" Width="155" Height="50"
         Margin="1057,499,0,0"></TextBox>
21       <Button x:Name="Testing" Content="Test" FontSize="20"
         HorizontalAlignment="Left" VerticalAlignment="Top" Width="100"
```

```
        Height="50" Margin="1084,325,0,0"
        RenderTransformOrigin="1.5,2.18"
        Click="Testing_Click"></Button>
22      <Button x:Name="Training" Content="Train" FontSize="20"
        HorizontalAlignment="Left" VerticalAlignment="Top" Width="100"
        Height="50" Margin="1083,569,0,0"
        RenderTransformOrigin="1.5,2.18"
        Click="Training_Click"></Button>
23      <Label Content="Frame" FontSize="20" HorizontalAlignment="Left"
        Margin="74,616,0,0" VerticalAlignment="Top"/>
24      <Image Name="GambarGerakan"  HorizontalAlignment="Left"
        VerticalAlignment="Top" Margin="1057,89,0,0" Width="155"
        Height="142"></Image>
25    </Grid>
26  </Window>
```

Kode Sumber 4.1 Tampilan Perangkat Lunak

4.3 Implementasi Perangkat Lunak

Pada subbab ini akan dibahas mengenai implementasi perangkat lunak dari kasus penggunaan ke dalam baris kode. Dijelaskan juga dengan fungsi yang dibutuhkan untuk menunjang perangkat lunak ini agar dapat berjalan sebagaimana mestinya. Implementasi ini dilakukan menggunakan Microsoft Visual Studio 2017 dengan bahasa pemrograman C# dan Python.

4.3.1 Implementasi Pendeteksian Skeleton Pengguna

Untuk menjalankan perangkat lunak ini tentunya membutuhkan perangkat keras Kinect 2.0, sehingga dibutuhkan suatu proses untuk mendeteksi *skeleton* pengguna. Sebelum mendeteksi *skeleton*, Kinect 2.0 harus diintegrasikan dengan program terlebih dahulu.

Kode sumber proses integrasi Kinect 2.0 dapat dilakukan seperti pada Kode Sumber 4.2. Untuk melakukan pengenalan gerakan tangan, perangkat lunak mendeteksi tubuh pengguna terlebih dahulu. Ketika tubuh pengguna sudah terdeteksi, perangkat lunak kemudian menggambarkan *skeleton* pengguna secara keseluruhan termasuk 15 *skeleton joints* yang akan digunakan dalam ekstraksi data dan menentukan posisi gerakan.

Kode sumber untuk mendeteksi *skeleton* pengguna dapat dilihat pada Kode Sumber 4.3.

```
1   _sensor = KinectSensor.GetDefault();
2
3   if (_sensor != null)
4   {
5      _sensor.Open();
6      _bodies = new Body[_sensor.BodyFrameSource.BodyCount];
7      _camera = _sensor.OpenMultiSourceFrameReader(FrameSourceTypes.Color
       | FrameSource-Types.Depth | FrameSourceTypes.Infrared |
       FrameSourceTypes.Body);
8      _camera.MultiSourceFrameArrived += Read-er_MultiSourceFrameArrived;
9      _reader = _sensor.BodyFrameSource.OpenReader();
10     _reader.FrameArrived += BodyReader_FrameArrived;
11     _recorder = new KinectCSVManager();
12  }
```

Kode Sumber 4.2 Kode Sumber Integrasi Kinect

```
1   void Reader_MultiSourceFrameArrived(object sender,
    MultiSourceFrameArrivedEventArgs e)
2   {
3      var reference = e.FrameReference.AcquireFrame();
4
5      #region Acquire Frame Color
6
7      // Color
8      using (var frame = reference.ColorFrameReference.AcquireFrame())
9      {
10       if (frame != null)
11       {
12          if (_mode == Mode.Color)
13          {
14             camera.Source = frame.ToBitmap();
15          }
16       }
17     }
18
19     using (var frame = reference.BodyFrameReference.AcquireFrame())
20     {
21       if (frame != null)
22       {
23          canvas.Children.Clear();
24          _bodies = new Body[frame.BodyFrameSource.BodyCount];
25          frame.GetAndRefreshBodyData(_bodies);
26
27          foreach (var body in _bodies)
28          {
29             if (body != null)
30             {
31                if (body.IsTracked)
32                {
33                   Joint shoulderRight =
                     body.Joints[JointType.ShoulderRight];
```

```
34                      Joint shoulderCenter = body.Joints[JointType.Neck];
35                      Joint shoulderLeft =
                        body.Joints[JointType.ShoulderLeft];
36                      Joint hip = body.Joints[JointType.SpineBase];
37                      Joint spine = body.Joints[JointType.SpineMid];
38                      Joint elbowRight = body.Joints[JointType.ElbowLeft];
39                      Joint elbowLeft = body.Joints[JointType.ElbowRight];
40                      Joint wristRight = body.Joints[JointType.WristLeft];
41                      Joint wristLeft = body.Joints[JointType.WristRight];
42                      Joint handRight = body.Joints[JointType.HandRight];
43                      Joint handLeft = body.Joints[JointType.HandLeft];
44                      Joint head = body.Joints[JointType.Head];
45
46                      // Menggambar Skeleton
47                      foreach (Joint joint in body.Joints.Values)
48                      {
49                          if (joint.TrackingState == TrackingState.Tracked)
50                          {
51                              //Proyeksi 3D point ke 2D screen space
52                              // 3D space point
53                              CameraSpacePoint jointPosition =
                                joint.Position;
54                              // 2D space point
55                              Point point = new Point();
56
57                              ColorSpacePoint colorPoint =
                                _sensor.CoordinateMapper.MapCameraPointToColor
                                Space(jointPosition);
58                              point.X = float.IsInfinity(colorPoint.X) ? 0 :
                                colorPoint.X;
59                              point.Y = float.IsInfinity(colorPoint.Y) ? 0 :
                                colorPoint.Y;
60
61                              Ellipse ellipse = new Ellipse
62                              {
63                                  Fill = Brushes.Blue,
64                                  Width = 30,
65                                  Height = 30
66                              };
67
68                              Canvas.SetLeft(ellipse, point.X -
                                ellipse.Width / 2);
69                              Canvas.SetTop(ellipse, point.Y -
                                ellipse.Height / 2);
70
71                              canvas.Children.Add(ellipse);
72                          }
73                      }
74                  }
75              }
76          }
77      }
78  }
79  }
```

Kode Sumber 4.3 Kode Sumber Deteksi *Skeleton* Pengguna

4.3.2 Implementasi Proses Menentukan Data dan Ekstraksi Data Akhir

Pada Kode Sumber 4.4 dijelaskan mengenai proses pengambilan fitur pada setiap lintasan gerakan tangan yang ditentukan oleh pengguna berdasarkan rancangan pada Subbab 3.2.1.1. Untuk mendapatkan titik tersebut dilakukan pengambilan data posisi sumbu X, Y, dan Z dari 15 titik yang telah dijelaskan pada Subbab 2.6. Lalu setiap data akan diambil 40 frame, sehingga akan menjadi 15 x 3 x 40 = 1800 data untuk setiap data bahasa isyarat. Setiap data akan disimpan dalam writer untuk nantinya siap ditulis ke dalam file berekstensi .csv.

Fitur yang didapat di atas kemudian disimpan ke dalam sebuah berkas berekstensi .csv. Sebelum disimpan, seluruh atribut yang ada di dalam fitur data diubah terlebih dahulu menjadi atribut tipe double. Kode sumber penyimpanan fitur data dapat dilihat pada Kode Sumber 4.5.

```
1   public void Update(Body body)
2   {
3      if (!IsRecording) return;
4      if (body == null || !body.IsTracked) return;
5
6      string path = Path.Combine(Folder, _current.ToString() +
       ".line");
7
8      using (StreamWriter writer = new StreamWriter(path))
9      {
10        StringBuilder line = new StringBuilder();
11
12        if (!_hasEnumeratedJoints)
13        {
14           _hasEnumeratedJoints = true;
15        }
16
17        if (_current < 40)
18        {
19           line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
             body.Joints[JointType.Head].Position.X,
             body.Joints[JointType.Head].Position.Y,
             body.Joints[JointType.Head].Position.Z));
20           line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
             body.Joints[JointType.Neck].Position.X,
```

```
            body.Joints[JointType.Neck].Position.Y,
            body.Joints[JointType.Neck].Position.Z));
21          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.SpineShoulder].Position.X,
            body.Joints[JointType.SpineShoulder].Position.Y,
            body.Joints[JointType.SpineShoulder].Position.Z));
22          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.ShoulderRight].Position.X,
            body.Joints[JointType.ShoulderRight].Position.Y,
            body.Joints[JointType.ShoulderRight].Position.Z));
23          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.ShoulderLeft].Position.X,
            body.Joints[JointType.ShoulderLeft].Position.Y,
            body.Joints[JointType.ShoulderLeft].Position.Z));
24          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.ElbowRight].Position.X,
            body.Joints[JointType.ElbowRight].Position.Y,
            body.Joints[JointType.ElbowRight].Position.Z));
25          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.ElbowLeft].Position.X,
            body.Joints[JointType.ElbowLeft].Position.Y,
            body.Joints[JointType.ElbowLeft].Position.Z));
26          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.WristRight].Position.X,
            body.Joints[JointType.WristRight].Position.Y,
            body.Joints[JointType.WristRight].Position.Z));
27          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.WristLeft].Position.X,
            body.Joints[JointType.WristLeft].Position.Y,
            body.Joints[JointType.WristLeft].Position.Z));
28          line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
            body.Joints[JointType.HandRight].Position.X,
            body.Joints[JointType.HandRight].Position.Y,
            body.Joints[JointType.HandRight].Position.Z));
            line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
29          body.Joints[JointType.HandLeft].Position.X,
            body.Joints[JointType.HandLeft].Position.Y,
            body.Joints[JointType.HandLeft].Position.Z));
            line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
30          body.Joints[JointType.HandTipRight].Position.X,
            body.Joints[JointType.HandTipRight].Position.Y,
            body.Joints[JointType.HandTipRight].Position.Z));
            line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
31          body.Joints[JointType.HandTipLeft].Position.X,
            body.Joints[JointType.HandTipLeft].Position.Y,
            body.Joints[JointType.HandTipLeft].Position.Z));
            line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
32          body.Joints[JointType.ThumbRight].Position.X,
            body.Joints[JointType.ThumbRight].Position.Y,
            body.Joints[JointType.ThumbRight].Position.Z));
            line.Append(string.Format("{0:0.000},{1:1.000},{2:2.000},",
33          body.Joints[JointType.ThumbLeft].Position.X,
            body.Joints[JointType.ThumbLeft].Position.Y,
            body.Joints[JointType.ThumbLeft].Position.Z));
        }
34      _status = 1;
35      writer.Write(line);
```

```
36          _current++;
37      }
38  }
39
```

Kode Sumber 4.4 Kode Sumber Menentukan Data

```
1   public void Stop()
2   {
3      _status = 0;
4      IsRecording = false;
5      _hasEnumeratedJoints = false;
6
7      Result = @"D:\TA\insert_dataset.csv";
8
9      using (StreamWriter writer = new StreamWriter(Result))
10     {
11        for (int index = 0; index < _current; index++)
12        {
13           string path = Path.Combine(Folder, index.ToString() +
           ".line");
14
15           if (File.Exists(path))
16           {
17              string line = string.Empty;
18
19              using (StreamReader reader = new StreamReader(path))
20              {
21                 line = reader.ReadToEnd();
22              }
23
24              writer.Write(line);
25           }
26        }
27
28
29        for (int i = 0; i < nama.Length; i++)
30        {
31           Debug.Write(nama[i]);
32           if (string.Equals(nama[i], MainWindow.kelasGerakan))
33           {
34              urutan = i;
35              break;
36           }
37           else if(i==nama.Length-1)
38           {
39              urutan = nama.Length;
40           }
41        }
42        writer.Write(string.Format("{0:0.000}", urutan));
43     }
44  }
```

Kode Sumber 4.5 Kode Sumber Menyimpan Hasil Ekstraksi Data dari *Skeleton* Pengguna

4.3.3 Implementasi Proses Training Data

Setelah data disimpan ke dalam file berekstensi .csv, data akan digabung dengan dataset utama. Dataset akan di *train* ke dalam jaringan CNN dan model jaringan yang telah di *train* tersebut akan disimpan ke dalam json.

```
import pandas as pd
import numpy as np
import keras
import time
import os

lenfile2 = open('D:\TA\\namagerakan.txt')
jumlahset = len(pd.read_csv('D:\TA\dataset.csv', header=None))
jumlahkelas = len(lenfile2.readlines())
lenfile2.close()

# get dataset
filenya = pd.read_csv('D:\TA\dataset.csv', header=None)
classes = filenya[1800]

classes = np.array(classes)

data =filenya.drop(labels=1800,axis=1)
data = np.array(data)

x_arr = np.zeros([jumlahset, 40, 15, 3],dtype="float32")

for j in range(len(data)):
   count =0
   ind =0
   dept= 0
   for i in range(0,len(data[j]),3):
      if count==15:
         dept += 1
         count = 0
      x_arr[j, dept, count, 0] = data[j,i]
      x_arr[j, dept, count, 1] = data[j,i+1]
      x_arr[j, dept, count, 2] = data[j,i+2]
      count+=1

class_cat = keras.utils.to_categorical(classes, jumlahkelas)

#cnn part
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D, Activation
from keras.callbacks import CSVLogger
from keras.callbacks import ReduceLROnPlateau
from keras.models import model_from_json
from keras import backend as K
from keras.optimizers import SGD

while True:
```

```
49
50      filemod = os.path.getmtime('D:\TA\dataset.csv')
51      if(os.path.exists('D:\TA\model.json')):
52          jsonmod = os.path.getmtime('D:\TA\model.json')
53      else: jsonmod = 1
54      selisih = jsonmod-filemod
55
56      # cek if dataset recently updated
57      if filemod>jsonmod or not(os.path.exists('D:\TA\model.json')):
58          lenfile = open('D:\TA\dataset.csv')
59          jumlahset = len(lenfile.readlines())
60          lenfile.close()
61          lenfile2 = open('D:\TA\\namagerakan.txt')
62          jumlahkelas =  len(lenfile2.readlines())
63          lenfile2.close()
64
65          # get dataset
66
67          filenya = pd.read_csv('D:\TA\dataset.csv', header=None)
68          classes = filenya[1800]
69          #1800->40(jumlah frame)x15(jumlah titik)x3(jumlah axis)
70          classes = np.array(classes)
71
72          data =filenya.drop(labels=1800,axis=1)
73          data = np.array(data)
74
75          x_arr = np.zeros([jumlahset, 40, 15, 3],dtype="float32")
76
77          for j in range(len(data)):
78              count =0
79              ind =0
80              dept= 0
81              for i in range(0,len(data[j]),3):
82                  if count==15:
83                      dept += 1
84                      count = 0
85              x_arr[j, dept, count, 0] = data[j,i]
86              x_arr[j, dept, count, 1] = data[j,i+1]
87              x_arr[j, dept, count, 2] = data[j,i+2]
88              count+=1
89
90          class_cat = keras.utils.to_categorical(classes, jumlahkelas)
91          #convert vector classes ke matrix kelas ukuran jumlahkelasx2
92
93          #cnn part
94          opt = SGD(lr=0.001)
95
96          model = Sequential()
97
98          # first set of CONV => ACTIVATION => POOL
99          model.add(Conv2D(200, (1, 1), padding="same",
100         data_format="channels_last",
101         input_shape=(40,15,3)))
102         model.add(Activation("tanh"))
103         model.add(MaxPooling2D(pool_size=(5, 5), strides=(2, 2)))
104
105         # second set of CONV => ACTIVATION => POOL
```

```
106         model.add(Conv2D(200, (1, 1), padding="same"))
107         model.add(Activation("tanh"))
108         model.add(MaxPooling2D(pool_size=(5, 5), strides=(2, 2)))
109 
110         # set of FC => ACTIVATION layers
111         model.add(Flatten())
112         model.add(Dense(500))
113         model.add(Activation("tanh"))
114 
115         # softmax classifier
116         model.add(Dense(jumlahkelas))
117         model.add(Activation("softmax"))
118 
119         model.compile(loss="categorical_crossentropy", optimizer=opt,
            metrics=["accuracy"])
120 
121         history = model.fit(x_arr,class_cat, batch_size=20,
            epochs=2500, verbose=1)
122 
123         scores = model.evaluate(x_arr, class_cat, verbose=1)
124         print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
125 
126         # serialize model to JSON
127         model_json = model.to_json()
128         with open("model.json", "w") as json_file:
129             json_file.write(model_json)
130         # serialize weights to HDF5
131         model.save_weights("model.h5")
132         print("Saved model to disk")
133 
134     time.sleep(1)
135 
```

Kode Sumber 4.6 Kode Sumber Implementasi Proses *Training* Data

4.3.4 Implementasi Proses Testing Data

Di awal proses *testing*, program akan *load* file json dari model yang telah di *train*. Lalu file *testing* data akan dicek ke dalam model yang telah di *train*. Hasil yang dikeluarkan akan diklasifikasi dan aplikasi akan memberi *output* berupa nama gerakan dan gambar dari gerakannya.

```
1   import pandas as pd
2   import numpy as np
3   import keras
4   import time
5   import os
6   import platform
7 
```

```
8   from keras.models import Sequential
9   from keras.layers import Dense, Dropout, Flatten
10  from keras.layers import Conv2D, MaxPooling2D, Activation
11  from keras.callbacks import CSVLogger
12  from keras.callbacks import ReduceLROnPlateau
13  from keras.models import model_from_json
14  from keras import backend as K
15  from keras.optimizers import SGD
16  opt = SGD(lr=0.001)
17
18  # load json and create model
19  json_file = open('D:\TA\model.json', 'r')
20  loaded_model_json = json_file.read()
21  # json_file.close()
22  loaded_model = model_from_json(loaded_model_json)
23  # load weights into new model
24  loaded_model.load_weights("D:\TA\model.h5")
25
26  loaded_model.compile(loss='categorical_crossentropy', optimizer=opt,
    metrics=['accuracy'])
27
28  while True:
29     filemod = os.path.getmtime('D:\TA\dataset.csv')
30     if(os.path.exists('D:\TA\model.json')):
31        jsonmod = os.path.getmtime('D:\TA\model.json')
32     else:
33        jsonmod = 1
34     selisih = jsonmod-filemod
35     # cek if dataset recently updated
36     if filemod<jsonmod:
37        json_file = open('D:\TA\model.json', 'r')
38        loaded_model_json = json_file.read()
39        # json_file.close()
40        loaded_model = model_from_json(loaded_model_json)
41        # load weights into new model
42        loaded_model.load_weights("D:\TA\model.h5")
43
44        loaded_model.compile(loss='categorical_crossentropy',
45        optimizer=opt, metrics=['accuracy'])
46
47     else:
48        print("Please train your dataset")
49        break
50
51     # cek if datatest exist
52     if os.path.exists('D:\TA\datatest.csv'):
53
54        #get datatest
55        filetest = pd.read_csv('D:\TA\\datatest.csv', header=None)
56        classes2 = filetest[1800]
57
58        classes2 = np.array(classes2)
59
60        data2 =filetest.drop(labels=1800,axis=1)
61        data2 = np.array(data2)
62
63        y_arr = np.zeros([1, 40, 15, 3],dtype="float32")
```

```
64
65          for j in range(len(data2)):
66              count =0
67              ind =0
68              dept= 0
69              for i in range(0,len(data2[j]),3):
70                  if count==15:
71                      dept += 1
72                      count = 0
73              y_arr[j, dept, count, 0] = data2[j,i]
74              y_arr[j, dept, count, 1] = data2[j,i+1]
75              y_arr[j, dept, count, 2] = data2[j,i+2]
76              count+=1
77
78          #testing
79          acc = loaded_model.predict_classes(y_arr,verbose=1)
80          K.clear_session()
81          del loaded_model
82          np.savetxt('hasil.txt', acc, fmt='%d', delimiter=';')
83          for j in range(len(acc)):
84              print("{}".format(acc[j]))
85
86          #delete datatest file
87          os.remove('D:\TA\datatest.csv')
88
89      time.sleep(1)
```

Kode Sumber 4.7 Kode Sumber Implementasi Proses *Testing* Data

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada perangkat yang dikembangkan. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsional secara keseluruhan. Pengujian dilakukan dengan beberapa skenario. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

5.1 Lingkungan Pembangunan

Dalam membangun perangkat lunak ini digunakan beberapa perangkat pendukung baik perangkat keras maupun perangkat lunak lainnya. Perangkat keras yang digunakan dalam pembuatan perangkat lunak ini adalah sebuah laptop yang memiliki spesifikasi sebagai berikut:

- Prosesor Intel(R) Core(TM) i5-7200U CPU @ 2.60GHz
- Memori (RAM) 8.00 GB
- GPU Nvidia GT940M
- Kinect Sensor 2.0

5.2 Lingkungan Pengujian

Pengujian dibagi menjadi 2 kelompok secara besar, yaitu kelompok pertama pengujian untuk mendapatkan parameter yang optimal. Pada pengujian yang pertama dilakukan pembagian terhadap data *training* dan data *testing* dari dataset awal yang berjumlah 1300 menjadi 1000 data *training* dan 300 data *testing*. Parameter yang diuji adalah sebagai berikut:

A1. Pengujian untuk mendapatkan jumlah filter

Filter adalah sebagian kecil data yang memiliki data yang mencolok. Semakin banyak jumlah *filter*, hasil fitur dari layer konvolusi akan semakin banyak, maka semakin cepat pula akurasi dari jaringan tersebut bertambah.

A2. Pengujian untuk mendapatkan ukuran dan stride filter.

Stride adalah jumlah langkah untuk mengaplikasikan filter. Semakin kecil stride dari data tersebut, filter akan melakukan lebih banyak proses konvolusi, memiliki parameter lebih banyak, sehingga akurasi akan meningkat.

A3. Pengujian untuk mendapatkan ukuran pool.
Dalam *pooling layer*, akan melakukan *downsampling*, yaitu merepresentasikan beberapa data dengan diwakili oleh 1 data.

A4. Pengujian untuk mendapatkan stride pool.
Stride dalam *pooling layer* adalah seberapa besar langkah tiap proses *pooling.*

A5. Pengujian untuk mendapatkan fungsi aktivasi.
Fungsi aktivasi sangat penting untuk membuat output dari konvolusi menjadi *non-linear*.

A6. Pengujian untuk mendapatkan jumlah dense layer.
Dense *layer* melakukan pembuatan *node* untuk memasukkan keluaran *layer* sebelumnya, lalu akan diproses dengan *weight* dari *dense layer* itu sendiri.

Setelah didapatkan parameter yang optimal, akan dilakukan pengujian langsung terhadap pengguna yang bagi menjadi 3 skenario berikut:

B1. Pengujian skenario 1
Pengujian akurasi yang dilakukan oleh penulis dengan menggunakan data *training* yang sudah ada sebelumnya.

B2. Pengujian skenario 2
Pengujian akurasi yang dilakukan oleh pengguna lain menggunakan data *training* yang sudah ada sebelumnya.

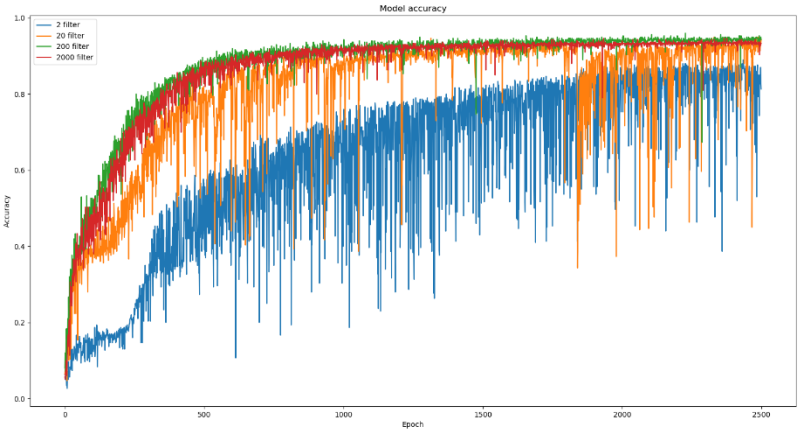
B3. Pengujian skenario 3
Pengujian yang dilakukan oleh pengguna lain dengan data *training* baru yang ditambahkan oleh pengguna tersebut.

5.3 Skenario Optimasi Variabel Jaringan CNN

CNN terdiri dari beberapa *layer* yang membangun jaringan itu sendiri. Tiap *layer* ini memiliki parameter tersendiri. Maka pada subbab ini akan dilakukan pengujian terhadap parameter yang digunakan di dalam setiap *layer*.

5.3.1 Jumlah Filter / Kernel

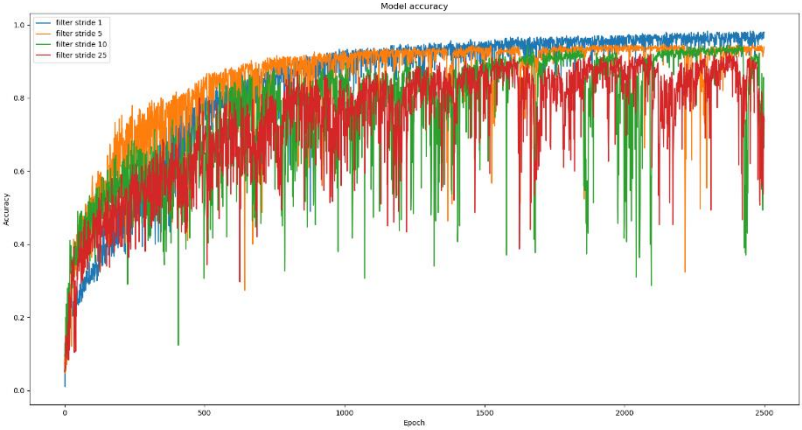
Dalam pengujian ini dilakukan perbandingan untuk jumlah filter / kernel. Filter yang dibandingkan berjumlah 2 (biru), 20 (oranye), 200 (hijau), dan 2000 filter (merah). Seperti yang dapat dilihat pada gambar 5.1, akurasi sudah bisa diatas 93% di saat jumlah filter 20, tapi memiliki varian akurasi yang jauh lebih besar. Hasil paling baik terlihat pada 200 filter, karena dapat menghasilkan akurasi yang maksimal dengan *epoch* yang lebih sedikit. Saat menggunakan 2000 filter, akurasinya sedikit lebih rendah dibandingkan dengan menggunakan 200 filter, dan pertambahan waktu tiap *epoch*-nya hampir 10x lipat.



Gambar 5.1 Grafik Pengujian Jumlah Filter/Kernel

5.3.2 Filter Stride

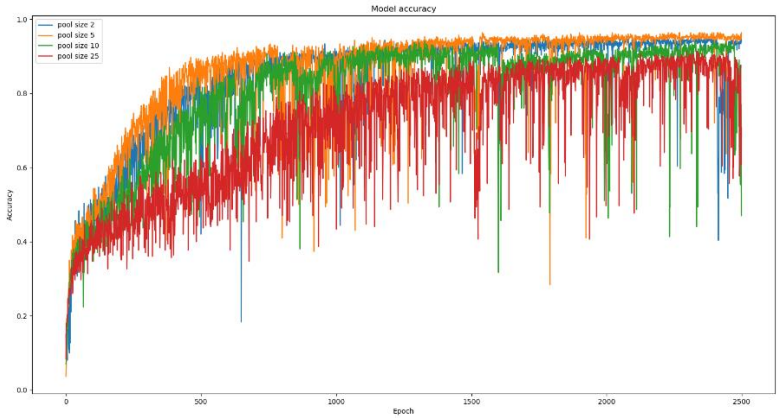
Dalam pengujian kali ini dilakukan perbandingan ukuran *stride* dari filter. Ukuran *stride* yang dibandingkan berjumlah 1 (biru), 5 (oranye), 10 (hijau), dan 25 (merah). Seperti yang dapat dilihat pada gambar 5.2, di akhir proses *training* filter dengan stride yang lebih kecil, yaitu 1, dapat mencapai akurasi yang lebih tinggi, meskipun dengan ukuran filter stride 5 akan mendapat akurasi yang lebih tinggi dengan *epoch* yang lebih sedikit.



Gambar 5.2 Grafik Pengujian Ukuran *Filter Stride*

5.3.3 Ukuran Pool

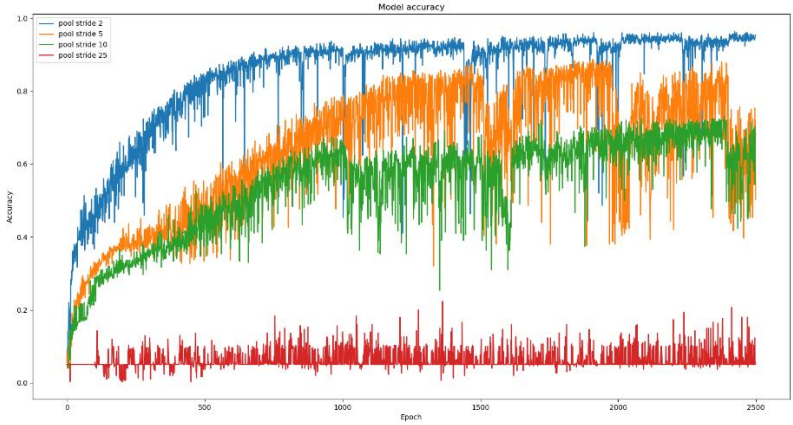
Dalam pengujian ini dilakukan perbandingan terhadap parameter ukuran *pool.* Ukuran *pool* yang dibandingkan berjumlah 2 (biru), 5 (oranye), 10 (hijau), dan 25 (merah). Dapat dilihat pada gambar, ukuran *pool* lebih kecil memiliki akurasi lebih besar. Namun pada ukuran *pool* yang lebih kecil, yaitu 2, akurasinya tidak dapat melebihi akurasi dari *pool* diatasnya yang berjumlah 5. Hal ini disebabkan oleh terlalu kecilnya ukuran *pool* sehingga saat melakukan *downsampling*, ada fitur penting yang tidak masuk ke dalam hasil *pooling.*



Gambar 5.3 Grafik Pengujian Ukuran *Pool Size*

5.3.4 Pool Stride

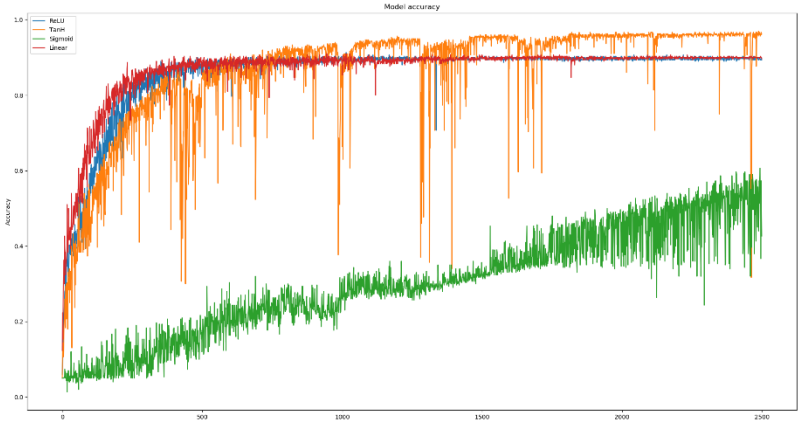
Pada pengujian ini dilakukan perbandingan terhadap ukuran *pool stride.* Ukuran *pool stride* yang digunakan berjumlah 2 (biru), 5 (oranye), 10 (hijau), dan 25 (merah). Seperti yang dapat dilihat pada gambar, akurasi menggunakan *stride* lebih kecil akan lebih besar, karena membuat hasil gambar dari *downsampling* memiliki ukuran lebih besar dan memiliki lebih banyak fitur penting. *Stride* dengan ukuran 2 memberi hasil yang konsisten dan akurasi yang paling besar.



Gambar 5.4 Grafik Pengujian Ukuran Pool Stride

5.3.5 Fungsi Aktivasi

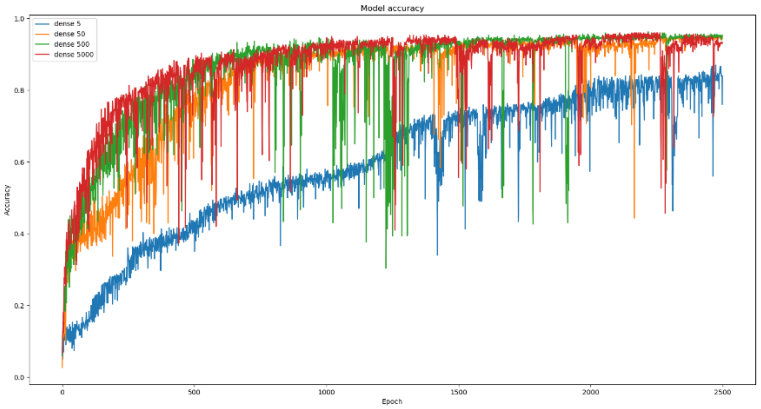
Pada pengujian ini dilakukan perbandingan fungsi aktivasi. Beberapa fungsi aktivasi yang sering digunakan antara lain fungsi identitas / linear, sigmoid, TanH, dan juga ReLU. Fungsi softmax sendiri biasanya hanya digunakan di akhir dari *fully-connected layer*, maka dari itu tidak akan dibandingkan dengan yang lainnya. Fungsi aktivasi yang dibandingkan adalah ReLU (biru), TanH (oranye), Sigmoid (hijau), dan Linear (merah). Seperti yang dapat dilihat pada gambar, untuk fungsi sigmoid peningkatan akurasinya terlihat jauh lebih lambat. Fungsi linear dan ReLU memiliki akurasi yang sudah cukup tinggi yang dicapai cukup cepat, pada 500 *epoch.* Untuk fungsi tanh akurasinya dapat menjadi lebih tinggi dari yang lainnya, namun variasi datanya cukup besar dan baru melewati akurasi dari yang lainnya saat hampir menyentuh 1000 *epoch*.



Gambar 5.5 Grafik Pengujian Fungsi Aktivasi

5.3.6 Jumlah Dense Layer

Pada pengujian in dilakukan perbandingan ukuran dari *dense layer*. Jumlah *Dense Layer* yang digunakan berjumlah 5 (biru), 50 (oranye), 500 (hijau), dan 5000 (merah). Semakin banyak *dense layer* akan memberi semakin banyak *node*-nya. Semakin banyak *node*, semakin banyak *weight* yang harus disesuaikan, maka jaringan pun akan lebih akurat. Seperti yang dapat dilihat pada gambar, jumlah *dense layer* 50 memiliki akurasi yang meningkat cukup jauh dari jumlah *dense layer* yang lebih sedikit, yaitu 5. Namun, penambahan jumlah *dense layer* melebihi itu tidak efektif karena tidak memberi dampak yang berarti terhadap akurasi jaringan keseluruhan.



Gambar 5.6 Grafik Pengujian Ukuran *Dense Layer*

5.4 Skenario Pengujian Bahasa Isyarat *Real Time*

Pengujian dilakukan terhadap 20 bahasa isyarat yang terdiri dari 14 gerakan dinamis dan 6 gerakan statis yang dipilih oleh penulis seperti yang dapat dilihat pada Gambar 2.1. Bahasa isyarat ini dipilih berdasarkan kemiripannya. Banyak dari gerakan tersebut memiliki arah gerakan yang sama, maupun awal atau akhir gerakan. Bahasa isyarat tersebut adalah sebagai berikut:

1. Abang
2. Adik
3. Anak
4. Anda
5. Baik
6. Bapak
7. Baris
8. Bayi
9. Buku
10. Doa
11. Halo
12. Ibu

13. Jahat
14. Jumpa
15. Maaf
16. Makan
17. Pagi
18. Rumah
19. Saya
20. Tambal

5.4.1 Pengujian Skenario B1 dan Analisis

Pada pengujian skenario B1, uji coba dilakukan sendiri oleh penulis yang mempunyai karakteristik tinggi badan 170 cm dengan menggunakan data gerakan sebanyak 100 data. Skenario dapat dilihat pada Tabel 5.1.

Tabel 5.1 Skenario Pengujian B1

Nama Skenario Pengujian	Pengujian Akurasi B1
Kode	SP-B1
Algoritma	Convolutional Neural Network
Jumlah Data	100 data
Penguji	Penulis memiliki tinggi 170 cm
Prosedur Pengujian	Penulis melakukan uji coba 20 gerakan bahasa isyarat pokok dimana masing-masing gerakan dilakukan uji coba sebanyak 5 kali
Hasil yang Diperoleh	Akurasi 78%

Kesalahan klasifikasi terjadi pada beberapa gerakan. Menurut pengamatan penulis berdasarkan skenario B1 yang dilakukan, kesalahan klasifikasi tersebut terjadi karena posisi dan gerakan yang dilakukan kurang sesuai dengan gerakan bahasa

isyarat yang diinginkan dan ada beberapa gerakan yang memiliki kemiripan posisi tangan.

5.4.2 Pengujian Skenario B2 dan Analisis

Pada pengujian skenario B2, uji coba dilakukan oleh pengguna yang mempunyai karakteristik tinggi badan 155 cm dengan menggunakan data gerakan sebanyak 100 data hasil implementasi yang dilakukan oleh pengguna lain. Skenario dapat dilihat pada Tabel 5.2.

Tabel 5.2 Skenario Pengujian B2

Nama Skenario Pengujian	Pengujian Akurasi B2
Kode	SP-B2
Algoritma	Convolutional Neural Network
Jumlah Data	100 data
Penguji	Pengguna memiliki tinggi 155 cm
Prosedur Pengujian	Pengguna melakukan uji coba 20 gerakan bahasa isyarat pokok dimana masing-masing gerakan dilakukan uji coba sebanyak 5 kali
Hasil yang Diperoleh	Akurasi 59%

Hasil yang didapatkan pada skenario B2 cukup baik namun akurasi yang didapatkan lebih rendah dibandingkan dengan skenario A, seperti yang dapat dilihat pada Tabel 5.5. Kesalahan klasifikasi terjadi pada beberapa gerakan.

Berdasarkan skenario B2 yang dilakukan, disamping gerakan yang dilakukan kurang sesuai dengan gerakan bahasa isyarat yang diinginkan (terlebih penguji belum terbiasa mempraktikkan bahasa isyarat), menurut pengamatan penulis perbedaan karakteristik tinggi badan yang signifikan antara penulis sebagai objek dalam pembuatan *dataset* dan penguji mempunyai pengaruh terhadap tingkat akurasi dari klasifikasi fitur dinamis.

Hal tersebut disebabkan karena tinggi badan pengguna turut andil dalam menentukan posisi gerakan seperti yang dijelaskan pada Subbab 4.3.2. Dan perbedaan karakteristik tinggi badan juga mempunyai efek terhadap identifikasi *skeleton joints* oleh Kinect 2.0 karena adanya perbedaan koordinat Kinect 2.0 dalam mendeteksi *skeleton joints*.

5.4.3 Pengujian Skenario B3 dan Analisis

Pada pengujian skenario B3, uji coba dilakukan oleh pengguna yang mempunyai karakteristik tinggi badan 155 cm dengan menggunakan data gerakan sebanyak 100 data yang dilakukan oleh pengguna lain. Skenario dapat dilihat pada Tabel 5.3.

Tabel 5.3 Skenario Pengujian B3

Nama Skenario Pengujian	Pengujian Akurasi B3
Kode	SP-B3
Algoritma	Convolutional Neural Network
Jumlah Data	100 data
Penguji	Pengguna memiliki tinggi 155 cm
Prosedur Pengujian	Pengguna melakukan uji coba 20 gerakan bahasa isyarat pokok dimana masing-masing gerakan dilakukan uji coba sebanyak 5 kali
Hasil yang Diperoleh	Akurasi 76%

Hasil yang didapatkan pada skenario B3 cukup baik. Akurasi yang didapatkan hampir sama seperti skenario B1, seperti yang dapat dilihat pada Tabel 5.6. Menurut pengamatan penulis, hasil klasifikasi yang rendah terjadi karena posisi dan gerakan yang dilakukan kurang sesuai dengan gerakan bahasa isyarat yang diinginkan dan terdapat beberapa gerakan yang memiliki kemiripan dengan gerakan lain.

Tabel 5.4 Terjemahan Isyarat Kata Skenario Pengujian B1

		Target Kelas																			
		Abang	Adik	Anak	Anda	Baik	Bapak	Baris	Bayi	Buku	Doa	Halo	Ibu	Jahat	Jumpa	Maaf	Makan	Pagi	Rumah	Saya	Tambal
Hasil Uji	Abang	2		2								3									
	Adik		5	2																	
	Anak	1		1																	
	Anda				5																
	Baik					3														1	
	Bapak						3									1					
	Baris							4						1				1			
	Bayi								5	1											
	Buku									3											
	Doa										4										
	Halo	2										2									
	Ibu						1						5			1					
	Jahat													4							
	Jumpa									1	1				5						
	Maaf						1									3					
	Makan																4				
	Pagi							1										4			
	Rumah																		5		
	Saya					2											1			4	
	Tambal																				5

Tabel 5.5 Terjemahan Isyarat Kata Skenario Pengujian B2

		Target Kelas																			
		Abang	Adik	Anak	Anda	Baik	Bapak	Baris	Bayi	Buku	Doa	Halo	Ibu	Jahat	Jumpa	Maaf	Makan	Pagi	Rumah	Saya	Tambal
Hasil Uji	Abang		1	1																	
	Adik		3																		
	Anak			1																	
	Anda		1		5															1	
	Baik					1														1	
	Bapak						3						4			1	1				
	Baris							4													
	Bayi								1												
	Buku								4	5					3						1
	Doa										5										
	Halo	1		3								5									
	Ibu	1					1						1								
	Jahat													4				4			
	Jumpa														2						
	Maaf	3					1							1		4	2				
	Makan																2				
	Pagi							1										1			
	Rumah																		5		
	Saya					4														3	
	Tambal																				4

Tabel 5.6 Terjemahan Isyarat Kata Skenario Pengujian B3

		Target Kelas																			
		Abang	Adik	Anak	Anda	Baik	Bapak	Baris	Bayi	Buku	Doa	Halo	Ibu	Jahat	Jumpa	Maaf	Makan	Pagi	Rumah	Saya	Tambal
Hasil Uji	Abang	3	1	2																	
	Adik	1	3	2																	
	Anak			1																	
	Anda		1		5																
	Baik					2														2	
	Bapak						3						2			1	1				
	Baris							4										1			
	Bayi								4												
	Buku								1	5											1
	Doa										5				1						
	Halo	1										5									
	Ibu						1						3								
	Jahat													5				1			
	Jumpa														4						
	Maaf						1									4	2				
	Makan																2				
	Pagi							1										3			
	Rumah																		5		
	Saya					3														3	
	Tambal																				4

5.5 Evaluasi

Subbab ini membahas mengenai evaluasi terhadap pengujian-pengujian yang telah dilakukan. Dalam hal ini evaluasi menunjukkan data hasil pengujian yang telah dipaparkan pada Subbab 5.3 dan 5.4. Evaluasi disampaikan dalam bentuk analisis hasil secara keseluruhan sebagai berikut:

1. Jaringan CNN yang optimum berdasarkan hasil dari pengujian A1 hingga A6 adalah sebagai berikut:
    - Jumlah *epoch* yang dilakukan yaitu 2500
    - Jumlah *filter* yang digunakan adalah 200
    - Ukuran & *stride filter* yang digunakan adalah 1 x 1
    - Ukuran *pool* yang digunakan adalah 5 x 5
    - *Stride pool* yang digunakan adalah 2 x 2
    - Fungsi Aktivasi yang digunakan adalah TanH
    - Jumlah *Dense Layer* yang digunakan adalah 200
2. Adanya beberapa gerakan bahasa isyarat yang hampir sama antara satu dengan yang lain menyebabkan kesalahan pada proses klasifikasi. Seperti yang dapat dilihat pada Gambar 5.7, gerakan bahasa isyarat yang menunjukkan kata “Saya” mempunyai kemiripan dengan gerakan bahasa isyarat yang menunjukkan kata “Baik”. Kemudian gerakan bahasa isyarat yang menunjukkan kata “Ibu” mempunyai kemiripan dengan gerakan bahasa isyarat yang menunjukkan kata “Bapak” dan “Maaf”.

Gambar 5.7 Kemiripan Gerakan dari Beberapa Bahasa Isyarat

3. Selain adanya beberapa gerakan bahasa isyarat dinamis yang hampir sama antara satu dengan yang lain, sensitifitas Kinect 2.0 dalam merekam koordinat masing-masing *skeleton joints* juga mempengaruhi tingkat akurasi klasifikasi gerakan bahasa isyarat.
4. Bahasa isyarat yang tidak mempunyai kemiripan gerakan secara signifikan antara gerakan satu dengan yang lainnya tidak menutup kemungkinan menimbulkan kesalahan klasifikasi gerakan bahasa isyarat. Analisis dapat dijelaskan sebagai berikut:
    1) Pada gerakan "Ibu" terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil gerakan "Bapak". Hal ini disebabkan karena kedua gerakan tersebut mempunyai gerakan yang serupa dan posisinya yang berdekatan. Begitu pun dengan sebaliknya.

2) Pada gerakan “Abang”, terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil “Adik” atau “Anak”. Kesalahan klasifikasi disebabkan karena ketiga gerakan tersebut memiliki gerakan awal yang serupa. Begitu pun dengan sebaliknya.
3) Pada gerakan “Halo”, terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil “Anak” dan “Abang”. Kesalahan klasifikasi ini disebabkan karena ketiga gerakan tersebut memiliki arah gerakan yang hampir sama, hanya berbeda pada bentuk tangannya saja. Begitu pun dengan sebaliknya.
4) Pada gerakan “Saya”, terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil “Baik”. Kesalahan klasifikasi disebabkan oleh gerakan yang sama persis, yang membedakan antar keduanya hanyalah bentuk tangan saja. Begitu pun dengan sebaliknya.
5) Pada gerakan “Buku” terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil “Bayi” atau “Jumpa”. Kesalahan klasifikasi disebabkan oleh gerakan dari ketiga kata tersebut hanya berpusat di satu daerah yang sama. Begitu pun dengan sebaliknya.
6) Pada gerakan “Pagi” terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil “Jahat”. Kesalahan klasifikasi disebabkan oleh gerakan akhir yang serupa. Begitupun dengan sebaliknya.
7) Pada gerakan “Makan” terdapat beberapa kesalahan klasifikasi yaitu mendapatkan hasil “Maaf” atau “Bapak”. Kesalahan klasifikasi disebabkan oleh gerakan dari ketiga kata tersebut hanya berpusat di satu daerah yang sama. Begitu pun dengan sebaliknya.

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1 Kesimpulan

Dari proses pengerjaan selama perancangan, implementasi, dan proses pengujian aplikasi yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Titik skeleton yang mewakilkan seluruh gerakan yang diperlukan dalam komputasi CNN cukup hanya meliputi tubuh bagian atas saja, yang berjumlah 15 titik.
2. Perangkat lunak yang dibangun pada Tugas Akhir ini dapat menerjemahkan bahasa isyarat dengan akurasi mencapai 78%.
3. Kesalahan klasifikasi banyak terjadi pada gerakan yang memiliki kemiripan dari arah gerakannya maupun bentuk tangannya.

6.2 Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Di antaranya adalah sebagai berikut:

1. Memperbanyak data *training* dari berbagai macam pengguna yang mempunyai karakteristik tubuh yang berbeda-beda maupun memperbanyak dengan variasi gerakan yang berbeda.
2. Menggunakan perangkat tambahan, seperti *Leap Motion*, untuk mengatasi kesalahan dalam klasifikasi yang dikarenakan oleh perbedaan gerakan yang minim dan hanya berbeda pada bentuk tangan saja.

3. Menambah fitur klasifikasi gambar, tidak hanya menggunakan *skeleton* sebagai fitur yang dipertimbangkan untuk klasifikasi, dapat menggunakan metode dan algoritma yang serupa (CNN).
4. Interaksi antar aplikasi dapat menggunakan *socket,* agar *delay* yang terjadi tidak terlalu besar.

DAFTAR PUSTAKA

[1] A. Basuki, M. Zikky, J. A. Nur Hasim and N. I. Ramadhan, "Sensor Gerak dengan Leap Motion untuk Membantu Komunikasi Tuna Rungu/Wicara," vol. 8, pp. A317-A321, 2016.

[2] I. H. A. Hasibuan, A. Mulyana and A. Brian O, "Perancangan dan Implementasi Aplikasi Penerjemah Bahasa Isyarat Menjadi Suara Berbasis Kinect Menggunakan Metode Dynamic Time Warping," pp. 1-7.

[3] A. A. S. Gunawan and A. Salim, "Pembelajaran Bahasa Isyarat dengan Kinect dan Metode Dynamic Time Warping," vol. 13, pp. 77-84, 2013.

[4] W. N. Khotimah, Y. A. Susanto and N. Suciati, "Combining Decision Tree and Back Propagation Genetic Algorithm Neural Network for Recognizing Word Gestures in Indonesian Sign Language using Kinect," *Journal of Theoritical and Applied Information Technology,* vol. 95, pp. 292-298, 2017.

[5] A. W. Yanuardi, S. Prasetio and P. P. J. Adi, "Indonesian Sign Language Computer Application for the Deaf," *2nd International Conference on Education Technology and Computer,* vol. 5, 2010.

[6] T. Anggita, "Pengenalan Bahasa Isyarat Indonesia dengan Metode Dynamic Time Warping (DTW) Menggunakan Kinect 2.0," Institut Teknologi Sepuluh Nopember, Surabaya, 2017.

[7] Y. Chen, B. Luo, Y.-L. Chen, G. Liang and X. Wu, "A Real-time Dynamic Hand Gesture Recognition System Using Kinect Sensor," *IEEE Conference on Robotics and Biometrics,* pp. 2026-2030, 2015.

[8] "Kinect," Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Kinect. [Accessed 7 June 2017].

[9] "Python," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Python_(programming_language). [Accessed 14 July 2018].

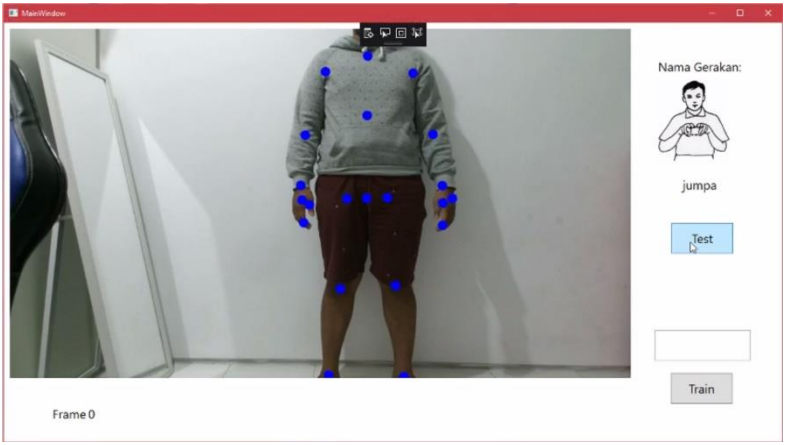
[10] "Keras," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Keras. [Accessed 25 November 2018].

[11] "Microsoft Visual Studio," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Microsoft_Visual_Studio. [Accessed 11 July 2017].

[12] "Convolutional Neural Network," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Convolutional_neural_network. [Accessed 6 July 2018].

[13] N. Priyadharsini and N. Rajeswari, "SIgn Language Recognition using Convolutional Neural Networks," *International Journal on Recent and Innovative Trends in Computing and Communication,* vol. 5, no. 6, pp. 625-628, 2017.

LAMPIRAN A
SCREENSHOT PERANGKAT LUNAK



[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Muhammad Fahmi Abdurrahman merupakan anak dari pasangan Bapak Desrial dan Ibu Titi Candra Sunarti. Lahir di Bogor pada tanggal 30 November 1995. Penulis menempuh Pendidikan formal dimulai dari TK Kartika III Bogor (2001-2002), SDN Polisi 4 Bogor (2002-2008), SMP Negeri 1 Bogor (2008-2011), SMA Negeri 1 Bogor (2011-2014), dan S1 Departemen Informartika ITS (2014-2018). Bidang studi yang diambil oleh penulis pada saat berkuliah di Departemen Informatika ITS adalah Interaksi Grafika dan Seni (IGS). Penulis aktif dalam organisasi dan UKM seperti Mahkamah Mahasiswa ITS (2015-2016) dan juga Victory Sepuluh Nopember Marching Corps (2014-2016), lalu juga dalam kepanitiaan seperti Integralistik Festival ITS (2015). Penulis memiliki hobi masak dan bermain game. Penulis dapat dihubungi melalui email m.fahmi@live.com.