SISTEM PENERJEMAH BAHASA ISYARAT OTOMATIS MENGGUNAKAN METODE DEEP LEARNING MODEL CONVOLUTIONAL NEURAL NETWORK (CNN)

SKRIPSI

sebagai salah satu syarat untuk memperoleh gelar Sarjana Terapan (S.Tr) di
Program Studi Teknik Informatika
Jurusan Teknologi Informasi

oleh
Rizkika Zakka Palindungan
NIM E41170164

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2021

SISTEM PENERJEMAH BAHASA ISYARAT OTOMATIS MENGGUNAKAN METODE DEEP LEARNING MODEL CONVOLUTIONAL NEURAL NETWORK (CNN)

SKRIPSI

sebagai salah satu syarat untuk memperoleh gelar Sarjana Terapan (S.Tr) di
Program Studi Teknik Informatika
Jurusan Teknologi Informasi

oleh
Rizkika Zakka Palindungan
NIM E41170164

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI JEMBER
2021

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
POLITEKNIK NEGERI JEMBER
JURUSAN TEKNOLOGI INFORMASI

SISTEM PENERJEMAH BAHASA ISYARAT OTOMATIS MENGGUNAKAN METODE DEEP LEARNING MODEL CONVOLUTIONAL NEURAL NETWORK (CNN)

RIZKIKA ZAKKA PALINDUNGAN (E41170164)

Ketua Penguji

Ratih Ayuninghemi, S.ST, M.Kom
NIP 19860802 201504 2 002

Sekretaris Penguji	Anggota Penguji
Aji Seto Arifianto, S.ST., M.T.	Mukhamad Angga G., S. Pd., M. Eng.
NIP 19851128 200812 1 002	NIP 19940812 201903 1 013

Dosen Pembimbing

Aji Seto Arifianto, S.ST., M.T.
NIP 19851128 200812 1 002

Mengesahkan,
Ketua Jurusan Teknologi Informasi

Hendra Yufit Riskiawan, S.Kom, M.Cs
NIP 19830203 200604 1 003

SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini :

Nama : Rizkika Zakka Palindungan

NIM : E41170164

Menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam Laporan Skripsi saya yang berjudul "Sistem Penerjemah Bahasa Isyarat Otomatis Menggunakan Metode Deep Learning Model Convolutional Neural Network (CNN)" merupakan gagasan dan hasil karya saya sendiri dengan arahan komisi pembimbing, dan belum pernah diajukan dalam bentuk apapun pada perguruan tinggi manapun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan dari penulis lain telah disebutkan dalam naskah dan dicantumkan dalam daftar pustaka di bagian akhir Laporan Skripsi ini.

Jember, 1 Agustus 2021

Rizkika Zakka Palindungan

NIM E41170164

PERNYATAAN
PERSETUJUAN PUBLIKASI
KARYA ILMIAH UNTUK KEPENTINGAN
AKADEMIS

Yang bertandatangan di bawah ini, saya:
Nama : Rizkika Zakka Palindungan
NIM : E41170164
Program Studi : Teknik Informatika
Jurusan : Teknologi Informasi

Demi pengembangan Ilmu Pengetahuan, saya menyetujui untuk memberikan kepada UPT. Perpustakaan Politeknik Negeri Jember, Hak Bebas Royalti Non-Eksklusif (Non-Exclusive Royalty Free Right) atas Karya Ilmiah **berupa Laporan Skripsi saya yang berjudul**:

SISTEM PENERJEMAH BAHASA ISYARAT OTOMATIS MENGGUNAKAN METODE DEEP LEARNING MODEL CONVOLUTIONAL NEURAL NETWORK (CNN)

Dengan Hak Bebas Royalti Non-Eksklusifini UPT. Perpustakaan Politeknik Negeri Jember berhak menyimpan, mengalih media atau format, mengelola dalam bentuk pangkalan data (Database), mendistribusikan karya dan menampilkan atau mempublikasikannya di Internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis atau penciptanya.

Saya bersedia untuk menanggung secara pribadi tanpa melibatkan pihak Politeknik Negeri Jember, Segala bentuk tuntutan hukum yang timbul atas Pelanggaran Hak Cipta dalam Karya ilmiah ini.
Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Jember
Pada Tanggal : 1 Agustus 2021
Yang menyatakan,

Nama : Rizkika Zakka Palindungan
NIM : E41170164

MOTTO

PERSEMBAHAN

Laporan skripsi ini saya persembahkan dengan rasa hormat kepada :

1. Orang tua tercinta terima kasih atas semua kasih sayang dan cintanya, dukungan baik moril maupun materil, serta doa yang tak henti dan pengorbanan yang tak terhingga.
2. Para pengajar Politeknik Negeri Jember khususnya dosen dan teknisi Program Studi Teknik Informatika yang telah memberikan banyak ilmu dan pengetahuan.
3. Almamater tercinta Politeknik Negeri Jember.

Sistem Penerjemah Bahasa Isyarat Otomatis Menggunakan Metode Deep Learning Model Convolutional Neural Network (CNN)

Sistem Penerjemah Bahasa Isyarat Otomatis Menggunakan Metode Deep Learning Model Convolutional Neural Network (CNN)

Rizkika Zakka Palindungan
Study Program of Informatics Engineering
Majoring of Information Technology
Program Studi Teknik Informatika
Jurusan Teknologi Informasi

ABSTRACT

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.

***Keywords** : Xxxxxx, Xxxxxx, Xxxxxx, Xxxxxx*

RINGKASAN

Sistem Penerjemah Bahasa Isyarat Otomatis Menggunakan Metode Deep Learning Model Convolutional Neural Network (CNN), Rizkika Zakka Palindungan, NIM E41170164, Tahun 2021, xx hlm., Teknologi Informasi, Politeknik Negeri Jember, Aji Seto Arifianto, S.ST., M.T. (Dosen Pembimbing).

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

PRAKATA

Puji syukur penulis panjatkan kehadirat Allah Swt. Atas berkat rahmat dan karunia-Nya sehingga penulisan laporan skripsi berjudul "Sistem Penerjemah Bahasa Isyarat Otomatis Menggunakan Metode Deep Learning Model Convolutional Neural Network (CNN)" dapat diselesaikan dengan baik. Laporan ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Terapan (S.Tr. Kom) di Program Studi Teknik Informatika Jurusan Teknologi Informasi Politeknik Negeri Jember.

Penulis menyampaikan penghargaan dan ucapan terima kasih yang sebesar-besarnya sebagai berikut:

1. Allah SWT yang telah memberikan petunjuk dan hidayah, kesehatan dan kemudahan dalam menyelesaikan skripsi ini.
2. Direktur Politeknik Negeri Jember.
3. Bapak Hendra Yufit Riskiawan, S.Kom, M.Cs selaku Ketua Jurusan Teknologi Informasi.
4. Ibu Trismayanti Dwi Puspitasari, S.Kom. M.Cs. selaku Ketua Program Studi Teknik Informatika.
5. Bapak Aji Seto Arifianto, S.ST., M.T. selaku Pembimbing.
6. Ibu Ratih Ayuninghemi, S.ST, M.Kom dan Bapak Mukhamad Angga Gumilang, S. Pd., M. Eng. Selaku penguji.
7. Orang tua, kakak, dan seluruh keluarga besar, terima kasih atas semangat dan dukungannya.
8. Rekan-rekanku dan semua pihak yang telah ikut membantu dalam pelaksanaan penelitian dan penulisan laporan ini.

Laporan Skripsi ini masih kurang sempurna, mengharapkan kritik dan saran yang sifatnya membangun guna perbaikan di masa mendatang. Semoga tulisan ini bermanfaat.

Jember, 1 Agustus 2021

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Manusia adalah makhluk sosial yang saling berkomunikasi dengan menggunakan bahasa. Bentuk komunikasi ini biasanya dilakukan secara verbal/lisan, yang artinya komunikasi dengan menggunakan kata-kata. Akan tetapi *The World Health Organization* (WHO) memberikan pernyataan bahwa lebih dari 5% populasi dunia atau 466 juta orang mengalami gangguan pendengaran. Diperkirakan pada tahun 2050 lebih dari 900 juta orang atau satu dari setiap sepuluh orang akan mengalami gangguan pendengaran. Dengan informasi tersebut dapat disimpulkan bahwa akan terjadi peningkatan populasi yang mengalami gangguan pendengaran. Komunitas yang mengalami gangguan pendengaran disebut sebagai tunarungu. Komunitas ini memiliki cara berkomunikasi sendiri tanpa menggunakan bahasa lisan yaitu dengan bahasa isyarat. Sedangkan pada umumnya bahasa isyarat sulit dipahami oleh masyarakat dan membuat komunitas tersebut merasa terasingkan di lingkungannya. Bahasa isyarat itu sendiri merupakan bahasa yang digunakan untuk berkomunikasi dengan menggunakan gerak bibir dan bahasa tubuh, termasuk ekspresi wajah dan pandangan mata. Mengatasi keterbatasan dalam hal berkomunikasi, teknologi *computer vision* bisa menjadi solusi. Teknologi ini mampu menjembatani komunikasi antar manusia melalui mesin. C*omputer vision* merupakan salah satu bidang yang memanfaatkan kecerdasan buatan untuk melatih komputer dalam menafsirkan dan memahami dunia visual. Teknologi ini memanfaatkan data dari kamera atau sensor yang kemudian diolah menggunakan model *deep learning*. Sehingga mesin mampu secara akurat mengidentifikasi dan mengklasifikasikan objek kemudian bereaksi terhadap apa yang mesin “lihat”.

Penelitian ini dilakukan dengan pendekatan berbasis *computer vision*. Pada pendekatan tersebut digunakan data yang berasal dari kamera untuk menangkap gerakan bahasa isyarat. Selanjutnya melakukan praproses seperti mengumpulkan data mentah dengan menggunakan kamera, deteksi objek, segmentasi, konversi warna, *reshaping*, *thresholding* dan operasi morfologi. Setelah data berhasil didapatkan, kemudian melakukan normalisasi untuk pembuatan model dataset. Normalisasi ini memastikan bahwa data yang berhasil dimodelkan bisa dengan

efektif dan efisien digunakan oleh sistem cerdas. Semua praproses tersebut dilakukan untuk memisahkan objek yang sangat penting dari noise. Sampai pada akhirnya sistem komputer dengan teknologi kecerdasan buatan mampu memanfaatkan model dari dataset tersebut dengan maksimal.

Untuk membuat kecerdasan buatan. Salah satu pendekatan yang bisa digunakan adalah menggunakan Jaringan Syaraf Tiruan yang terinspirasi dari jaringan syaraf pada manusia. Konsep tersebut kemudian dikembangkan lebih lanjut dalam *Deep Learning*. *Deep Learning* telah menjadi salah satu topik hangat dalam dunia *Machine Learning* karena kapabilitasnya yang signifikan dalam memodelkan berbagai data kompleks seperti citra dan suara. Metode *Deep Learning* yang saat ini memiliki hasil paling signifikan dalam pengenalan citra adalah *Convolutional Neural Network* (CNN). Hal tersebut dikarenakan CNN berusaha meniru sistem pengenalan citra pada visual *cortex* manusia sehingga memiliki kemampuan mengolah informasi citra.

Sebernarnya konsep penelitian dan pengembangan sistem komputer vision untuk penerjemah bahasa isyarat sudah ada beberapa yang telah dilakukan antara lain : Sistem Pengenalan Bahasa Isyarat Indonesia Dengan Menggunakan Metode Fuzzy K-Nearest Neighbor (Gafar, 2017), Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network (Bakti., 2019), Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network (Fadillah, 2020). Dari beberapa penelitian tersebut masih ditemukan keterbatasan dan kekurangan yang bisa diperbaiki seperti kemampuan sistem yang hanya bisa menerjemahkan bahasa isyarat sederhana, input data yang hanya berupa foto dari model peraga yang diambil secara manual atau tidak *realtime* berasal dari video kamera/webcam, tidak melakukan penghilangan *area* wajah dan leher didalam proses pengolahan citra untuk deteksi objek tangan, dan hasil akurasi yang rendah dalam menerjemahkan bahasa isyarat.

Sebagai perbaikan dan pengembangan sistem lebih lanjut, penelitian ini akan menggunakan Bahasa Isyarat Indonesia (BISINDO) sebagai jenis bahasa isyarat yang dipakai didalam dataset. Selanjutnya peneliti akan menggunakan framework *Mediapipe* untuk mendeteksi objek tangan manusia dan mendapatkan landmark,

hal ini bertujuan untuk meningkatkan kemampuan dan akurasi dalam mendeteksi bahasa isyarat. Kemudian pengambilan citra akan dilakukan dengan menggunakan satu kamera *webcam (selfie)* yang nantinya data citra akan diterjemahkan secara *realtime* ketika objek tangan terdeteksi oleh sistem. Pada proses pengolahan data citra dari kamera akan dilakukan proses pengaturan kecerahan gambar, membalik *(flip)* gambar, konversi warna, *reshaping* citra dan *cropping* data citra. Algoritma yang akan digunakan pada penelitian ini adalah CNN yaitu model arsitektur *Convolutional Neural Network*, algoritma ini dipilih berdasarkan tingkat kemampuan dan akurasi untuk menerjemahkan abjad dan nomor, hal ini dibuktikan dari penelitian berjudul "Automatic recognition of Colombian car license plates using convolutional neural networks and Chars74k database" dengan objek input berupa gambar karakter yang diambil dari pelat mobil Kolombia yang memiliki akurasi di atas 98% (D E Arroyo-Pérez., 2020).

1.2 Rumusan Masalah

Berdasarkan uraian dari latar belakang, terdapat beberapa permasalahan yang bisa dirumuskan yaitu sebagai berikut :

1. Bagaimana implementasi framework mediapipe dalam mendeteksi objek tangan pada data citra dari kamera video?
2. Bagaimana proses pengolahan dan normalisasi data citra digital?
3. Bagaimana implementasi metode *Deep Learning Model Convolutional Neural Network* dalam menerjemahkan gerakan bahasa isyarat?

Batasan masalah yang ada dalam penelitian ini yaitu sebagai berikut :

1. Objek penelitian adalah daerah anggota badan manusia dari pergelangan sampai ujung jari tangan.
2. Pencahayaan ruangan harus terang. Objek penelitian berupa tangan harus masuk dalam frame kamera.
3. Kamera yang digunakan minimal memiliki resolusi sebesar 480 x 360 *pixels*.

1.3 Tujuan Penelitian

Adapun tujuan penelitian ini adalah sebagai berikut :

1. Untuk mengetahui implementasi framework mediapipe dalam mendeteksi objek tangan.
2. Untuk mengetahui teknik pengolahan dan normalisasi data citra yang tepat.
3. Untuk mengetahui hasil metode *Deep Learning Model Convolutional Neural Network* dalam memahami dan menerjemahkan gerakan bahasa isyarat.

1.4 Manfaat Penelitian

Adapun manfaat penelitian ini adalah sebagai berikut :

1. Meningkatkan efisiensi komputasi pada sistem sehingga memiliki tingkat fungsional dan akurasi yang baik didalam menerjemahkan bahasa isyarat.
2. Memberikan data yang sesuai dengan lingkungan arsitektur dari mechine learning dalam proses data training dan prediksi gerakan bahasa isyarat.
3. Mampu mengembangkan software untuk memudahkan masyarakat luas dalam memahami bahasa isyarat dengan bantuan mechine learning.

BAB 2. TINJAUAN PUSTAKA

2.1 Bahasa Isyarat

Bahasa isyarat merupakan jenis komunikasi non verbal karena merupakan bahasa yang tidak menggunakan suara tetapi menggunakan bentuk dan arah tangan, pergerakan tangan, bibir, badan serta ekspresi wajah untuk menyampaikan maksud dan pikiran dari seorang penutur. Kaum tunarungu adalah kelompok utama yang menggunakan bahasa ini. Bahasa isyarat biasanya pengkombinasian dari bentuk, orientasi dan gerak tangan, lengan, tubuh serta ekspresi wajah untuk mengungkapkan isi pikiran (Zuhir dan Amri, 2019).

Di Indonesia, ada dua sistem dari bahasa isyarat yang digunakan yaitu: Bahasa Isyarat Indonesia (BISINDO) dan Sistem Isyarat Bahasa Indonesia (SIBI). SIBI adalah bahasa isyarat yang diperkenalkan secara awal oleh Alm. Anton Widyatmoko yang merupakan mantan kepala sekolah SLB/B Widya Bakti di Semarang dalam proses penciptaannya tidak melalui musyawarah dan persetujuan dari Gerakan Kesejahteraan Tunarungu Indonesia (GERKATIN) tetapi berkolaborasi dengan mantan kepala sekolah SLB/B di Jakarta dan Surabaya dengan hasil akhir sebuah kamus SIBI. BISINDO adalah bahasa isyarat yang mengadopsi nilai budaya asli Indonesia dan mudah dapat digunakan untuk berkomunikasi diantara kaum tunarungu dalam kehidupan sehari - hari. Kecepatan dan kepraktisannya dari BISINDO membuat lebih mudah untuk memahami dan mengerti bagi kaum tunarungu walaupun tidak mengikuti faedah tata bahasa dari bahasa Indonesia (Yunanda, 2018).

2.1.1 Sejarah Bahasa Isyarat Indonesia (BISINDO)

Menurut Laura Lesmana Wijaya selaku Ketua Pusat Bahasa Isyarat Indonesia (Pusbisindo), Bisindo dapat diartikan sebagai sebuah terminologi yang digunakan untuk menunjuk pada bahasa isyarat alami yang digunakan oleh komunitas Tuli di Indonesia. Berdasarkan penjelasan tersebut dapat disimpulkan bahwa sejarah Bisindo sejalan dengan kemunculan bahasa isyarat alami yang terdapat di Indonesia (Yohans, Arjawa dan Punia., 2019).

Kemunculan bahasa isyarat alami diyakini telah berlangsung sejak tahun 1933 ketika sekolah khusus Tuli pertama yaitu Sekolah Luar Biasa (SLB)/B Cicendo, Bandung, Jawa Barat berdiri. Selain itu, terdapat pula sekolah khusus Tuli lainnya yang berdiri pada tahun-tahun berikutnya seperti SLB/B Dena Upakara, Wonosobo, Jawa Tengah (sekolah khusus perempuan) pada tahun 1938, SLB/B Don Bosco, Wonosobo, Jawa tengah (sekolah khusus laki- laki) pada tahun 1955, dan SLB/B Santi Rama (Jakarta) pada tahun 1970 (Tim Produksi Bahasa Isyarat Jakarta, 2014: vii). Penjelasan ini diperkuat dengan keberadaan bahasa isyarat Jakarta yang variasinya berasal dari pencampuran bahasa isyarat asli, termasuk bahasa isyarat yang digunakan oleh orang orang Tuli yang pernah mendapatkan pendidikan formal di sekolah khusus Tuli tersebut (Tim Produksi Bahasa Isyarat Jakarta, 2014: vii).

Perkembangan bahasa isyarat alami di Indonesia tidak serta merta mendapatkan pengakuan oleh pemerintah Indonesia. Sistem Isyarat Bahasa Indonesia (SIBI) merupakan sarana komunikasi yang terlebih dahulu diakui oleh pemerintah Indonesia. Pengakuan dan pembakuan atas penggunaan SIBI secara resmi ditetapkan pada tahun 1994 melalui Keputusan Mendikbud RI Nomor 0161/U/1994.

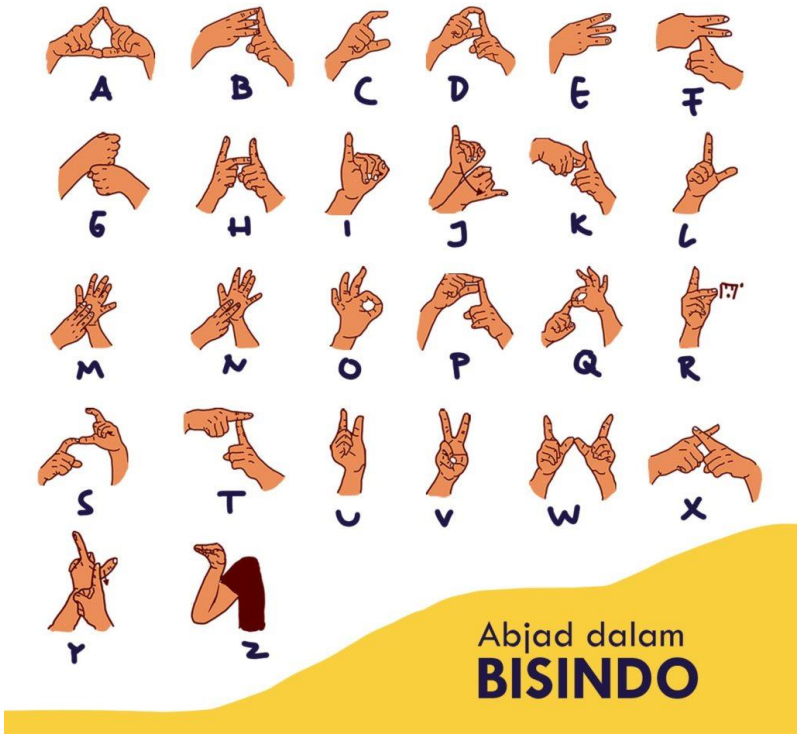
Disebarluaskan dan dibakukannya penggunaan SIBI sebagai sistem isyarat buatan yang bersifat nasional tidak sepenuhnya diterima oleh komunitas Tuli. Komunitas Tuli menilai bahwa keberadaan SIBI tidak merepresentasikan bahasa isyarat asli Indonesia, terdapat berbagai bentuk isyarat yang tidak sesuai dengan isyarat yang berkembang di komunitas Tuli. Salah satu isyarat yang banyak diterapkan pada kamus SIBI, yaitu isyarat yang terdapat pada sistem isyarat American Sign Language (ASL).

Permasalahan dan pertentangan atas penggunaan SIBI menjadi salah satu faktor yang memengaruhi munculnya penggunaan istilah Bisindo. Laura Lesmana Wijaya menjelaskan bahwa penggunaan istilah Bahasa Isyarat Indonesia atau disingkat Bisindo dimulai pada awal tahun 2000. Istilah tersebut muncul melalui pelaksanaan kongres yang dilaksanakan oleh Gerkatin. Penentuan istilah tersebut

digunakan untuk menunjuk pada bahasa isyarat alami yang berkembang di komunitas Tuli.

2.1.2 Isyarat Abjad Dalam Bisindo

Berikut adalah gambar gerakan bahasa isyarat dari abjad mulai A hingga Z yang diterjemahkan kedalam Bahasa Isyarat Indonesian (BISINDO).



Gambar 2.1 Gesture Bahasa Isyarat Abjad (BISINDO)

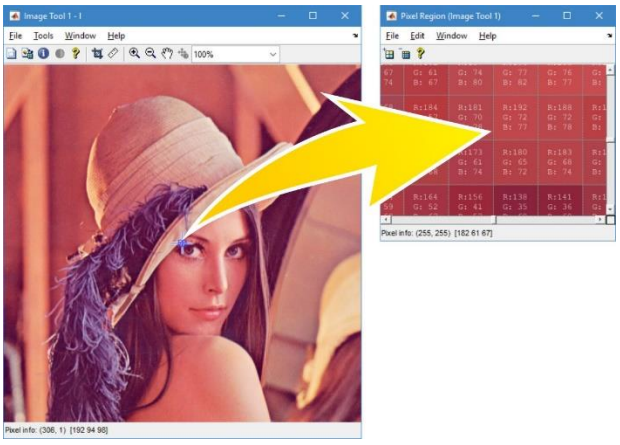
2.1.3 Isyarat Nomor Dalam Bisindo

Berikut adalah gambar gerakan bahasa isyarat dari nomor mulai 1 hingga 10 yang diterjemahkan kedalam Bahasa Isyarat Indonesian (BISINDO).

Gambar 2.2 Gesture Bahasa Isyarat Angka (BISINDO)

2.2 Citra Digital

Citra digital merupakan representasi dari fungsi intensitas cahaya dalam bentuk diskrit pada bidang dua dimensi. Citra tersusun oleh sekumpulan piksel (*picture element*) yang memiliki koordinat (x,y) dan amplitudo f(x,y). Koordinat (x,y) menunjukkan letak/posisi piksel dalam suatu citra, sedangkan amplitudo f(x,y) menunjukkan nilai intensitas warna citra. Representasi citra digital beserta piksel penyusunnya ditunjukkan pada Gambar berikut ini.



Gambar 2.3 Citra dan piksel penyusunnya

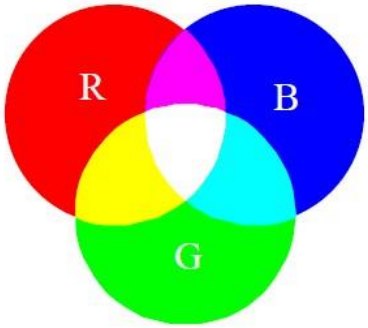
2.3 Jenis Citra Digital

Nilai suatu *pixel* memiliki nilai dalam rentang tertentu, dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbedabeda tergantung dari jenis warnanya. Berikut adalah jenis-jenis citra berdasarkan nilai *pixel*-nya, sebagai berikut:

2.3.1 Citra RGB

RGB (*Red, Green, Blue*) merupakan model perpaduan warna yang didasari pada tiga warna dasar yaitu *Red, Green, Blue* yang kemudian dikombinasikan bersama- sama untuk menghasilkan perpaduan warna.

Jangkauan warna RGB adalah mulai dari range 0 – 255. Penggunaan warna RGB untuk warna yang berbasiskan elektronik seperti *Camera,* Komputer, Televisi dan berbagai gadget lainnya.



Gambar 2.4 Warna RGB

Dari range warna yang dihasilkan oleh masing – masing R = 0 – 255, G = 0-255 dan B = 0 – 255 yang apabila digabungkan komposisinya menghasilkan warna yang baru. Sehingga warna yang mampu dihasilkan oleh ketiga kombinasi di atas adalah 256^3 sebanyak 16.777.216 kombinasi warna.

Y. Ming (1988) yang memperkenalkan sebuah metode Normalisasi RGB. Dimana warna dari sebuah pixel diproporsikan dengan semua nilai RGB. Konsep ini digunakan untuk mengatasi adanya perbedaan intentitas yang terdapat pada objek yang sama. Y. Ming merumuskan konsep Normalisasi RGB ini sebagai berikut :

$$r = \frac{R}{R+G+B} \quad (2.1)$$

$$g = \frac{G}{R+G+B} \quad (2.2)$$

$$b = \frac{B}{R+G+B} \quad (2.3)$$

Dan selanjutnya Michael J. Jones James M. Rehg pada tahun 1999 memperkenalkan warna model histogram untuk membedakan antara warna kulit manusia atau bukan warna kulit manusia.

2.3.2 Citra Grayscale

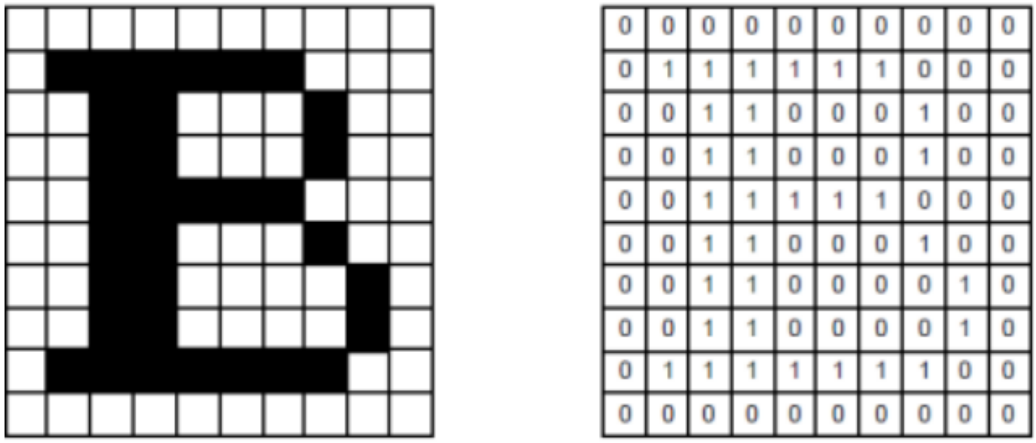
Citra grayscale mempunyai kemungkinan warna hitam untuk nilai minimal dan warna putih untuk nilai maksimal. Banyaknya warna tergantung pada jumlah bit yang disediakan di memori untuk menampung kebutuhan warna tersebut. Semakin besar jumlah bit warna yang disediakan di memori, maka semakin halus gradasi warna yang terbentuk.

Gambar 2.5 Warna Grayscale

Nilai warna untuk Grayscale (derajat keabuan) adalah nilai warna yang menunjukkan kehitaman dengan nilai 28 – 1 = 255 untuk image 8 bit.

2.3.3 Citra Biner

Citra biner adalah citra yang hanya mempunyai dua nilai derajat keabuan, hanya hitam dan putih. Piksel-piksel objek bernilai 1 dan piksel-piksel latar belakang bernilai 0. Pada waktu menampilkan gambar, 1 adalah putih sedangkan 0 adalah hitam (Erwin, Dedi, dan Ikhwan., 2015).



0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	0	0	0	0	1	0
0	0	1	1	0	0	0	0	1	0
0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

Gambar 2.6 Huruf “B” dan citra dalam biner

2.4 Pengolahan Citra Digital

Pengolahan Citra Digital adalah merupakan proses yang bertujuan untuk memanipulasi dan menganalisis citra dengan bantuan komputer. Baik citra yang berdimensi 2 atau citra 3 dimensi.

Kegiatan yang dilakukan dalam pengolahan citra dibagi menjadi dua bagian, yang pertama perbaikan kualitas terhadap sebuah citra agar mata manusia mampu menginterpretasi dengan baik. Perbaikan ini termasuk Image Enhancement agar mendapatkan citra yang lebih baik dari citra sebelum dilakukan pengolahan. Dan Kedua pengolahan citra bekerja untuk mendapatkan dan mengolah informasi yang terdapat pada suatu citra untuk keperluan pengenalan objek secara otomatis. Sebagai contoh aplikatifnya, image detection, skin detection, Pengenalan Pola dan masih banyak lainnya.

2.4.1 Preprocessing

Preprocessing adalah proses awal dilakukannya perbaikan suatu citra untuk menghilangkan noise. *Preprocessing* merupakan suatu proses untuk menghilangkan bagian-bagian yang tidak diperlukan pada gambar input untuk proses selanjutnya. Beberapa proses yang dapat dilakukan pada tahap *preprocessing* antara lain, proses binerisasi, segmentasi, dan normalisasi.

2.4.2 Segmentasi citra

Segmentasi citra adalah proses pengolahan citra yang bertujuan memisahkan wilayah (*region*) objek dengan wilayah latar belakang agar objek mudah dianalisis dalam rangka mengenali objek yang banyak melibatkan persepsi visual (Destyningtias, 2010).

2.4.3 Normalisasi

Normalisasi adalah proses mengubah ukuran citra, baik menambah atau mengurangi, menjadi ukuran yang ditentukan tanpa menghilangkan informasi penting dari citra tersebut. Dengan adanya proses normalisasi maka ukuran semua citra yang akan diproses menjadi seragam.

2.5 Visi komputer

Visi komputer (*Computer Vision*) adalah salah satu teknologi yang paling banyak dipakai pada zaman ini. Teknologi visi komputer ini merupakan salah satu bidang dari teknologi *Artificial Intelligence*. Visi komputer juga merupakan dan kumpulan dari metode-metode untuk mendapatkan, memproses, menganalisis suatu gambar atau dalam arti lain visi komputer, merupakan kumpulan metode-metode yang digunakan untuk menghasilkan angka-angka atau simbol-simbol yang didapat dari gambar yang diambil dari dunia nyata agar komputer dapat mengerti apa makna dari gambar tersebut. Inti dari teknologi visi komputer adalah untuk menduplikasi kemampuan penglihatan manusia kedalam benda elektronik sehingga benda elektronik dapat memahami dan mengerti arti dari gambar yang dimasukkan (Milan Sonka, Vaclav Hlavac and Roger Boyle - 2008).

2.6 Gradient Descent

Gradient Descent merupakan salah satu algoritma yang paling populer dalam melakukan optimasi pada model jaringan syaraf tiruan *(Artificial neural network* / ANN*)*. Algoritma ini adalah cara yang paling sering dipakai dalam berbagai macam model pembelajaran. Ketika akan melatih sebuah model, akan dibutuhkan sebuah

loss function yang dapat memungkinkan peneliti untuk mengukur kualitas dari setiap bobot atau parameter tertentu. Tujuan dari pengoptimalan ini yaitu untuk menentukan parameter manakah yang mampu meminimalkan *loss function* (Ruder, 2017).

Gradient Descent bekerja dengan meminimalkan fungsi (θ) yang mempunyai parameter θ dengan memperbarui parameter ke suatu arah yang menurun. *Gradient descent* mempunyai *learning rate* (η) yang digunakan untuk menentukan langkah yang akan diambil untuk mencapai pada titik minimum. Hal ini, dapat digambarkan bahwa suatu objek akan seperti menuruni sebuah bukit dengan langkah tersebut sehingga mencapai pada bagian lembah (titik minimum).

Stochastic Gradient Descent (SGD) merupakan metode *gradient descent* yang melakukan *update parameter* untuk setiap data pelatihan x(i) dan label y(i) dan mempunyai persamaan dasar berikut :

$$\theta = \theta - \eta . \nabla_\theta J(\theta; x^i; y^i) \quad 2.4$$

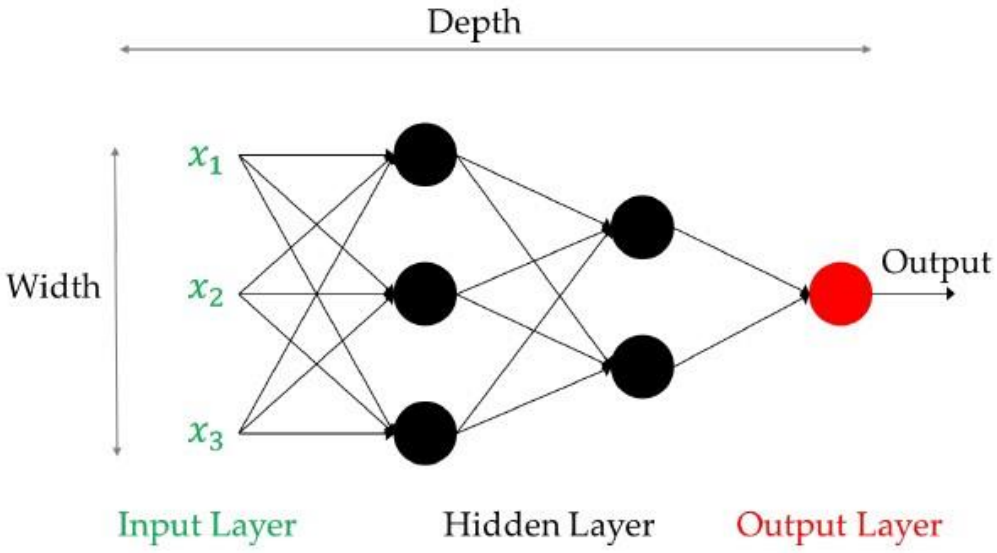
SGD seringkali melakukan *update*/pembaruan dengan varians yang tinggi, sehingga menyebabkan fungsi objektif meningkat secara tidak beraturan. Di satu sisi, hal ini dapat membuat *loss function* melompat ke titik minimal yang baru dan mempunyai potensi untuk melompat ke nilai minimum yang tidak pasti. Namun, hal ini dapat dicegah dengan mengurangi nilai *learning rate*, dan hasil SGD akan menuruni *loss function* ke titik minimum dengan optimal.

2.7 Artificial Neural Network

Artificial neural network adalah salah satu algoritma *supervised learning* yang populer dan bisa juga digunakan untuk *semi-supervised* atau *unsupervised learning* (Amir Atiya, 1994). Walaupun tujuan awalnya adalah untuk mensimulasikan jaringan saraf biologis, jaringan tiruan ini sebenenarnya simulasi yang terlalu disederhanakan, artinya simulasi yang dilakukan tidak mampu menggambarkan kompleksitas jaringan biologis manusia.

Artificial Neural Network (selanjutnya disingkat ANN), menghasilkan model yang sulit dibaca dan dimengerti oleh manusia karena memiliki banyak layer (kecuali *single perceptron*) dan sifat **non-linear** (merujuk pada fungsi aktivasi).

Pada bidang riset ini, ANN disebut agnostik - kita percaya, tetapi sulit membuktikan kenapa konfigurasi parameter yang dihasilkan *training* bisa benar. Konsep matematis ANN itu sendiri cukup *solid*, tetapi *interpretability* (tingkat pemahaman dan 'insight') model rendah menyebabkan kita tidak dapat menganalisa proses inferensi(penyimpulan) yang terjadi pada model ANN. Secara matematis, ANN ibarat sebuah graf. ANN memiliki neuron/*node (vertex)*, dan sinapsis *(edge)*. Karena memiliki struktur seperti graf, operasi pada ANN mudah dijelaskan dalam notasi aljabar linear. Sebagai gambaran, ANN berbentuk seperti Gambar 2.7 (*deep neural network*, salah satu varian arsitektur). *Depth* (kedalaman) ANN mengacu pada jumlah layer. Sementara width (lebar) ANN mengacu pada jumlah unit pada *layer*.

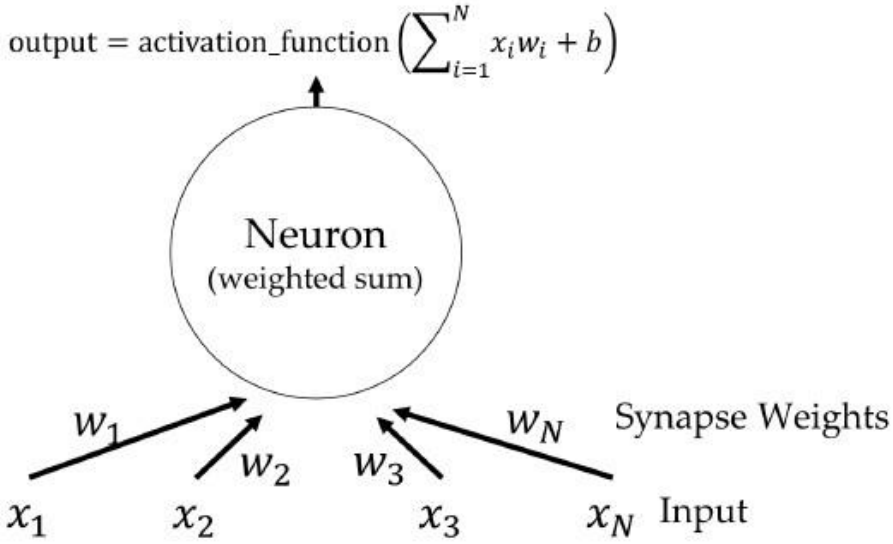


Gambar 2.7 Deep Neural Network

2.7.1 Single Perceptron

Bentuk terkecil (minimal) sebuah ANN adalah *single perceptron* yang hanya terdiri dari sebuah neuron. Sebuah neuron diilustrasikan pada Gambar 2.8. Secara matematis, terdapat *feature vector* x yang menjadi *input* bagi neuron tersebut (*feature vector* merepresentasikan suatu *data point*, *event* atau *instans*). Neuron akan memproses *input* x melalui perhitungan jumlah perkalian antara nilai *input* dan *synapse weight*, yang dilewatkan pada fungsi non linear. Pada *training*, yang

dioptimasi adalah nilai *synapse weight (learning parameter)*. Selain itu, terdapat juga bias *b* sebagai kontrol tambahan (materi *steepest gradient descent*). Output dari neuron adalah hasil fungsi aktivasi dari perhitungan jumlah perkalian antara nilai *input* dan *synapse weight*. Ada beberapa macam fungsi aktivasi, misal ***step function, sign function, rectifier*** dan ***sigmoid function***. Bila diplot menjadi grafik, fungsi ini memberikan bentuk seperti huruf S.



Gambar 2.8 Single Perceptron

$$\sigma(u) = \frac{1}{1+e^{-u}} \quad (2.5)$$

Perhatikan kembali, Gambar 2.8 sesungguhnya adalah operasi aljabar linear. Single perceptron dapat dituliskan kembali sebagai berikut :

$$o = f(x.w + b) \quad (2.6)$$

Dimana o adalah *output* dan f adalah fungsi non-linear yang dapat diturunkan secara matematis (*differentiable non-linear function* - selanjutnya disebut "fungsi non linear" saja.). Bentuk ini tidak lain dan tidak bukan adalah persamaan model linear yang ditransformasi dengan fungsi non-linear. Secara filosofis, ANN bekerja mirip dengan model linear, yaitu mencari *decision boudary*. Apabila beberapa model non-linear ini digabungkan, maka kemampuannya akan menjadi lebih hebat. Yang menjadikan ANN "spesial" adalah penggunaan fungsi non-linear.

Untuk melakukan pembelajaran *single perceptron*, *training* dilakukan menggunakan ***perceptron training rule***. Prosesnya sebagai berikut:

1. Inisiasi nilai *synapse weights*, bisa *random* ataupun dengan aturan tertentu.
2. Lewatkan input pada neuron, kemudian kita akan mendapatkan nilai *output*. Kegiatan ini disebut ***feedforward***.
3. Nilai *output (actual output)* tersebut dibandingkan dengan *desired output*.
4. Apabila nilai output sesuai dengan *desired output*, tidak perlu mengubah apa-apa.
5. Apabila nilai *output* tidak sesuai dengan *desired output*, hitung nilai *error (loss)* kemudian lakukan perubahan terhadap *learning parameter (synapse weight)*.
6. Ulangi langkah-langkah ini sampai tidak ada perubahan nilai *error*, nilai *error* kurang dari sama dengan suatu *threshold* (biasanya mendekati 0), atau sudah mengulangi proses latihan sebanyak **T** kali (*threshold*).

Error function diberikan pada persamaan 2.7 (dapat diganti dengan absolute value) dan perubahan *synapse weight* diberikan pada persamaan 2.8, dimana y melambangkan d*esired output*, $\mathrm{o} = \mathrm{f}(\mathrm{x}\,.\,\mathrm{w} + \mathrm{b})$ melambangkan actual output untuk x sebagai input. μ disebut sebagai learning rate.

$$E(w) = (y - 0)^2 \qquad 2.7$$

$$\Delta w_i = \mu(y - 0)x_1 \qquad 2.8$$

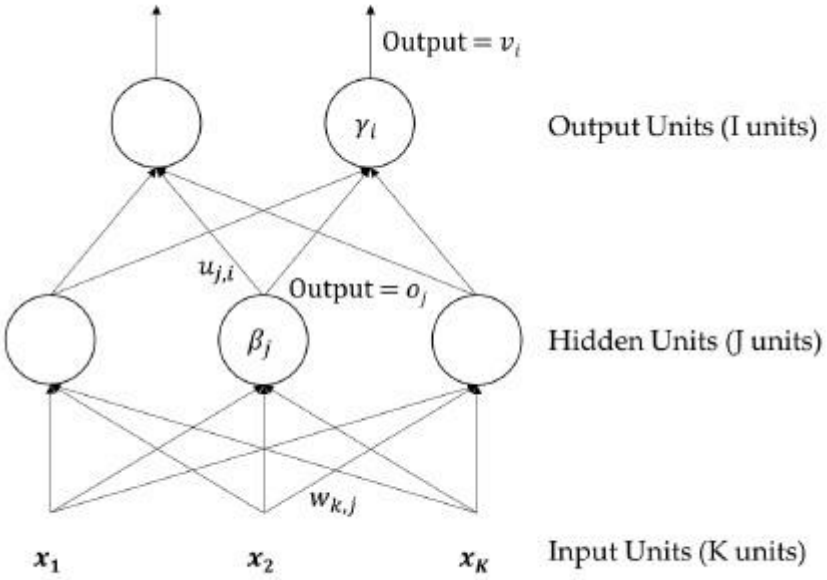
Hasil akhir pembelajaran *(learning)* adalah konfigurasi *synapse weight*. Saat klasifikasi, kita melewatkan *input* baru pada jaringan yang telah dibangun, kemudian tinggal mengambil hasilnya. Pada contoh kali ini, seolah-olah *single perceptron* hanya dapat digunakan untuk melakukan *binary classification* (hanya ada dua kelas, nilai 0 dan 1). Untuk multi-class classification, kita dapat menerapkan berbagai strategi, misal thresholding, i.e., nilai output 0 - 0.2 mengacu pada kelas pertama, 0.2 - 0.4 untuk kelas kedua, dst.

2.7.2 Multilayer Perceptron

Multilayer perceptron (MLP) yang juga dikenal sebagai *feedforward neural network* secara literal memiliki beberapa *layers*. Pada lecture note ini, secara umum ada tiga *layers*: *input*, *hidden*, dan *output layer*. *Input layer* menerima *input* (tanpa melakukan operasi apapun), kemudian nilai *input* (tanpa dilewatkan ke fungsi aktivasi) diberikan ke *hidden units*. Pada *hidden units*, *input* diproses dan dilakukan perhitungan hasil fungsi aktivasi untuk tiap-tiap neuron, lalu hasilnya diberikan ke

layer berikutnya. *Output* dari *input layer* akan diterima sebagai *input* bagi *hidden layer*. Begitupula seterusnya *hidden layer* akan mengirimkan hasilnya untuk *output layer*.

Kegiatan ini dinamakan *feed forward*. Hal serupa berlaku untuk *artificial neural network* dengan lebih dari tiga *layers*. Parameter neuron dapat dioptimisasi menggunakan metode *gradient-based optimization*. Perlu diperhatikan, MLP adalah gabungan dari banyak fungsi *non-linear*. Gabungan banyak fungsi *non-linear* ini lebih hebat dibanding single perceptron. Masing-masing neuron terkoneksi dengan semua neuron pada *layer* berikutnya. Konfigurasi ini disebut sebagai ***fully connected***. MLP pada umumnya menggunakan konfigurasi *fully connected*.



Gambar 2.9 Multilayer Perceptron

$$o_j = \sigma\left(\sum_{k=1}^{k} x_k w_{k,j} + \beta_j\right) \quad 2.9$$

$$v_i = \sigma\left(\sum_{j=1}^{j} o_j u_{j,i} + \gamma_i\right) = \sigma\left(\sum_{j=1}^{j} \sigma\left(\sum_{k=1}^{k} x_k w_{k,j} + \beta_j\right) u_{j,i} + \gamma_i\right) \quad 2.10$$

Perhatikan persamaan 2.9 dan 2.10 untuk menghitung *output* pada *layer* yang berbeda. u,w adalah *learning parameters*. β,γ melambangkan *noise* atau *bias*. K adalah banyaknya *input units* dan J adalah banyaknya *hidden units*.

Persamaan 2.10 dapat disederhanakan penulisannya sebagai persamaan 2.11. Persamaan 2.11 terlihat relatif lebih "elegan". Sehingga ANN dapat direpresentasikan dengan notasi operasi aljabar.

$$V = \sigma(oU + \gamma) = \sigma(\sigma(xU + \beta)U + \gamma) \quad 2.11$$

Untuk melatih MLP, algoritma yang umumnya digunakan adalah **backpropagation**. Arti kata *backpropagation* sulit untuk diterjemahkan ke dalam bahasa Indonesia. Peneliti memperbaharui parameter (*synapse weights*) secara bertahap (dari *output* ke *input* layer, karena itu disebut *backpropagation*) berdasarkan *error/loss* (*output* dibandingkan dengan *desired output*). Intinya adalah mengkoreksi *synapse weight* dari output layer ke *hidden layer*, kemudian *error* tersebut dipropagasi ke layer sebelum-sebelumnya. Artinya, perubahan *synapse weight* pada suatu layer dipengaruhi oleh perubahan synapse weight pada layer setelahnya. Backpropagation tidak lain dan tidak bukan adalah metode *gradient-based optimization* yang diterapkan pada ANN.

Pertama-tama diberikan pasangan *input* (x) dan *desired output* (y) sebagai *training data*. Untuk meminimalkan *loss*, algoritma *backpropagation* menggunakan prinsip *gradient descent*. Cara menurunkan *backpropagation* menggunakan teknik *gradient descent*, yaitu menghitung *loss* ANN pada Gambar 2.9 **yang menggunakan fungsi aktivasi sigmoid**.

Error, untuk MLP diberikan oleh persamaan 2.12 (untuk satu data point), dimana I adalah banyaknya *output unit* dan θ adalah kumpulan *weight matrices* (semua *parameter* pada MLP). Kami inggatkan kembali perhitungan *error* bisa juga menggunakan nilai absolut.

$$E(\theta) = \frac{1}{2}\sum_{i=1}^{I}(y_i - v_i)^2 \quad 2.12$$

Proses penurunan untuk melatih MLP, *error/loss* diturunkan terhadap tiap learning parameter. Diferensial $u_{j,i}$ diberikan oleh turunan sigmoid function.

$$\frac{\delta E(\theta)}{\delta u_{j,i}} = (y_i - v_i)\frac{\delta v_i}{\delta u_{j,i}}$$

$$= (y_i - v_i)v_i(1 - v_i)o_j \quad 2.13$$

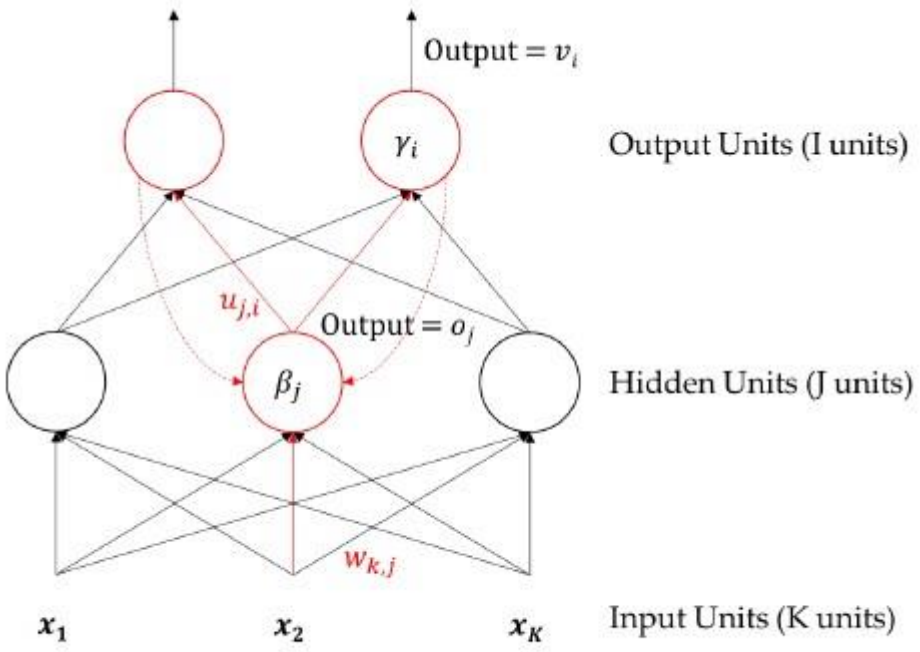
Diferensial $w_{k,j}$ diberikan oleh turunan *sigmoid function*

$$\frac{\delta \mathrm{E}(\theta)}{\delta w_{\mathrm{k,j}}} = \sum_{i=1}^{I}(y_i - v_i)\frac{\delta v_i}{\delta w_{\mathrm{k,j}}}$$

$$= \sum_{i=1}^{I}(y_i - v_i)\frac{\delta v_i}{\delta o_j}\frac{\delta o_j}{\delta w_{\mathrm{k,j}}}$$

$$= \sum_{i=1}^{I}(y_i - v_i)\,(v_i(1 - v_i)u_{j,i})(o_j(1 - o_j)x_k) \qquad 2.14$$

Perhatikan, *diferensial* $w_{k,j}$ memiliki $\sum$ sementara $u_{j,i}$ tidak ada. Hal ini disebabkan karena $u_{j,I}$ hanya berkorespondensi dengan satu *output* neuron. Sementara $w_{k,j}$ berkorespondensi dengan banyak *output* neuron. Dengan kata lain, nilai $w_{k,j}$ mempengaruhi hasil operasi yang terjadi pada banyak *output* neuron, sehingga banyak neuron mempropagasi error kembali ke $w_{k,j}$. Ilustrasi diberikan pada Gambar 2.10.



Gambar 2.10 Multilayer Perceptron 2

Metode penurunan serupa dapat juga digunakan untuk menentukan perubahan β dan . Jadi proses *backpropagation* untuk kasus Gambar 2.9 dapat diberikan seperti pada Gambar 2.11 dimana η adalah *learning rate*. Untuk *artificial neural network* dengan lebih dari 3 layers juga bisa menurunkan persamaannya. Secara umum, proses melatih ANN (apapun variasi arsitekturnya) mengikuti *framework perceptron training rule*.

(2) Hidden to Output

$$v_i = \sigma\left(\sum_{j=1}^{J} o_j u_{j,i} + \gamma_i\right)$$

(3) Output to Hidden

$$\delta_i = (y_i - v_i)v_i(1 - v_i)$$
$$\Delta u_{j,i} = -\eta(t)\delta_i o_j$$
$$\Delta\gamma_i = -\eta(t)\delta_i$$

(1) Input to Hidden Layer

$$o_j = \sigma\left(\sum_{k=1}^{K} x_k w_{k,j} + \beta_j\right)$$

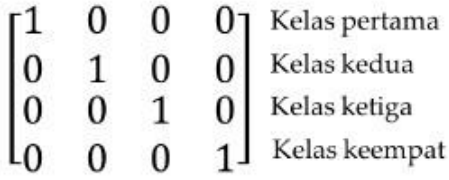
(4) Hidden to Input

$$\varphi_j = \sum_{i=1}^{I} \delta_i u_{j,i} o_j(1 - o_j)$$
$$\Delta w_{k,j} = -\eta(t)\varphi_j x_k$$
$$\Delta\beta_j = -\eta(t)\varphi_j$$

Gambar 2.11 Proses latihan menggunakan backpropagation

2.7.3 Multi-class Classification

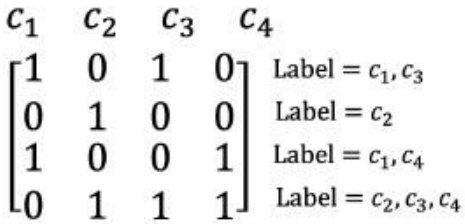
Multilayer perceptron dapat memiliki *output unit* berjumlah lebih dari satu. Seumpama mempunyai empat kelas, dengan demikian peneliti dapat merepresentasikan keempat kelas tersebut sebagai empat *output units*. Kelas pertama direpresentasikan dengan *unit* pertama, kelas kedua dengan *unit* kedua, dst. Untuk C kelas, kita dapat merepresentasikannya dengan C *output units*. Kita dapat merepresentasikan data harus dimasukkan ke kelas mana menggunakan *sparse vector*, yaitu bernilai 0 atau 1. Elemen ke-i bernilai 1 apabila data masuk ke kelas c_i, sementara nilai elemen lainnya adalah 0 (ilustrasi pada Gambar 2.12).Output ANN dilewatkan pada suatu fungsi softmax yang melambangkan probabilitas *class-assignment*, i.e., kita ingin output agar semirip mungkin dengan *sparse vector* (*desired output*). Pada kasus ini, *output* ANN adalah sebuah distribusi yang melambangkan input di-*assign* ke kelas tertentu. Cross entropy cocok digunakan sebagai utility function ketika output berbentuk distribusi.



Gambar 2.12 Ilustrasi representasi desired output pada multi-class classification

2.7.4 Multi-label Classification

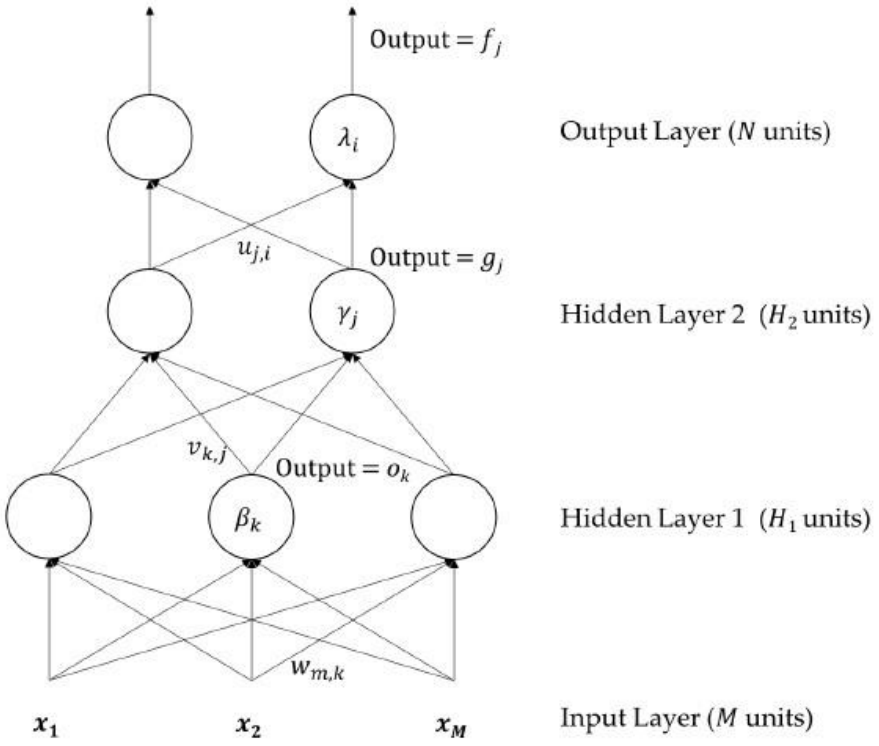
Seperti halnya *multi-class classification*, kita dapat menggunakan sejumlah C neuron untuk merepresentasikan C kelas pada *multi-label classification* Perbedaan *multi-class* dan *multilabel* terletak pada cara interpretasi *output* dan evaluasi *output*. Pada umumnya, *layer* terakhir diaktivasi dengan fungsi sigmoid, dimana tiap neuron n_i merepresentasikan probabilitas suatu dapat diklasifikasikan sebagai kelas c_i atau tidak. *Cross entropy* juga cocok untuk mengevaluasi (dan melatih) *multi-label classification*.



Gambar 2.13 Ilustrasi representasi desired output pada multi-label classification

2.7.5 Deep Neural Network

Deep Neural Network (DNN) adalah *artificial neural network* yang memiliki banyak *layer*. Pada umumnya, deep neural network memiliki lebih dari 3 *layers (input layer,* N *hidden layers, output layer*), dengan kata lain adalah MLP dengan lebih banyak *layer*. Karena ada relatif banyak *layer*, disebutlah *deep*. Proses pembelajaran pada DNN disebut sebagai *deep learning*11. Jaringan neural *network* pada DNN disebut *deep neural network*. Perhatikan Gambar 2.14 yang memiliki 4 layers.



Gambar 2.14 Deep Neural Network

Cara menghitung *final output* sama seperti MLP, diberikan pada persamaan berikut dimana adalah *noise* atau *bias*.

$$f_i = \left(\sum_{j=1}^{H_2} u_{j,i}\, \sigma\left(\sum_{k=1}^{H_1} u_{k,j}\, \sigma\left(\sum_{m=1}^{M} x_m\, w_{m,k} + \beta_k\right) + \gamma_j\right) + \lambda_i\right) \quad 2.15$$

Cara melatih *deep neural network*, salah satunya dapat menggunakan backpropagation. Hanya perlu menurunkan rumusnya saja. Hasil proses penurunan dapat dilihat pada Gambar 2.15.

(3) Hidden 2 to Output

$$f_i = \sigma\left(\sum_{j=1}^{H_2} g_j u_{j,i} + \lambda_i\right)$$

(4) Output to Hidden 2

$$\delta_i = (y_i - f_i) f_i (1 - f_i)$$
$$\Delta u_{j,i} = -\eta(t)\delta_i g_j$$
$$\Delta \lambda_i = -\eta(t)\delta_i$$

(2) Hidden 1 to Hidden 2

$$g_j = \sigma\left(\sum_{k=1}^{H_1} o_k v_{k,j} + \gamma_j\right)$$

(5) Hidden 2 to Hidden 1

$$\varphi_j = \sum_{i=1}^{N} \delta_i u_{j,i} g_j (1 - g_j)$$
$$\Delta v_{k,j} = -\eta(t)\varphi_j o_k$$
$$\Delta \gamma_j = -\eta(t)\varphi_j$$

(1) Input to Hidden Layer

$$o_k = \sigma\left(\sum_{m=1}^{M} x_m w_{m,k} + \beta_k\right)$$

(6) Hidden 1 to Input

$$\mu_k = \sum_{j=1}^{H_2} \varphi_j v_{k,j} o_k (1 - o_k)$$
$$\Delta w_{m,k} = -\eta(t)\mu_k x_m$$
$$\Delta \beta_k = -\eta(t)\beta_k$$

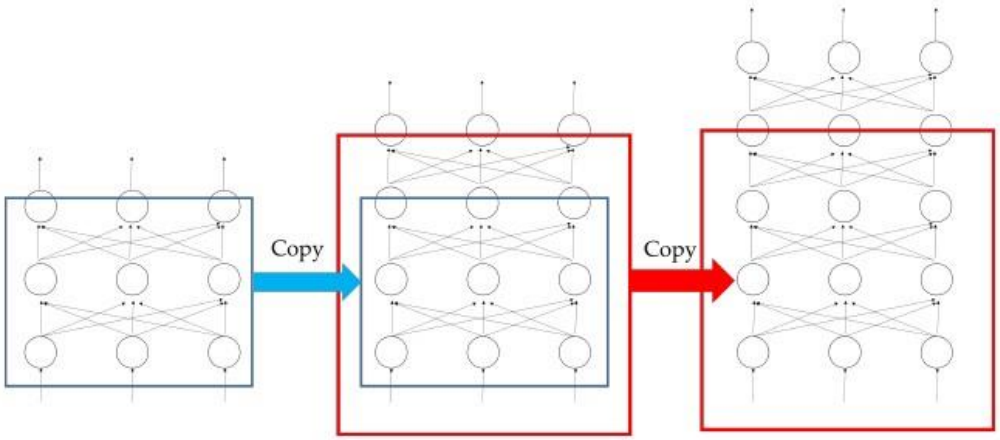
Gambar 2.15 Proses latihan DNN menggunakan backpropagation

Deep network terdiri dari banyak *layer* dan *synapse weight*, karenanya estimasi parameter susah dilakukan. Arti filosofisnya adalah susah/lama untuk menentukan relasi antara *input* dan *output*. Walaupun deep learning sepertinya kompleks, tetapi entah kenapa dapat bekerja dengan baik untuk permasalahan praktis. *Deep learning* dapat menemukan relasi "tersembunyi" antara input dan output, yang tidak dapat diselesaikan menggunakan *multilayer perceptron* (3 layers).

Banyak orang percaya *deep neural network* lebih baik dibanding *neural network* yang lebar tapi sedikit layer, karena terjadi lebih banyak transformasi. Maksud lebih banyak transformasi adalah kemampuan untuk merubah input menjadi suatu representasi (tiap hidden layer dapat dianggap sebagai salah satu bentuk representasi input) dengan langkah hierarchical. Seperti contoh permasalahan XOR, permasalahan *non-linearly separable* pun dapat diselesaikan apabila kita dapat mentransformasi data (representasi data) ke dalam bentuk *linearly separable* pada ruang yang berbeda. Keuntungan utama deep learning adalah mampu merubah data dari *non-linearly separable* menjadi *linearly separable* melalui serangkaian transformasi (*hidden layers*). Selain itu, deep

learning juga mampu mencari *decision boundary* yang berbentuk *non-linear*, serta mengsimulasikan interaksi *non-linear* antar fitur.

Karena memiliki banyak parameter, proses latihan ANN pada umumnya lambat. Ada beberapa strategi untuk mempercepat pembelajaran menggunakan deep learning, misalnya: regularisasi, *successive learning*, dan penggunaan *autoencoder*. Sebagai contoh, arti *successive learning* adalah jaringan yang dibangun secara bertahap. Misal kita latih ANN dengan 3 *layers*, kemudian kita lanjutkan 3 *layers* tersebut menjadi 4 *layers*, lalu kita latih lagi menjadi 5 *layers*, dst (mulai dari hal kecil). Ilustrasinya dapat dilihat pada Gambar 2.16. Menggunakan *deep learning* harus hati-hati karena pembelajaran cenderung *divergen* (artinya, *minimum squared error* belum tentu semakin rendah seiring berjalannya waktu - *swing* relatif sering).



Gambar 2.16 Contoh successive learning

2.7.6 Regularization and Dropout

Pada model linear. Model harus mampu mengeneralisasi dengan baik (kinerja baik pada *training* data dan *unseen examples*). Peneliti dapat menambahkan fungsi regularisasi untuk mengontrol kompleksitas ANN. Regularisasi pada ANN cukup *straightforward* seperti regularisasi pada model linear.

Selain itu, agar ANN tidak “bergantung” pada satu atau beberapa *synapse weights* saja, peneliti dapat menggunakan *dropout*. **Dropout** berarti me-nol-kan nilai *synapse weights* dengan nilai rate tertentu. Misalkan me-nol-kan nilai 30%

synapse weights (dropout rate= 0,3) secara random. Hal ini dapat dicapai dengan teknik *masking*, yaitu mengalikan *synapse weights* dengan suatu *mask*.

Ingat kembali ANN secara umum, persamaan 2.16 dimana W adalah *synapse weights*, x adalah input (dalam pembahasan saat ini, dapat merepresentasikan *hidden state* pada suatu *layer*), b adalah *bias* dan f adalah fungsi aktivasi (*non-linear*). Peneliti bisa buat suatu *mask* untuk *synapse weights* seperti pada persamaan 2.17, dimana p adalah vektor dan pi = [0,1] merepresentasikan *synapse weight* diikutsertakan atau tidak. r% (*dropout rate*) elemen vektor p bernilai 0. Biasanya p diambil dari *bernoulli distribution*. Kemudian, saat *feed forward*, mengganti *synapse weights* menggunakan *mask* seperti pada persamaan 2.18. Saat menghitung *backpropagation*, turunan fungsi juga mengikut sertakan *mask* (*gradient di-mask*). Teknik *regularization* dan *dropout* sudah menjadi metode yang cukup "standar" dan diaplikasikan pada berbagai macam arsitektur.

$$o = f(x \,.\, W + b) \tag{2.16}$$

$$w` = p \,.\, W \tag{2.17}$$

$$o = f(x \,.\, W` + b) \tag{2.18}$$

2.8 Convolutional Neural Network

Kemampuan utama *convolutional neural network* (CNN) adalah arsitektur yang mampu mengenali informasi prediktif suatu objek (gambar, teks, potongan suara, dsb) walaupun objek tersebut dapat diposisikan dimana saja pada input. Kontribusi CNN adalah pada *convolution* dan *pooling layer*. *Convolution* bekerja dengan prinsip *sliding window* dan *weight sharing* (mengurangi kompleksitas perhitungan). *Pooling layer* berguna untuk merangkum informasi informatif yang dihasilkan oleh suatu *convolution* (mengurangi dimensi). Pada ujung akhir CNN, kita lewatkan satu vektor hasil beberapa operasi *convolution* dan *pooling* pada *multilayer perceptron (feed-forward neural network)*, dikenal juga sebagai *fully connected layer*, untuk melakukan suatu pekerjaan, e.g., klasifikasi. Pada umumnya CNN tidak berdiri sendiri, dalam artian CNN biasanya digunakan (dikombinasikan) pada arsitektur yang lebih besar.

2.8.1 Activation Layer

Activation layer pada umumnya merupakan sebuah fungsi aktivasi di bagian atas sebuah *output layer*. Tujuan utama dari penggunaan fungsi aktivasi ini adalah untuk membentuk *non-linearity* (ketidaklinieran) pada *neural network*. Tanpa adanya fungsi aktivasi, *neural network* hanya akan melakukan transformasi linier dari input ke output. Secara matematis, fungsi aktivasi ditulis sebagai berikut.

$$x^l = f(x^{l-1}) \quad 2.19$$

Beberapa jenis fungsi aktivasi yang umum digunakan adalah RelU *(Rectified-Linear Unit)*, Sigmoid dan TanH *(Hyperbolic Tangent)*.

Sigmoid

Fungsi Sigmoid akan memetakan input ke interval [0, 1]. Fungsi sigmoid didefinisikan sebagai berikut.

$$\sigma(u) = \frac{1}{1+exp^{-u}} \quad 2.20$$

TanH

Fungsi TanH atau *hyperbolic tangent function* merupakan fungsi transformasi linier dari sigmoid ke interval [-1, 1]. Fungsi TanH dapat didefinisikan sebagai berikut.

$$tanh\, x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad 2.21$$

RelU

Permasalahan yang ada pada fungsi sigmoid dan tanh yaitu output yang mudah jenuh ke angka 0 atau 1 pada sigmoid dan +1 dan -1 pada tanh. Output yang jenuh akan membuat hilangnya gradien dan menyebabkan turunnya kecepatan training dan memungkinkan untuk menjebak model pada *area local minimum*.

RelU diperkenalkan pada tahun 2010 untuk meningkatkan kecepatan *konvergensi* dari *training neural networks* (V. Nair, 2010). RelU memiliki kelebihan tidak akan jenuh pada suatu nilai dan memungkinkan untuk melakukan *training neural network* berukuran besar layaknya CNN. Relu didefinisikan sebagai berikut.

$$f(x) = \max(0, x) \quad 2.22$$

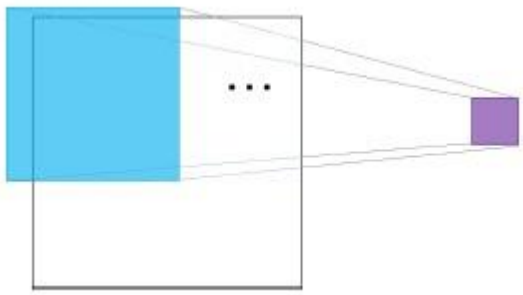
Pada CNN, layer output pada umumnya merupakan *feature maps* dimensi dua. Pada persamaan 2.22 , x adalah matrix dan max adalah elemen *twise* yang diterapkan pada setiap elemen matriks. Varian lain dari RelU yaitu LeakyRelU memungkinkan adanya nilai yang kecil, bukan nol meski unit sedang tidak dalam keadaan aktif. Leaky RelU didefinisikan sebagai berikut.

$$\begin{cases} x, x > 0 \\ 0.01x \quad selain x > 0 \end{cases} \tag{2.23}$$

2.8.2 Convolution

Motivasi CNN adalah untuk mampu mengenali aspek yang informatif pada regional tertentu (lokal). Dibanding meng-*copy* mesin pembelajaran beberapa kali untuk mengenali objek pada banyak regional, ide lebih baik adalah untuk menggunakan *sliding window*. Setiap operasi pada window bertujuan untuk mencari aspek lokal yang paling informatif.

Ilustrasi diberikan oleh Gambar 2.17. Warna biru merepresentasikan satu *window*, kemudian kotak ungu merepresentasikan aspek lokal paling informatif (disebut *filter*) yang dikenali oleh *window*. Dengan kata lain, kita mentransformasi suatu window menjadi suatu nilai numerik (*filter*). Kita juga dapat mentransformasi suatu *window* (regional) menjadi d nilai numerik (d channels, setiap elemen berkorespondensi pada suatu *filter*). *Window* ini kemudian digeser-geser sebanyak T kali, sehingga akhirnya kita mendapatkan vektor dengan panjang d x T. Keseluruhan operasi ini disebut sebagai *convolution*.

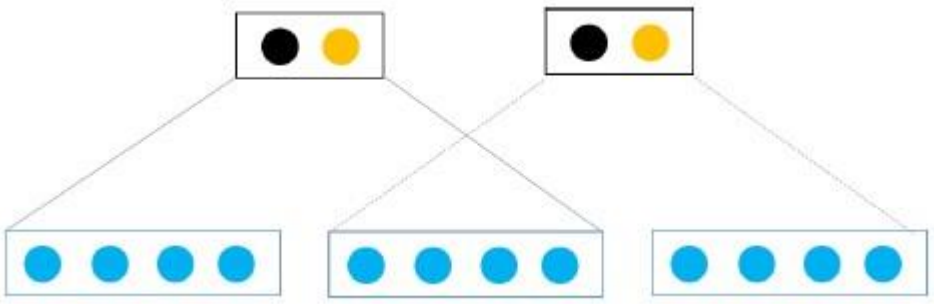


Gambar 2.17 Sliding window

Pada Gambar 2.18 menggunakan *window* selebar 2, satu *window* mencakup 2 data; i.e., window$_1$ = (X_1, X_2), window$_2$ = (X_2; X_3), … . Untuk suatu input X. Kita juga dapat mempergunakan *stride* sebesar s, yaitu seberapa banyak data yang

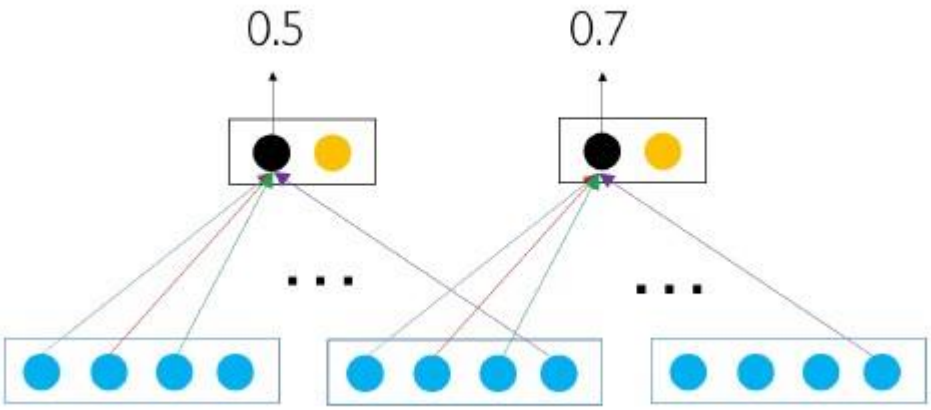
digeser untuk *window* baru. Contoh yang diberikan memiliki *stride* sebesar satu. Apabila memiliki *stride*= 2, maka akan menggeser sebanyak 2 data setiap langkah, i.e., window$_1$ = (X_1, X_2), window$_2$ = (X_3, X_4), … .

Berikut contoh dalam bentuk 1-D pada Pada Gambar 2.18. Warna biru merepresentasikan *feature vector* (*regional*) untuk suatu input (e.g., *regional* pada suatu gambar, kata pada kalimat, dsb). Pada contoh ini, setiap 2 input ditransformasi menjadi vektor berdimensi 2 (2-*channels*); menghasilkan vektor berdimensi 4 (2 *window* x 2).



Gambar 2.18 1D Convolution

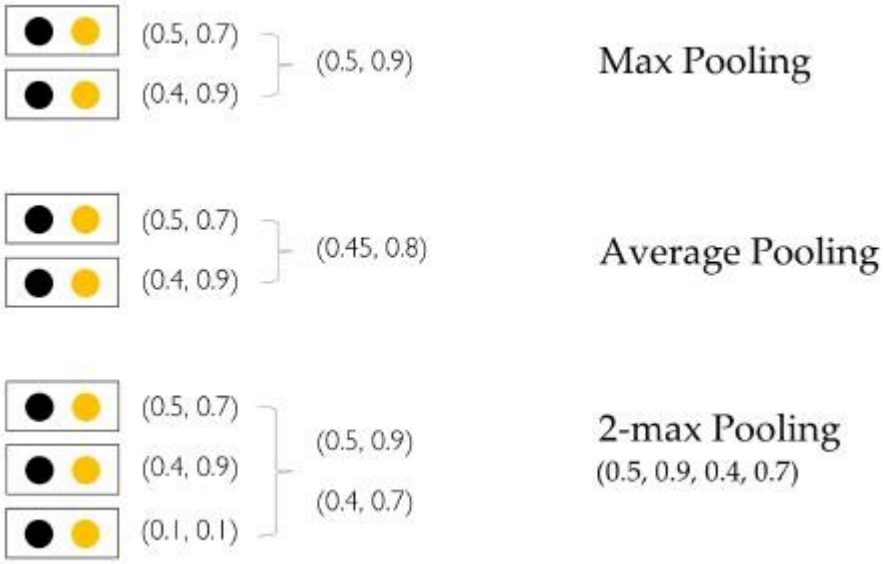
Selain *sliding window* dan *filter*, *convolutional layer* juga mengadopsi prinsip *weight sharing*. Artinya, *synapse weights* untuk suatu filter adalah sama walau filter tersebut dipergunakan untuk berbagai *window*. Sebagai ilustrasi, perhatikan Gambar 2.19, warna yang sama pada *synapse weights* menunjukkan *synapse weights* bersangkutan memiliki nilai (*weight*) yang sama. Tidak hanya pada *filter* hitam, hal serupa juga terjadi pada *filter* berwarna oranye (i.e., *filter* berwarnya oranye juga memenuhi prinsip *weight sharing*). Walaupun memiliki konfigurasi bobot *synapse weights* yang sama, unit dapat menghasilkan output yang berbeda untuk input yang berbeda. Konsep *weight sharing* ini sesuai dengan pernyataan bahwa konfgurasi parameter untuk mengenali karakteristik informatif untuk satu objek bernilai sama walau pada lokasi yang berbeda. Dengan *weight sharing*, parameter neural network juga menjadi lebih sedikit dibanding menggunakan *multilayer perceptron (feed-forward neural network).*



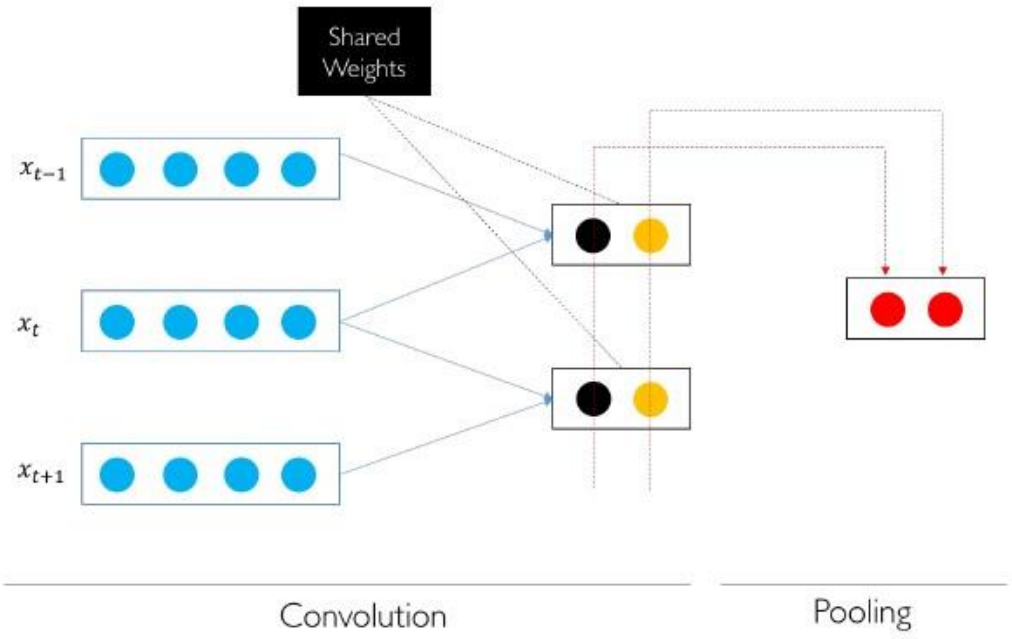
Gambar 2.19 Konsep weight sharing

2.8.3 Pooling

Pada tahap convolution, setiap k-sized window diubah menjadi satu vektor berdimensi d (yang dapat disusun menjadi matriks D). Semua vektor yang dihasilkan pada tahap sebelumnya dikombinasikan (pooled) menjadi satu vektor c. Ide utamanya adalah mengekstrak informasi paling informatif (semacam meringkas). Ada beberapa teknik *pooling*, diantaranya: *max pooling*, *average pooling*, dan *K-max pooling*; diilustrasikan pada Gambar 2.20. *Max pooling* mencari nilai maksimum untuk setiap dimensi vektor. *Average pooling* mencari nilai rata-rata tiap dimensi. *K-max* pooling mencari K nilai terbesar untuk setiap dimensinya (kemudian hasilnya digabungkan). Gabungan operasi *convolution* dan *pooling* secara konseptual diilustrasikan pada Gambar 2.21.

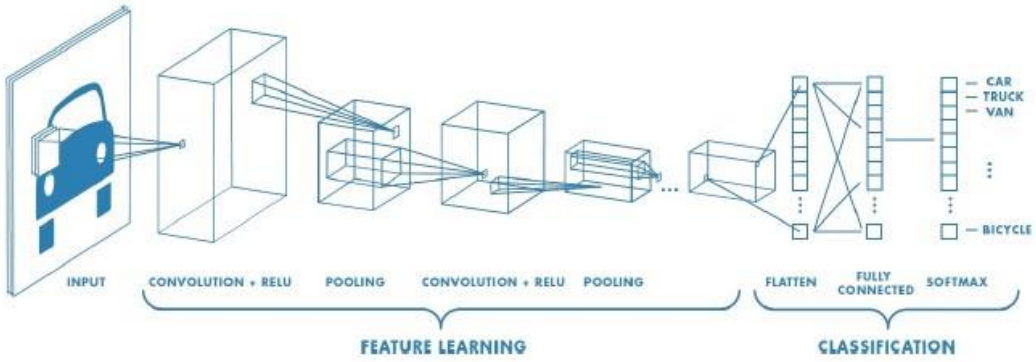


Gambar 2.20 Contoh pooling



Gambar 2.21 Convolution dan pooling

Setelah melewati berbagai operasi *convolution* dan *pooling*, pembuat akan memiliki satu vektor yang kemudian dilewatkan pada *multilayer perceptron (fully connected)* untuk melakukan sesuatu (tergantung permasalahan), misal klasifikasi gambar, klasifikasi sentimen, dsb (Ilustrasi pada Gambar 2.22).



Gambar 2.22 Convolutional Neural Network

BAB 3. METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian dilakukan selama rentang waktu 12 bulan (satu tahun), dimulai dari bulan Mei 2020 sampai Selesai di Lab Rekayasa Perangkat Lunak Jurusan Teknologi Informasi Politeknik Negeri Jember. Penelitian hanya dilakukan di satu tempat yaitu di Gedung Jurusan Teknologi Informasi Politeknik Negeri Jember. Berikut adalah jadwal pengerjaan penelitian yang akan dilakukan pada Tabel 3.1.

Tabel 3.1 Jadwal Pengerjaan Penelitian

Jenis Penelitian	Bulan Ke											
	1	2	3	4	5	6	7	8	9	10	11	12
Studi Permasalahan												
Studi Pustaka dan Literatur												
Pengumpulan Data												
Pengembangan Sistem												
Hasil Penelitian												
Analisis Hasil												

3.2 Alat dan Bahan

3.2.1 Alat Penelitian

Pada penelitian ini digunakan alat berupa perangkat keras dan perangkat lunak sebagai berikut :

a. Perangkat Keras

Perangkat keras yang digunakan antara lain satu unit laptop, pc dan kamera webcam dengan detail sebagi berikut :

1. Acer Aspire E5-476G-58V
    a) Screen Size 14 inch
    b) Processor Intel(R) Core™ i5-8250 1.6 GHz with Turbo Boost up to 3.4 GHz
    c) NVIDIA (R) Geforce(R) MX130 with 2 GB VRAM
    d) 8 GB DDR4 Memory
    e) 1000 GB HDD
2. Webcam :
    a) Internal HDR webcam Acer Aspire E5-476G-58V
    b) Eksternal USB webcam

b. Perangkat Lunak

Perangkat lunak yang digunakan antara lain :

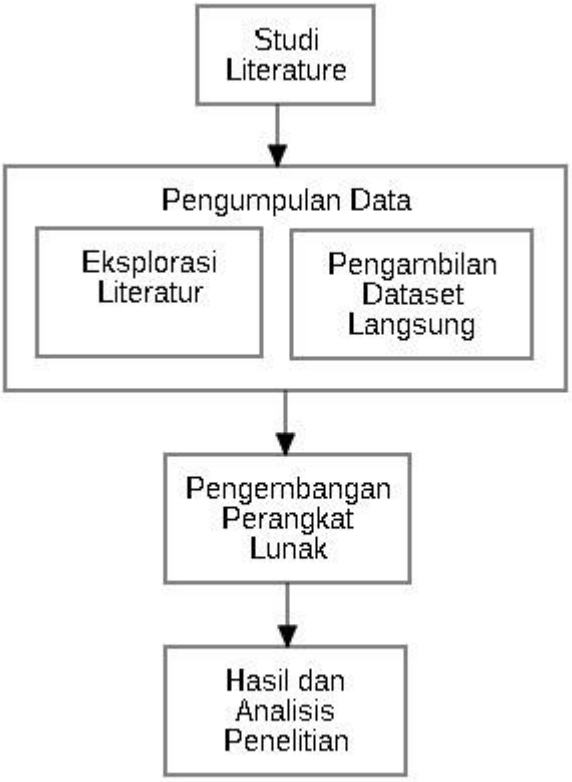
1. Operating System Windows 10 64-bit
2. JetBrains PyCharm Edu x64
3. Python 3.7 ++
4. OpenCV 4.5.2.54
5. Keras 2.4.3
6. Matplotlib 3.4.2
7. Mediapipe 0.8.6
8. Numpy 1.19.5
9. Tensorflow 2.5.0
10. Sklearn

3.2.2 Bahan Penelitian

Data sampel (training dan validasi) citra Tangan Bahasa Isyarat Indonesia (BISINDO) dari huruf A-Z dan 1-10.

3.3 Metode Penelitian

Pada metode penelitian ini peneliti melakukan beberapa tahapan yaitu studi literatur, pengumpulan data, tahapan pengembangan sistem, hasil, dan analisis penelitian seperti pada Gambar 3.1.



Gambar 3.1 Block diagram tahapan metode penelitian

Gambar 3.1 menunjukkan tahap-tahap yang dilakukan pada penelitian ini. Tahapannya terdiri atas studi literatur, pengumpulan data dan tahapan pengembangan sistem. Penjelasan mengenai setiap tahapan akan dijelaskan sebagai berikut :

a. Studi Literatur

Penelitian ini diawali dengan tahap studi literatur untuk mencari referensi-referensi atau teori-teori yang sesuai dengan permasalahan dan solusi penelitian. Adapun referensi yang peneliti pelajari meliputi :

1. Informasi tentang sejarah dan bentuk gerak Bahasa Isyarat Indonesia (BISINDO)
2. Penelitian yang mirip dalam pengembangan sistem komputer vision untuk penerjemah bahasa isyarat
3. Ekstraksi fitur warna untuk deteksi objek
4. Konsep Deep Learning Convolutional Neural Network

b. Pengumpulan Data

Penelitian ini menggunakan data dan informasi akurat untuk menunjang proses penelitian agar berjalan efektif dan efisien. Berikut ini metode pengumpulan data yang dilakukan peneliti :

1. Eksplorasi dan Studi Literatur

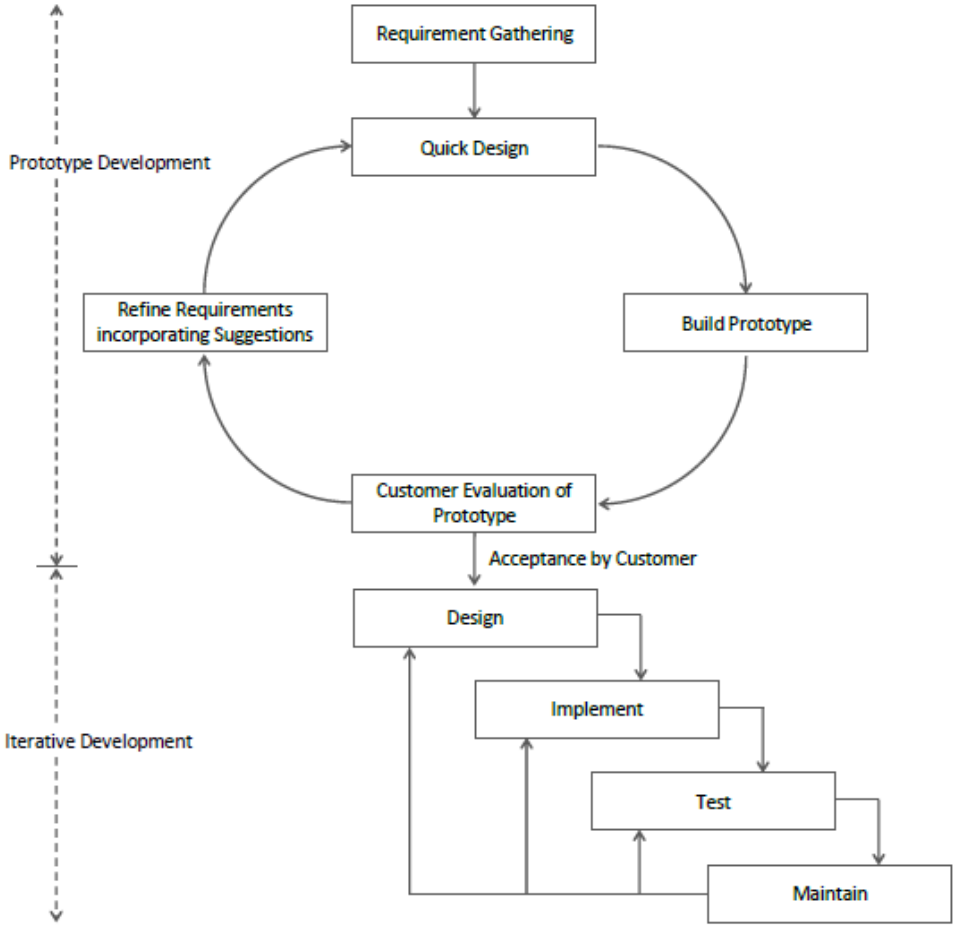
   Untuk menghasilkan output yang akurat dalam mengembangkan suatu sistem penerjemah bahasa isyarat. Peneliti membaca, mempelajari dan membandingkan literatur yang memuat model, struktur data dan metode yang diterapkan pada sistem penerjemah bahasa isyarat yang terdahulu.

2. Pengambilan Dataset Langsung

   Pada tahap ini peneliti melakukan pengambilan dataset secara langsung berupa citra Bahasa Isyarat Indonesia (BISINDO) untuk digunakan sebagai data peneltian.

c. Tahapan Pengembangan Perangkat Lunak

Metode yang digunakan dalam pengembangan sistem penerjemah bahasa isyarat adalah metode *prototype*. Metode ini merupakan suatu paradigma baru dalam metode pengembangan perangkat lunak dimana merevolusi metode pengembangan perangkat lunak yang lama yaitu sistem sekuensial yang biasa dikenal dengan nama SDLC atau waterfall development model.



Gambar 3.2 Tahapan Metode Prototype

Berikut adalah Tahapan – tahapan Proses Pengembangan dalam Model Prototype, yaitu :

1. Pengumpulan kebutuhan

   Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

2. Membangun prototyping

   Membangun prototyping dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat input dan format output).

3. Evaluasi protoptyping

Evaluasi ini dilakukan oleh pelanggan, apakah prototyping yang sudah dibangun sudah sesuai dengan keinginan pelanggan atau belum. Jika sudah sesuai, maka langkah selanjutnya akan diambil. Namun jika tidak, prototyping direvisi dengan mengulang langkah-langkah sebelumnya.

4. Mengkodekan sistem

   Dalam tahap ini prototyping yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.

5. Menguji sistem

   Setelah kode program selesai testing dapat dilakukan. Testing memfokuskan pada logika internal dari perangkat lunak, fungsi eksternaldan mencari segala kemungkinan kesalahan dan memriksa apakah sesuaidengan hasil yang diinginkan.

6. Pemeliharaan

   Pemeliharaan mencakup koreksi dari berbagai error yang tidak ditemukan pada tahap-tahap terdahulu, perbaikan atas implementasi dan pengembangan unit sistem, serta pemeliharaan program.

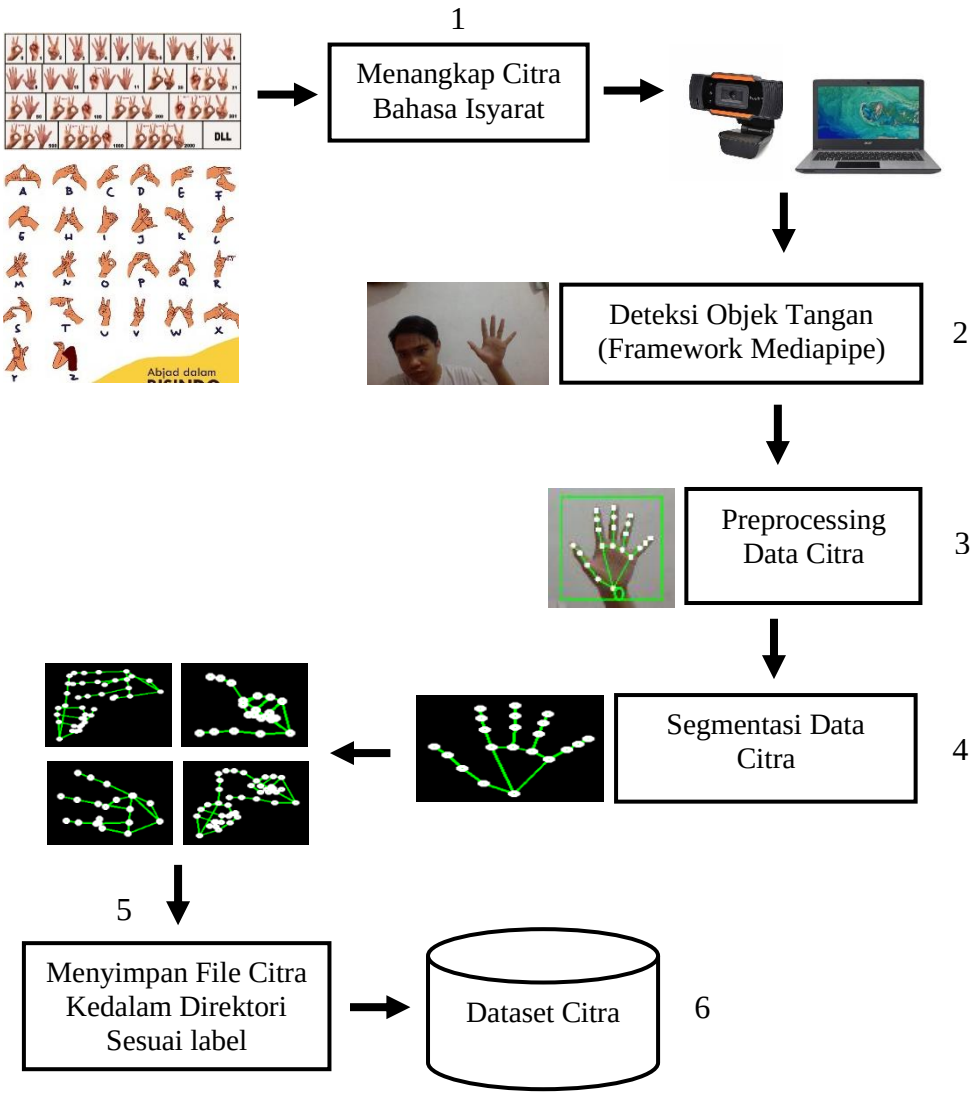
3.4 Jenis Data

3.4.1 Data Primer

Data Primer dalam penelitian ini adalah data gambar atau sampel gambar Bahasa Isyarat Indonesia (BISINDO) dengan proses pengambilan langsung yang dilakukan oleh peneliti.

3.5 Block Diagram System

Dalam sistem ini hanya terdapat user tunggal sebagai aktornya. Alur proses sistem ini dapat digambarkan dalam bentuk diagram block. Berikut ini digambarkan mengenai diagram *block* sistem pada Gambar *3.3*, Gambar 3.4 dan Gambar *3.5*.



Gambar 3.3 Block Diagram Proses Mendapatkan Dataset Citra

Pada Gambar *3.3* Block diagram proses mendapatkan dataset citra memiliki beberapa tahapan, seperti berikut :

a. Tahap Pertama

   Kamera webcam dan laptop menangkap *gesture* Bahasa Isyarat indonesia yang diperagakan oleh peneliti. Terdapat satu kamera yang dipakai yaitu kamera webcam eksternal. Posisi kamera berada didepan user atau disebut *selfie*.

b. Tahap Kedua

Untuk melakukan pelacakan tangan dan jari dengan ketelitian tinggi, peneliti menggunakan framework machine learning bernama “Mediapipe”. Framework ini menggunakan pembelajaran mesin (ML) untuk menyimpulkan 21 landmark 3D tangan hanya dari satu bingkai secara realtime, yang nantinya 21 landmark tersebut akan digunakan untuk proses-proses penting seperti mendapatkan *region of interest* pada gambar.

c. Tahap Ketiga

Sistem melakukan proses preprocessing pada data citra yang berhasil ditangkap oleh kamera. Preprocessing adalah proses dimana citra yang telah masuk ke sistem akan diproses melalui beberapa tahap, meliputi :

1. Crop Image

   Crop image merupakan penghapusan bagian sudut dari suatu gambar untuk memotong/mengambil/mengeluarkan sebagian isi dari gambar guna memperoleh hasil yang diinginkan.

2. Resize

   Proses resize citra digunakan untuk mengurang jumlah piksel dari citra inputan.

3. Operator Laplacian

   Operator Laplacian merupakan operator turunan yang digunakan untuk mencari edge pada suatu citra. Laplacian sering diterapkan pada gambar yang telah dihaluskan terlebih dahulu dengan filter Gaussian seperti untuk mengurangi sensitivitasnya terhadap noise. Operator ini biasanya menggunakan satu gambar keabuan sebagai input dan output.

d. Tahap Keempat

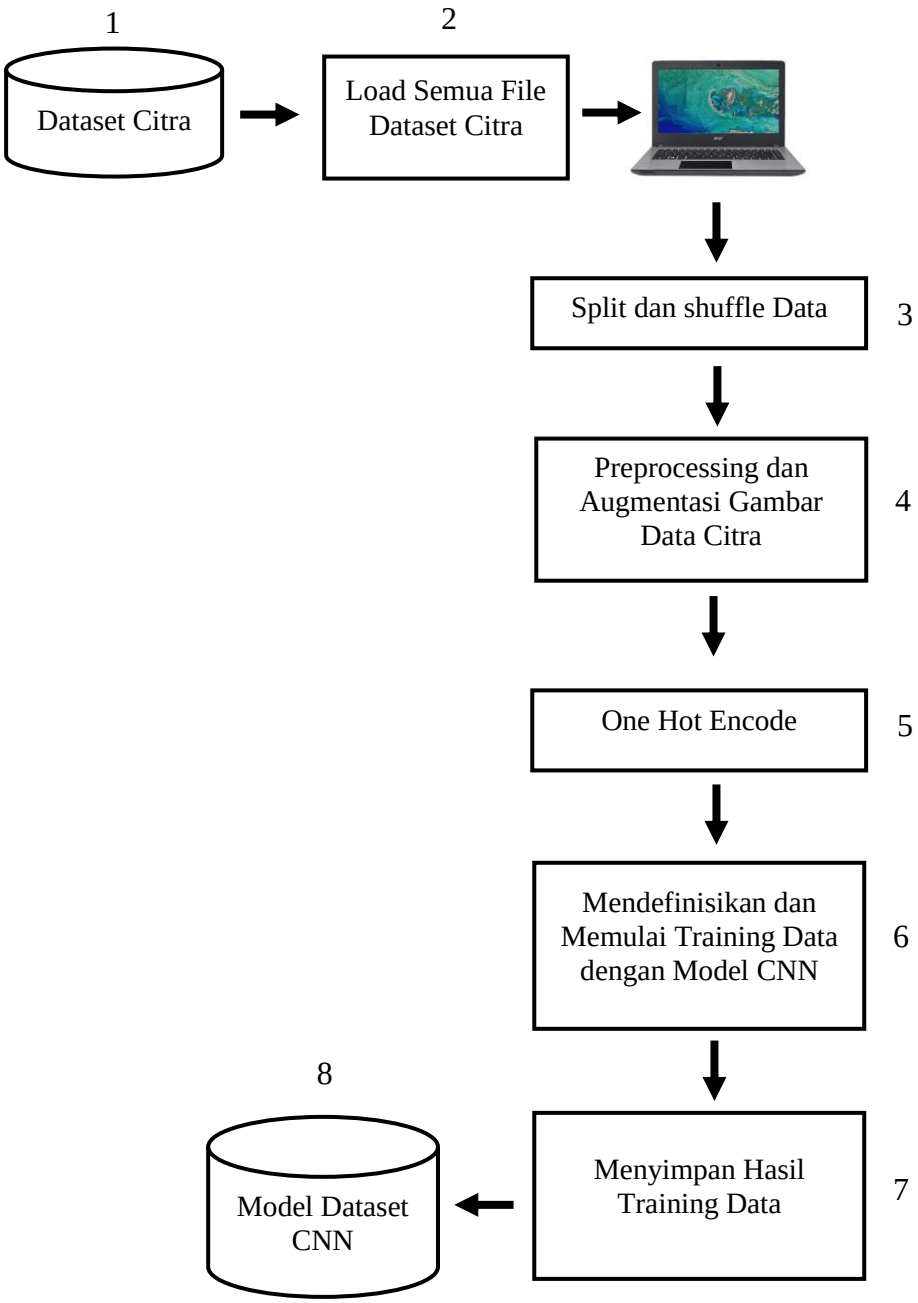
Melakukan proses segmentasi citra untuk memisahkan objek penelitian dengan latar yang terdapat dalam sebuah citra. Proses tersebut meliputi mendapatkan data landmark hasil proses *hand detection* Mediapipe, membuat *binding box* disekitar objek tangan yang sudah terdeteksi, dan membuat ulang gambar landmark tangan dengan background hitam sesuai *binding box* dan hasil proses *hand detection* Mediapipe.

e. Tahap Kelima

   Menyimpan file gambar hasil proces *preprocessing* dan segmentasi citra kedalam folder yang bernama sesuai dengan label (kelas/kelompok *gesture*).

f. Tahap Keenam

   Dataset merupakan kumpulan data citra yang akan digunakan untuk proses data training dalam pembuatan data model deep learning (CNN). Pada penelitian ini terdapat 36 direktori/kelas dimana setiap kelas berisi 1016 data citra sehingga total dataset yang digunakan sebanyak 36.576.



Gambar 3.4 Block Diagram Proses Training Dataset Citra

Pada Gambar 3.4 Block diagram proses training dataset citra memiliki beberapa tahapan, seperti berikut :

a. Tahap Pertama

   Pada penelitian ini terdapat 36 direktori/kelas dimana setiap kelas berisi 1016 data citra sehingga total dataset yang digunakan sebanyak 36.576.

b. Tahap Kedua

Memanggil semua file beserta dengan labelnya yang ada dalam direktori dataset, kemudian sistem akan membuat array dataset dan array label secara berurutan. Array tersebut akan digunakan untuk proses komputasi oleh komputer.

c. Tahap Ketiga

Dengan bantuan fungsi pada *scikit-learn*, peneliti akan memisahkan *array* dataset menjadi 3 bagian utama yaitu *training* (23.424), *testing* (7.320) dan *validation* (5856).

d. Tahap Keempat

Sistem melakukan proses preprocessing pada data citra yang berhasil ditangkap oleh kamera. Preprocessing adalah proses dimana citra yang telah masuk ke sistem akan diproses melalui beberapa tahap, meliputi :

1. Resize

   Proses resize citra digunakan untuk mengurang jumlah piksel dari citra inputan.

2. Reshaping Image Array

   Dikarenakan Keras Tensorflow hanya menerima data gambar berupa gambar dengan 1 channel maka diperlukan proses untuk merubah bentuk array. Bentuk array adalah jumlah elemen dalam setiap dimensi. Dengan reshaping array peneliti bisa dapat menambah atau menghapus dimensi atau mengubah jumlah elemen di setiap dimensi.

3. Konversi Citra Grayscale

   Citra yang ditampilkan dari citra jenis ini terdiri atas warna abu-abu, bervariasi pada warna hitam pada bagian yang intensitas terlemah dan warna putih pada intensitas terkuat. Citra grayscale disimpan dalam format 8 bit untuk setiap sample pixel, yang memungkinkan sebanyak 256 intensitas. Format ini sangat membantu dalam pemrograman karena manipulasi bit yang tidak terlalu banyak. Untuk mengubah citra berwarna yang mempunyai nilai matrik masing-masing R, G dan B menjadi citra grayscale dengan nilai X,

maka konversi dapat dilakukan dengan mengambil rata-rata dari nilai R, G dan B.

4. Histogram Equalization

   Histogram Equalization adalah suatu metode dalam pengolahan citra dengan penyesuaian kontras menggunakan histogram citra. Metode ini biasanya meningkatkan kontras global dari banyak gambar. OpenCV memiliki fungsi untuk melakukan ini, cv2.equalizeHist(). Inputnya hanya gambar skala abu-abu dan outputnya adalah gambar histogram yang disamakan.

5. Augmentasi Data

   Augmentasi data adalah suatu proses dalam pengolahan data gambar, augmentasi merupakan proses mengubah atau memodifikasi gambar sedemikian rupa sehingga komputer akan mendeteksi bahwa gambar yang diubah adalah gambar yang berbeda, namun manusia masih dapat mengetahui bahwa gambar yang diubah tersebut adalah gambar yang sama (L. Perez and J. Wang, 2017). Augmentasi dapat meningkatkan akurasi dari model CNN yang dilatih karena dengan augmentasi model mendapatkan data-data tambahan yang dapat berguna untuk membuat model yang dapat melakukan generalisasi dengan lebih baik. Augmentasi yang dilakukan pada penelitian ini adalah pergeseran lebar dengan nilai batas sebanyak 10%, pergeseran Ketinggian dengan nilai batas sebanyak 10%, *zoom* dengan nilai batas sebanyak 20%, intensitas geser dengan nilai batas sebanyak 10%, dan rotasi dengan nilai batas sebanyak 10°.

e. Tahap Kelima

   One Hot Encoding adalah salah satu metode encoding. Metode ini merepresentasikan data bertipe kategori sebagai vektor biner yang bernilai integer 0 dan 1, dimana semua elemen akan bernilai 0 kecuali satu elemen yang bernilai 1, yaitu elemen yang memiliki nilai kategori tersebut.

f. Tahap Keenam

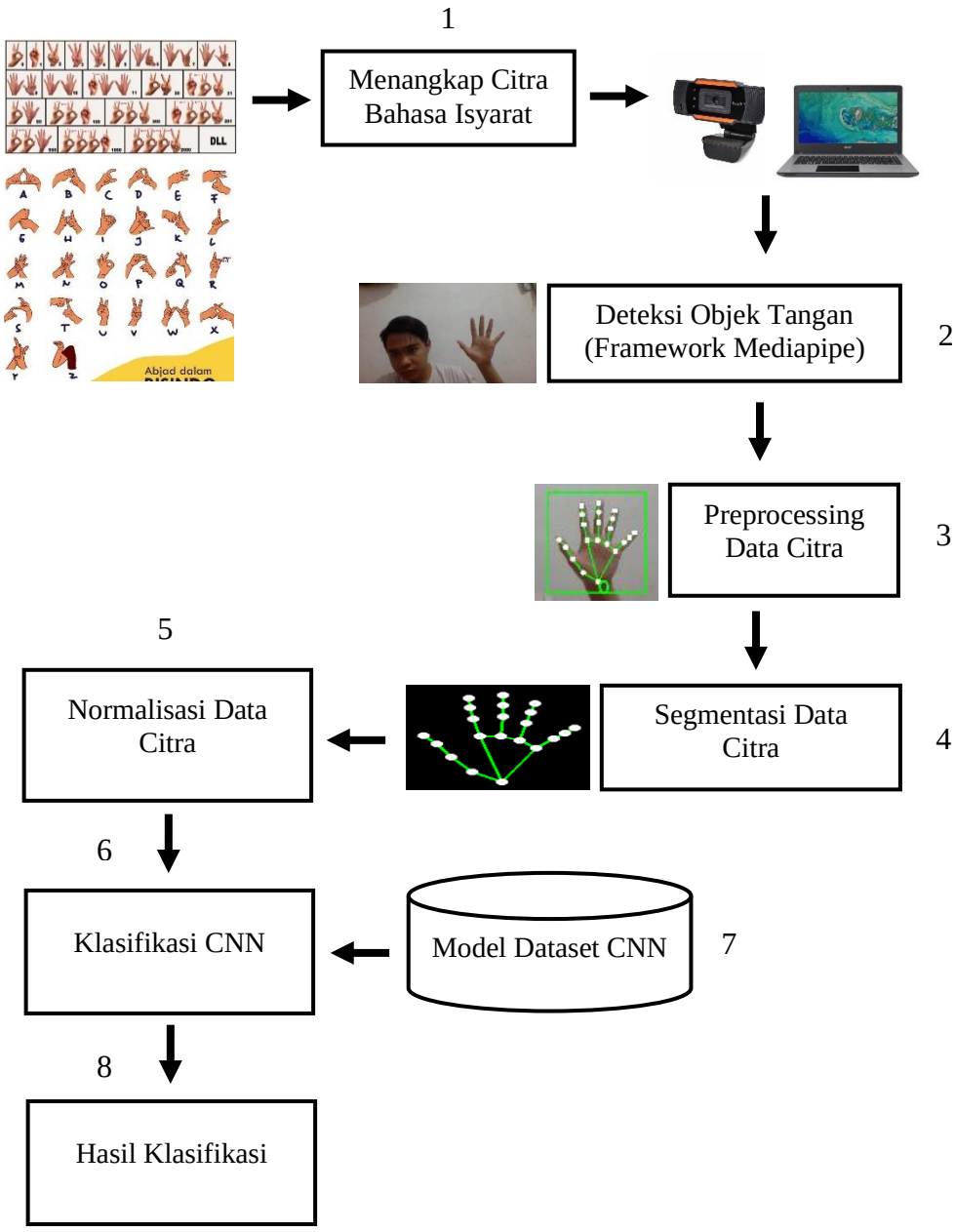
Melakukan Proses Training data dengan menggunakan metode algoritma CNN (Convolutional Neural Network).

g. Tahap Ketujuh

Menyimpan hasil proses training data dengan format file tertentu (model_trained_all.h5).

h. Tahap Kedelapan

File model yang sudah berhasil dibuat selanjutnya bisa digunakan oleh mechine learning untuk proses klasifikasi tanpa harus melakukan training data lagi.



Gambar 3.5 Block Diagram Proses Klasifikasi Citra

Pada Gambar *3.5* Block diagram proses klasifikasi citra dengan menggunakan algoritma model CNN memiliki beberapa tahapan, seperti berikut :

a. Tahap Pertama

Kamera webcam dan laptop menangkap gesture Bahasa Isyarat indonesia yang diperagakan oleh peneliti. Terdapat satu kamera yang dipakai yaitu kamera webcam eksternal. Posisi kamera berada didepan user atau disebut selfie.

b. Tahap Kedua

Untuk melakukan pelacakan tangan dan jari dengan ketelitian tinggi, peneliti menggunakan framework machine learning bernama “Mediapipe”. Framework ini menggunakan pembelajaran mesin (ML) untuk menyimpulkan 21 landmark 3D tangan hanya dari satu bingkai secara realtime, yang nantinya 21 landmark tersebut akan digunakan untuk proses-proses penting seperti mendapatkan region of interest pada gambar.

c. Tahap Ketiga

Sistem melakukan proses preprocessing pada data citra yang berhasil ditangkap oleh kamera. Preprocessing adalah proses dimana citra yang telah masuk ke sistem akan diproses melalui beberapa tahap, meliputi :

1. Crop Image

   Crop image merupakan penghapusan bagian sudut dari suatu gambar untuk memotong/mengambil/mengeluarkan sebagian isi dari gambar guna memperoleh hasil yang diinginkan.

2. Resize

   Proses resize citra digunakan untuk mengurang jumlah piksel dari citra inputan.

3. Operator Laplacian

   Operator Laplacian merupakan operator turunan yang digunakan untuk mencari edge pada suatu citra. Laplacian sering diterapkan pada gambar yang telah dihaluskan terlebih dahulu dengan filter Gaussian seperti untuk mengurangi sensitivitasnya terhadap noise. Operator ini biasanya menggunakan satu gambar keabuan sebagai input dan output.

d. Tahap Keempat

Melakukan proses segmentasi citra untuk memisahkan objek penelitian dengan latar yang terdapat dalam sebuah citra. Proses tersebut meliputi mendapatkan data landmark hasil proses hand detection Mediapipe, membuat binding box disekitar objek tangan yang sudah terdeteksi, dan

membuat ulang gambar landmark tangan dengan background hitam sesuai binding box dan hasil proses hand detection Mediapipe.

e. Tahap Kelima

Dikarenakan Keras Tensorflow hanya menerima data gambar berupa gambar dengan 1 channel maka diperlukan proses untuk merubah bentuk array. Bentuk array adalah jumlah elemen dalam setiap dimensi. Dengan reshaping array peneliti bisa dapat menambah atau menghapus dimensi atau mengubah jumlah elemen di setiap dimensi.

f. Tahap Keenam

Inisialisasi proses prediksi oleh deep learning CNN terhadap citra input dan model dataset untuk mengklasifikasikan citra dan menerjemahkannya kedalam bentuk tulisan.

g. Tahap Ketujuh

File model yang sudah berhasil dibuat bisa digunakan oleh mechine learning untuk proses klasifikasi tanpa harus melakukan training data lagi.

h. Tahap Kedelapan

Hasil klasifikasi yang didapatkan akan disimpan dan ditampilkan di layar monitor.

CNN Model Architecture.

Jaringan saraf yang dirancang untuk pengenalan karakter bahasa isyarat Indonesia (Abjad dan Nomor) terdiri dari 4 lapisan konvolusi diikuti oleh fungsi aktivasi Relu dan 2 lapisan Maxpooling. Diikuti dengan *dropout*, *flatten*, *density layer* yang diakhiri dengan *softmax activation layer* dengan 36 output (26 huruf dan 10 angka), yang akan menunjukkan persentase klasifikasi yang diperoleh untuk setiap kelas.

Penelitian ini menggunakan gambar input 32x32 piksel, operasi konvolusi pertama diterapkan dengan filter 5x5 untuk menghasilkan 60 peta fitur ukuran 28x28. operasi konvolusi kedua diterapkan yang filternya 5x5 untuk menghasilkan 60 peta fitur ukuran 24x24. Filter max pooling 2x2 diterapkan dan menghasilkan peta fitur ukuran 12x12. Selanjutnya melakukan 2 operasi konvolusi terakhir dengan ukuran dan jumlah filter yang sama yaitu 3x3 dan 30, menghasilkan peta

fitur 30 dengan ukuran 8x8. Kemudian menerapkan filter max pooling 2x2 menghasilkan peta fitur 4x4. Setelah tahap ekstraksi fitur, dilakukan klasifikasi dari *layer flatten*, *dense*, *dropout* dan *fully connected*. CNN ini diakhiri dengan fungsi softmax, yang memberikan probabilitas untuk melakukan klasifikasi masing-masing karakter. Berikut adalah gambaran struktur model CNN.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 60)        1560
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 60)        90060
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 60)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 30)        16230
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 30)          8130
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 4, 30)          0
_________________________________________________________________
dropout (Dropout)            (None, 4, 4, 30)          0
_________________________________________________________________
flatten (Flatten)            (None, 480)               0
_________________________________________________________________
dense (Dense)                (None, 500)               240500
_________________________________________________________________
dropout_1 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_1 (Dense)              (None, 36)                18036
=================================================================
```

Gambar 3.6 CNN Model Architecture

BAB 4. HASIL DAN PEMBAHASAN

4.1 Pengumpulan data

…

4.2 Pengolahan Citra

…

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

….

5.2 Saran

….

DAFTAR PUSTAKA

Khoiruddin M. Analisis Human Skin Detection Menggunakan Hsv dengan Salt-And Pepper Noise Reduction. Published online 2016.

Sosial I, Dan T, Di D, et al. Bahasa Isyarat Indonesia Dalam Proses. Published online 2013:1-15.

Gumelar G, Hafiar H, Subekti P. Bahasa Isyarat Indonesia Sebagai Budaya Tuli Melalui. Inf Kaji Ilmu Komun. 2018;48(1):65-78.

Breva Yunanda A, Mandita F, Primasetya Armin A. Pengenalan Bahasa Isyarat Indonesia (BISINDO) Untuk Karakter Huruf Dengan Menggunakan Microsoft Kinect. Fountain Informatics J. 2018;3(2):41. doi:10.21111/fij.v3i2.2469

Putra JWG. Pengenalan Konsep Pembelajaran Mesin dan Deep Learning. 2019;(July):1-235. https://www.researchgate.net/publication/323700644

Ilmiah J, Fisip M, Volume U, et al. Jurnal Ilmiah Mahasiswa FISIP Unsyiah Volume 4, Nomor 3, Agustus 2019 www.jim.unsyiah.ac.id/FISIP. 2019;4:1-15.

Fadillah RZ. Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network. Fak Sains dan Ilmu Komputer, Progr Stud Ilmu Komputer, Univ Pertamina. Published online 2020.

Bagus M, Bakti S, Pranoto YM. Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network. Published online 2019:11-16.

Gafar AA, Sari JY. Sistem Pengenalan Bahasa Isyarat Indonesia dengan Menggunakan Metode Fuzzy K-Nearest Neighbor. J Ultim. 2018;9(2):122-128. doi:10.31937/ti.v9i2.671

Hu J, Kuang Y, Liao B, Cao L, Dong S, Li P. A Multichannel 2D Convolutional Neural Network Model for Task-Evoked fMRI Data Classification. Comput Intell Neurosci. 2019;2019(i). doi:10.1155/2019/5065214

Bhatnagar S, Agrawal S. Hand Gesture Recognition for Indian Sign Language: A Review. Int J Comput Trends Technol. 2015;21(3):121-122. doi:10.14445/22312803/ijctt-v21p122

Rahim MA, Islam MR, Shin J. Non-touch sign word recognition based on dynamic hand gesture using hybrid segmentation and CNN feature fusion. Appl Sci. 2019;9(18). doi:10.3390/app9183790

Pinto RF, Borges CDB, Almeida AMA, Paula IC. Static Hand Gesture Recognition Based on Convolutional Neural Networks. J Electr Comput Eng. 2019;2019. doi:10.1155/2019/4167890