

**SISTEM PENERJEMAH BAHASA ISYARAT MENGGUNAKAN  
METODE *DEEP LEARNING MODEL MULTICHANNEL 2D*  
*CONVOLUTIONAL NEURAL NETWORK (M2D CNN)***

**SKRIPSI**



Oleh

**Rizkika Zakka Palindungan**

**NIM E41170164**

**PROGRAM STUDI TEKNIK INFORMATIKA  
JURUSAN TEKNOLOGI INFORMASI  
POLITEKNIK NEGERI JEMBER**

**2020**

**SISTEM PENERJEMAH BAHASA ISYARAT MENGGUNAKAN  
METODE *DEEP LEARNING MODEL MULTICHANNEL 2D*  
*CONVOLUTIONAL NEURAL NETWORK (M2D 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  
2020**

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN  
POLITEKNIK NEGERI JEMBER  
JURUSAN TEKNOLOGI INFORMASI**

---

**HALAMAN PENGESAHAN PROPOSAL**

- |                               |   |
|-------------------------------|---|
| 1. Judul                      | : Sistem Penerjemah Bahasa Isyarat<br>Menggunakan Metode Deep Learning<br>Model Multichannel 2D Convolutional<br>Neural Network (M2D CNN) |
| 2. Identitas Pelaksana        | :   |
| a. Nama                       | : Rizkika Zakka Palindungan   |
| b. NIM                        | : E41170164   |
| c. Jurusan/Program Studi      | : Teknologi Informasi/Teknik Informatika  |
| 3. Lokasi                     | : Politeknik Negeri Jember  |
| 4. Identitas Dosen Pembimbing | :   |
| a. Nama Lengkap               | : Aji Seto Arifianto, S.ST., M.T.   |
| b. NIP                        | : 19851128 200812 1 002   |
| c. Jurusan/Program Studi      | : Teknologi Informasi/Teknik Informatika  |
| 5. Lama Kegiatan              | : dua belas (12) bulan  |
- 

**Jember, 27 Juni 2020**

**Menyetujui,**

**Dosen Pembimbing**

**Pelaksana**

Aji Seto Arifianto, S.ST., M.T.

NIP 19851128 200812 1 002

Rizkika Zakka Palindungan

NIM E41170164

**Mengetahui,**

**Ketua Program Studi Teknik Informatika**

Trismayanti Dwi P, S.Kom, M.Cs.

NIP 19900227 201803 2 001

## DAFTAR ISI

HALAMAN JUDUL.....	ii
HALAMAN PENGESAHAN PROPOSAL .....	iii
DAFTAR ISI.....	iv
DAFTAR GAMBAR .....	vi
DAFTAR TABEL.....	vii
BAB 1. PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	3
1.3 Tujuan Penelitian.....	4
1.4 Manfaat Penelitian.....	4
BAB 2. TINJAUAN PUSTAKA .....	5
2.1 State Of The Art .....	5
2.2 Referensi dan Penelitian Yang Sudah Ada.....	6
2.2.1 Sistem Pengenalan Bahasa Isyarat Indonesia Dengan Menggunakan Metode Fuzzy K-Nearest Neighbor. ....	6
2.2.2 Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network.....	6
2.2.3 Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network.....	7
2.3 Kekurangan Penelitian Sebelumnya.....	7
2.3.1 Sistem Pengenalan Bahasa Isyarat Indonesia Dengan Menggunakan Metode Fuzzy K-Nearest Neighbor. ....	7
2.3.2 Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network.....	8
2.3.3 Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network.....	8
2.4 Kelebihan Dari Penelitian Yang Diambil.....	8
2.5 Bahasa Isyarat.....	9
2.5.1 Sejarah Bahasa Isyarat Indonesia (BISINDO).....	10

2.5.2	Isyarat Abjad Dalam Bisindo .....	11
2.6	Pengolahan Citra Digital .....	12
2.6.1	RGB Color Spaces .....	12
2.6.2	HSV Color Space .....	13
2.6.3	Citra Grayscale .....	14
2.6.4	Deteksi Warna Kulit .....	14
2.7	Gradient Descent .....	15
2.8	Artificial Neural Network .....	16
2.8.1	Single Perceptron .....	17
2.8.2	Multilayer Perceptron .....	19
2.8.3	Multi-class Classification .....	23
2.8.4	Multi-label Classification .....	24
2.8.5	Deep Neural Network .....	24
2.8.6	Regularization and Dropout .....	27
2.9	Convolutional Neural Network .....	28
2.9.1	Activation Layer .....	29
2.9.2	Convolution .....	30
2.9.3	Pooling .....	32
BAB 3.	Metode penelitian .....	35
3.1	Waktu dan Tempat Penelitian .....	35
3.2	Alat dan Bahan .....	35
3.2.1	Alat Penelitian .....	35
3.2.2	Bahan Penelitian .....	36
3.3	Metode Penelitian .....	36
3.4	Jenis Data .....	40
3.4.1	Data Primer .....	40
3.5	Block Diagram System .....	40
DAFTAR PUSTAKA	.....	48

## DAFTAR GAMBAR

Gambar 2.1 Gesture Bahasa Isyarat Indonesia (BISINDO).....	11
Gambar 2.2 Warna RGB .....	12
Gambar 2.3 Warna HSV .....	13
Gambar 2.4 Warna Grayscale .....	14
Gambar 2.5 Deep Neural Network.....	17
Gambar 2.6 Single Perceptron .....	18
Gambar 2.7 Multilayer Perceptron.....	20
Gambar 2.8 Multilayer Perceptron 2.....	22
Gambar 2.9 Proses latihan menggunakan backpropagation .....	23
Gambar 2.10 Ilustrasi representasi desired output pada multi-class classification	24
Gambar 2.11 Ilustrasi representasi desired output pada multi-label classification	24
Gambar 2.12 Deep Neural Network.....	25
Gambar 2.13 Proses latihan DNN menggunakan backpropagation.....	26
Gambar 2.14 Contoh successive learning .....	27
Gambar 2.15 Sliding window .....	30
Gambar 2.16 1D Convolution .....	31
Gambar 2.17 Konsep weight sharing .....	32
Gambar 2.18 Contoh pooling.....	33
Gambar 2.19 Convolution dan pooling.....	33
Gambar 2.20 Convolutional Neural Network .....	34
Gambar 3.1 Block diagram tahapan metode penelitian .....	37
Gambar 3.2 Tahapan Metode Prototype .....	39
Gambar 3.3 Block diagram proses proses mendapatkan dataset .....	41
Gambar 3.4 Block diagram proses data latih (training) .....	43
Gambar 3.5 Block diagram proses klasifikasi citra .....	44
Gambar 3.6 Model arsitektur Multichannel 2D Convolutional Neural Network .	47

## **DAFTAR TABEL**

Tabel 2.1 State Of The Art.....	5
Tabel 3.1 Jadwal Pengerjaan Penelitian.....	35

## 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. *Computer 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, *thresholding*, *filtering* 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 (alfabet dan angka), input data yang hanya berupa foto dari model peraga yang diambil secara manual atau tidak *realtime* berasal dari video kamera/webcam dan tidak semua penelitian melakukan penghilangan *area* wajah dan leher didalam proses pengolahan citra untuk deteksi objek tangan.

Sebagai perbaikan dan pengembangan sistem lebih lanjut, penelitian ini akan menggunakan Bahasa Isyarat Indonesia (BISINDO) sebagai jenis bahasa isyarat yang dipakai didalam dataset. Selanjutnya akan menambah jumlah kamera *input* menjadi 2 buah untuk mendeteksi tangan dari 2 sisi yaitu sisi depan dan sisi samping

hal ini bertujuan untuk meningkatkan kemampuan dan akurasi dalam mendeteksi bahasa isyarat yang lebih kompleks. Pada proses pengolahan data citra dari kamera akan dilakukan penghilangan bagian wajah dan leher dengan menggunakan bantuan *Haar Cascade Classifier*, kemudian melakukan *skin mask* atau deteksi warna kulit untuk memfokuskan pada objek tangan. Algoritma yang akan digunakan pada penelitian ini adalah salah satu model pengembangan dari CNN yaitu model arsitektur *Multichannel 2D Convolutional Neural Network*, algoritma ini dipilih berdasarkan tingkat kemampuan dan akurasi untuk mengolah 2 atau lebih data input citra dari sebuah objek dengan sudut pandang yang berbeda hal ini dibuktikan dari penelitian berjudul *A Multichannel 2D Convolutional Neural Network Model for Task-Evoked fMRI Data Classification* dengan objek input berupa gambar otak dari 3 sisi yaitu sumbu x, y, z yang memiliki akurasi 83.20% (Hu., 2019).

## 1.2 Rumusan Masalah

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

1. Bagaimana proses ekstraksi fitur warna untuk mendeteksi warna kulit dan objek tangan pada data citra dari kamera video?
2. Bagaimana proses pengolahan citra dari 2 kamera masukan?
3. Bagaimana implementasi metode *Deep Learning Model Multichannel 2D 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 (bagian dalamnya disebut telapak tangan).
2. Subjek peraga berada didalam ruangan dengan warna background putih dan baju lengan panjang yang tidak sama atau mirip dengan warna kulit.
3. Pencahayaan ruangan harus terang dan tidak menggunakan cahaya lampu yang memiliki warna dasar, karena akan mengganggu deteksi kulit.
4. Kamera yang digunakan minimal memiliki resolusi sebesar 640 x 480 *pixels*.

### **1.3 Tujuan Penelitian**

Adapun tujuan penelitian ini adalah sebagai berikut :

1. Untuk mengetahui teknik pengolahan citra yang tepat dalam ekstraksi fitur warna dan deteksi objek tangan data citra yang berasal dari kamera.
2. Untuk mengetahui penyesuaian data citra dari 2 kamera masukan.
3. Untuk mengetahui hasil metode *Deep Learning Model Multichannel 2D 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 machine learning dalam proses data training dan prediksi gerakan bahasa isyarat.
3. Mampu mengembangkan software untuk memudahkan masyarakat luas dalam memahami bahasa isyarat dengan bantuan machine learning.

## BAB 2. TINJAUAN PUSTAKA

### 2.1 State Of The Art

Tabel 2.1 State Of The Art

	Penelitian 1	Penelitian 2	Penelitian 3	Penelitian 4
Judul	Sistem Pengenalan Bahasa Isyarat Indonesia dengan Menggunakan Metode Fuzzy K-Nearest Neighbor	Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network	Model Penerjemah Bahasa Isyarat Indonesia (BISINDO) Menggunakan Convolutional Neural Network	Sistem Penerjemah Bahasa Isyarat Menggunakan Metode Deep Learning Model Multichannel 2D Convolutional Neural Network (M2D CNN)
Penulis	Agum Agidutama Gafar	Mochamad Bagus Setiyo Bakti dan Yuliana Melita Pranoto	Riestiyya Zain Fadillah	Rizkika Zakka Palindungan
Tahun	2017	2019	2020	2020
Metode	Fuzzy K- Nearest Neighbor (FKNN)	Convolutional Neural Network (CNN)	Convolutional Neural Network (CNN)	Multichannel 2D Convolutional Neural Network (M2D CNN)
Objek	Citra Bahasa Isyarat Tangan (SIBI)	Citra Bahasa Isyarat Tangan (SIBI)	Citra Bahasa Isyarat Tangan (BISINDO)	Citra Bahasa Isyarat Tangan (BISINDO)

## **2.2 Referensi dan Penelitian Yang Sudah Ada**

Konsep penelitian dan pengembangan sistem komputer vision untuk penerjemah bahasa isyarat yang telah dilakukan antara lain :

### **2.2.1 Sistem Pengenalan Bahasa Isyarat Indonesia Dengan Menggunakan Metode Fuzzy K-Nearest Neighbor.**

Proses pengenalan pada Sistem Pengenalan Bahasa Isyarat Indonesia Dengan Menggunakan Metode Fuzzy K-Nearest Neighbor (Gafar, 2017) dibangun menggunakan metode Fuzzy K-Nearest Neighbor (FKNN), metode ini memiliki dua keunggulan utama daripada algoritma K-Nearest Neighbor yaitu mampu mempertimbangkan sifat ambigu dari tetangga jika ada dan membuat sebuah interface akan memiliki derajat nilai keanggotaan pada setiap kelas sehingga akan lebih memberikan kekuatan atau kepercayaan suatu instance yang berada pada suatu kelas. Dengan menerapkan metode Fuzzy K-Nearest Neighbor (FKNN) pada proses klasifikasi bahasa isyarat, maka proses klasifikasi bisa dilakukan dengan lebih objektif. Penelitian ini memilih metode Fuzzy K-Nearest Neighbor (FKNN) untuk pengenalan Sistem Isyarat Bahasa Indonesia (SIBI). Hasil Sistem pengenalan bahasa isyarat Indonesia dengan menggunakan metode Fuzzy K-Nearest Neighbor (KNN) diperoleh nilai akurasi sebesar 88,8%. Dengan melakukan percobaan sebanyak 5 kali, masih ada beberapa huruf yang konsisten belum bisa dikenali, seperti huruf C, E, L, U dan V yang cenderung memiliki nilai kemiripan yang dekat.

### **2.2.2 Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network.**

Dalam Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network (Bakti., 2019) membahas pengenalan isyarat angka SIBI dengan menggunakan metode Convolutional Neural Network (CNN). Arsitektur CNN yang digunakan adalah arsitektur LeNet. Arsitektur CNN diproses dalam 3 tahap, 25 epoch, 50 epoch dan 100 epoch. Berdasarkan percobaan yang dilakukan nilai akurasi yang didapat terus meningkat dalam tiap tahapnya, mulai 67.66%, 89.44% sampai nilai akurasi tertinggi dalam proses training sebesar 96.44%. Begitupun dalam proses prediksi data juga mengalami kenaikan dalam tiap tahapnya, mulai 79.23%, 90.45% sampai

didapatkan nilai akurasi tertinggi dalam prediksi data 98.89%. Dari total 90 data testingset, hanya sekali keasalahan dalam prediksi data.

### 2.2.3 Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network.

Penelitian Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network (Fadillah, 2020) menghasilkan model penerjemah alfabet Bisindo menggunakan implementasi algoritma CNN dengan akurasi 94.38% dan 30% sebagai upaya untuk meningkatkan aksesibilitas Tuli pengguna Bisindo. Model penerjemah tersebut merupakan model terpilih dari dua model yang telah dikembangkan dengan menggunakan pendekatan yang berbeda. Model A dikembangkan menggunakan pemindahan arsitektur dari source model (Model ASL) dan Model B dikembangkan menggunakan pemindahan arsitektur beserta learning parameter yang tersimpan dalam source model tersebut. Dengan percobaan yang dilakukan dalam penelitian ini, pemindahan learning parameter dari Model ASL kepada Model B mengurangi durasi learning yang signifikan namun menyebabkan Model B hanya dapat mempelajari kemiripan fitur antara source dan target domain (huruf yang memiliki kemiripan antara ASL dan Bisindo). Oleh karena itu, pemindahan arsitektur tanpa menggunakan learning parameter merupakan metode pengembangan yang sesuai untuk kasus seperti ini, sehingga parameter-transfer masih belum dapat mengatasi keberagaman gestur ASL dan Bisindo.

## 2.3 Kekurangan Penelitian Sebelumnya

Dari penelitian dan karya tulis yang telah ada sebelumnya masih ditemukan keterbatasan dan kekurangan. Hal tersebut akan dijadikan sebagai acuan untuk melakukan pengembangan dan penelitian yang lebih lanjut. Sehingga menjadi gagasan, ide dan inovasi dalam penelitian baru.

### 2.3.1 Sistem Pengenalan Bahasa Isyarat Indonesia Dengan Menggunakan Metode Fuzzy K-Nearest Neighbor.

Dalam penelitian ini ditemukan beberapa kekurangan yang bisa dikembangkan untuk penelitian selanjutnya, yaitu sebagai berikut :

- a. Tingkat akurasi dalam melakukan prediksi masih rendah meskipun bahasa isyarat yang dipakai sederhana.
- b. Tidak ada objek deteksi dalam proses pengolahan citra.
- c. Metode ekstraksi fitur sangat simple dan sedikit sehingga data/informasi yang diambil dari citra tidak efisien dan relevan sebagai parameter dalam proses klasifikasi.
- d. Input citranya masih dalam bentuk file gambar / manual tidak secara *realtime* dengan menggunakan video kamera

### 2.3.2 Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network.

Dalam penelitian ini ditemukan beberapa kekurangan yang bisa dikembangkan untuk penelitian selanjutnya, yaitu sebagai berikut :

- a. Bahasa isyarat yang digunakan masih sederhana yaitu hanya bahasa isyarat angka dengan menggunakan satu tangan.
- b. Sudut pandang kamera hanya satu yaitu dari depan objek / model peraga.
- c. Input citranya masih dalam bentuk file gambar / manual tidak secara *realtime* dengan menggunakan video kamera

### 2.3.3 Model Penerjemah Bahasa Isyarat Indonesia (Bisindo) Menggunakan Convolutional Neural Network.

Dalam penelitian ini ditemukan beberapa kekurangan yang bisa dikembangkan untuk penelitian selanjutnya, yaitu sebagai berikut :

- a. Metode ekstraksi fitur sangat simple dan sedikit serta input citra hanya dalam bentuk gambar tangan tidak menggunakan gambar model *full body*.
- b. Input citranya masih dalam bentuk file gambar / manual tidak secara *realtime* dengan menggunakan video kamera.

## 2.4 Kelebihan Dari Penelitian Yang Diambil

Dalam proses pengembangan dan penelitian yang lebih lanjut. Peneliti akan melakukan upaya untuk memperbaiki kekurangan dari penelitian-penelitian sebelumnya. Adapun upaya tersebut dapat dijabarkan sebagai berikut :

- a. Input citra akan ditingkatkan dari pada penelitian sebelumnya yaitu dengan menggunakan 2 buah kamera dan data yang diolah dalam bentuk video yang diambil secara *realtime*.
- b. Proses pengolahan citra lebih kompleks dengan adanya deteksi objek untuk menghilangkan area wajah dan leher serta ekstraksi warna kulit untuk memfokuskan dalam pengambilan objek tangan.
- c. Karena data input berasal dari 2 kamera yang berbeda (sudut pandang) maka algoritma yang akan digunakan pada penelitian ini menggunakan model pengembangan dari CNN 2D yaitu model arsitektur *Multichannel 2D Convolutional Neural Network*, berbeda dengan CNN 2D biasa algoritma ini memiliki kemampuan dan akurasi untuk mengolah 2 atau lebih data input citra dari sebuah objek dengan sudut pandang yang berbeda.

## 2.5 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).

#### 2.5.1 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.

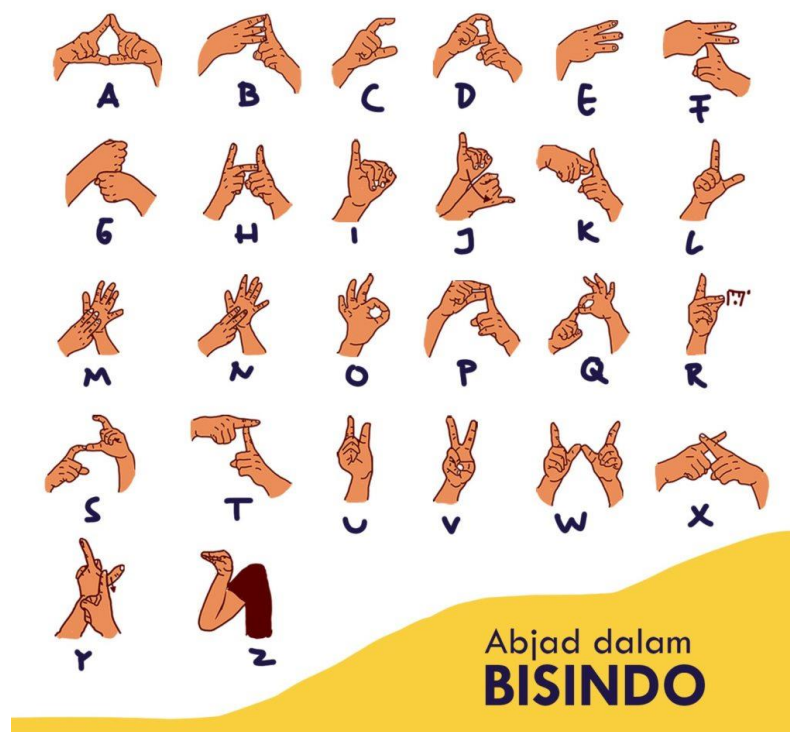
Disebarluaskan dan dibakukannya penggunaan SIBI sebagai sistem isyarat buatan yang bersifat nasional tidak sepenuhnya diterima oleh komunitas Tuli. Komunitas Tuli menilai bahwa keberadaan SIBI tidak merepresentasikan bahasa

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.

#### 2.5.2 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 Indonesia (BISINDO)

## 2.6 Pengolahan Citra Digital

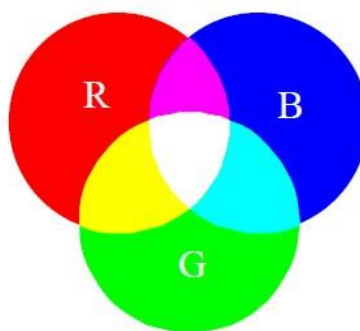
Pengolahan Citra Digital adalah merupakan proses yang bertujuan untuk memanipulasi dan menganalisis citra dengan bantuan komputer. Baik citra yang berdimensi 2 atau citra 3 dimensi.

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.

### 2.6.1 RGB Color Spaces

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 berbasis elektronik seperti Camera, Komputer, Televisi dan berbagai gadget lainnya.



Gambar 2.2 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 :

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

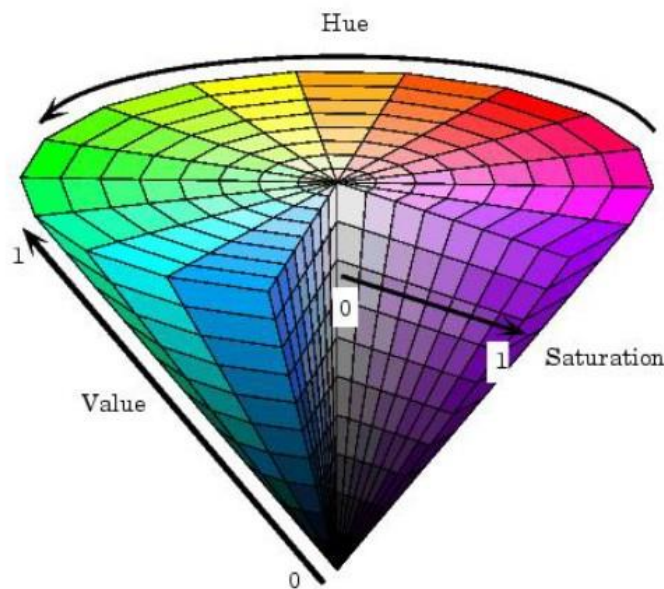
$$g = \frac{G}{R+G+B} \quad 2.2$$

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

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.

#### 2.6.2 HSV Color Space

HSV merupakan singkatan dari Hue Saturation Value, yaitu pendekatan perspektif pandangan perspektif warna yang mampu ditangkap oleh panca indra manusia. Hal ini dikemukakan oleh Y. Ming (1988) manusia warna yang mendekati dengan mata manusia.



Gambar 2.3 Warna HSV

Berdasarkan gambar di atas, dijelaskan bahwa nilai Hue adalah nilai rotasi atau nilai sudut putaran yang diberikan nilai dari 0° sampai dengan 360°. Sedangkan

Saturation adalah merupakan nilai Intensitas yang bernilai 0 – 1.0 yang merupakan bilangan real.

Hue merupakan warna sebenarnya yang benar benar mendekati persepsi mata manusia seperti merah, violet, kuning, warna yang kemerahan, warna kehijauan, warna kebiruan dan seterusnya dan Saturation atau terkadang disebut juga dengan Chroma yaitu warna kekuatan, kemurnian atau kepekatan dari sebuah warna. Dan Value digunakan untuk menentukan tingkat kecerahan dari sebuah warna yang bernilai persentasi antara 0 sampai dengan 100%, dimana warna dengan Value 0 merupakan hitam dan 100% adalah putih.

Formula yang digunakan untuk mengkonversi nilai RGB ke HSV menggunakan rumus berikut ini :

$$V = \frac{1}{3} (R + G + B) \quad 2.4$$

$$S = 1 - \frac{3}{(R+G+B)} [\min(R + G + B)] \quad 2.5$$

$$H = \begin{cases} 0, & B \leq G \\ 360 - 0, & B > G \end{cases} \quad 2.6$$

$$\vartheta = \cos^{-1} \left\{ \frac{1/2[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}} \right\} \quad 2.7$$

### 2.6.3 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 2.4 Warna Grayscale

Nilai warna untuk Grayscale (derajat keabuan) adalah nilai warna yang menunjukkan kehitaman dengan nilai  $2^8 - 1 = 255$  untuk image 8 bit.

### 2.6.4 Deteksi Warna Kulit

Penelitian yang dilakukan oleh Putra Pandu Adikara, Muh, Arif Rahman dan Edi Santosa (2014), bahwa telah menjelaskan segmentasi warna kulit dalam

mode warna HSV sebagaimana yang didefinisikan oleh Yanjiang & Baozang. Yanjiang mendefinisikan warna kulit manusia adalah menggunakan rumusan :

$$\text{Nilai Hue} = 0 \leq H \leq 50$$

$$\text{Nilai Saturation} = 0.20 \leq S \leq 0.68$$

$$\text{Nilai Value} = 0.35 \leq V \leq 1.0$$

Berdasarkan nilai tersebut, setelah melakukan iterasi konversi warna dari RGB ke HSV maka selanjutnya adalah melakukan pengecekan terhadap nilai warna HSV tersebut di atas. Dimana pengecekan akan memberikan hasil warna kulit dan bukan kulit dari manusia.

Proses yang dilakukan sama dengan pengambilan nilai – nilai sebelumnya dan menggunakan operator logika AND dimana ketiga syarat nilai Hue, Saturation dan Value nya terpenuhi.

## 2.7 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 ( $\theta$ ) yang mempunyai parameter  $\theta$  dengan memperbarui parameter ke suatu arah yang menurun. *Gradient descent* mempunyai *learning rate* ( $\eta$ ) 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 :

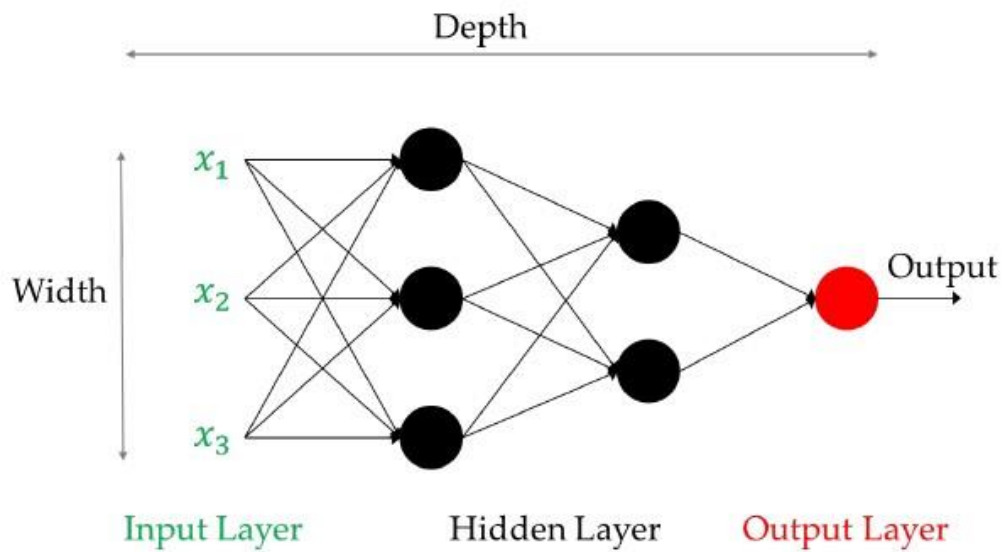
$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; x^i; y^i) \quad 2.8$$

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 menurun *loss function* ke titik minimum dengan optimal.

## 2.8 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 sebenarnya 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 2.5 (*deep neural network*, salah satu varian arsitektur). *Depth* (kedalaman) ANN mengacu pada jumlah layer. Sementara *width* (lebar) ANN mengacu pada jumlah unit pada *layer*.

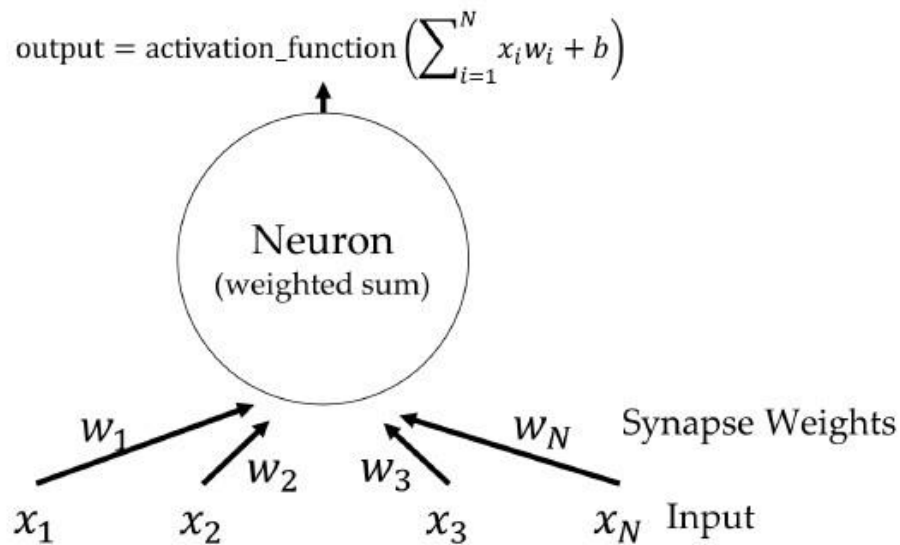


Gambar 2.5 Deep Neural Network

### 2.8.1 Single Perceptron

Bentuk terkecil (minimal) sebuah ANN adalah *single perceptron* yang hanya terdiri dari sebuah neuron. Sebuah neuron diilustrasikan pada Gambar 2.6. 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 2.6 Single Perceptron

$$\sigma(u) = \frac{1}{1+e^{-u}} \quad 2.9$$

Perhatikan kembali, Gambar 2.6 sesungguhnya adalah operasi aljabar linear. Single perceptron dapat dituliskan kembali sebagai berikut :

$$o = f(x \cdot w + b) \quad 2.10$$

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 2.11 (dapat diganti dengan absolute value) dan perubahan *synapse weight* diberikan pada persamaan 2.12, dimana  $y$  melambangkan *desired output*,  $o = f(x \cdot w + b)$  melambangkan actual output untuk  $x$  sebagai input.  $\mu$  disebut sebagai learning rate.

$$E(w) = (y - o)^2 \quad 2.11$$

$$\Delta w_i = \mu(y - o)x_1 \quad 2.12$$

Hasil akhir pembelajaran (*learning*) adalah konfigurasi *synapse weight*.

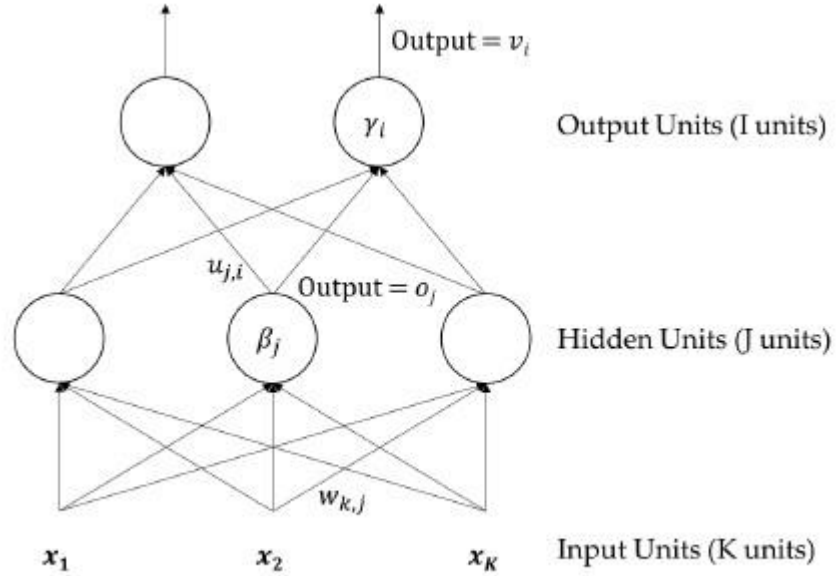
Saat klasifikasi, kita melewati *input* baru pada jaringan yang telah dibangun, kemudian tinggal mengambil hasilnya. Pada contoh kali ini, seolah-olah *single perceptron* hanya dapat digunakan untuk melakukan *binary classification* (hanya ada dua kelas, nilai 0 dan 1). Untuk multi-class classification, 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.

### 2.8.2 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 2.7 Multilayer Perceptron

$$o_j = \sigma\left(\sum_{k=1}^K x_k w_{k,j} + \beta_j\right) \quad 2.13$$

$$v_i = \sigma\left(\sum_{j=1}^J o_j u_{j,i} + \gamma_i\right) = \sigma\left(\sum_{j=1}^J \sigma\left(\sum_{k=1}^K x_k w_{k,j} + \beta_j\right) u_{j,i} + \gamma_i\right) \quad 2.14$$

Perhatikan persamaan 2.13 dan 2.14 untuk menghitung *output* pada *layer* yang berbeda.  $u, w$  adalah *learning parameters*.  $\beta, \gamma$  melambangkan *noise* atau *bias*.  $K$  adalah banyaknya *input units* dan  $J$  adalah banyaknya *hidden units*.

Persamaan 2.14 dapat disederhanakan penulisannya sebagai persamaan 2.15. Persamaan 2.15 terlihat relatif lebih “elegant”. Sehingga ANN dapat direpresentasikan dengan notasi operasi aljabar.

$$V = \sigma(oU + \gamma) = \sigma(\sigma(xU + \beta)U + \gamma) \quad 2.15$$

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 2.7 yang menggunakan fungsi aktivasi sigmoid.

*Error*, untuk MLP diberikan oleh persamaan 2.16 (untuk satu data point), dimana  $I$  adalah banyaknya *output unit* dan  $\theta$  adalah kumpulan *weight matrices* (semua *parameter* pada MLP). Kami ingatkan kembali perhitungan *error* bisa juga menggunakan nilai absolut.

$$E(\theta) = \frac{1}{2} \sum_{i=1}^I (y_i - v_i)^2 \quad 2.16$$

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

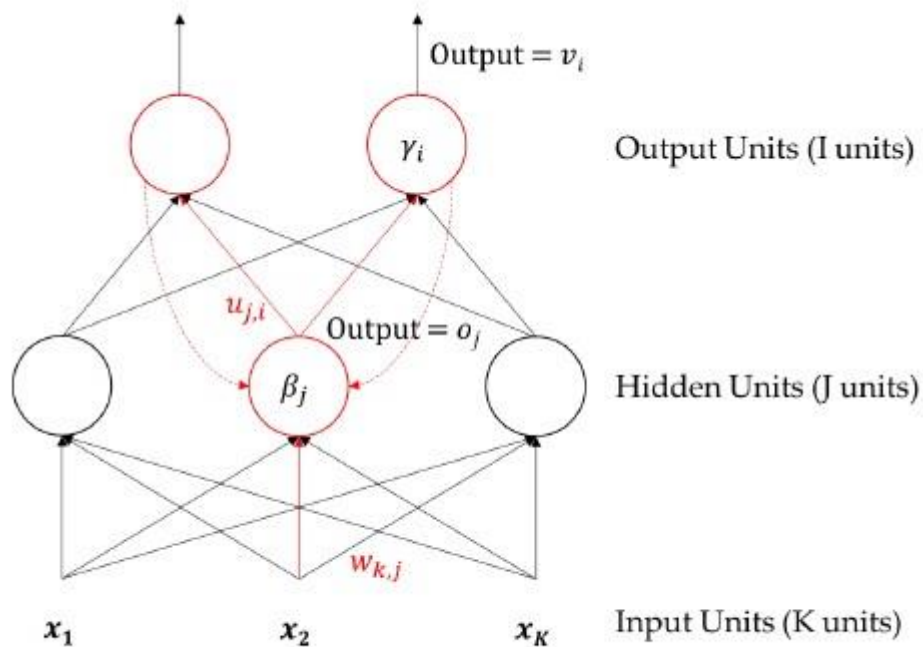
$$\begin{aligned} \frac{\delta E(\theta)}{\delta u_{j,i}} &= (y_i - v_i) \frac{\delta v_i}{\delta u_{j,i}} \\ &= (y_i - v_i) v_i (1 - v_i) o_j \end{aligned} \quad 2.17$$

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

$$\begin{aligned} \frac{\delta E(\theta)}{\delta w_{k,j}} &= \sum_{i=1}^I (y_i - v_i) \frac{\delta v_i}{\delta w_{k,j}} \\ &= \sum_{i=1}^I (y_i - v_i) \frac{\delta v_i}{\delta o_j} \frac{\delta o_j}{\delta w_{k,j}} \\ &= \sum_{i=1}^I (y_i - v_i) (v_i (1 - v_i) u_{j,i}) (o_j (1 - o_j) x_k) \end{aligned} \quad 2.18$$

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 2.8.



Gambar 2.8 Multilayer Perceptron 2

Metode penurunan serupa dapat juga digunakan untuk menentukan perubahan  $\beta$  dan  $\gamma$ . Jadi proses *backpropagation* untuk kasus Gambar 2.7 dapat diberikan seperti pada Gambar 2.9 dimana  $\eta$  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**

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

**(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 2.9 Proses latihan menggunakan backpropagation

**2.8.3 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 2.10). 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.

$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	Kelas pertama
	Kelas kedua
	Kelas ketiga
	Kelas keempat

Gambar 2.10 Ilustrasi representasi desired output pada multi-class classification

#### 2.8.4 Multi-label Classification

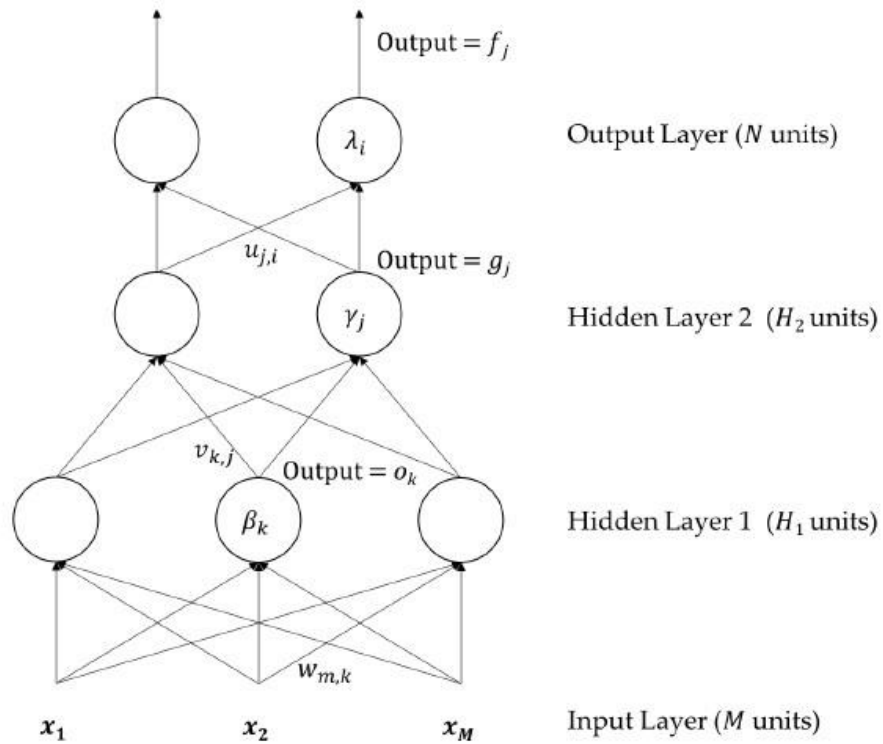
Seperti halnya *multi-class classification*, kita dapat menggunakan sejumlah  $C$  neuron untuk merepresentasikan  $C$  kelas pada *multi-label classification*. Perbedaan *multi-class* dan *multilabel* terletak pada cara interpretasi *output* dan evaluasi *output*. Pada umumnya, *layer* terakhir diaktivasi 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*.

$c_1$	$c_2$	$c_3$	$c_4$	
1	0	1	0	Label = $c_1, c_3$
0	1	0	0	Label = $c_2$
1	0	0	1	Label = $c_1, c_4$
0	1	1	1	Label = $c_2, c_3, c_4$

Gambar 2.11 Ilustrasi representasi desired output pada multi-label classification

#### 2.8.5 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*<sup>11</sup>. Jaringan *neural network* pada DNN disebut *deep neural network*. Perhatikan Gambar 2.12 yang memiliki 4 *layers*.



Gambar 2.12 Deep Neural Network

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

$$f_i = (\sum_{j=1}^{H_2} u_{j,i} \sigma(\sum_{k=1}^{H_1} u_{k,j} \sigma(\sum_{m=1}^M x_m w_{m,k} + \beta_k) + \gamma_j) + \lambda_i) \quad 2.19$$

Cara melatih *deep neural network*, salah satunya dapat menggunakan backpropagation. Hanya perlu menurunkan rumusnya saja. Hasil proses penurunan dapat dilihat pada Gambar 2.13.



**(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**

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

**(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**

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

**(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**

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

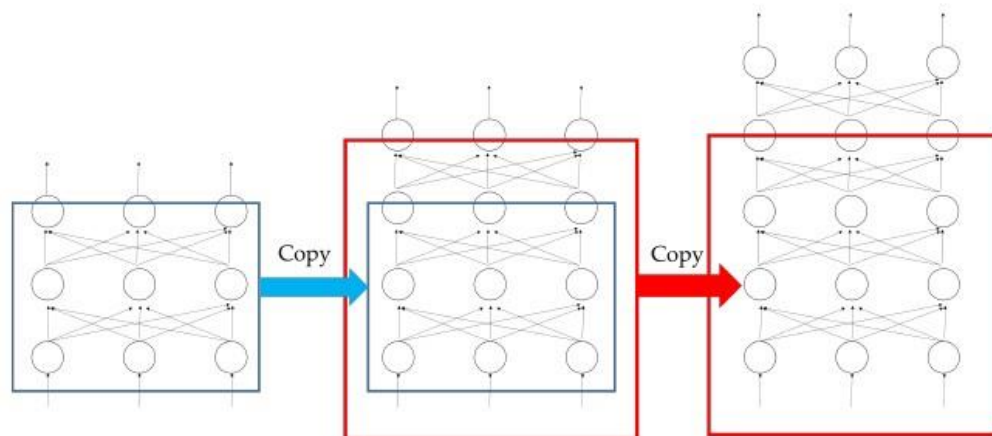
Gambar 2.13 Proses latihan DNN menggunakan backpropagation

*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 mensimulasikan 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 2.14. 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 2.14 Contoh successive learning

#### 2.8.6 Regularization and Dropout

Pada model linear. Model harus mampu menggeneralisasi 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 2.20 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 2.21, dimana  $p$  adalah vektor dan  $p_i = [0,1]$  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 2.22. 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.

$$o = f(x \cdot W + b) \quad 2.20$$

$$w' = p \cdot W \quad 2.21$$

$$o = f(x \cdot W' + b) \quad 2.22$$

## 2.9 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.

### 2.9.1 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.

$$x^l = f(x^{l-1}) \quad 2.23$$

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.

$$\sigma(u) = \frac{1}{1+exp^{-u}} \quad 2.24$$

#### **TanH**

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

$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad 2.25$$

#### **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.

$$f(x) = \max(0, x) \quad 2.26$$

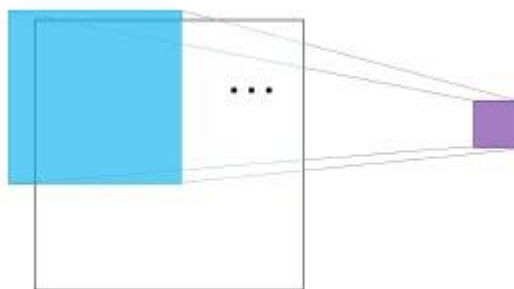
Pada CNN, layer output pada umumnya merupakan *feature maps* dimensi dua. Pada persamaan 2.26 ,  $x$  adalah matrix dan  $\max$  adalah elemen *twice* yang diterapkan pada setiap elemen matriks. Varian lain dari ReLU yaitu LeakyReLU memungkinkan adanya nilai yang kecil, bukan nol meski unit sedang tidak dalam keadaan aktif. Leaky ReLU didefinisikan sebagai berikut.

$$\begin{cases} x, & x > 0 \\ 0.01x & \text{selain } x > 0 \end{cases} \quad 2.27$$

### 2.9.2 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 2.15. 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  $d \times T$ . Keseluruhan operasi ini disebut sebagai *convolution*.

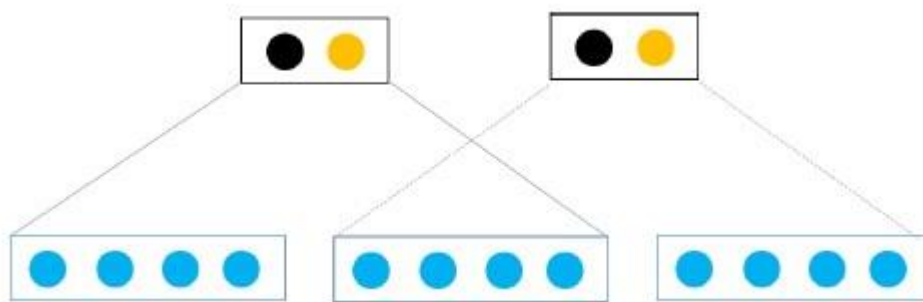


Gambar 2.15 Sliding window

Pada Gambar 2.16 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

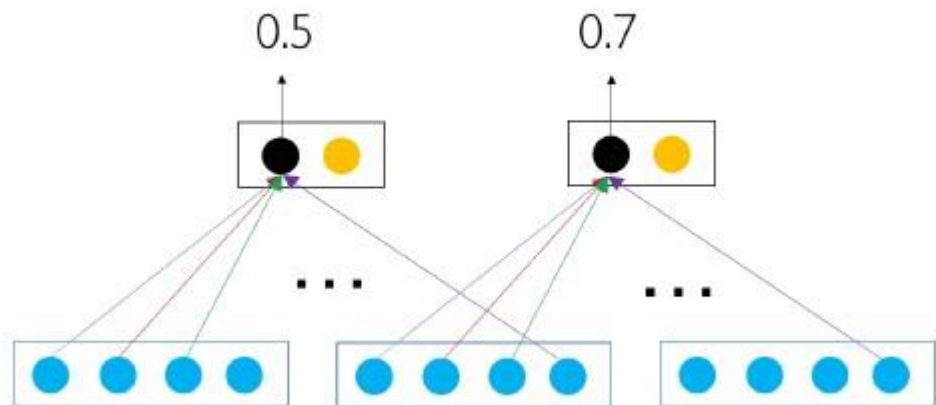
digeser untuk *window* baru. Contoh yang diberikan memiliki *stride* sebesar satu. Apabila memiliki *stride*= 2, maka akan menggeser sebanyak 2 data setiap langkah, i.e.,  $window_1 = (X_1, X_2)$ ,  $window_2 = (X_3, X_4)$ , ... .

Berikut contoh dalam bentuk 1-D pada Pada Gambar 2.16. 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 2.16 1D Convolution

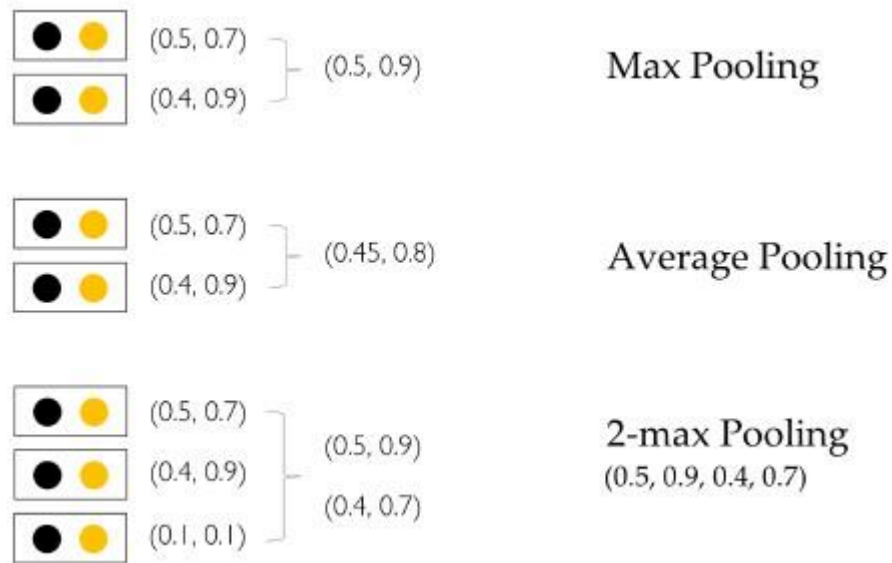
Selain *sliding window* dan *filter*, *convolutional layer* juga mengadopsi prinsip *weight sharing*. Artinya, *synapse weights* untuk suatu filter adalah sama walau filter tersebut dipergunakan untuk berbagai *window*. Sebagai ilustrasi, perhatikan Gambar 2.17, warna yang sama pada *synapse weights* 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* berwarna 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 konfigurasi 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)*.



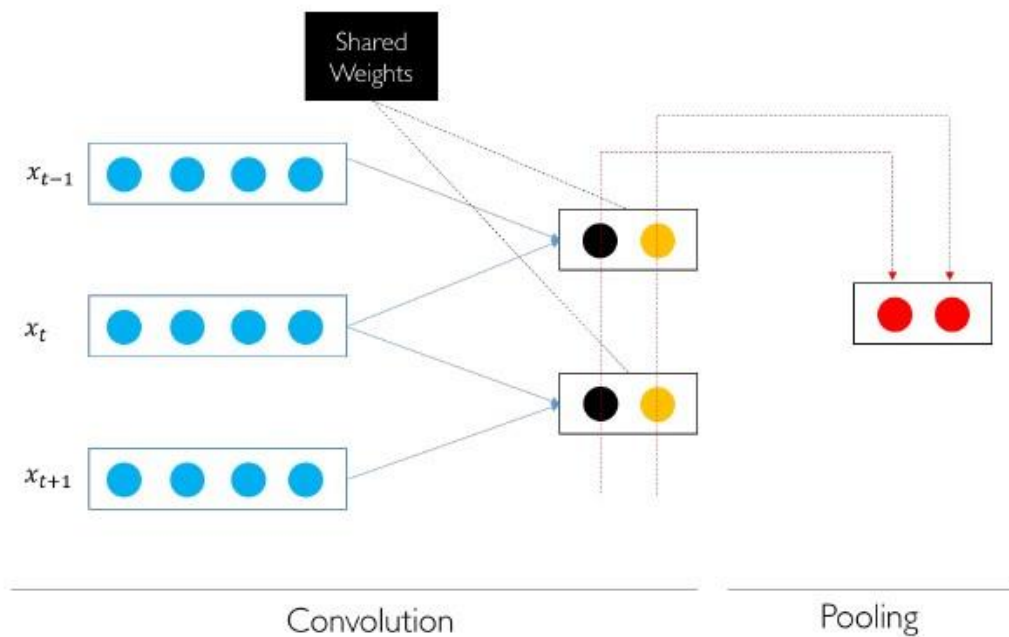
Gambar 2.17 Konsep weight sharing

### 2.9.3 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 2.18. *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 2.19.



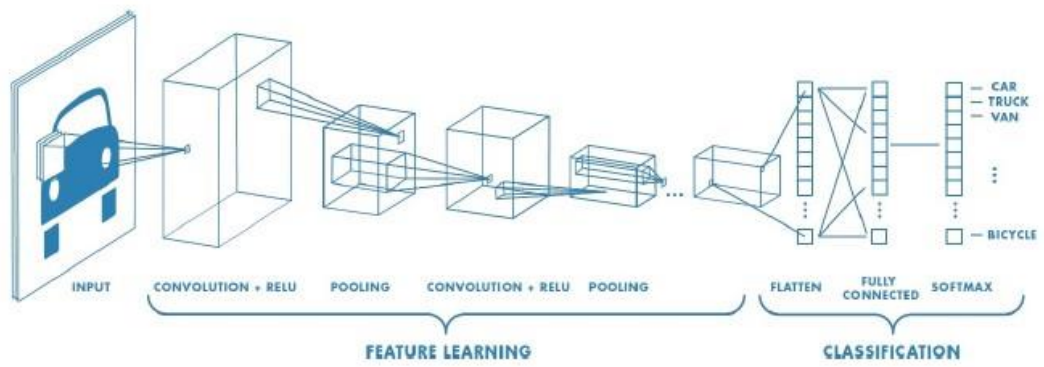
Gambar 2.18 Contoh pooling



Gambar 2.19 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 2.20).





Gambar 2.20 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 sebagai berikut :

1. Acer Aspire E5-476G-58V

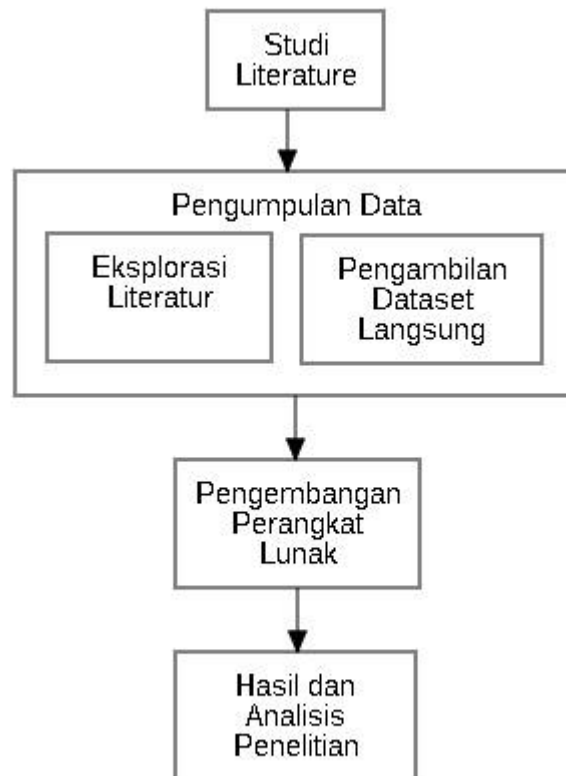
- a) Screen Size 14 inch
    - b) Processor Intel(R) Core™ i5-8250 1.6 GHz with Turbo Boost up to 3.4 GHz
    - c) NVIDIA (R) Geforce(R) MX130 with 2 GB VRAM
    - d) 8 GB DDR4 Memory
    - e) 1000 GB HDD
  - 2. PC
  - 3. Webcam sebanyak 2 buah
    - a) Internal HDR webcam Acer Aspire E5-476G-58V
    - 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
  - 5. TensorFlow 2 with Keras
  - 6. PyQt5

### 3.2.2 Bahan Penelitian

Data sampel (training dan validasi) citra Tangan Bahasa Isyarat Indonesia (BISINDO).

### 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

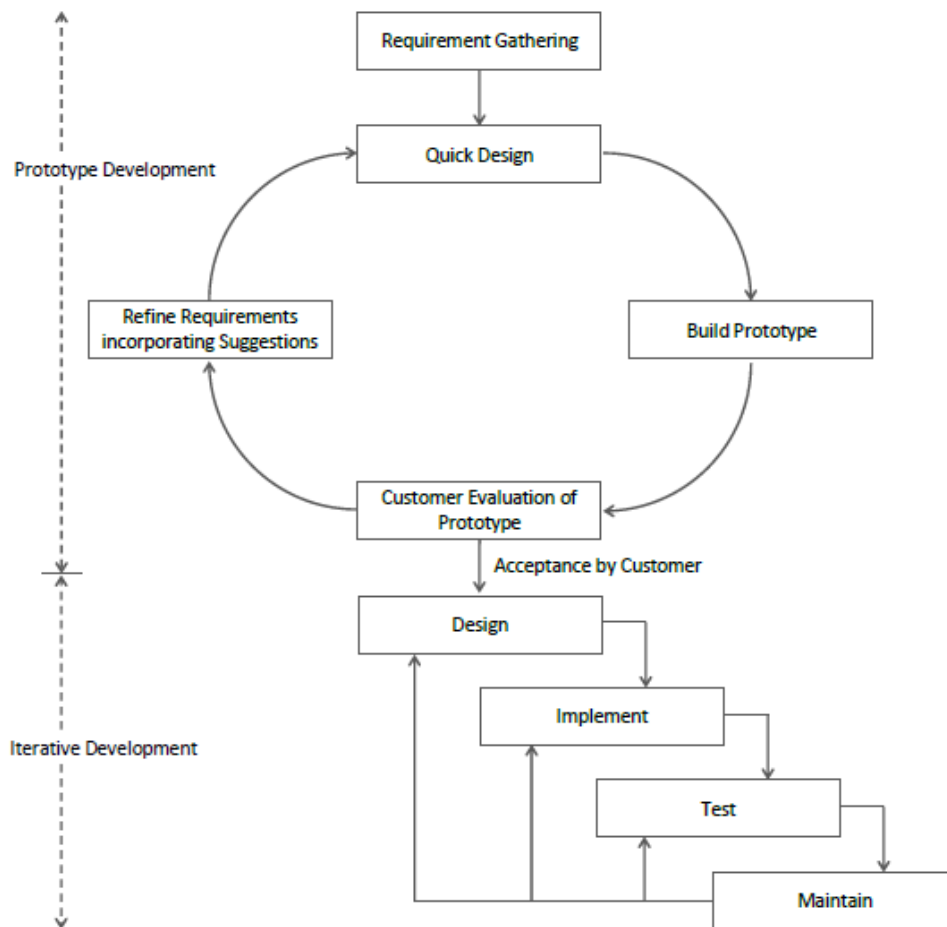
Untuk menghasilkan output yang akurat dalam mengembangkan suatu sistem penerjemah bahasa isyarat. Peneliti membaca, mempelajari dan membandingkan literatur yang memuat model, struktur data dan metode yang diterapkan pada sistem penerjemah bahasa isyarat yang terdahulu.

2. Pengambilan Dataset Langsung

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

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, mengidentifikasi 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 prototyping

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 disepakati 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 eksternal dan mencari segala kemungkinan kesalahan dan memeriksa apakah sesuai dengan 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.

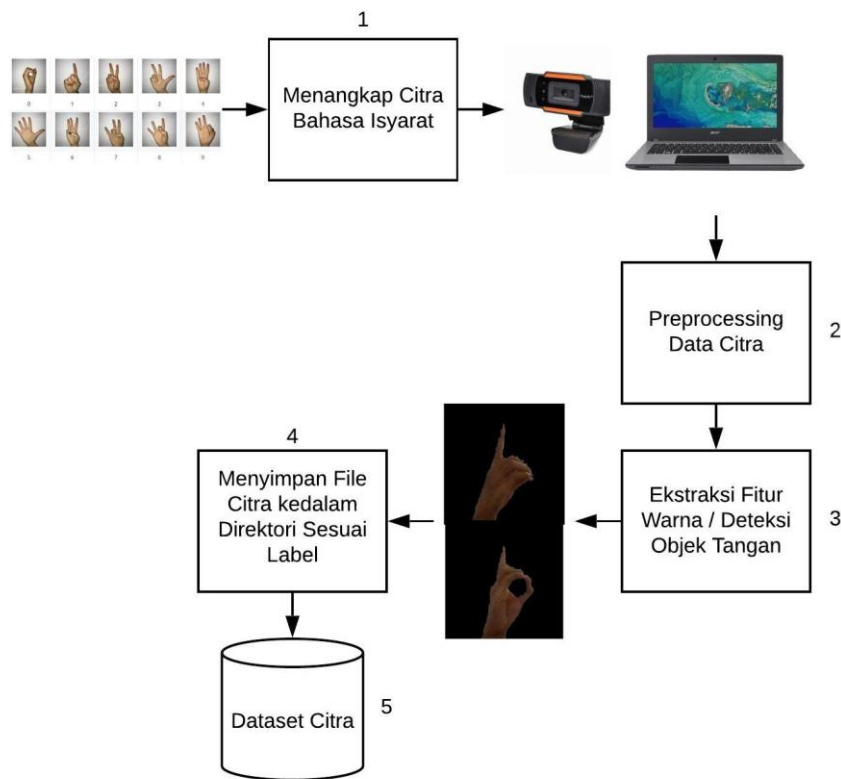
### **3.4 Jenis Data**

#### **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 alir (flowchart). Berikut ini digambarkan mengenai diagram alir sistem pada Gambar 3.3, 3.4, dan 3.5.



Gambar 3.3 Block diagram proses proses mendapatkan dataset

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 2 kamera yang dipakai yaitu kamera webcam internal (build in) laptop dan kamera webcam eksternal. Posisi kamera berada didepan dan disamping kanan user.

b. Tahap Kedua

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 mengurangi jumlah piksel dari citra inputan

3. Konversi Warna Dari BGR ke HSV

HSV merupakan singkatan dari Hue Saturation Value, yaitu pendekatan perspektif pandangan perspektif warna yang mampu ditangkap oleh panca indra manusia.

c. Tahap Ketiga

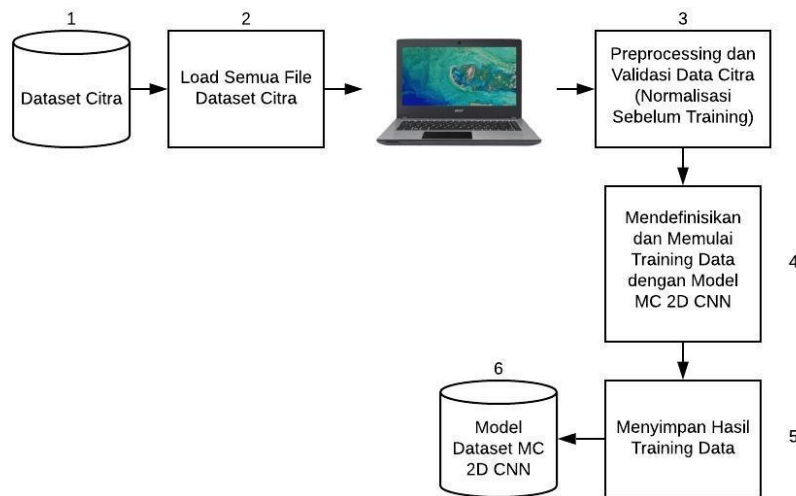
Melakukan proses deteksi objek wajah dan leher untuk dihilangkan dari *region of interest* dengan bantuan *Haar Cascade Classifier*. Selanjutnya melakukan deteksi warna kulit (Skin Mask) untuk mendeteksi objek tangan. Skin Mask adalah segmentasi warna kulit dalam mode warna HSV.

d. Tahap Keempat

Menyimpan file gambar hasil proses pengolahan citra kedalam folder yang bernama sesuai dengan label (kelas/kelompok *gesture*) yang diberikan dan sesuai dengan tipe kamera input (kamera 1 atau 2).

e. Tahap Kelima

Dataset merupakan kumpulan data citra yang akan digunakan untuk proses data training dalam pembuatan data model deep learning (M2D CNN)



Gambar 3.4 Block diagram proses data latih (training)

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

a. Tahap Pertama

Dataset merupakan kumpulan data citra yang akan digunakan untuk proses data training dalam pembuatan data model deep learning (M2D CNN)

b. Tahap Kedua

Memanggil semua file yang ada dalam direktori dataset untuk kemudian dilakukan proses komputasi oleh komputer.

c. Tahap Ketiga

Tahap ini bertujuan untuk menyesuaikan file citra terhadap lingkungan komputasi didalam Tensorflow dan Keras. Image processing ini adalah proses dimana citra yang telah masuk ke sistem akan diproses melalui beberapa tahap, meliputi :

1. Resize

Proses resize citra digunakan untuk mengurangi jumlah piksel dari citra inputan

2. Konversi Warna Dari BGR ke 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.

d. Tahap Keempat

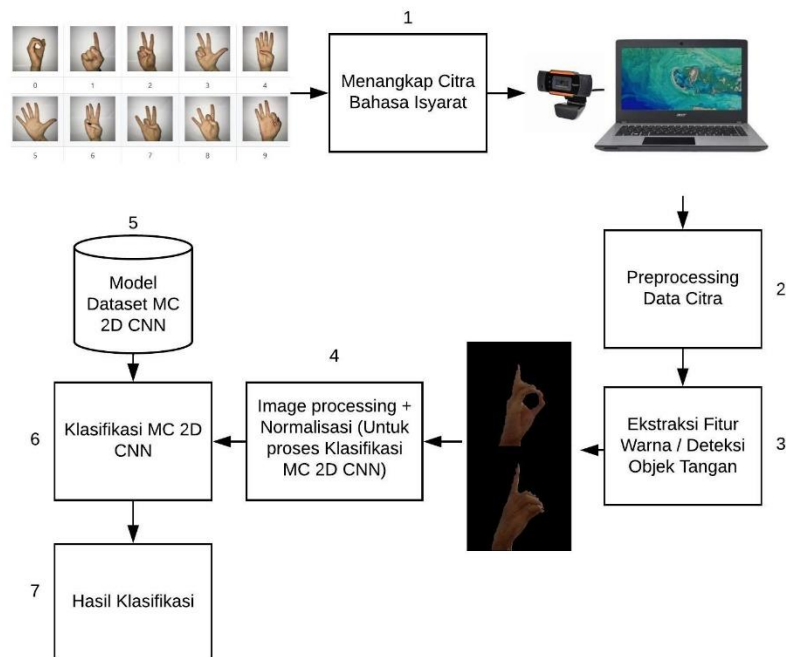
Melakukan Proses Training data dengan menggunakan metode algoritma M2D CNN (Multichannel 2D Convolutional Neural Network).

e. Tahap Kelima

Menyimpan hasil proses training data dengan format file tertentu (modelSignLanguageV1.h5).

f. Tahap Enam

File model yang sudah berhasil dibuat selanjutnya bisa digunakan oleh machine 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 M2D CNN memiliki beberapa tahapan, seperti berikut :

a. Tahap Pertama

Kamera webcam dan laptop menangkap *gesture* Bahasa Isyarat Indonesia yang diperagakan oleh peneliti. Terdapat 2 kamera yang dipakai yaitu kamera webcam internal (build in) laptop dan kamera webcam eksternal. Posisi kamera berada di depan dan disamping kanan user.

b. Tahap Kedua

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 mengurangi jumlah piksel dari citra inputan

3. Konversi Warna Dari BGR ke HSV

HSV merupakan singkatan dari Hue Saturation Value, yaitu pendekatan perspektif pandangan perspektif warna yang mampu ditangkap oleh panca indra manusia.

c. Tahap Ketiga

Melakukan proses deteksi objek wajah dan leher untuk dihilangkan dari *region of interest* dengan bantuan *Haar Cascade Classifier*. Selanjutnya melakukan deteksi warna kulit (Skin Mask) untuk mendeteksi objek tangan. Skin Mask adalah segmentasi warna kulit dalam mode warna HSV.

d. Tahap Keempat

Tahap ini bertujuan untuk menyesuaikan file citra terhadap lingkungan komputasi didalam Tensorflow dan Keras. Image processing ini adalah proses dimana citra yang telah masuk ke sistem akan diproses melalui beberapa tahap, meliputi :

1. Resize

Proses resize citra digunakan untuk mengurangi jumlah piksel dari citra inputan

2. Konversi Warna Dari BGR ke 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.

e. Tahap Kelima

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

f. Tahap Keenam

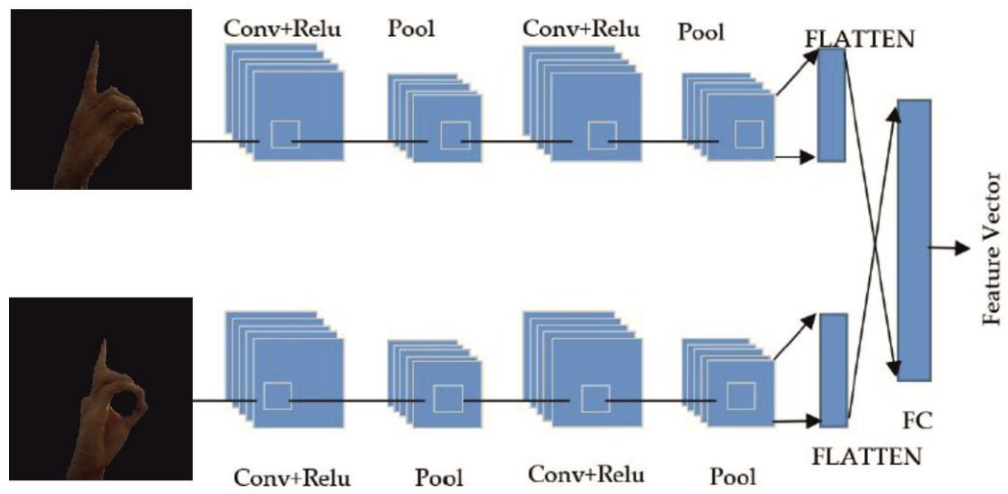
Inisialisasi proses prediksi oleh deep learning M2D CNN terhadap citra input dan model dataset untuk mengklasifikasikan citra dan menerjemahkannya kedalam bentuk tulisan.

g. Tahap Ketujuh

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

**M2D CNN Model Architecture.**

Seperti disebutkan di atas, citra *gesture* dilihat dari dua sisi yaitu sisi depan dan sisi samping. Peneliti menggunakan dua CNN 2D dan masing-masing kemudian digabungkan (dua arsitektur CNN 2D) secara paralel menjadi satu buah CNN multichannel 2D (M2D CNN). Setiap CNN 2D menggunakan satu jenis gambar sesuai dengan tipe kamera sebagai input dan melakukan komputasi konvolusi secara mandiri. Output dari kedua bagian CNN 2D diratakan dan digabungkan menjadi fitur 1D secara seri, yang merupakan input ke jaringan saraf yang terhubung penuh untuk pembelajaran lebih lanjut. Akhirnya, CNN M2D mengeluarkan hasil klasifikasi. Arsitektur model M2D CNN ditunjukkan pada Gambar 3.6.



Gambar 3.6 Model arsitektur Multichannel 2D Convolutional Neural Network

## 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](http://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