

**DETEKSI GAMBAR GESTUR KOSAKATA BAHASA ISYARAT  
INDONESIA DENGAN *CONVOLUTIONAL NEURAL NETWORK***

**SKRIPSI**

Diajukan Sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana Strata Satu

Program Studi Sistem Informasi



**PROGRAM STUDI SISTEM INFORMASI  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH  
JAKARTA  
2020 M / 1441 H**



**DETEKSI GAMBAR GESTUR KOSAKATA BAHASA ISYARAT  
INDONESIA DENGAN *CONVOLUTIONAL NEURAL NETWORK***

**SKRIPSI**

Diajukan Sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana Strata Satu



**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH**

**JAKARTA**

**2020 M / 1441 H**

## **HALAMAN JUDUL**

# **DETEKSI GAMBAR GESTUR KOSAKATA BAHASA ISYARAT INDONESIA DENGAN *CONVOLUTIONAL NEURAL NETWORK***



**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS SAINS DAN TEKNOLOGI**

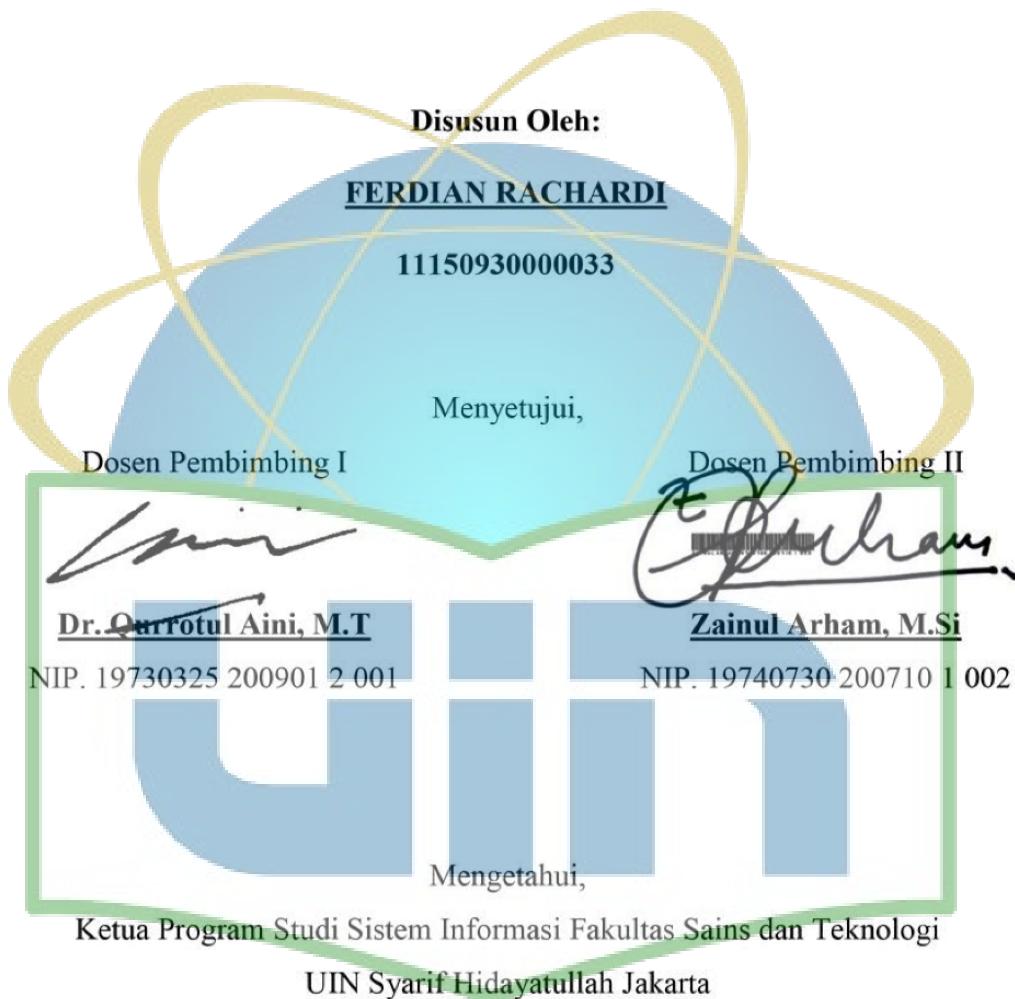
**UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH**

**JAKARTA**

**2020 M / 1441 H**

## LEMBAR PENGESAHAN SKRIPSI

### DETEKSI GAMBAR GESTUR KOSAKATA BAHASA ISYARAT INDONESIA DENGAN *CONVOLUTIONAL NEURAL NETWORK*



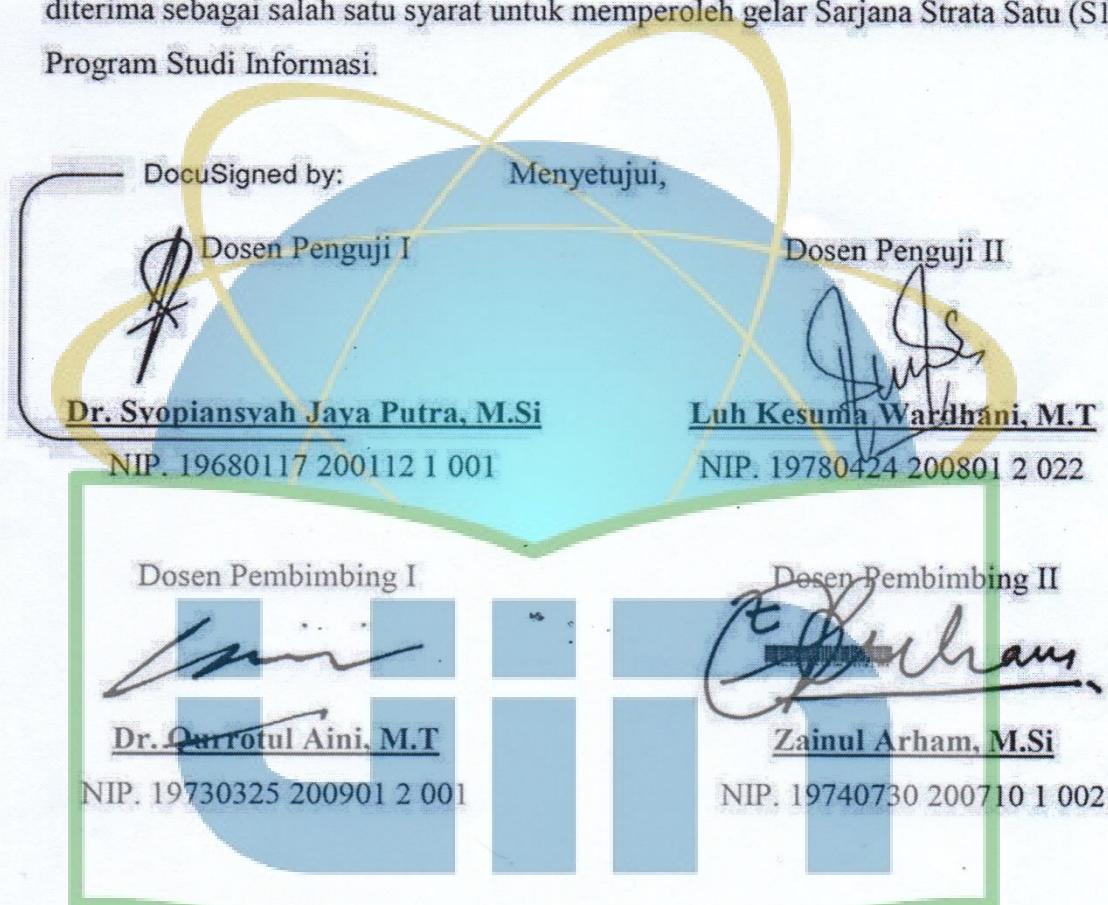
*[Signature]*

**Aang Subiyakto, PhD**

NIP. 19760219 200710 1 002

## PENGESAHAN UJIAN

Skripsi yang berjudul **Deteksi Gambar Gestur Kosakata Bahasa Isyarat Indonesia dengan Convolutional Neural Network** telah diuji dan dinyatakan lulus dalam sidang munaqosyah Fakultas Sains dan teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta pada hari Senin tanggal 5 Oktober 2020. Skripsi telah diterima sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu (S1) Program Studi Informasi.



KEMENTERIAN AGAMA  
FAKULTAS SAINS DAN TEKNOLOGI  
UIN SYARIF HIDAYATULLAH  
JAKARTA

**Prof. Dr. Lily Surraya Eka Putri,  
M.Env.Stud.**  
NIP. 19690404 200501 2 005

Ketua Prodi Sistem Informasi  
**Aang Subiyakto, PhD**  
NIP. 19760219 200710 1 002

## PERNYATAAN

DENGAN INI SAYA MENYATAKAN BAHWA SKRIPSI INI BENAR-BENAR HASIL KARYA SENDIRI DAN BELUM PERNAH DIAJUKAN SEBAGAI SKRIPSI ATAU KARYA ILMIAH PADA PERGURUAN TINGGI ATAU LEMBAGA MANAPUN.





## ABSTRAK

**Ferdian Rachardi – 11150930000033**, Deteksi Gambar Gestur Kosakata Bahasa Isyarat Indonesia dengan *Convolutional Neural Network* di bawah bimbingan **Qurrotul Aini dan Zainul Arham**.

Bahasa Isyarat Indonesia atau BISINDO adalah salah satu bahasa alternatif yang digunakan oleh penyandang disabilitas dan tumbuh berasal dari kaum tuli sehingga penggunaanya berbasis visual. Namun, terdapat ratusan ribu kosakata bahasa Indonesia yang dapat diwakili dengan gestur bahasa isyarat, ditambah jumlah populasi penyandang tunarungu di Indonesia hanya berjumlah 6.952.797 dari 207.839.035 orang, sehingga mengakibatkan bahasa isyarat menjadi sesuatu yang asing dan sulit mengartikan bagi sebagian orang yang normal atau awam. Tujuan penelitian ini untuk membantu orang awam dalam mengklasifikasi dan mendeteksi gestur isyarat pada kosakata bahasa isyarat secara langsung berbasis *mobile*. Untuk dapat mengenali variasi gestur isyarat maka dibutuhkan teknik pembelajaran klasifikasi seperti *machine learning* dengan teknik *supervised learning*. Pengembangan dari penelitian ini menggunakan metode *convolutional neural network* dengan bantuan teknik dari arsitektur *single shot detector* sebagai objek deteksi dan arsitektur *mobilenet* untuk klasifikasi. Objek yang digunakan adalah 32 kosakata gestur isyarat dari lirik lagu ‘Bidadari Tak Bersayap’ dengan total *dataset* berjumlah 17.600 citra, kemudian dibagi menjadi 2 bagian model berdasarkan sifat data bias dan non-bias yang berjumlah 8 dan 24 kelas. Penelitian ini menghasilkan prediksi model bias sebesar 15 dari 16, sedangkan model non bias sebesar 36 dari 48 prediksi yang benar dengan jumlah akurasi pengujian berbasis *real-time* pada *mobile* sebesar 93,75% dan 75%. Manfaat dari penelitian ini adalah menghasilkan model yang dapat mengklasifikasikan dan menerjemahkan gestur isyarat secara langsung menggunakan perangkat kamera *smartphone* untuk orang awam.

**Kata kunci:** Bahasa Isyarat Indonesia, *Machine Learning*, *Supervised Learning*, *Convolutional Neural Network*, Klasifikasi.

V BAB + xviii Halaman + 145 Halaman + 60 Gambar + 28 Tabel + Daftar Pustaka + Lampiran

Pustaka Acuan (50, 1989 - 2019)

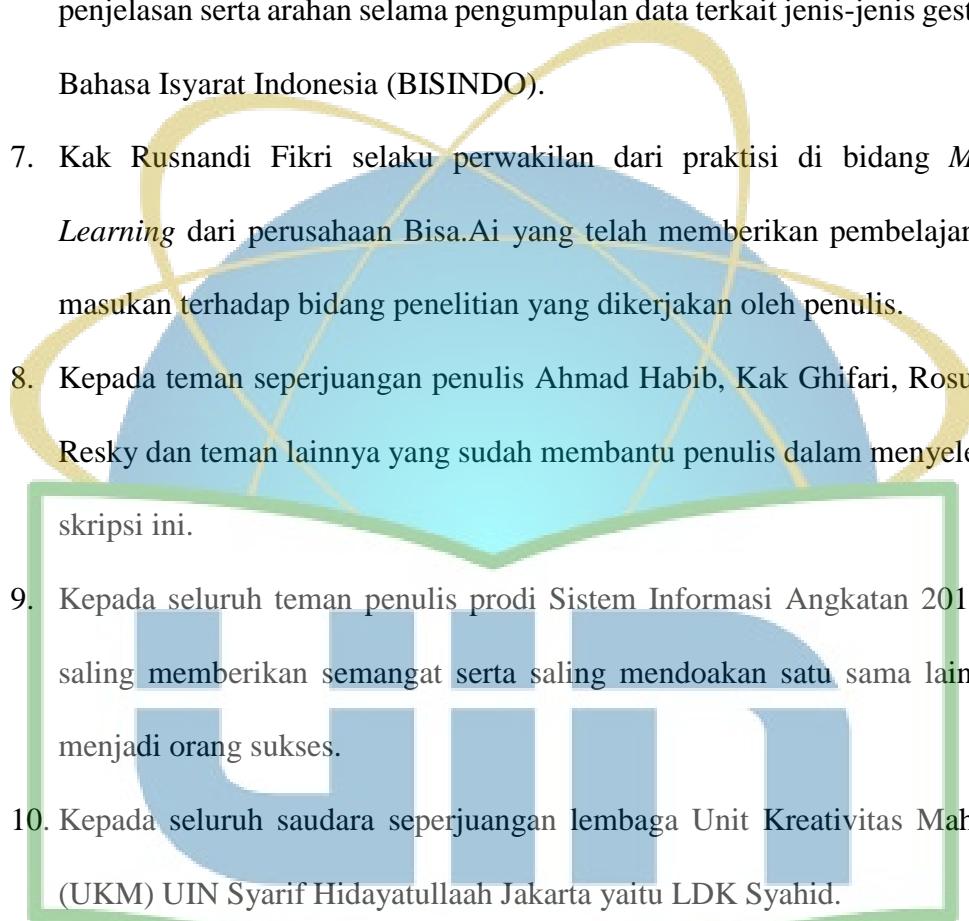


## KATA PENGANTAR

*Assalaamu'alaikum Warahmatullaah Wabarakaaatuh*

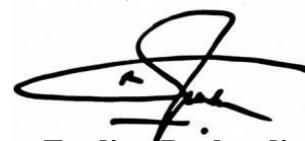
Puji syukur penulis panjatkan kepada Allah *Subhanahu Wa Ta'alaa*, karena atas nikmat dan rahmat-Nya sehingga penulis dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Komputer Program Studi Sistem Informasi Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta. Proses penyelesaian skripsi ini tidak lepas dari berbagai bantuan, dukungan, saran, dan kritik yang telah penulis dapatkan, oleh karena itu dalam kesempatan ini peneliti ingin mengucapkan terima kasih kepada:

1. Ibu Prof. Dr. Lily Surraya Eka Putri, M.Env.Stud. selaku dekan Fakultas Sains dan Teknologi.
2. Bapak Aang Subiyakto, PhD. selaku ketua Program Studi Sistem Informasi dan Ibu Nidaul Hasanati,S.T., MMSI. selaku sekretaris Program Studi Sistem Informasi.
3. Ibu Dr. Qurotul Aini, M.T. selaku dosen pembimbing I dan Bapak Zainul Arham, M.Si. selaku dosen pembimbing II yang secara kooperatif telah meluangkan waktu dan memberikan bimbingan, bantuan, semangat, dan motivasi kepada penulis dalam menyelesaikan skripsi ini.
4. Seluruh dosen, staf karyawan Fakultas Sains dan Teknologi yang telah memberikan bantuan dan kerjasama yang terjalin dari awal perkuliahan.
5. Orang tua dan keluarga penulis yang senantiasa mendoakan, dan mendukung penulis dalam mengerjakan skripsi baik secara moril maupun material.

- 
6. Kakak Gerkatin selaku perwakilan dari Komunitas Gerakan untuk Kesejahteraan Tunarungu Indonesia (GERKATIN) di Jakarta khususnya kepada Pak Irwan, Kak Aldila dan Ibu Silvana Tenrisara selaku dosen peneliti dari Lembaga Riset Bahasa Isyarat FIB UI, Depok yang telah memberikan penjelasan serta arahan selama pengumpulan data terkait jenis-jenis gestur kata Bahasa Isyarat Indonesia (BISINDO).
  7. Kak Rusnandi Fikri selaku perwakilan dari praktisi di bidang *Machine Learning* dari perusahaan Bisa.Ai yang telah memberikan pembelajaran dan masukan terhadap bidang penelitian yang dikerjakan oleh penulis.
  8. Kepada teman seperjuangan penulis Ahmad Habib, Kak Ghifari, Rosuli, Ari, Resky dan teman lainnya yang sudah membantu penulis dalam menyelesaikan skripsi ini.
  9. Kepada seluruh teman penulis prodi Sistem Informasi Angkatan 2015 yang saling memberikan semangat serta saling mendoakan satu sama lain untuk menjadi orang sukses.
  10. Kepada seluruh saudara seperjuangan lembaga Unit Kreativitas Mahasiswa (UKM) UIN Syarif Hidayatullah Jakarta yaitu LDK Syahid.

Penulis menyadari bahwa dalam skripsi ini masih banyak kekurangan, oleh karena itu penulis sangat mengharapkan saran dan kritik dari pembaca. Penulis berharap laporan ini dapat memberikan manfaat bagi yang membaca.

Jakarta, 10 Juli 2020



**Ferdian Rachardi**  
11150930000033

## DAFTAR ISI

<b>HALAMAN JUDUL .....</b>	ii
<b>LEMBAR PENGESAHAN SKRIPSI .....</b>	iii
<b>PENGESAHAN UJIAN.....</b>	iv
<b>PERNYATAAN.....</b>	v
<b>ABSTRAK .....</b>	vii
<b>KATA PENGANTAR.....</b>	ix
<b>DAFTAR ISI.....</b>	xi
<b>DAFTAR GAMBAR.....</b>	xv
<b>DAFTAR TABEL .....</b>	xviii
<b>BAB 1 PENDAHULUAN .....</b>	1
1.1 Latar Belakang.....	1
1.2 Identifikasi Masalah.....	7
1.3 Rumusan Masalah.....	7
1.4 Batasan Masalah .....	8
1.5 Tujuan Penelitian .....	9
1.6 Manfaat Penelitian .....	9
1.6.1 Bagi Penulis.....	9
1.6.2 Bagi Universitas .....	10
1.7 Metodologi Penelitian.....	10
1.7.1 Metode Pengumpulan Data .....	10
1.7.2 Pemodelan dengan <i>Machine Learning</i> .....	11
1.8 Sistematika Penulisan .....	11
<b>BAB 2 TINJAUAN PUSTAKA.....</b>	15

2.1 Artificial Intelligence (AI) .....	15
2.1.1 Machine Learning .....	16
2.1.2 Deep Learning.....	18
2.1.3 Klasifikasi.....	20
2.1.4 Objek Deteksi.....	20
2.2 Jaringan Syaraf Tiruan ( <i>Artificial Neural Network - ANN</i> ).....	22
2.2.1 Komponen Jaringan Syaraf ( <i>Neural Network</i> ).....	23
2.2.2 Arsitektur Jaringan .....	24
2.2.3 Fungsi Aktivasi ( <i>Activation Function</i> ).....	26
2.3 Convolutional Neural Network (CNN).....	28
2.3.1 Convolution Layer.....	30
2.3.2 Pooling Layer.....	32
2.3.3 Fully Connected Layer .....	33
2.3.4 Dropout Regulation.....	34
2.3.5 Softmax Classifier .....	34
2.3.6 Cross Entropy Loss Function .....	36
2.3.7 Proses Forward Propagation pada CNN .....	36
2.3.8 Proses Backward Propagation pada CNN.....	37
2.3.9 Stochastic Gradient Descent (SGD) .....	38
2.4 Arsitektur Objek Deteksi .....	40
2.4.1 Single Shot Detector (SSD).....	40
2.4.2 MobileNet.....	42
2.5 Evaluasi.....	43

2.5.1 <i>Intersection Over Union</i> (IOU) .....	43
2.5.2 <i>Mean Average Precision</i> (mAP) .....	44
<b>2.6 Citra Digital .....</b>	<b>46</b>
2.6.1 Tipe Citra Digital .....	48
2.6.2 Pengolahan Citra Digital ( <i>Image Processing</i> ).....	50
<b>2.7 Tools .....</b>	<b>51</b>
2.7.1 Python.....	51
2.7.2 Android.....	52
2.7.3 <i>Library</i> .....	53
<b>2.8 Populasi dan Teknik <i>Sampling</i> .....</b>	<b>55</b>
<b>2.9 Bahasa Sebagai Alat Komunikasi.....</b>	<b>57</b>
2.9.1 Pengertian <b>Komunikasi Nonverbal</b> .....	58
2.9.2 Pengertian Bahasa Isyarat .....	59
2.9.3 Jenis Bahasa Isyarat .....	61
2.10 Penelitian Sejenis .....	63
2.11 Ranah Penelitian .....	81
<b>BAB 3 METODOLOGI PENELITIAN .....</b>	<b>82</b>
3.1 Metode Pengumpulan Data.....	82
3.1.1 Studi Pustaka.....	82
3.1.2 Wawancara .....	82
3.1.3 Observasi .....	84
3.1.4 Kuesioner .....	89
3.2 Kerangka Penelitian.....	95

3.3 Tahapan Penelitian.....	97
<b>BAB 4 ANALISIS DAN PERANCANGAN SISTEM .....</b>	<b>100</b>
4.1 Tahapan Data <i>Preprocessing</i> .....	100
4.1.1 Data <i>Resizing</i> .....	100
4.1.2 Data <i>Augmentation</i> .....	100
4.1.3 Data <i>Labeling</i> atau <i>Annotasi</i> .....	102
4.2 Perancangan Model .....	103
4.2.1 Konversi Data.....	104
4.2.2 Pemodelan dan Pelatihan .....	107
4.2.3 Evaluasi .....	121
4.2.4 Perbandingan Evaluasi .....	129
4.3 Perancangan Aplikasi.....	130
4.3.1 Perancangan <i>User Interface</i> .....	130
4.3.2 Perencanaan Pengujian.....	131
4.3.3 Simulasi dan Hasil.....	133
4.3.4 Interpretasi.....	136
4.4 Implikasi Hasil.....	139
4.5 <i>Limitation Of Study</i> .....	141
<b>BAB 5 PENUTUP .....</b>	<b>143</b>
4.6 Kesimpulan.....	143
4.7 Saran .....	144
<b>DAFTAR PUSTAKA.....</b>	<b>145</b>
<b>LAMPIRAN</b>	

## DAFTAR GAMBAR

Gambar 1.1 Presentase Penduduk Normal dan Tunarungu .....	2
Gambar 1.2 Presentase Penduduk Disabilitas Menurut Tipe Daerah .....	3
Gambar 2.1 Perbedaan Pemrograman Tradisional dan <i>Machine Learning</i> .....	17
Gambar 2.2 Ilustrasi Penerapan Objek Deteksi .....	21
Gambar 2.3 Ilustrasi Neuron dengan Model Matematis .....	22
Gambar 2.4 Struktur Neural pada JST .....	23
Gambar 2.5 Struktur <i>Single Layer Neural Network</i> .....	25
Gambar 2.6 <i>Multiple Layers Neural Network</i> .....	25
Gambar 2.7 Fungsi Aktivasi Linear .....	26
Gambar 2.8 Fungsi Aktivasi <i>Sigmoid</i> .....	27
Gambar 2.9 Fungsi Aktivasi <i>Tanh</i> .....	27
Gambar 2.10 <i>Fungsi Aktivasi ReLU</i> .....	28
Gambar 2.11 Jaringan Arsitektur <i>Convolutional Neural Network</i> .....	29
Gambar 2.12 Proses Operasi Konvolusi .....	32
Gambar 2.13 Proses <i>Pooling Layer</i> .....	32
Gambar 2.14 Lapisan <i>Fully-Connected</i> .....	33
Gambar 2.15 Proses Sebelum dan Sesudah <i>Dropout</i> .....	34
Gambar 2.16 Jaringan <i>Backpropagation</i> .....	37
Gambar 2.17 Ilustrasi <i>Learning Rate</i> .....	39
Gambar 2.18 Ilustrasi Model SSD dengan Jaringan Dasar.....	40
Gambar 2.19 Sebelum NMS dan Sesudah NMS .....	41

Gambar 2.20 Ilustrasi Arsitektur <i>MobileNet</i> .....	43
Gambar 2.21 Ilustrasi dari IOU.....	44
Gambar 2.22 <i>Confusion Matrix</i> .....	44
Gambar 2.23 Contoh Citra Biner .....	49
Gambar 2.24 Contoh Citra <i>Grayscale</i> .....	49
Gambar 2.25 Contoh Citra Berwarna ( <i>RGB</i> ) .....	50
Gambar 2.26 Logo <i>Library NumPy</i> .....	53
Gambar 2.27 Logo <i>Library Pandas</i> .....	54
Gambar 2.28 Logo <i>Library Tensorflow</i> .....	55
Gambar 2.29 Huruf Abjad Sistem Isyarat Bahasa Indonesia (SIBI) .....	62
Gambar 2.30 Huruf Abjad Bahasa Isyarat Indonesia (BISINDO).....	63
Gambar 2.31 Ranah Penelitian.....	81
Gambar 3.1 Kamera <i>Smartphone</i> Lenovo A7700.....	84
Gambar 3.2 Kerangka Penelitian .....	96
Gambar 3.3 Tahapan Penelitian .....	97
Gambar 4.1 Hasil Augmentasi Data Citra.....	101
Gambar 4.2 Proses Annotasi Data ( <i>Labeling</i> ).....	102
Gambar 4.3 <i>Flowchart</i> Proses Model .....	103
Gambar 4. 4 Proses Model SSD-MobileNet.....	113
Gambar 4. 5 Matriks Input <i>Layer</i> dan <i>Filter</i> .....	115
Gambar 4.6 <i>Output Layer</i> dari Proses Konvolusi .....	116
Gambar 4.7 <i>Ouput</i> Citra dari <i>Average Pooling</i> .....	117
Gambar 4.8 Matriks dari Input <i>Pooling Layer</i> dan <i>Output Layer</i> .....	117

Gambar 4.9 Contoh Penerapan dalam <i>Fully Connected Layer</i> .....	118
Gambar 4.10 <i>Loss function</i> ketika Proses <i>Training</i> .....	119
Gambar 4.11 Hasil Total <i>Loss Function</i> dari Proses <i>Training</i> .....	120
Gambar 4.12 (a) Sebelum dan (b) Sesudah <i>Non Maximum Supression</i> .....	121
Gambar 4.13 Ilustrasi Evaluasi Gambar Uji .....	121
Gambar 4.14 Kurva Interpolasi.....	123
Gambar 4.15 Hasil Grafik <i>Loss</i> Skenario 1 .....	124
Gambar 4.16 Hasil Evaluasi Skenario 1 .....	125
Gambar 4.17 Hasil Evaluasi Skenario 2 Model Bias .....	127
Gambar 4.18 Hasil Grafik <i>Loss</i> Skenario 2 Model Bias .....	127
Gambar 4.19 Hasil Grafik <i>Loss</i> pada Skenario 2 Model Non Bias .....	128
Gambar 4.20 Hasil Evaluasi Skenario 2 Model Non Bias .....	128
Gambar 4.21 Desain <i>User Interface</i> untuk Aplikasi Pengujian .....	131
Gambar 4.22 Proses Implementasi dan Pengujian di Aplikasi .....	132
Gambar 4.23 Hasil Akurasi Benar dari Implementasi Objek Deteksi pada <i>Smartphone</i> .....	134
Gambar 4.24 Hasil Akurasi Salah dari Implementasi Objek Deteksi pada <i>Smartphone</i> .....	134

## DAFTAR TABEL

Tabel 2.1 Perbandingan Penelitian Sejenis <i>Deep Learning</i> .....	64
Tabel 2.2 Perbandingan Penelitian Sejenis Klasifikasi Gestur Bahasa Isyarat.....	74
Tabel 3.1 Waktu dan Tempat Penelitian .....	84
Tabel 3. 2 Spesifikasi <i>Hardware</i> .....	85
Tabel 3. 3 Spesifikasi <i>Software</i> dan <i>Library</i> .....	85
Tabel 3.4 Tabel Pembagian <i>Dataset</i> .....	86
Tabel 3.5 Dataset Gestur Kosakata BISINDO .....	87
Tabel 3.6 Daftar Pertanyaan Kuesioner tentang Pengetahuan .....	89
Tabel 3.7 Daftar Pertanyaan Kuesioner tentang Pengenalan .....	90
Tabel 3.8 Responden Mengetahui Ragam Jenis Bahasa Isyarat Indonesia .....	91
Tabel 3.9 Tingkat Pengetahuan dari Jumlah Responden Mengenai Bahasa Isyarat Indonesia .....	91
Tabel 3.10 Jumlah Responden Menjawab Benar dari Jenis Bahasa Isyarat di Indonesia .....	91
Tabel 3.11 Jumlah Responden Pernah Membaca Artikel tentang Ragam Jenis Bahasa Isyarat .....	92
Tabel 3.12 Jumlah Penilaian Responden Menjawab Benar dan Salah pada Gestur Huruf Bahasa Isyarat Indonesia .....	93
Tabel 3.13 Jumlah Penilaian Responden Menjawab Benar dan Salah pada Gestur Kosakata Bahasa Isyarat Indonesia .....	93

Tabel 3.14 Responden Tertarik untuk Mempelajari dan Mengenal Ragam Jenis Gestur Bahasa Isyarat Indonesia .....	94
Tabel 3.15 Tanggapan Responden Mengenai Aplikasi untuk Mengenali Ragam Jenis Gestur Bahasa Isyarat Indonesia .....	94
Tabel 3.16 Tanggapan Responden Mengenai Pentingnya Aplikasi.....	94
Tabel 4.1 Struktur Bagan Arsitektur dari MobileNet.....	114
Tabel 4.2 Contoh Penerapan Neuron Terakhir pada <i>Softmax Classifier</i> .....	119
Tabel 4.3 Identifikasi <i>Confusion Matrix</i> .....	122
Tabel 4.4 Akumulasi dari <i>Precision</i> dan <i>Recall</i> .....	122
Tabel 4.5 Detail Skenario Pertama .....	124
Tabel 4.6 Detail Skenario Kedua .....	126
Tabel 4.7 Hasil Pengujian <b>Model Bias dengan 8 kelas</b> .....	135
Tabel 4.8 Hasil Pengujian Non Bias dengan 24 Kelas.....	135
Tabel 4.9 Hasil Perbandingan Metode .....	138



# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Saat ini dunia telah masuk pada perkembangan revolusi industri 4.0 dengan transformasi pada segala aspek terutama pemanfaatan dari perkembangan *Artificial Intelligence* (AI), salah satunya seperti *machine learning* (Nurhikmat & Purwaningsih, 2018).

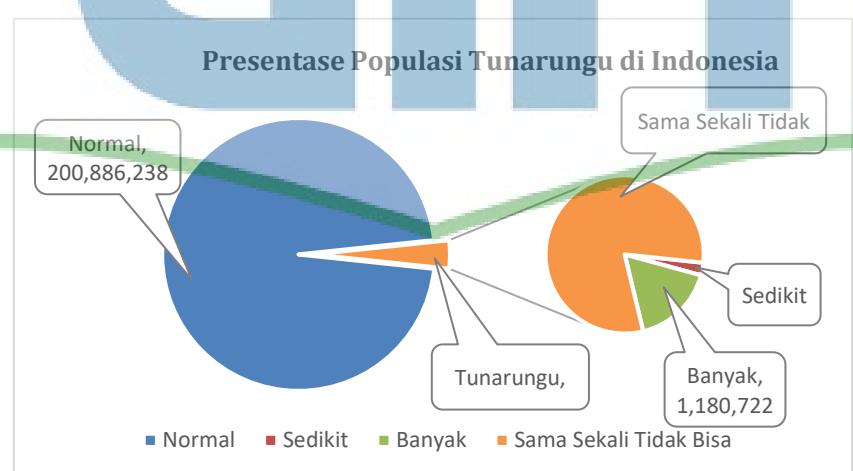
Dengan berkembangnya *machine learning* yang mempelajari sekumpulan data sederhana maka munculah salah satu model sub-bidangnya lebih dalam yaitu *deep learning*, merupakan suatu algoritma yang terinspirasi dan dianalogikan dengan model struktur dan fungsi otak manusia yang biasa disebut jaringan saraf (*Neuron*) tiruan (Brownlee, 2016).

Untuk mengklasifikasi dan menghasilkan prediksi dengan akurasi yang tinggi, memerlukan suatu teknik yang baik yaitu dengan cara hierarki dan berskala besar serta koneksi yang kuat, maka dapat dikatakan pemanfaatan konvolusi jaringan saraf atau *Convolutional Neural Network* (CNN) merupakan metode algoritma yang sesuai dalam spesialisasi bidang gambar dan suara (LeCun, Bengio, & Hinton, 2015).

Adapun mengklasifikasi nilai fungsi dari semua nilai input yang kemungkinan ada maka diperlukan teknik pembelajaran atau yang biasanya dikenal *supervised learning* atau label diberikan kepada algoritma sebagai dasar kebenaran (Mirsky et al, 2018).

Sebagaimana pada beberapa penelitian sebelumnya yang berkaitan dengan klasifikasi pada *machine learning* di berbagai bidang yang umumnya menggunakan metode *convolutional neural network* dengan hasil klasifikasi yang cukup tinggi seperti pada bidang biometrik oleh Michele *et al.* (2019); bidang transportasi Arfian (2018); bidang kesenian oleh Salsabila (2018), Nurhikmat & Purwaningsih (2018); bidang botani oleh Putri (2018), Kusumaningrum (2018), Shafira (2018), Dzulqarnain *et al.* (2019); dan bidang bahasa oleh Dewa *et al.* (2018). Hal ini menunjukkan algoritma CNN mampu menyelesaikan masalah klasifikasi dengan baik dan benar.

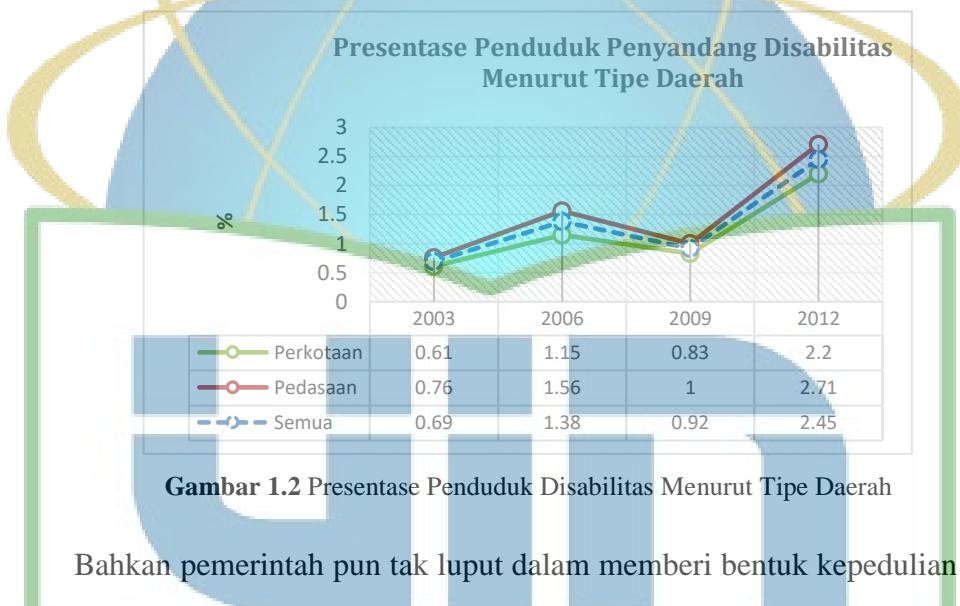
Disisi lain, bahasa utama yang digunakan oleh penyandang disabilitas yaitu bahasa isyarat, namun bahasa tersebut memiliki banyak keterbatasan komunikasi untuk dimengerti oleh orang awam dikarenakan populasi disabilitas di Indonesia berjumlah sedikit atau minoritas dibanding populasi penduduk normal sebagaimana data Survei Penduduk Antar Sensu (SUPAS) tahun 2015 yang dikeluarkan oleh BPS pada Gambar 1.1.



**Gambar 1.1** Presentase Penduduk Normal dan Tunarungu

Pada Gambar 1.1 menjelaskan bahwa terdapat 6.952.797 tunarungu dari 200.886.238 penduduk normal sehingga mempengaruhi penggunaan bahasa isyarat yang terbilang sedikit dan masih asing bagi orang normal.

Selaras dengan survei penduduk yang dilakukan oleh Kementerian Kesehatan tahun 2014 mengeluarkan buletin yang berjudul ‘Situasi Penyandang Disabilitas’ yang menyatakan bahwa presentase jumlah populasi data penduduk Indonesia mengalami perubahan tiap tahunnya antara daerah perkotaan maupun pedesaan, seperti pada Gambar 1.2 (Kementerian Kesehatan, 2014).



Bahkan pemerintah pun tak luput dalam memberi bentuk kepedulian kepada mereka berupa hak dan kewajiban penyandang disabilitas terdapat dalam peraturan UU RI No. 8 Tahun 2016, dimana dalam pembukaannya tercantum kalimat, “..*Negara Kesatuan Republik Indonesia menjamin kelangsungan hidup setiap warga Negara, termasuk para penyandang disabilitas yang mempunyai kedudukan hukum dan hak asasi manusia yang sama sebagai warga Negara Indonesia..*”.

Di sisi lain, pemerintah juga memberikan hak kebebasan dalam berinteraksi maupun berkomunikasi yang terdapat pada pasal 26 ayat ke-1 tentang hak

kebebasan yang menyatakan, “*Penyandang disabilitas memiliki hak dalam bersosialisasi dan berinteraksi dalam kehidupan berkeluarga, bermasyarakat, dan bernegara tanpa rasa takut..*”

Menurut Tolba & Elons (2013) bahwa banyak penyandang tunarungu tersebut belum dapat membaca atau menulis seperti pada umumnya sehingga mereka sulit untuk berkomunikasi. Selaras dengan apa yang dinyatakan Ibu Silva saat diwawancara bahwasanya kebanyakan teman tuli atau penyandang tunarungu kesulitan untuk dapat berkomunikasi dengan orang awam ketika berinteraksi langsung serta mereka pun kesulitan dalam hal membaca maupun menulis sebagai alternatif lain baginya untuk berinteraksi.

Adapun di Indonesia memiliki 2 standar sistem bahasa isyarat yang biasa dijumpai yaitu SIBI (Sistem Isyarat Bahasa Indonesia) dan BISINDO (Bahasa Isyarat Indonesia). Berdasarkan Peraturan Kementerian Pendidikan dan Kebudayaan RI Nomor 0161/U/1994 tentang Pembakuan Sistem Bahasa Isyarat Bahasa Indonesia Bagi Kaum Tuna Rungu bahwasanya, SIBI merupakan bahasa isyarat yang telah diresmikan oleh pemerintah sesuai dengan kurikulum yang berlaku dan sebagai standar mengajar di Sekolah Luar Biasa (SLB).

Namun menurut Demas (2017), kehadiran SIBI tidak menjawab solusi dari komunikasi yang efektif tapi dikemas dengan bahasa yang terlalu baku dan kaku sehingga membuat penyandang disabilitas kesulitan dalam berkomunikasi. Hal ini pun diutarakan juga oleh Ibu Silva, bahwa disebabkan dengan kemudahan yang ada pada sistem bahasa isyarat BISINDO, kebanyakan teman tuli lebih mudah memahami dan mengenal untuk berinteraksi sehari-hari dibandingkan SIBI. Oleh

karena itu, perlunya pengembangan penelitian lebih jauh tentang BISINDO untuk orang awam agar dapat mengerti ketika berinteraksi kepada teman tuli.

Dalam kemudahan yang ditawarkan oleh BISINDO, baik karakter setiap huruf, kosakata, ejaan dan tata bahasa memudahkan bagi orang awam dalam belajar maupun menerjemahkan bahasa isyarat tersebut ketika berkomunikasi dengan orang-orang difabel. Namun dengan jumlah kosakata pada bahasa isyarat dengan masing-masing mewakili gestur tertentu tiap katanya, sehingga diharuskan tiap kosakata dihafal dan dipahami maksud dari gestur tersebut.

Bahkan, total jumlah populasi yang terdata pada kosakata BISINDO adalah 814 gestur per kata, kemudian diturunkan menjadi beberapa tingkat dan kategori yang terdiri atas tingkat I - III. Banyaknya jumlah kata tersebut, jika diklasifikasikan secara tradisional atau manual akan membutuhkan waktu lama dan tenaga yang besar bagi orang awam untuk mengingat dan mengklasifikasi gestur kosakata tersebut.

Maka dengan hadirnya *machine learning* diharapkan membantu proses klasifikasi gestur kosakata pada bahasa isyarat indonesia (BISINDO) dan dapat dilakukan secara otomatis kepada yang menggunakan bahasa isyarat dengan waktu yang relatif singkat serta memiliki keakuratan yang cukup tinggi tanpa perlu menerjemahkan arti isyarat dari masing-masing gestur secara manual.

Bahkan beberapa penelitian mengenai klasifikasi bahasa isyarat telah dilakukan oleh Masood, Thuwal, & Srivastava (2018) yang mengidentifikasi huruf abjad dan angka pada *American Sign Language* (ASL) dengan hanya metode CNN dimana jumlah data yang digunakan mencapai 14.781 data citra yang terdiri

atas seluruh huruf abjad (A-Z) dan angka (0-9), dari hasil prediksi penelitian didapatkan nilai prediksi pada masing-masing huruf abjad dan angka sehingga kisaran total bernilai 94,68%, kemudian penelitian Bheda & Radpour (2017) menggunakan algoritma *Deep Convolutional Neural Network* pada bahasa isyarat Amerika (ASL) pada 11 dan 26 kelas menghasilkan akurasi sebesar 97% dan 82,5%, lalu penelitian Rakhman et al (2010) dan Borman et al (2017) menggunakan perpaduan algoritma *HaarClassifier* dan *K Nearest Neighbor* (KNN) pada SIBI dan BISINDO menghasilkan akurasi sebesar 89,68% dan 91,8%, serta penelitian Raheja, Mishra, & Chaudhary (2016) menggunakan algoritma *SVM* pada 4 kosakata ISL menghasilkan akurasi sebesar 97,5%.

Dari berbagai penelitian yang berkaitan dengan bahasa isyarat, sayangnya belum ada penelitian untuk objek deteksi dan mengklasifikasikan bahasa isyarat BISINDO yang berfokus pada klasifikasi gestur pada kosakata bahasa isyarat dengan algoritma *Convolutional Neural Network* (CNN). Berdasarkan penjelasan sebelumnya bahwa penelitian ini menggunakan teknik *supervised learning* dimana setiap kosakata dari bahasa isyarat BISINDO akan dilabelisasi berdasarkan gestur tangan yang ada pada setiap gambar.

Penelitian ini menggunakan *dataset* berdasarkan dokumentasi pengumpulan gambar pribadi oleh peneliti menggunakan alat kamera. Kemudian, *dataset* tersebut terdiri atas 32 jenis gestur atau label per kosakata dan menggunakan 500 gambar berdasarkan karakter tiap kosakata dengan jumlah total *dataset* sekitar 17.600 gambar. Pemilihan kosakata berdasarkan pada syair lagu yang digunakan sebagai bahan pengujian yaitu “Bidadari Tak Bersayap” dari Anji.

Oleh karena itu, berdasarkan berbagai sebab dari penjelasan sebelumnya, maka peneliti tertarik untuk melakukan penelitian yang berjudul, “**Deteksi Gambar Gestur Kosakata Bahasa Isyarat Indonesia dengan Convolutional Neural Network**”.

## 1.2 Identifikasi Masalah

Berdasarkan latar belakang permasalahan maka didapatkan rumusan masalah yang diangkat dalam penelitian ini. Penulis mengidentifikasikan masalah-masalah yang dijelaskan sebagai berikut:

- a. Belum banyak orang yang mengetahui arti dari jenis-jenis gestur tangan untuk kosakata pada bahasa isyarat sehingga sulitnya interaksi penyandang disabilitas atau tunarungu (teman tuli) dengan orang awam menggunakan bahasa kesehariannya yaitu Bahasa Isyarat Indonesia (BISINDO).
- b. Telah banyak dilakukan penelitian mengenai penerapan model klasifikasi untuk huruf dan angka dari gestur isyarat tidak secara langsung sehingga beban jumlah kelas menjadi lebih banyak untuk diklasifikasikan dan mempersulit orang awam untuk menerjemahkannya.
- c. Belum diketahui faktor apa saja yang mempengaruhi hasil klasifikasi dan akurasi terbaik pada penerapan model klasifikasi gambar gestur bahasa isyarat Indonesia secara langsung menggunakan *Convolutional Neural Network* (CNN).

## 1.3 Rumusan Masalah

Berdasarkan identifikasi masalah maka rumusan masalah yang dibahas dalam penelitian ini adalah:

- a. Bagaimana penerapan model *deep learning* menggunakan *Convolutional Neural Network* (CNN) untuk objek deteksi dan klasifikasi gambar gestur pada kosakata Bahasa Isyarat Indonesia (BISINDO) ?
- b. Bagaimana kinerja atau performa yang dihasilkan dari model *deep learning* menggunakan CNN untuk klasifikasi dan deteksi gestur kosakata Bahasa Isyarat Indonesia (BISINDO)?
- c. Apa saja faktor yang mempengaruhi hasil tingkat akurasi yang terbaik pada penerapan klasifikasi dan deteksi objek pada gambar kosakata Bahasa Isyarat Indonesia (BISINDO)?

#### 1.4 Batasan Masalah

- a. Jenis bahasa isyarat yang akan diklasifikasi pada penelitian ini yaitu Bahasa Isyarat Indonesia (BISINDO) yang meliputi 32 kosakata dari lirik lagu “Bidadari Tak Bersayap”.
- b. Jenis citra yang digunakan adalah JPEG (Joint Photographic Experts Group).
- c. Algoritma yang digunakan hanya metode klasifikasi gambar dan objek deteksi yaitu Convolutional Neural Network (CNN) dengan arsitektur SSD MobileNet.
- d. Metode pengumpulan data yang digunakan yaitu kuesioner, observasi dan wawancara.
- e. Perancangan model pada penelitian ini menggunakan bahasa pemrograman Python versi 3 dengan bantuan library Numpy, CV2, Scikit-learn, Tensorflow dan Tensorboard.

- f. Kemudian aplikasi dibangun pada *platform* Android dengan bahasa pemrograman *Java*.

## 1.5 Tujuan Penelitian

Berdasarkan rumusan masalah sebelumnya maka tujuan dalam penelitian ini adalah untuk:

- a. Melakukan penerapan model objek deteksi dan klasifikasi gambar pada gestur kosakata Bahasa Isyarat Indonesia (BISINDO) dengan *Convolutional Neural Network* (CNN).
- b. Mendapatkan hasil klasifikasi dan akurasi dari model objek deteksi yang menggunakan *Convolutional Neural Network* (CNN) dengan memprediksi kosakata Bahasa Isyarat Indonesia (BISINDO) pada aplikasi.
- c. Mendapatkan faktor yang mempengaruhi tingkat akurasi penerapan objek deteksi dan klasifikasi gambar dengan kosakata Bahasa Isyarat Indonesia (BISINDO).

## 1.6 Manfaat Penelitian

### 1.6.1 Bagi Penulis

- a. Menerapkan ilmu teknik pengolahan citra (*image processing*), objek deteksi dan klasifikasi dengan menggunakan algoritma *Convolutional Neural Network* dalam merancang simulasi sistem pada klasifikasi gambar jenis Kata Bahasa Isyarat Indonesia (BISINDO).

- b. Memahami tentang bahasa pemrograman yang digunakan oleh penulis, yaitu *Python* dan Java dan jenis gestur kosakata BISINDO.

### 1.6.2 Bagi Universitas

- a. Mengetahui kemampuan mahasiswa dalam menguasai materi pelajaran yang diperoleh di bangku kuliah.
- b. Mengenali kemampuan mahasiswa dalam menerapkan ilmunya dan sebagai bahan evaluasi.
- c. Memberikan gambaran tentang kesiapan mahasiswa dalam menghadapi dunia kerja yang sebenarnya.

### 1.6.3 Bagi Masyarakat

- d. Mempermudah kemampuan masyarakat untuk menerjemahkan gestur isyarat menjadi sebuah kosakata
- e. Mempercepat waktu dalam mengklasifikasikan gestur secara langsung tanpa mencari secara manual pada kamus.
- f. Meningkatkan kepedulian dalam komunikasi bagi orang normal terhadap bahasa isyarat yang digunakan oleh penyandang disabilitas seperti tunawicara, tunarungu, dan lain-lain.

## 1.7 Metodologi Penelitian

### 1.7.1 Metode Pengumpulan Data

Dalam penulisan ini metodologi pengumpulan data yang digunakan adalah sebagai berikut:

- a. Wawancara

Mengumpulkan data mengenai bahasa isyarat Indonesia dengan mewawancara langsung kepada pihak yang bersangkutan terkait objek skripsi yaitu Ibu Silva Tenrisara Isma Putri dari LRBI FIB UI dan Kak Aldilla dari Kopi Tuli.

b. Observasi

Mengumpulkan dan memperbanyak *dataset* berjumlah 17.600 citra menggunakan kamera *smartphone* yang direkam oleh penulis.

c. Kuesioner

Penulis menyebarluaskan kuesioner untuk mengetahui seberapa besar pengetahuan masyarakat umum mengenai bahasa isyarat serta harapan kegunaan dari aplikasi ini.

### 1.7.2 Pemodelan dengan *Machine Learning*

Dalam penelitian ini, penulis menggunakan metode pengembangan yang berasal dari Ariel (2019) terdiri atas pengumpulan *dataset*, *pre-processing* data, konfigurasi model CNN, pengujian model dengan data uji, implementasi pada aplikasi, evaluasi dan menggunakan *tools* bahasa Python versi 3 dengan bantuan *library* Augmentor, Numpy, Tensorflow dan Tensorboard.

## 1.8 Sistematika Penulisan

Dalam penelitian ini pembahasan dalam lima bab yang secara singkat akan diuraikan sebagai berikut:

## BAB 1 PENDAHULUAN

Pada bab ini terdiri dari delapan sub bab yang berisikan latar belakang masalah, identifikasi masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penulisan, dan sistematika penulisan.

## BAB 2 TINJAUAN PUSTAKA

Pada bab ini, berisikan literatur yang dijadikan acuan dalam penelitian dan menguraikan teori yang terkait dengan konsep *deep learning* dengan *Convolutional Neural Network* untuk Bahasa Isyarat Indonesia (BISINDO).

## BAB 3 METODOLOGI PENELITIAN

Bab ini berisikan pemaparan metode yang penulis gunakan dalam proses penelitian yang terdiri atas pengumpulan data, populasi data, kerangka dan tahapan penelitian.

## BAB 4 HASIL DAN PEMBAHASAN

Dalam bab ini, menjelaskan tahapan proses pengumpulan data, data *preprocessing*, pembuatan model klasifikasi dan objek deteksi *deep learning* dengan CNN, perancangan desain *interface* serta aplikasi, dan menguraikan uji coba aplikasi serta hasil yang diperoleh.

## BAB 5 PENUTUP

Pada bab ini menjelaskan tentang kesimpulan dari hasil penelitian yang didapat dan saran yang dapat digunakan untuk pengembangan aplikasi yang lebih baik lagi di masa datang.



## BAB 2

### TINJAUAN PUSTAKA

#### 2.1 Artificial Intelligence (AI)

Pada tahun 1956, terdapat seorang tokoh terkenal dari Massachusetts Institute of Technology (MIT) adalah John McCarthy, yang pertama kali mengusulkan suatu istilah tentang “*Artificial Intelligence*” di mana beliau menjelaskan tujuan yang dimaksud dengan ilmu kecerdasan buatan atau *Artificial Intelligence* (AI) adalah, ‘*The goal of AI is to develop machines that behave as though they were intelligent. It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable*’.

Diterjemahkan dalam pengertian yang dimaksud John McCarthy yang dikutip dari Budiharto (2018), *Artificial Intelligence* merupakan cabang dari ilmu komputer yang berfokus pada pengembangan mesin yang berperilaku cerdas. Perpaduan ilmu sains dan teknik yang membuat mesin memiliki kecerdasan, khususnya pada program komputer yang cerdas’.

Dimulai dari pemaparan yang disampaikan oleh McCarthy dan peneliti lainnya, kecerdasan buatan menjadi berkembang yang awal mulanya berada pada tingkat penelitian di laboratorium menjadi semakin berkembang pesat sampai pada produk-produk bisnis teknologi besar maupun kecil, sehingga disebabkan pengaruh yang sangat dirasakan oleh pemakai. Beberapa pakar mendefinisikan

pengertian dari kecerdasan buatan atau *Artificial Intelligence* (AI) yang dikutip dari (Amrizal & Aini, 2013), sebagai berikut:

1. Schalkoff (1990): AI adalah suatu bidang studi yang berusaha menerangkan dan meniru perilaku cerdas dalam bentuk proses komputasi.
2. Rich dan Knight (1991): AI adalah studi tentang cara membuat komputer melakukan sesuatu, di mana saat itu, orang dapat melakukannya lebih baik.
3. Luger dan Stubblefield (1993): AI adalah cabang ilmu komputer yang berhubungan dengan otomasi perilaku yang cerdas.

Dari penjelasan yang diungkapkan beberapa ahli tersebut dapat disimpulkan bahwa kecerdasan buatan atau *Artificial Intelligence* (AI) merupakan suatu bidang dari cabang ilmu komputer yang berfokus pada peniruan, penangkapan, perilaku, pemodelan serta penyimpanan yang ada pada kecerdasan manusia di dalam sebuah sistem teknologi informasi sehingga sistem dapat menjembatani proses pengambilan keputusan secara langsung yang biasanya hanya dapat dilakukan oleh manusia.

### 2.1.1 *Machine Learning*

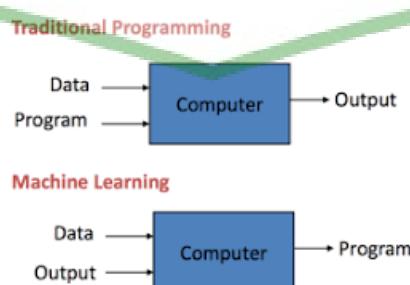
Salah satu sub-bidang yang terdapat pada *Artificial Intelligence* (AI) adalah *Machine Learning* di mana pertama kali didefinisikan oleh Arthur Samuel pada tahun 1959 yang dikutip oleh Nurhikmat & Purwaningsih (2018), *machine learning* merupakan salah satu dari bidang ilmu komputer yang memberikan pembelajaran kepada komputer untuk mengetahui sesuatu tanpa pemrograman yang jelas.

Sedangkan pendapat lain menyatakan bahwa *machine learning* dapat didefinisikan sebagai metode komputasi berdasarkan pengalaman untuk

meningkatkan performa atau membuat prediksi yang akurat (Mohri, Rostamizadeh, & Talwalkar, 2012). Di dalam definisi pengalaman tersebut dimaksudkan informasi yang ada sebelumnya yang telah tersedia dan bisa dijadikan data pembelajar. Adapun dalam definisi lain dari *machine learning* yang dijabarkan oleh beberapa ahli dalam Wahyono (2018) dengan bukunya yang berjudul “*Fundamental Of Python For Machine Learning*”, sebagai berikut:

1. Mitchel (1997): Komputer yang memiliki kemampuan melakukan belajar dari pengalaman terhadap tugas-tugasnya dan mengalami peningkatan kinerja.
2. Budiharto (2016): Tipe dari kecerdasana buatan yang menyediakan komputer dengan kemampuan untuk belajar dari data, tanpa secara eksplisit harus mengikuti instruksi terprogram.

Dapat disimpulkan bahwa pengertian yang dimaksud tentang *machine learning* adalah suatu bidang ilmu komputer yang merupakan cabang dari *Artificial Intelligence* (AI) yang memiliki kemampuan pembelajaran dari pengalaman yang ada seperti data-data atau informasi yang sudah ada sebelumnya untuk meningkatkan performa ataupun membuat prediksi yang tepat dan akurat.



**Gambar 2.1** Perbedaan Pemrograman Tradisional dan *Machine Learning*

Dalam keterangan pada Gambar 2.1 dijelaskan secara singkat menurut Brownlee (2016) bahwa terdapat perbedaan antara konsep pemrograman tradisional dengan pemrograman menggunakan algoritma *machine learning*. Dari sisi metode pembelajaran *machine learning* yang diungkapkan bahwa algoritma *machine learning* dapat dikategorikan dalam beberapa skenario (Brownlee, 2016) yaitu:

1. *Supervised Learning*

Pada penggunaan tipe *Supervised Learning*, ialah teknik pembelajaran mesin atau *machine learning* dengan memasukkan data latih untuk melakukan kegiatan pembelajaran yang telah diberikan tiap label, kemudian membuat prediksi dari data yang telah diberi label.

2. *Unsupervised Learning*

Pada penggunaan tipe *Unsupervised Learning*, ialah teknik pembelajaran mesin yang memasukkan data latih tanpa diberikan label, kemudian mencoba untuk mengelompokkan data berdasarkan karakteristik-karakteristik yang diketahui.

3. *Reinforcement Learning*

Pada penggunaan tipe *Reinforcement Learning*, ialah teknik yang mengajarkan bagaimana cara bertindak untuk menghadapi suatu masalah, di mana tindakan tersebut mempunyai dampak sehingga metode ini diterapkan pada agen cerdas agar dapat menyesuaikan dengan kondisi lingkungannya.

### 2.1.2 *Deep Learning*

Dalam suatu penelitian yang dijelaskan oleh Deng & Yu (2013) menerangkan apa yang dimaksud dengan konsep *Deep Learning*, bahwasanya konsep tersebut

merupakan salah satu teknik pada *machine learning* yang memanfaatkan banyak *layer* pengolahan informasi nonlinier untuk melakukan ekstraksi fitur, baik pengenalan pola, dan klasifikasi.

Ditambah lagi penjelasan lain yang diungkapkan oleh Wehle (2017) bahwa *deep learning* adalah, “*a form of machine learning that can utilize either supervised or unsupervised algorithms, or both..*”, dan diungkapkan kembali, “..*deep learning has recently seen a surge in popularity as way to accelerate the solution of certain types of difficult computer problems, most notably in the computer vision and natural language processing (NLP) fields.*”.

Diterjemahkan bahwa yang dimaksud *deep learning* adalah suatu cabang pada bidang *machine learning* yang bisa memanfaatkan algoritma dari *supervised* atau *unsupervised*, maupun keduanya sehingga dapat mempercepat pencarian solusi dari bermacam-macam masalah komputer, khususnya pada bidang *computer vision* dan *natural language processing (NLP)*.

Adapun dalam definisi lain, *deep learning* merupakan cabang ilmu dari *machine learning* yang berbasis Jaringan Syaraf Tiruan (JST) atau bisa dikatakan dari JST tersebut yang mengajarkan komputer untuk dapat melakukan tindakan yang dianggap alami oleh manusia dan direpresentasikan dengan membentuk arsitektur jaringan syaraf yang mempunyai banyak *layer* (lapisan), misalkan pembelajaran dari contoh atau disebut *supervised learning* (Putri, 2018).

Dapat disimpulkan dari pengertian yang dijelaskan oleh beberapa ahli yakni, *deep learning* merupakan suatu cabang teknik dari *machine learning* yang memanfaatkan Jaringan Syaraf Tiruan (JST) sehingga memiliki banyak lapisan-

lapisan *layer* yang mengelola informasi nonlinier sehingga meningkat perecepatan dalam pencarian solusi dari berbagai masalah seperti regresi klasifikasi, dan sebagainya.

### 2.1.3 Klasifikasi

Salah satu metode algoritma yang umum digunakan pada konsep *Machine Learning* maupun *Deep Learning* untuk mencari solusi dari permasalahan yang sering muncul ialah, metode klasifikasi. Menurut beberapa ahli pun memaparkan definisi masing-masing terhadap istilah klasifikasi tersebut, yaitu sebagai berikut:

1. Klasifikasi diartikan sebagai suatu proses untuk memperoleh model ataupun fungsi yang melukiskan dan membedakan kelas data maupun konsep yang mempunyai tujuan memprediksi kelas untuk data yang tidak dikenali kelasnya (Han & Kamber, 2006).
2. Klasifikasi merupakan suatu pekerjaan menilai objek data untuk memasukkannya ke dalam kelas tertentu dari sejumlah kelas yang tersedia (Rahmatullah, Amrizal & Rozy, 2018).

Jadi dapat disimpulkan dari beberapa definisi tersebut bahwa klasifikasi merupakan suatu pekerjaan dalam memperoleh model atau fungsi untuk menilai objek data atau memprediksi data pada kelas sesuai dengan perolehan kelas yang telah tersedia sebelumnya.

### 2.1.4 Objek Deteksi

Menurut Gong *et al.* (2019), objek deteksi adalah proses pengenalan yang dapat membedakan objek dari latar belakang dan objek mencurigakan lainnya, seperti mobil dan jalan. Sedangkan menurut Michelluci & Moolayil (2019), objek

deteksi ialah pengenalan *instance* objek dalam gambar dan menentukan objek yang diketahui dengan tanda kotak pembatas di sekitaran objek.

Kemudian, detector objek akan menampilkan daftar objek yang terdeteksi dengan beberapa informasi terkait yang terdiri atas (Vasilev *et al.*, 2019):

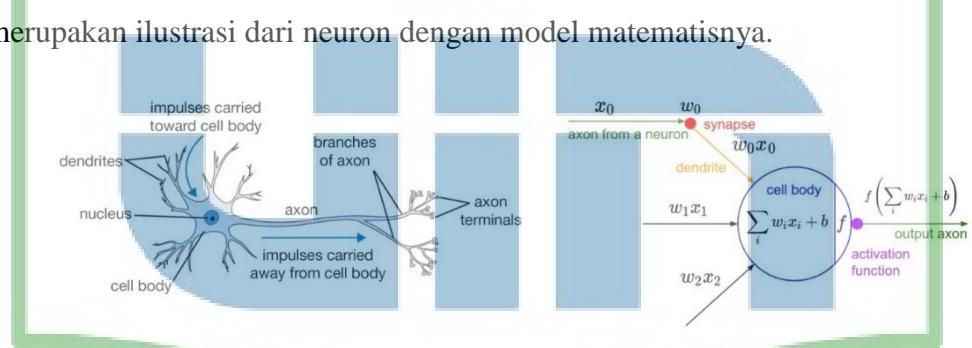
1. Kelas objek (Misal: orang, mobil, hewan, dan sebagainya)
2. Probabilitas (skor kepercayaan) dengan rentang sekitar 0–1 yang menandai seberapa yakin detektor terhadap objek yang berada di lokasi tersebut layaknya *output* klasifikasi biasa.
3. Koordinat wilayah segi empat dari gambar tempat objek berada, kotak tersebut biasanya disebut *ground truth* atau kotak pembatas sebagaimana pada Gambar 2.2.



Gambar 2.2 Ilustrasi Penerapan Objek Deteksi

## 2.2 Jaringan Syaraf Tiruan (*Artificial Neural Network - ANN*)

Yang dimaksudkan dari Jaringan Syaraf Tiruan (JST) atau disebut *Artificial Neural Network* (ANN) menurut Nurhikmat & Purwaningsih (2018) adalah sebuah model komputasi paralel yang meniru fungsi dari sistem jaringan syaraf biologi pada otak manusia sehingga dari representasi otak manusia terdiri atas milyaran *neuron* yang saling terpaut dan terhubung. Adapun definsi lain dari Kusumadewi (2003), *Neural Network* adalah salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut. Hubungannya biasa disebut *Synapses*, serta komponennya terdiri atas satu inti sel yang melakukan pemrosesan informasi, satu akson (*axon*) dan minimal satu *dendrit* sebagai penerima informasi. Di samping itu, *dendrit* juga menyertai akson sebagai keluaran dari sebuah pemrosesan informasi. Gambar 2.3 merupakan ilustrasi dari neuron dengan model matematisnya.



Gambar 2.3 Ilustrasi Neuron dengan Model Matematis

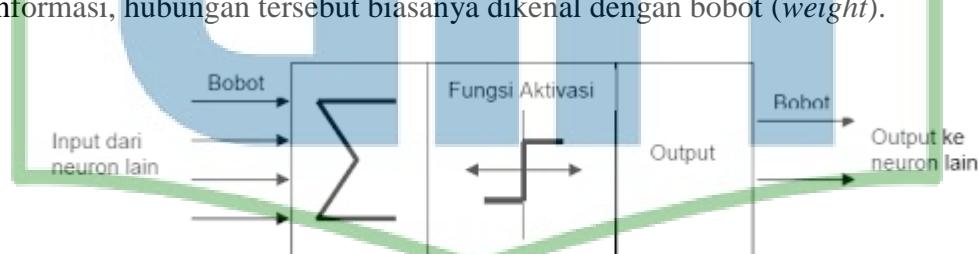
Permulaan dari cara kerja yang ada pada sistem syaraf yang tertera di Gambar 2.3 adalah sinyal masuk yang melalui dendrit menuju *cell body*, lalu sinyal tersebut diproses dalam *cell body* berdasarkan fungsi tertentu (*Summation Proses*). Apabila sinya hasil proses melewati dari nilai ambang batas (*threshold*) dalam jumlah tertentu, sinyal akan membangunkan *neuron* untuk melanjutkan sinyal, sedangkan bila dibawah nilai ambang batas maka sinyal akan dihalangi (*inhibited*). Sehingga

sinyal yang diteruskan menuju ke *axon* dan akhirnya menuju ke *neuron* lain melewati *synapses*.

Secara sederhana analogi tersebut bisa diartikan bahwa setiap neuron menerima input dan melakukan operasi dot dengan sebuah bobot (*weight*), menjumlahkannya (*weighted sum*) dan menambahkan bias sehingga hasil dari model operasi matematis tersebut akan dijadikan parameter dari fungsi aktivasi (*activation function*) yang akan dijadikan *output* dari neuron tersebut.

### 2.2.1 Komponen Jaringan Syaraf (*Neural Network*)

Komponen yang terdapat pada jaringan syaraf atau *neural network* rata-rata memiliki bentuk yang sama, walaupun beberapa tipe *neural network* lain terdapat perbedaan. Seperti yang dijelaskan pada poin sebelumnya komponen jaringan syaraf merepresentasikan fungsi dari jaringan syaraf pada otak manusia dengan dianalogikan model algoritma secara matematis. Komponen tersebut terdiri atas beberapa neuron yang saling berhubungan dengan melakukan transformasi informasi, hubungan tersebut biasanya dikenal dengan bobot (*weight*).



Gambar 2.4 Struktur Neural pada JST

Pada Gambar 2.4 diilustrasikan struktur neuron pada *neural network* (Kiki, 2003), di mana input diproses dengan suatu fungsi yang menjumlahkan nilai berbobot dari semua input yang masuk sehingga hasil dari penjumlahan kemudian dikenakan suatu fungsi aktivasi (*activation function*) yang berguna untuk memutuskan neuron dapat diaktifkan atau tidak dengan cara membandingkan nilai

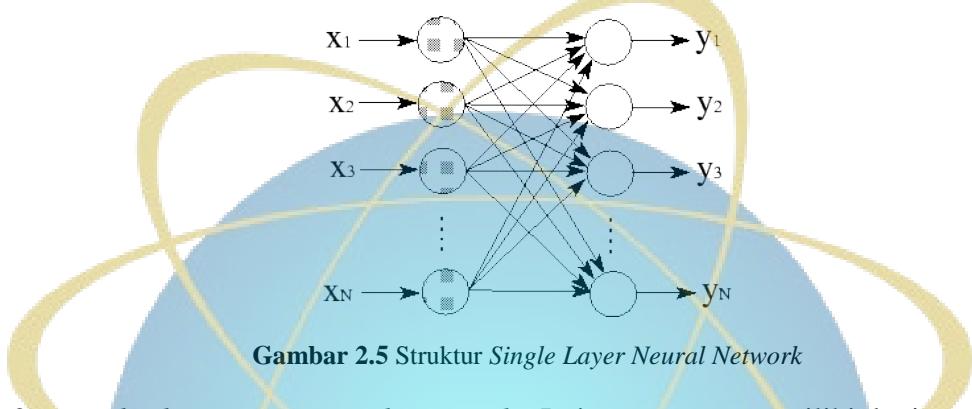
ambang batas (*threshold*) tertentu. Jika neuron aktif, maka neuron dapat mengirimkan *output* melalui nilai-nilai bobot (*weight*) ke semua neuron berikutnya yang saling terhubung. Secara sederhana, dapat ditunjukkan struktur dari komponen yang dimiliki oleh *Neural Network*, yaitu sebagai berikut:

1. Input terdiri atas variabel independen ( $X_1, X_2, X_3, \dots, X_n$ ) yaitu suatu sinyal yang dapat masuk ke sel syaraf.
2. Bobot (*weight*) yang terdiri atas beberapa bobot ( $W_1, W_2, W_3, \dots, W_n$ ) saling berhubungan dengan masing-masing *node*.
3. Ambang batas (*Threshold*) adalah nilai ambang batas internal dari *node*. Besaran nilai mempengaruhi aktivasi dari *output node*  $y$ .
4. Fungsi aktivasi (*Activation Function*) ialah operasi matematika yang dikenal pada sinyal *output*  $y$ .

### 2.2.2 Arsitektur Jaringan

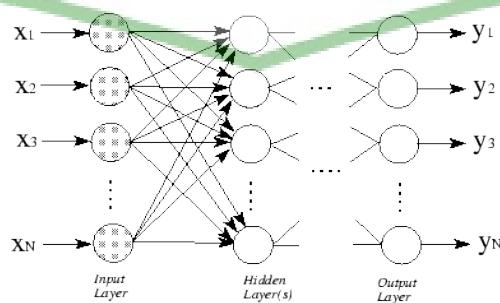
*Neural Network* memiliki beragam variasi bentuk khususnya pada neuron-neuron yang terletak pada lapisan yang sama pun memiliki keadaan yang sama. Faktor yang menentukan sifat suatu neuron adalah nilai bobot (*weight*) dan fungsi aktivasi dari neuron, serta setiap lapisan pada neuron memiliki fungsi aktivasi yang sama. Setiap arsitektur memiliki ciri khas masing-masing, seperti yang ada pada ANN dari yang paling sederhana terdiri atas satu neuron (*single neuron*) sampai yang paling rumit menjadi multi neuron (*multiple neuron*) dalam satu lapis (*single layer*) sampai jaringan *multiple neuron* di dalam *multiple layers*. Arsitektur *neural network* dapat dibagi berdasarkan jumlah lapisannya (Hermawan, 2006), di antaranya sebagai berikut:

1. *Single Layer Neural Network*: Jaringan dengan lapisan yang tunggal dan terdiri atas 1 lapisan *input* dan 1 lapisan *output*. Setiap neuron yang ada di dalam lapisan *input* selalu terhubung dengan setiap neuron yang terdapat pada lapisan *output*. Jaringan ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi.



**Gambar 2.5 Struktur Single Layer Neural Network**

2. *Multiple Layers Neural Network*: Jaringan yang memiliki lapisan dalam berjumlah lebih dari satu atau **jamak** serta memiliki ciri khas tertentu yaitu terdapat 3 jenis lapisan yang terdiri atas *input*, lapisan *output*, dan lapisan tersembunyi (*hidden layer*). Jaringan yang mempunyai lapisan banyak ini dapat menyelesaikan permasalahan yang jauh lebih kompleks dibandingkan jaringan berlapis tunggal. Namun, proses pelatihan (*training*) biasanya membutuhkan waktu yang cukup lama.



**Gambar 2.6 Multiple Layers Neural Network**

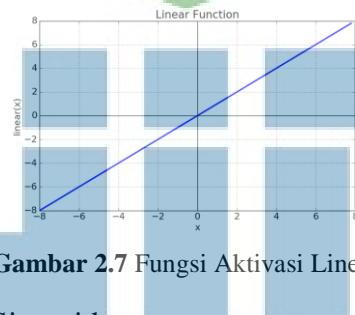
### 2.2.3 Fungsi Aktivasi (*Activation Function*)

Menurut Nurhikmat & Purwaningsih (2018), yang dimaksud dengan fungsi aktivasi (*activation function*) ialah fungsi yang menggambarkan suatu hubungan antara tingkat internal (*summation function*) yang berbentuk linear atau pun non-linear, di mana tujuan dari hal tersebut untuk menentukan apakah neuron perlu diaktifkan atau tidak. Beberapa fungsi aktivasi yang biasa digunakan dalam *Neural Network*, sebagai berikut:

1. Fungsi Aktivasi: Linear

Fungsi dari aktivasi linear adalah suatu fungsi yang mempunyai nilai *output* sama dengan nilai *input*. Apabila suatu neuron memakai aktivasi linear, keluaran dari neuron tersebut merupakan *weighted sum* dari *input* + bias.

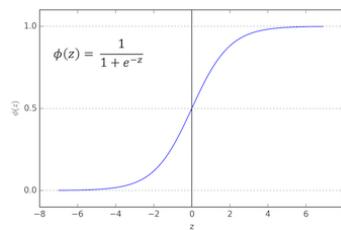
Ilustrasi contoh grafik dari fungsi linear terdapat pada Gambar 2.7.



Gambar 2.7 Fungsi Aktivasi Linear

2. Fungsi Aktivasi: Sigmoid

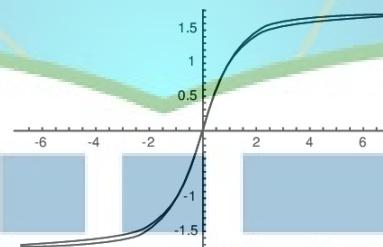
Yang dimaksudkan dari fungsi aktivasi sigmoid adalah fungsi *non-linear*, di mana *input* pada fungsi aktivasi berupa bilangan *real* dan *output*-nya memiliki *range* antara angka nol sampai satu (0–1) yang diilustrasikan contoh grafik dari Gambar 2.8.



Gambar 2.8 Fungsi Aktivasi Sigmoid

### 3. Fungsi Aktivasi: Tanh

Fungsi aktivasi Tanh juga termasuk fungsi *non-linear*, di mana *input* dari fungsi aktivasi berupa bilangan *real* dan *output* dari fungsi memiliki kisaran nilai antara -1 sampai 1. Kelebihan dalam fungsi ini adalah *output* nya dapat *zero-centered*, sedangkan kekurangannya ialah dapat mematikan *gradient*. Ilustrasi contoh grafik diperlihatkan pada Gambar 2.9.

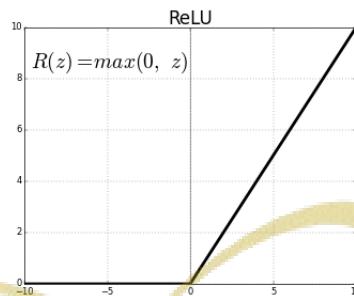


Gambar 2.9 Fungsi Aktivasi Tanh

### 4. Fungsi Aktivasi: ReLU

Fungsi aktivasi dari ReLU (*Rectified Linear Unit*) menggunakan nilai ambang batas (*threshold*) dari nilai 0 hingga tidak terbatas (*infinite*), di mana fungsi ini memiliki *input* dari neuron-neuron bernilai bilangan negatif maka fungsi akan menerjemahkan bilangan yang bernilai negatif menjadi nilai 0 dan jika input bernilai bilangan positif sehingga *output*-nya adalah nilai aktivasi itu sendiri. Kelebihan dari fungsi aktivasi ini adalah mempercepat proses konfigurasi yang dapat dikerjakan dengan *Stochastic Gradient*

*Descent (SGD)* dibanding dengan fungsi aktivasi lain seperti sigmoid dan tanh. Ilustrasi contoh grafik pada Gambar 2.10.



Gambar 2.10 Fungsi Aktivasi ReLU

### 2.3 Convolutional Neural Network (CNN)

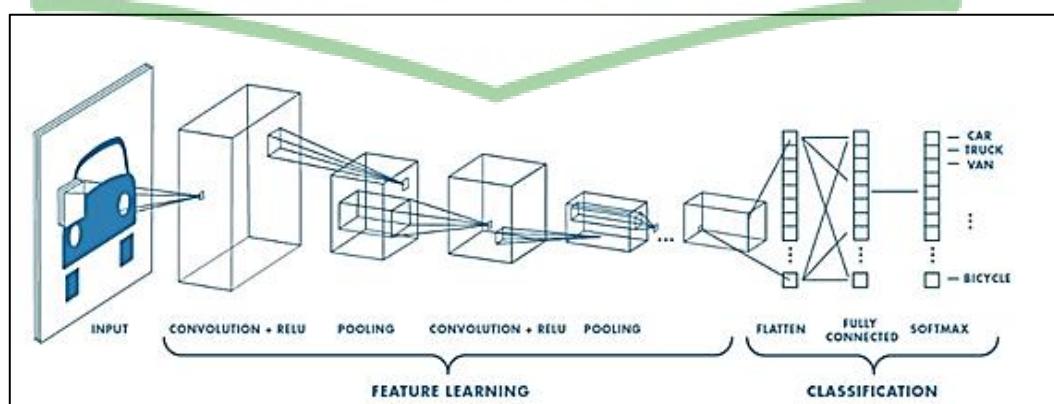
Salah satu metode yang cukup terkenal saat ini disebabkan akurasi yang dihasilkan dari pemanfaatannya untuk memprediksi sesuatu dengan tingkat keakuratan yang tinggi adalah *Convolutional Neural Network* atau disingkat menjadi CNN. Beberapa peneliti telah banyak melakukan pemanfaatan metode CNN untuk menghasilkan prediksi yang memiliki tingkat akurasi tinggi, sehingga terdapat beberapa pengertian yang didefinisikan oleh para praktisi atau tenaga ahli yang berkecimpung pada *deep learning* yang menggunakan metode CNN, di antaranya sebagai berikut:

1. CNN ialah suatu jenis jaringan neural khusus di mana bertujuan untuk pengolahan data yang memiliki topologi seperti *grid* atau struktur kotak (Y LeCun *et al.*, 1989).
2. CNN adalah suatu metode pengembangan yang berawal dari *Multi Layer Perceptron* (MLP) yang di mana penggunaan tersebut didesain untuk mengolah data dua dimensi (W. S. E. Putra, 2016).

3. CNN merupakan suatu penggunaan metode yang berbasiskan jaringan syaraf dengan operasi matematis (konvolusi) sebagai pengganti perkalian matriks yang umum berada pada minimal satu lapisan (*layer*) (Goodfellow, 2016).

Dari pengertian beberapa ahli tersebut dapat disimpulkan oleh peneliti bahwasanya *Convolutional Neural Network* (CNN) adalah suatu metode pengembangan yang berawal dari MLP untuk pengolahan data khususnya pada dua dimensi seperti gambar dan suara dengan berlandaskan jaringan syaraf (*neural network*) di mana kaidah-kaidahnya mengikuti operasi matematis yang biasa disebut dengan konvolusi pada jumlah lapisan tertentu. Tujuan penggunaan CNN tidak lain untuk mengklasifikasikan data yang berlabel dengan menggunakan metode *supervised learning* sebagaimana yang telah dijelaskan pada teori sebelumnya.

Dalam penggunaan secara teknis, CNN merupakan sebuah arsitektur yang dapat dilatih dan memiliki tahapan-tahapan tertentu. Adanya masukan (*input*) dan keluaran (*output*) di mana dari tiap tahap tersebut terdiri atas beberapa *array* yang biasa disebut *feature map* atau fitur peta. Pada setiap tahapannya terdiri atas tiga *layer* yaitu konvolusi, fungsi aktivasi *layer* dan *pooling layer*. Gambar 2.11 merupakan ilustrasi jaringan arsitektur dari *Convolutional Neural Network*.



Gambar 2.11 Jaringan Arsitektur *Convolutional Neural Network*

Dari Gambar 2.11 dijelaskan bahwa penggunaan arsitektur CNN (Yann LeCun, Kavukcuoglu, & Farabet, 2010) sesuai dengan apa yang digunakan oleh Lecun dkk, di mana input dari CNN merupakan berbentuk citra dengan ukuran tertentu. Pada tahap awalan disebut dengan tahapan konvolusi, penggunaan kernel dengan ukuran tertentu menjadi tolak ukur utama kemudian jumlah dari kernel yang digunakan tergantung dari jumlah fitur yang dihasilkan. Berikutnya, *output* dari tahap pertama menuju ke tahap kedua akan dikenakan fungsi aktivasi (*activation function*) dengan penggunaan fungsi tanh atau yang biasa digunakan, *Rectifier Linear Unit* (ReLU). Setelah itu, dari fungsi aktivasi akan melalui proses *sampling* atau *pooling* di mana *output* dari proses *pooling* ialah ukuran citra yang telah berkurang sesuai dari *pooling mask* yang digunakan.

### 2.3.1 Convolution Layer

Lapisan konvolusi (*Convolution Layer*) merupakan bagian dari tahap pada arsitektur CNN sehingga yang dimaksud dengan operasi konvolusi adalah suatu operasi dua fungsi argumen yang bernilai *real* atau nyata. Operasi tersebut merapkan fungsi keluaran atau *output* sebagai *feature map* dari *input* citra, keduanya memiliki nilai yang bersifat riil. Operasi konvolusi dapat dirumuskan sebagai berikut:

$$s(t) = (x * w)(t) \quad (2.1)$$

Fungsi  $s(t)$  memberikan *output* tunggal yaitu *feature maps*, di mana argument pertama yaitu *input* yang variabel-nya  $x$  dan argumen kedua  $w$  sebagai *Kernel* atau *filter*. Namun jika dilihat input bersifat citra dua dimensi maka dapat diasumsikan bahwa  $t$  sebagai piksel lalu menggantinya dengan  $i$  dan  $j$ . Oleh karena itu, operasi

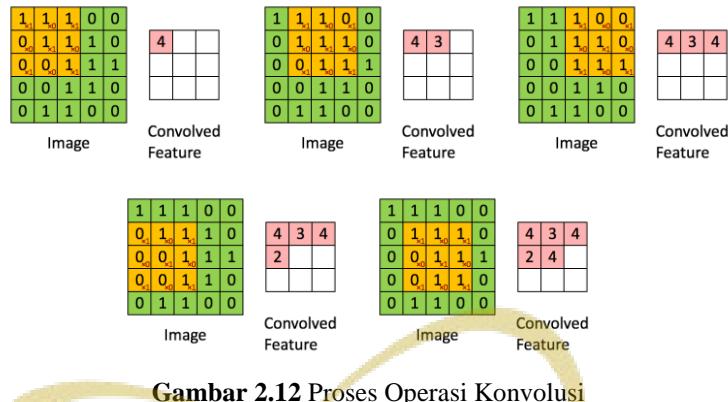
konvolusi ke dalam *input* lebih dari satu dimensi dirumuskan menjadi sebagai berikut:

$$S(i, j) = (K * I)(i, j) = \sum_{i=1}^m \sum_{j=1}^n I(i - m, j - n)K(m, n) \quad (2.2)$$

Persamaan 2.2 adalah perhitungan dasar pada operasi konvolusi dengan  $i$  dan  $j$  ialah piksel dari citra, di mana perhitungan bersifat komutatif dan muncul saat  $K$  sebagai *Kernel* atau *filter*, lalu  $I$  sebagai input dan *Kernel* yang dapat dibalik relatif terhadap input. Operasi konvolusi ini dapat dilihat sebagai perkalian matriks antara citra input dan *kernel* sehingga keluarannya dihitung dengan *dot product*. Adapun penentuan volume *output* dapat ditentukan dari tiap-tiap lapisan input dengan ukuran spasial  $W_1$  pada gambar input dengan *hyperparameters* yang berguna untuk persamaan 2.3 dengan fungsinya menghitung banyaknya neuron aktivasi dalam sekali gambar *output*. Persamaannya adalah:

$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1 \quad (2.3)$$

Persamaan 2.3 merupakan operasi perhitungan untuk mendapatkan *output* dari *layer* gambar input menjadi gambar yang baru dengan ukuran volume yang berbeda, dengan  $W_2$  berfungsi sebagai ukuran volume *output*,  $W_1$  sebagai ukuran volume input,  $F$  sebagai ukuran *filter*,  $P$  sebagai nilai *padding*, dan  $S$  berfungsi sebagai ukuran pergeseran *filter* dalam konvolusi input citra .

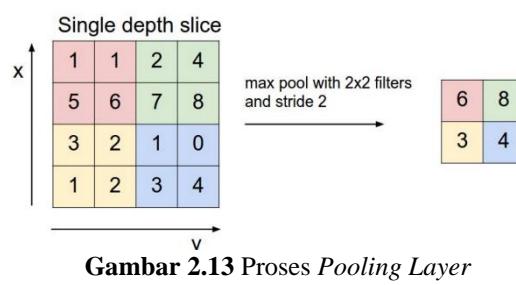


Gambar 2.12 Proses Operasi Konvolusi

Pada Gambar 2.12 terdapat ilustrasi untuk memahami secara intuisi dari konsep konvolusi, di mana istilah matematis yang ada dalam pengolahan citra sehingga dapat dianalogikan seperti adanya sebuah *Kernel* (kotak kuning) pada sebuah citra pada semua *offset* sedangkan kotak berwarna hijau secara keseluruhan adalah citra yang akan dikonvolusi. Tahapan konvolusi biasanya berawal dari letak sudut kiri atas citra yang dilambangkan *Kernel* (kotak kuning) kemudian bergerak ke arah sudut kanan bawah.

### 2.3.2 Pooling Layer

*Pooling layer* terletak setelah operasi *convolution layer*, sehingga prinsip yang dimiliki *pooling layer* yaitu terdiri atas sebuah *filter* dengan ukuran dan *stride* tertentu dengan cara bergeser pada seluruh area *feature map* di mana teknik *pooling* sering digunakan pengguna adalah *Max pooling* dan *Average Pooling*, sebagaimana contoh pada Gambar 2.13.



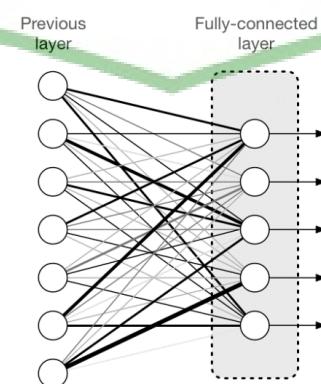
Gambar 2.13 Proses Pooling Layer

Tujuan dari proses *pooling* adalah untuk mengurangi dimensi dari *feature map* (*down sampling*) sehingga meningkatkan kecepatan komputasi yang disebabkan parameter yang harus ter-*update* semakin kecil dan sedikit serta kegunaan lain untuk mengatasi *overfitting*.

### 2.3.3 Fully Connected Layer

Letak dari lapisan *Fully Connected* setelah *pooling layer* di mana lapisan tersebut memiliki neuron aktivasi dari lapisan sebelumnya perlu diubah menjadi data satu dimensi sebelum dapat terhubung ke semua neuron di lapisan selanjutnya yaitu *fully connected layer*, tersebut layaknya jaringan syaraf tiruan (JST). Tujuan dari lapisan *fully connected* untuk mentransformasi pada dimensi data agar data dapat diklasifikasikan secara linear. Perbedaan mendasar antara lapisan *fully connected* dengan lapisan *konvolusi biasa* ialah terletak pada hubungan neuron, jika *konvolusi biasa* neuron terhubung hanya dari daerah input, sementara lain lapisan *fully connected* mempunyai neuron yang seluruhnya terhubung. Pada kedua lapisan tersebut yaitu lapisan *konvolusi* dan *fully connected* masih menggunakan operasi produk *dot*, sehingga fungsi dari keduanya tidak jauh berbeda (Danakusumo, 2017).

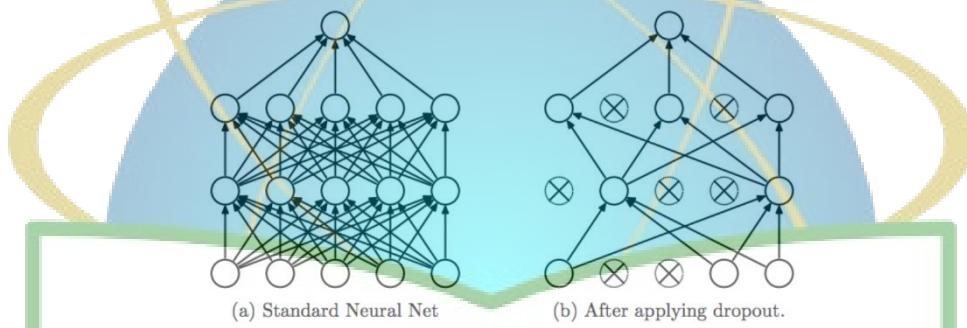
Ilustrasi dari konsep lapisan *fully connected* ditunjukkan pada Gambar 2.14.



**Gambar 2.14** Lapisan *Fully-Connected*

### 2.3.4 Dropout Regulation

*Dropout* merupakan suatu teknik regulasi pada jaringan syaraf untuk menentukan beberapa neuron secara *random* dan beberapa neuron tidak dipakai selama proses pelatihan, dengan arti lain neuron-neuron tertentu akan dibuang secara acak. Pada keadaan tersebut menandakan kontribusi neuron yang terbuang akan diberhentikan sementara dan bobot (*weight*) baru juga tidak digunakan pada neuron saat melakukan *backpropagation*. Gambar 2.15 adalah ilustrasi dari proses *neural network* sebelum dan sesudah menggunakan teknik *dropout*.



**Gambar 2.15** Proses Sebelum dan Sesudah *Dropout*

Pada Gambar 2.15 dijelaskan bahwa pada bagian (a) proses *neural network* yang memiliki dua lapisan *hidden layers* dengan belum melakukan teknik *dropout* sehingga seluruh neuron aktivasi terpakai, sedangkan bagian (b) *neural network* telah melakukan teknik regulasi *dropout* sehingga menyebabkan beberapa neuron *aktivasi* tidak digunakan lagi sementara. Tujuan dari hal tersebut ialah agar memudahkan performa dari model CNN dalam melatih data klasifikasi dan mengurangi potensi terjadnya *overfitting* (Krizhevsky, Sutskever, & Hinton, 2012).

### 2.3.5 Softmax Classifier

Pengertian dari *softmax classifier* adalah suatu bentuk lain dari algoritma *logistic regression* yang bisa digunakan untuk mengklasifikasikan lebih dari dua

kelas atau banyak (multinomial), di mana standar dari klasifikasi yang biasa dilakukan algoritma dari *logistic regression* ialah tugas untuk klasifikasi kelas biner (Putri, 2018). Ditambah lagi bahwasanya *softmax* ialah sebuah fungsi yang mengubah K-dimensi vektor ‘x’ yang bernilai sebenarnya menjadi vektor dengan bentuk yang sama namun bernilai dalam rentang 0 – 1 atau yang jumlahnya 1. Fungsi dari *softmax* ini biasanya berada pada *layer* yang memiliki *neural network* dan terletak didalam *layer* terakhir yang bertujuan untuk menghasilkan *output* (Salsabila, 2018).

Secara umum, neuron pada *softmax* menerima input kemudian melakukan pembobotan dan penambahan bias, akan tetapi neuron tersebut tidak menerapkan fungsi aktivasi (*activation function*) melainkan hanya menggunakan fungsi *softmax*. Bentuk persamaan rumus dari *softmax* adalah sebagai berikut.

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (2.4)$$

Pada persamaan 2.4 dijelaskan bahwa notasi  $f_j$  adalah hasil fungsi untuk setiap elemen ke- $j$  pada vector keluaran kelas. Argumen  $z$  merupakan hipotesis yang diberikan oleh model pelatihan agar dapat diklasifikasi oleh fungsi *softmax*. Pada hasil *softmax* memberikan nilai yang lebih intuitif dan mempunyai interpretasi probabilistik lebih baik disbanding algoritma klasifikasi lain, serta memungkinkan untuk dapat menghitung probabilitas semua label, sehingga dari label yang ada akan diambil nilai yang bersifat riil dan dirubah menjadi vektor dengan nilai kisaran nol dan satu apabila semua dijumlahkan bernilai satu.

### 2.3.6 Cross Entropy Loss Function

*Loss Function* atau *Cost Function* adalah fungsi yang mendeskripsikan kerugian yang terkait dengan semua kemungkinan yang dihasilkan oleh suatu model (Nurhikmat & Purwaningsih, 2018). Tugas dari *loss function* dimulai ketika model pembelajaran memberikan pesan kesalahan yang harus diperhatikan, sehingga *loss function* yang optimal adalah ketika fungsi yang menghasilkan error diharapkan mendapat nilai yang paling rendah.

Penggunaan *crossentropy* adalah saat suatu model memiliki kelas yang banyak sehingga dibutuhkannya cara mengukur perbedaan antara probabilitas hasil hipotesis dan probabilitas kebenaran yang asli serta selama pelatihan, banyak algoritma yang perlu menyesuaikan parameter sehingga perbedaan yang ada perlu diminimalisir. Gambaran secara umum dari teknik ini yaitu meminimalisir kemungkinan *log negative* dari *dataset* yang merupakan ukuran langsung dari hasil prediksi model.

### 2.3.7 Proses Forward Propagation pada CNN

*Forward Propagation* atau yang bisa diartikan propogasi ke depan ialah suatu teknik pada proses pengolahan data pada *input* yang berjalan melewati tiap neuron pada *hidden layer* hingga menuju *output layer* di mana nanti akan dihitung jumlah *error*-nya. Berikut merupakan persamaan dari *forward propagation*:

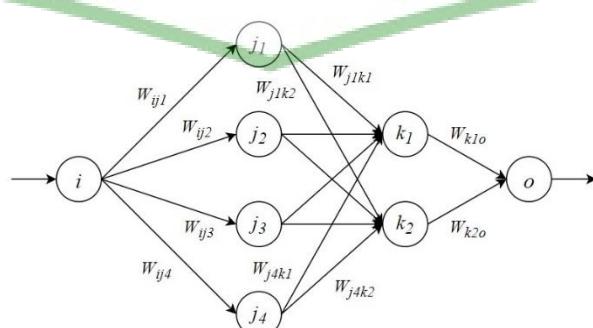
$$dot_j = \sum_i^3 w_{ji}x_i + b_j \quad (2.5)$$

$$h_j = \sigma(dot_j) = \max(0, dot_j) \quad (2.6)$$

Persamaan 2.5 dan 2.6 ialah bentuk contoh *forward pass* yang ada pada arsitektur pertama di mana penggunaannya memakai fungsi aktivasi, ReLU. Dari keterangan tersebut dapat diketahui jika  $i$  adalah *node* pada input *layer* (contoh : 3 *node* input),  $j$  adalah *node* pada *hidden layer* sedangkan  $h$  merupakan *output* dari *node* pada *hidden layer*.

### 2.3.8 Proses *Backward Propagation* pada CNN

Sebagaimana penjelasan dari Kusumadewi yang dikutip dari penilitian tentang *deep learning* oleh Salsabila (2018), *Backpropagation* adalah suatu bentuk algoritma *supervised learning* yang biasanya dipakai oleh *perceptron* dengan lapisan yang banyak dan bertujuan untuk mengubah nilai bobot yang terhubung dengan memperdayakan neuron-neuron yang terdapat pada lapisan tersembunyi (*hidden layer*) (Salsabila, 2018), di mana pemanfaatan dari algoritma ini adalah keterbalikan dari konsep *forward* yaitu, memakai *error output* untuk mengubah nilai bobot dalam arah mundur (*backward*). Adapun syarat dalam mendapatkan nilai *error* tersebut, tahap *forward propagation* dikerjakan terlebih dahulu, kemudian neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi yang dapat diturunkan. Gambar 2.16 adalah gambar dari konsep *backward propagation*.



Gambar 2.16 Jaringan *Backpropagation*

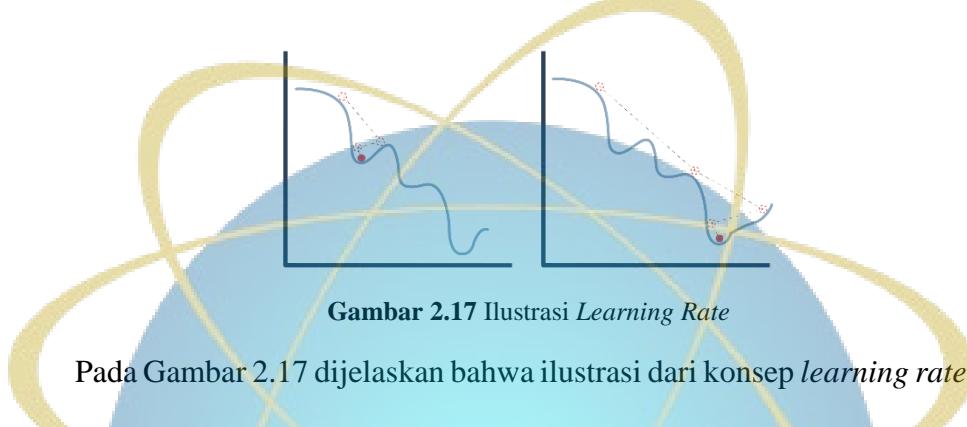
Arsitektur pada jaringan *back-propagation* pada Gambar 2.16 menunjukkan bahwa *neural network* yang terdiri atas 1 unit neuron pada lapisan input (*i*), 4 neuron pada *hidden layer* pertama ( $j_1, j_2, j_3, j_4$ ), kemudian 2 neuron pada *hidden layer* kedua ( $k_1, k_2$ ) dan 1 unit neuron pada lapisan *output* (*o*), di mana terdapat bobot (*weight*) yang menghubungkan lapisan *input* (*i*) dengan neuron pertama pada *hidden layer* pertama adalah  $W_{ij1}, W_{ij2}, W_{ij3}, W_{ij4}$ . Kemudian, bobot yang menghubungkan *hidden layer* pertama dengan neuron kedua pada *hidden layer* kedua adalah  $W_{j1k1}, W_{j1k2}, W_{j2k1}, W_{j2k2}, W_{j3k1}, W_{j3k2}, W_{j4k1}, W_{j4k2}$ . Selanjutnya, nilai bobot neuron pada *hidden layer* kedua terhubung kepada lapisan *output* (*o*) adalah  $W_{k1o}, W_{k2o}$ . Sisanya terdapat bobot bias yang berbalik arah dari lapisan *output* hingga kepada lapisan *input*.

### 2.3.9 Stochastic Gradient Descent (SGD)

Pada sebagian besar algoritma pelatihan yang berkaitan dengan *propogation* memanfaatkan *gradient* dari fungsi kesalahan di mana fungsinya tersebut agar dapat menentukan perubahan bobot dalam rangka menimalkir fungsi pada kesalahan yang ada sehingga bisa mendapatkan optimasi yang optimal pada *neural network* (Nurhikmat & Purwaningsih, 2018).

Adapun makna secara sederhana, dasar-dasar dari algoritma ini untuk mengurangi inisial *weight* dengan sebagian dari nilai *gradient* yang telah didapatkan dengan cara kerjanya adalah *gradient descent* melakukan cara menimalkan fungsi  $J(\theta)$  yang memiliki parameter  $\theta$  dengan meng-*update* parameter ke suatu arah menurun. Dengan sebab algoritma ini dapat menemukan parameter yang dapat menimalkan *loss function* (Ruder, 2016). Namun untuk mengetahui dan

menentukan langkah-lankah yang dapat diambil agar mencapai titik minimum kesalahan maka *Gradient descent* memerlukan teknik khusus yaitu *Learning Rate* ( $\eta$ ). Sebagaimana tergambar pada gambar dibawah ini, apabila suatu objek akan menuruni sebuah bukit dengan mengetahui langkah yang tepat dari nilai *learning rate* yang sesuai maka akan tecapai pada lembah atau titik minimum.



Pada Gambar 2.17 dijelaskan bahwa ilustrasi dari konsep *learning rate* seperti suatu grafik yang tidak linear namun terdapat beberapa jumlah nilai minimum pada grafik sehingga untuk menentukan nilai minimum diperlukannya suatu momentum agar bola dapat terhempas dan mencapai nilai minimum secara akurat.

*Stochastic Gradient Descent* (SGD) adalah metode *gradient descent* yang senantiasa melakukan *update* parameter terus menerus untuk setiap data pelatihan  $x(i)$  serta label  $y(i)$  dan memiliki persamaan 2.7.

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; x^{(i)}; y^{(i)}) \quad (2.7)$$

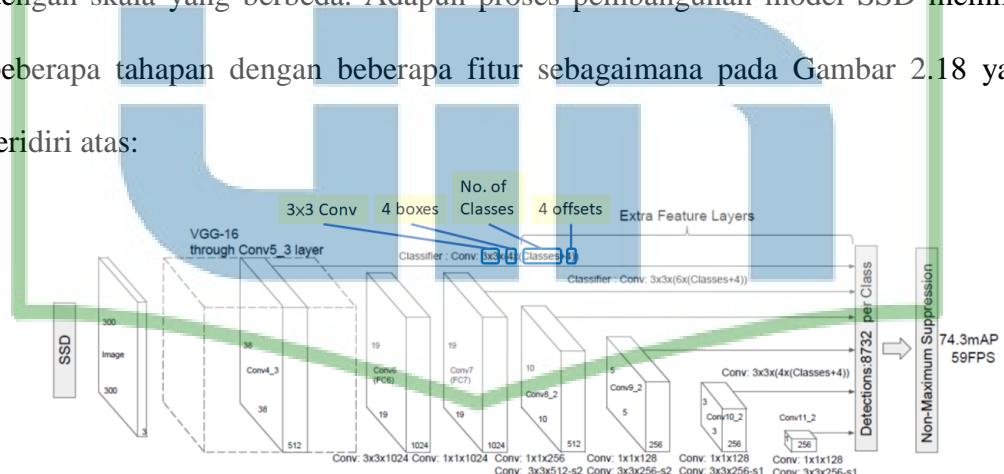
Dikarenakan SGD selalu melakukan *update* dengan varians tinggi maka menyebabkan fungsi objektif meningkat secara tidak beraturan. Disisi lain, hal ini memungkinkan *loss function* melompat ke titik minimal yang baru dan berpotensi melompat ke minimum yang tidak pasti nilainya. Namun hal tersebut dapat dicegah dengan penggunaan teknik *learning rate*, dan SGD pun akan menurunkan *loss function* ke titik minimum dengan nilai yang optimal.

## 2.4 Arsitektur Objek Deteksi

### 2.4.1 Single Shot Detector (SSD)

*Single Shot* atau *Multibox Detector* adalah arsitektur untuk model objek deteksi yang dapat mendeteksi banyak objek dalam satu gambar yang dikembangkan oleh Liu *et al.* (2016). Model ini didesain berbasis *real-time* dan memiliki kecepatan deteksi yang lebih tinggi dibanding dengan arsitektur *You Only Look Once* (YOLO) yang menggunakan prinsip *single shot detector* dan seakurat *Faster R-CNN* yang menggunakan prinsip jaringan wilayah proposal atau *Region Proposal Network* (RPN).

Pada dasarnya, SSD hanya memerlukan gambar input dan *ground truth box* untuk tiap objek pelatihan, lalu evaluasi dilakukan dengan *default box* yang mempunyai aspek rasio yang berbeda untuk tiap lokasinya di beberapa *feature maps* dengan skala yang berbeda. Adapun proses pembangunan model SSD memiliki beberapa tahapan dengan beberapa fitur sebagaimana pada Gambar 2.18 yang terdiri atas:



Gambar 2.18 Ilustrasi Model SSD dengan Jaringan Dasar

#### 1. Ekstraksi *Feature Map*

Model ini terdiri tujuh lapisan, lapisan pertama merupakan tahapan awal untuk input gambar lalu ke ekstraksi *feature map*. Pada ekstraksi pertama dalam model dengan *default* SSD menggunakan jaringan dasar yang berbasis

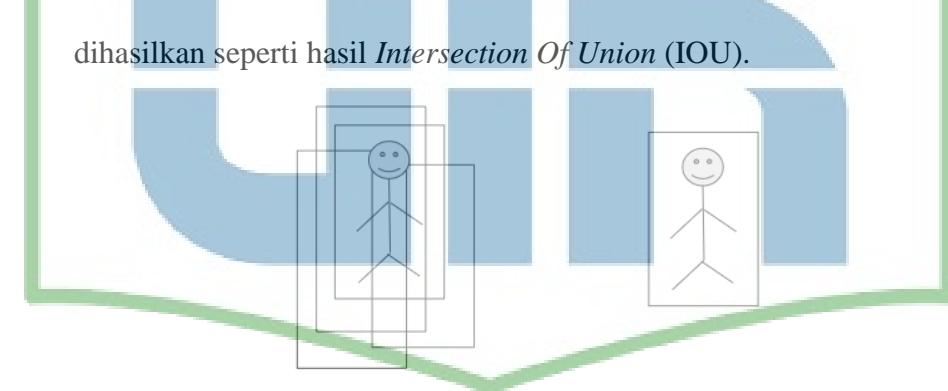
VGG-16 (16 *Layer*) kemudian dilanjutkan dengan enam lapisan tambahan berikutnya, fungsi enam lapisan tersebut untuk melakukan prediksi tambahan yang ukuran skalanya berbeda dan semakin mengecil fitur yang digunakan untuk tiap *layer*-nya.

## 2. *Multiscale Feature Map*

Ekstraksi dilakukan dengan penambahan *layer* fitur konvolusi pada jaringan dasar model konvolusi. Pada layer ini ukuran akan mengecil secara progresif untuk dapat melakukan deteksi dengan skala berbeda.

## 3. *Non Maximum Supression*

Tahap ini digunakan untuk memilih satu entitas seperti kotak pembatas (*bounding box*) dari banyak entitas yang *overlap* dengan cara menyeleksi pada hasil tertentu berdasarkan nilai ambang batas atau *threshold*. Secara umum, kriteria nilai tersebut didapatkan dari angka probabilitas yang dihasilkan seperti hasil *Intersection Of Union* (IOU).



**Gambar 2.19** Sebelum NMS dan Sesudah NMS

Adapun metode yang digunakan berdasarkan perbandingan hasil kinerja akurasi model pada data uji PASCAL VOC (*Visual Object Classes*) dengan

menggunakan berbagai arsitektur atau metode selain SSD yang dilakukan oleh Liu *et al.* (2016) sebagaimana pada Gambar 2.20 dan Gambar 2.21.

Method	data	mAP	aero	bike	bird	boat	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	
Fast	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
Fast	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
Faster	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster	07+12+COCO	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300	07	68.0	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46.0	67.8	69.2	76.6	82.1	77.0	72.5	41.2	64.2	69.1	78.0	68.5
SSD300	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83.0	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74.0
SSD300	07+12+COCO	79.6	80.9	86.3	79.0	76.2	57.6	87.3	88.2	88.6	60.5	85.4	76.7	87.5	89.2	84.5	81.4	55.0	81.9	81.5	85.9	78.9
SSD512	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
SSD512	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
SSD512	07+12+COCO	81.6	86.6	88.3	82.4	76.0	66.3	88.6	88.9	89.1	65.1	88.4	73.6	86.5	88.9	85.3	84.6	59.1	85.0	80.4	87.4	81.2

Gambar 2. 20 Perbandingan nilai mAP

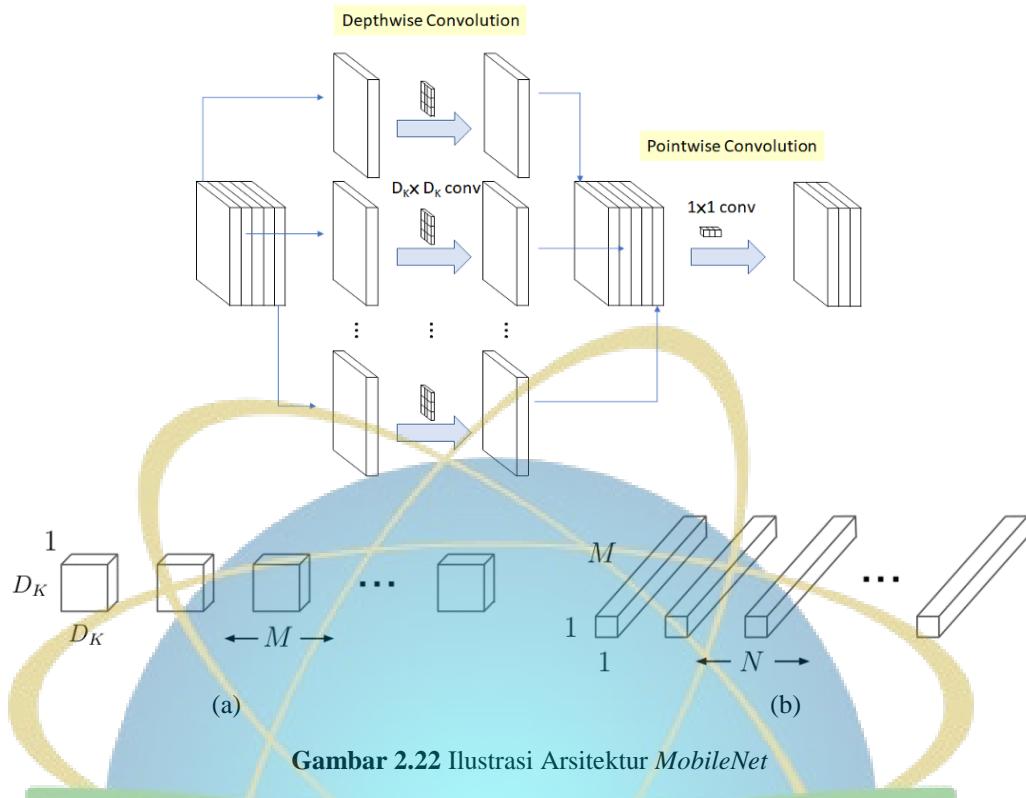
Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Gambar 2. 21 Perbandingan nilai FPS, batch size, box dan resolusi

Dari Gambar 2.20 dan Gambar 2.21 menunjukkan bahwa nilai SSD memiliki nilai mAP yang tinggi dibanding dengan metode lain. Akan tetapi, metode SSD tergantung dengan resolusi input gambar, semakin tinggi nilai resolusi maka semakin tinggi nilai akurasi namun memiliki kelemahan dalam kecepatan prediksi dikarenakan jumlah parameter atau box yang meningkat sehingga menyebabkan kecepatan menjadi menurun.

#### 2.4.2 MobileNet

Model pengembangan yang dilakukan oleh Howard *et al.* (2017) merupakan model yang bertujuan mengurangi beban jumlah operasi konvolusi agar proses ekstraksi fitur menjadi lebih cepat serta mengurangi beban perangkat keras. Teknik yang digunakan dalam mengurangi jumlah operasi adalah *depthwise convolution* dan *pointwise convolution* sebagaimana diilustrasikan pada Gambar 2.20.



Gambar 2.22 Ilustrasi Arsitektur *MobileNet*

Pada Gambar 2.22 menggambarkan bahwa (a) *depthwise convolution* bekerja dengan memisahkan *layer* tiap *depth* lalu dilakukan konvolusi pada tiap *layer* *depth*-nya, hasil dari tiap *layer* lalu digabung kembali menjadi satu, apabila telah digabung maka dilakukan (b) *pointwise convolution* yang menggunakan *filter* konvolusi  $1 \times 1$  untuk menghasilkan keluaran dengan dimensi yang baru.

## 2.5 Evaluasi

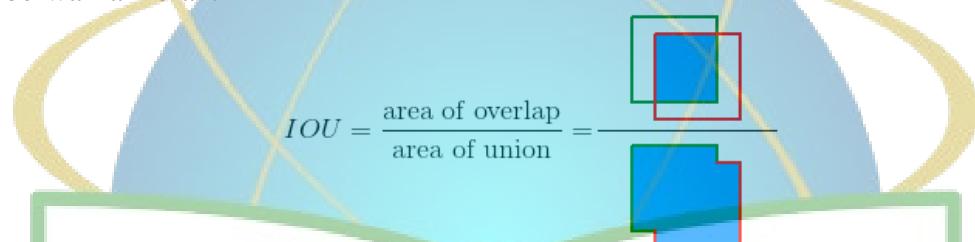
### 2.5.1 Intersection Over Union (IOU)

Menurut Padilla *et al.*(2020), *Intersection Over Union* (IOU) merupakan pengukuran berdasarkan Jaccard Indeks yang mengevaluasi tumpang tindih antara dua kotak pembatas (*bounding box*), dengan membandingkan selisih antara  $B_{gt}$  sebagai kotak pembatas yang benar (*ground-truth bounding box*) dan  $B_p$  kotak pembatas yang diprediksi (*predicted bounding box*).

Tujuannya untuk mengetahui deteksi yang bernilai valid *True Positive* atau *False Positive* dengan mengambil nilai IOU tertinggi dari selisih area yang *overlap* dengan area persatuan di antara kedua kotak pembatas, sebagaimana pada persamaan 2.8.

$$IOU = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})} \quad (2.8)$$

Sedangkan, pada Gambar 2.23 mengilustrasikan IOU di antara *ground-truth bounding box* ( $B_{gt}$ ) yang berwarna hijau dan *predicted bounding box* ( $B_p$ ) yang berwarna merah.



Gambar 2.23 Ilustrasi dari IOU

### 2.5.2 Mean Average Precision (mAP)

Suatu performa dapat diukur untuk menentukan baik atau tidaknya dengan model klasifikasi yang bisa terlihat dari parameter pengukuran seperti tingkat akurasi, *precision* dan *recall*, di mana untuk mengoperasikan perhitungan tersebut dibutuhkannya sebuah matriks yang disebut *confusion matrix* (S. J. Putra et al., 2019).

		Actual Value (as confirmed by experiment)	
		positives	negatives
Predicted Value (predicted by the test)	positives	<b>TP</b> True Positive	<b>FP</b> False Positive
	negatives	<b>FN</b> False Negative	<b>TN</b> True Negative

Gambar 2.24 Confusion Matrix

Berdasarkan Gambar 2.24 terdapat beberapa definisi dalam pengukuran klasifikasi yaitu ‘*True Positive*’ (TP) merupakan nilai yang diprediksi benar oleh model dan memiliki nilai aktual yang benar, ‘*False Positive*’ (FP) merupakan nilai yang diprediksi benar oleh model dan memiliki beberapa nilai aktual yang keliru, ‘*False Negative*’ (FN) ialah nilai yang diprediksi benar oleh model namun ada nilai aktual yang belum diprediksi atau prediksi bernilai salah, dan ‘*True Negative*’ (TN) adalah nilai yang diprediksi keliru oleh model dan memiliki nilai aktual yang benar, namun dalam konsep objek deteksi ini tidak berlaku.

Sedangkan, dalam konsep pengukuran model objek deteksi memiliki konsep dasar yang berbeda untuk dapat mengukur hasil prediksi klasifikasi dan objek deteksi (Padilla et al., 2020), yaitu:

1. *True Positive* (TP): Memiliki ciri deteksi dan klasifikasi yang benar, dengan nilai IOU  $\geq$  ambang batas atau *threshold*.
2. *False Positive* (FP): Memiliki ciri klasifikasi benar namun nilai IOU  $< \text{threshold}$  dan memiliki deteksi yang keliru atau berlipat ganda.
3. *False Negative* (FN): Memiliki ciri objek yang belum terdeteksi dan klasifikasi yang salah dengan nilai IOU  $\geq \text{threshold}$ .
4. *True Negative* (TN): Tidak digunakan.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \quad (2.9)$$

Persamaan akurasi 2.9 befungsi sebagai parameter sebagaimana keakuratan suatu model melakukan objek deteksi. Sementara itu, untuk menghitung tingkat presisi dapat digunakan persamaan 2.10.

$$Precision = \frac{TP}{TP + FP} \quad (2.10)$$

Dengan presisi mendeskripsikan seberapa besar keakuratan suatu model dalam meprediksi kejadian positif ( $P$ ) pada serangkaian kegiatan prediksi. Selain dari pada presisi dan akurasi, untuk dapat melihat lebih detail kinerja suatu sistem, *recall* atau sensifitas sistem terhadap suatu kelas juga dapat dihitung menggunakan persamaan 2.11.

$$Recall = \frac{TP}{TP + FN} \quad (2.11)$$

Agar mudah mengidentifikasi nilai rata-rata *precision* (*Average Precision*) maka diperlukannya plot kurva *precision-recall* (PR), namun terlebih dahulu mencari tahu nilai *precision interpolation* pada setiap objek. Tujuannya untuk mengurangi dampak goyangan ‘wiggles’ yang disebabkan oleh variasi kecil dalam peringkat deteksi (Everingham *et al.*, 2010). Formula *precision interpolation* dan *average precision* (AP) sebagaimana pada persamaan 2.12 dan 2.13.

$$p_{interp}(r) = \max p(\tilde{r}) \quad (2.12)$$

$$AP = \frac{1}{11} \sum_{r \in \{0.0, 0.1, \dots, 0.9, 1\}} \rho_{interp}(r) \quad (2.13)$$

## 2.6 Citra Digital

Dalam mendefinisikan citra digital, terdapat beberapa penjelasan dari beberapa ahli yaitu sebagai berikut:

1. Citra digital adalah representasi dua dimensi yang berisi kumpulan piksel-piksel atau titik-titik yang berwarna dari dunia visual serta menyangkut

berbagai macam disiplin ilmu baik mencakup seni, *human vision*, astronomi, teknik, dan sebagainya (Tsoi & Pearson, 1991).

2. Citra ialah *array* dari bermacam nilai di mana sebuah nilai tersebut merupakan sekumpulan angka yang mendeskripsikan atribut dari piksel di dalamnya (Mukherjea & Foley, 1996).
3. Citra digital adalah suatu sinyal diskrit berbentuk 2 dimensi, adapun secara matematis yakni sinyal dapat dipresentasikan sebagai fungsi dari kumpulan variabel 2 dimensi (Wolfram, 2002).

Jadi dapat disimpulkan bahwa citra digital merupakan suatu bentuk representasi dua dimensi yang terdapat piksel-piksel di mana isi dari piksel tersebut merupakan suatu *array* yang memiliki bermacam nilai dengan kumpulan angka sehingga mendeskripsikan suatu gambar grafik tertentu.

Sebuah citra digital dapat dinyatakan dalam bentuk matriks dua dimensi  $f(x,y)$  yang terdiri atas kolom  $M$  dan baris  $N$ , perpotongan antara kolom dan baris bisa disebut piksel, sehingga variabel dari ‘x’ dan ‘y’ adalah koordinat piksel pada matriks sedangkan ‘f’ ialah derajat intensitas piksel. Nilai yang dimiliki pada koordinat  $(x,y)$  mewakili intensitas atau warna dari piksel pada titik tersebut.

Sebuah citra dapat dituliskan dalam sebuah matriks, sebagai berikut:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0, M - 1) \\ f(1,0) & f(1,1) & \cdots & f(1, M - 1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N - 1,0) & f(N - 1,1) & \cdots & f(N - 1, M - 1) \end{bmatrix} \quad (2.14)$$

Berdasarkan persamaan 2.15 maka suatu citra pada  $f(x,y)$  dapat dituliskan ke dalam fungsi matematis seperti berikut:

$$0 \leq x \leq M - 1 \quad (2.15)$$

$$0 \leq y \leq N - 1$$

$$0 \leq f(x, y) \leq G - 1$$

Dengan  $M$  adalah banyaknya jumlah baris pada *array* citra,  $N$  adalah banyaknya jumlah kolom pada *array* citra dan  $G$  adalah skala nilai keabuan pada citra. Namun, jika diperhatikan citra digital terdapat titik-titik kecil berbentuk segi empat yang membentuk citra tersebut sehingga satuan terkecil dari sebuah citra digital biasanya disebut dengan *picture element*, *pixel*, atau pel. Oleh karena itu jumlah piksel per-satuan panjang akan mempengaruhi resolusi dari citra di mana semakin banyak nilai piksel maka semakin tinggi tingkat resolusi serta makin halus dan tajam daripada citra digital yang terlihat.

### 2.6.1 Tipe Citra Digital

1. **Citra Biner:** merupakan citra yang memiliki 2 nilai khusus untuk setiap pikselnya yaitu antara nilai 0 atau 1 (Hidayatullah, 2017). Nilai 0 mewakili warna hitam sedangkan pada nilai 1 mewakili warna putih sehingga jenis citra ini hanya memerlukan 1-bit agar dapat menyimpan nilai pada setiap piksel, ditambah lagi waktu pemroses menjadi lebih cepat dan dapat dilakukan dengan berbagai operasi logika seperti AND, OR, NOT, XOR. Dalam penggunaan

citra ini biasanya dibutuhkan untuk proses segmentasi citra atau proses *masking*, contoh citra biner seperti Gambar 2.25.



Gambar 2.25 Contoh Citra Biner

2. **Citra Grayscale:** merupakan suatu citra yang mempunyai 1 kanal sehingga yang ditampilkan pada citra hanya berbentuk nilai intensitas atau bisa disebut dengan derajat keabuan (Hidayatullah, 2017). Dengan sebab tersebut, citra *grayscale* mempunyai tempat penyimpanan yang jauh lebih hemat apalagi jenis citra ini bisa disebut juga sebagai 8-bit *image* dikarenakan pada setiap nilai piksel membutuhkan penyimpanan sebesar 8-bit. Gambar 2.26 adalah contoh dari citra *grayscale*.



Gambar 2.26 Contoh Citra *Grayscale*

3. **Citra Berwarna (RGB):** merupakan citra yang terdapat 3 buah kanal warna didalamnya di mana terbentuknya citra tersebut berasal dari komponen merah atau *red* (R), hijau atau *green* (G), biru atau *blue* (B) kemudian dimodelkan dalam ruang warna menjadi RGB (Hidayatullah, 2017). Istilah RGB

merupakan standar yang digunakan untuk menampilkan citra berwarna baik pada layar televisi maupun layar komputer. Adapun citra berwarna lain seperti CMYK (*Cyan, Magenta, Yellow, Black*), HSV (*Hue, Saturation, Value*), YcbCr (*Luma, Chroma blue, Chroma red*), dan Lab ( $L^*a^*b$ ), citra berwarna lain memiliki keutamaan sendiri tergantung pada pemanfaatan untuk hal tertentu. Adapun dalam citra berwarna membutuhkan penyimpanan sebesar 24-bit, sehingga pada masing-masing piksel di setiap kanal mempunyai kemungkinan nilai 256, yaitu nilai sekitar 0-255. Oleh sebab itu, setiap piksel pada satu kanal membutuhkan 8-bit data maka citra kanal memiliki 3 buah kanal sehingga untuk satu piksel memerlukan  $3 \times 8$  bit = 24 bit. Dengan kombinasi warna yang dimiliki, citra berwarna mempunyai potensi jumlah warna yang bervariatif sebanyak  $256 \times 256 \times 256 = 2^{24} = 16.777.216$  variasi warna. Gambar 2.27 adalah contoh dari citra berwarna (RGB).



Gambar 2.27 Contoh Citra Berwarna (RGB)

### 2.6.2 Pengolahan Citra Digital (*Image Processing*)

Pengolahan citra digital atau *image processing* adalah suatu proses dalam mengolah piksel-piksel pada citra digital untuk tujuan tertentu. Sebagaimana pada penjelasan sebelumnya, citra digital termasuk gambar yang dihasilkan dari proses komputer, kamera, dan sebagainya. Pengolahan citra digital diproses dengan

memanfaatkan algoritma tertentu, dipresentasikan dengan matriks, sehingga memanipulasi elemen-elemen matriks yang berupa piksel (A'la, 2016).

Tujuan dari pengolah citra untuk memperbaiki kualitas gambar baik dari aspek radiometrik seperti mengatur kontras, translasi, skala, transformasi geometrik. Pengolahan citra dilakukan juga untuk bisa mendapatkan informasi atau deskripsi suatu objek atau pengenalan objek yang terkandung pada citra tersebut (Hermawati, 2013).

## 2.7 Tools

### 2.7.1 Python

Dalam sejarahnya, bahasa pemrograman python dirilis pertama kali oleh Guido Van Rossum di *Scitchting Mathematisch Centrum* Belanda pada tahun 1991.

Sebab mula dia menggunakan **nama Python dikarenakan ia penggemar grup komdei Inggris bernama Monty Python** (Wahyono, 2018). Dari versi awal yaitu Python versi 1.0 pada tahun 1994 **telah berkembang sampai saat ini menjadi 2 jenis Python dengan versi berbeda dan terbaru yaitu versi 3.7 dan versi 2.7.**

Perbedaan mendasar pada Python versi 2.0 mempunyai *Python Enhacement Proposal* (PEP) dengan kemampuan dalam pemberian tuntutan informasi bagi para pengguna serta peningkatan dukungan untuk *Unicode* dan berbagai macam fitur-fitur *programmatical* yang berguna untuk manajemen memori.

Sedangkan pada Python versi 3.0 adalah pengembangan dari Python 2.0 dengan berfokus pada *codebase* dan menghilangkan *redundancy*, bahkan perbedaan terbesarnya yaitu membuat *statement print* dalam fungsi yang *built-in* dan ditambah

pengingkatan dukungan yang beralih ke Python 3.0 sehingga berkurangnya dukungan Python 2.0.

### 2.7.2 Android

Beberapa sistem operasi (*Operating System / OS*) pada *platform smartphone* yang bersifat *mobile* dan menjadi salah satu sistem operasi yang menjadi *trending* saat ini adalah Android, yaitu sebuah sistem operasi yang *open source* berasal dari Google. Adapun beberapa pendapat dari para ahli mengenai OS Android adalah:

1. Menurut Safaat (2012), Android merupakan sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Android umum digunakan di *smartphone* dan juga *tablet PC*, fungsinya pun sama seperti sistem operasi Symbian di Nokia, iOS di Apple dan BlackBerry OS (Safaat, 2012).
2. Adapun menurut para pihak dari pengembang (*developer*) Android bahwasanya, *Android* adalah software untuk perangkat mobile yang meliputi sistem operasi, *middleware* dan aplikasi inti. *Android* dilengkapi dengan *Android SDK* (*Software Development Kit*) yang menyediakan *tools* dan mendukung kebutuhan *API* (*Application Programming Interface*) yang diperlukan untuk mengembangkan aplikasi pada *platform* Android dengan menggunakan bahasa pemrograman Java (Android Developers, 2018).
3. Sedangkan keterangan dari Mulyana, *Android* merupakan sebuah *platform* untuk perangkat bergerak (*mobile devices*) yang semakin popular. Bahkan, beberapa perusahaan riset telah menobatkan android sebagai jawara ponsel

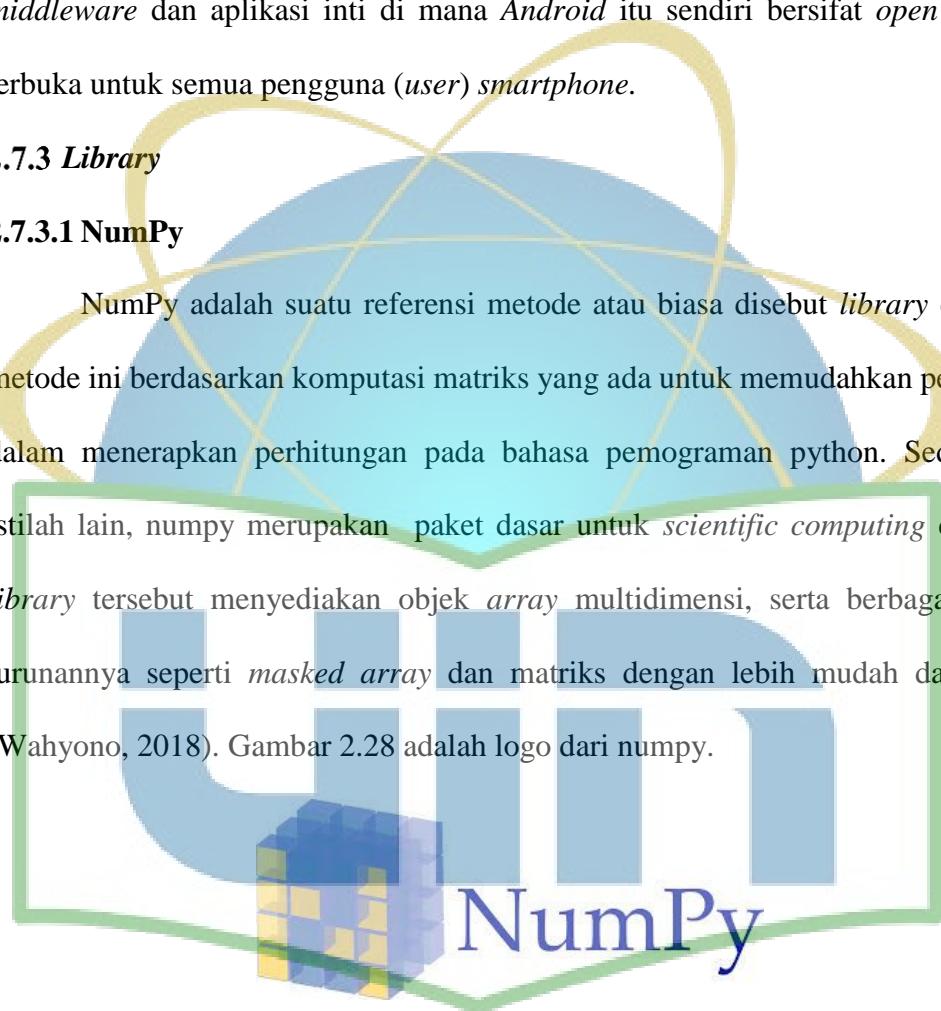
pintar (*smartphone*) melebihi *platform* yang lain, seperti *Symbian* atau *AppleOS* (Mulyana, 2012).

Jadi, dapat disimpulkan pengertian OS *Android* adalah sebuah sistem operasi yang merupakan *platform* untuk *mobile devices* yang meliputi sistem operasi, *middleware* dan aplikasi inti di mana *Android* itu sendiri bersifat *open source*, terbuka untuk semua pengguna (*user*) *smartphone*.

### **2.7.3 Library**

#### **2.7.3.1 NumPy**

NumPy adalah suatu referensi metode atau biasa disebut *library* di mana metode ini berdasarkan komputasi matriks yang ada untuk memudahkan pengguna dalam menerapkan perhitungan pada bahasa pemrograman python. Sedangkan istilah lain, numpy merupakan *paket dasar untuk scientific computing* di mana *library* tersebut menyediakan objek *array* multidimensi, serta berbagai objek turunannya seperti *masked array* dan matriks dengan lebih mudah dan cepat (Wahyono, 2018). Gambar 2.28 adalah logo dari numpy.



**Gambar 2.28 Logo Library NumPy**

### 2.7.3.2 Pandas

Pandas merupakan salah satu dari *library* pada bahasa pemrograman python yang menyediakan struktur data yang cepat, fleksibel, dan ekspresif dengan tujuan untuk membuat data relasional dan melabelkan data menjadi lebih mudah (McKinney & Team, 2015). Gambar 2.29 merupakan logo dari pandas.



Gambar 2.29 Logo Library Pandas

### 2.7.3.3 Tensorflow

Tensorflow adalah sebuah *framework* pada *machine learning* yang dibangun secara *opensource* dengan bantuan dukungan dari perusahaan teknologi terbesar di dunia, Google. Secara umum bahwasanya Tensorflow menyediakan antarmuka untuk mengekspresikan bermacam-macam algoritma *machine learning* secara fleksibel dan bisa dijalankan pada beragam sistem (Abadi et al., 2016).

Apabila ditujukan untuk aplikasi dalam pengenalan objek (*object recognition*) tensorflow dapat melakukan beragam jenis algoritma, termasuk didalamnya untuk proses *training* dan *inference* yang digunakan pada berbagai model *deep neural network*. Ditambah lagi, Tensorflow pun juga dapat termasuk sebagai *library* atau pustaka untuk *data science* paling popular dengan berbagai

jumlah pengembang dan dukungan komunitas yang begitu besar. Gambar 2.30 adalah logo dari *library Tensorflow*.



**Gambar 2.30 Logo Library Tensorflow**

## 2.8 Populasi dan Teknik *Sampling*

Menurut Indrawan & Yaniawati (2014), Populasi adalah kumpulan dari keseluruhan elemen yang dijadikan objek penelitian kemudian ditarik sebuah kesimpulan. Dalam berbagai penelitian ataupun survei, penggunaan sampel adalah sebuah konsekuensi logis dari keterbatasan sumber daya manusia, tenaga, waktu, serta biaya. Sedangkan menurut Sugiyono (2013), teknik pengambilan sampel (*sampling*) dikelompokkan menjadi dua, yaitu:

### 1. *Probability Sampling*

Merupakan teknik *sampling* dengan kaidah-kaidah peluang yang sama bagi setiap anggota populasi untuk dipilih menjadi sampel (Nursiyono, 2015).

Teknik ini terdiri atas:

#### a. *Simple Random Sampling*

Pengambilan anggota sampel dari populasi dilakukan secara acak tanpa memperhatikan strata dalam populasi tersebut (Sugiyono, 2013).

Ditambah, teknik paling mudah dan dapat digunakan dalam populasi yang relative homogen (Nursiyono, 2015)

#### b. *Proportionate Stratified Random Sampling*

Teknik untuk populasi yang memiliki anggota non homogeny dan memiliki strata secara proposional.(Sugiyono, 2013)

c. *Cluster Sampling*

Teknik untuk menentukan sampel pada objek yang memiliki data atau sumber yang luas, bahkan sering dilakukan dengan dua tahap yaitu menentukan wilayah sampel dan menentukan orang-orang yang ada pada wilayah tersebut secara *sampling*.

2. *Nonprobability Sampling*

Keterbalikan dari sebelumnya, cari ini menggunakan sampel tanpa kaidah-kaidah peluang sehingga hasil bersifat subyektif dikarenakan setiap elemen populasi tidak memiliki peluang yang sama untuk dipilih menjadi sampel

(Nursiyono, 2015). Teknik ini terdiri atas:

a. *Convenience Sampling*

Teknik pengambilan sampel berdasarkan kemudahan, dikarenakan pengambilan secara langsung menghubungi unit-unit pengambilan sampel yang mudah dijumpai. Teknik ini tidak mewakili populasi dan hanya cocok untuk penelitian yang sifatnya eksploratif atau untuk *pilot study* (Nursiyono, 2015).

b. *Quota Sampling*

Teknik berdasarkan kuota atau menentukan jumlah sampel terlebih dahulu dimana sampel ini biasanya sering digunakan dalam survei opini publik (Nursiyono, 2015).

c. *Purposive Sampling*

Pengambilan sampel berdasarkan kriteria tertentu, pemikiran atau pengetahuan pengambil sampel (Nursiyono, 2015). Pemilihan kelompok subjek bedasarkan ciri-ciri atau sifat tertentu yang dianggap punya hubungan erat dengan sifat-sifat populasi yang telah ditentukan sebelumnya.

Berdasarkan penjelasan sebelumnya, teknik pengambilan sampel mempengaruhi hasil kesimpulan data. Adapun menurut Indrawan & Yaniawati (2014) menjelaskan batasan untuk uji statistic akan efektif jika sampel digunakan berjumlah 30 sampai 60, ataupun 120 sampai 250.

## 2.9 Bahasa Sebagai Alat Komunikasi

Secara umum, pengertian dari bahasa adalah sebuah bentuk percakapan; sedangkan dalam ruang lingkup wacana linguistik bahasa dapat dimaksudkan sebagai suatu simbol bunyi bermakna dan berartikulasi yang bersifat arbitrer dan konvensional (Sobur, 2006). Bahkan diantara banyaknya perbedaan bahasa merupakan suatu tanda kebesaran dari Allah *Subhanahu wa ta'ala* sebagaimana yang tecantum dalam Al-Quran pada surat Ar Rum ayat 22, hal ini ditunjukkan pada firman-Nya yang berbunyi.

وَمِنْ آيَاتِهِ خَلْقُ السَّمَاوَاتِ وَالْأَرْضِ وَاحْتِلَافُ الْسَّمَاءِ كُمٌ وَالْأَوَانِ كُمٌ إِنَّ فِي ذَلِكَ لَآيَاتٍ لِّلْعَالَمِينَ

Artinya: “Dan di antara tanda-tanda kekuasaan-Nya ialah menciptakan langit dan bumi dan berlain-lainan bahasamu dan warna kulitmu. Sesungguhnya pada yang demikian itu benar-benar terdapat tanda-tanda bagi orang-orang yang mengetahui.”  
(Q.s Ar Rum:22)

Dari pengertian tersebut diartikan bahwasanya bahasa merupakan suatu ungkapan yang memiliki ragam jenis dan mengandung maksud untuk menyampaikan sesuatu kepada orang lain. Tujuan dari hal tersebut adalah agar seseorang bisa dipahami dan dimengerti oleh lawan bicara melalui bahasa yang diungkapkan dan memiliki kapabilitas untuk menyatakan lebih daripada apa yang dimaksud ada disampaikan.

Dengan bahasa, manusia bisa berkomunikasi antar sesama sehingga manusia biasa disebut sebagai makhluk sosial yang saling membutuhkan antar lainnya, oleh sebab itu fungsi daripada bahasa amat begitu penting bagi kehidupan manusia terutama sebagai fungsi komunikatif. Namun dalam rujukan pada definisi diatas, bahasa hanya dapat dilakukan jika organ pendengaran dan bicara kita berfungsi, sehingga informasi yang berupa simbol sandi konseptual secara vocal dapat tersampaikan kepada penerima pesan. Bahasa pun terbatas disesuaikan dengan penggunaan pada komunitas di mana bahasa tersebut harus disetujui dan dipahami bersama pengertiannya sehingga dikenal perbedaan bahasa bergantung pada tiap kebudayaan atau kelompok manusia yang menggunakannya.

### 2.9.1 Pengertian Komunikasi Nonverbal

Definisi sederhana dari komunikasi nonverbal yang dijelaskan oleh Mulyana (2015) adalah sebagai komunikasi tanpa kata-kata atau dengan selain kata-kata yang digunakan. Ditambah dalam penjelasan dari Hardjana (2003), komunikasi verbal bisa dikatakan suatu bentuk penyampaian pesan yang dibungkus tanpa adanya kata-kata.

Definisi lain menurut Malandro dan Barker yang dikutip dari Sunarwinadi (1993) bahwasanya komunikasi antar budaya memberikan kesan memiliki batasan-batasan dalam berkomunikasi nonverbal yang berarti komunikasi tanpa adanya kata-kata, dan terjadi jika tiap individu yang saling berinteraksi tanpa adanya suara atau setiap hal yang dilakukan oleh seseorang yang diberi makna oleh orang lain bisa dengan ekspresi wajah, sentuhan, waktu, gerak isyarat, bau, perilaku mata dan lain-lain.

Dari kesimpulan beberapa tokoh tersebut dapat disimpulkan definisi yang dimaksud dari komunikasi nonverbal ialah suatu bentuk penyampaian pesan komunikasi yang dilakukan oleh seseorang atau individu kepada individu yang lain tanpa menggunakan kata-kata dan suara namun memiliki makna tertentu seperti ekspresi dari mimik wajah, kontak mata, sentuhan waktu, gerak isyarat, dan lain-lainnya.

### 2.9.2 Pengertian Bahasa Isyarat

Definisi bahasa isyarat yang diungkapkan oleh Yunanda, dkk. adalah bahasa yang digunakan oleh orang yang memiliki berkebutuhan khusus untuk berkomunikasi dengan cara manual, bahasa gerakan tubuh, dan gerakan bibir daripada menggunakan bunyi dan suara untuk berkomunikasi (Yunanda, Mandita, & Armin, 2018). Sedangkan yang dimaksud dari orang yang berkebutuhan khusus (tunarungu) ialah biasanya pengguna utama dari bahasa isyarat untuk praktiknya dengan cara mengkombinasikan bentuk tangan, gerak tangan, lengan serta gerak tubuh dan ekspresi pada wajah untuk saling berinteraksi dan menyatakan apa yang dipikiran mereka baik kepada sesama atau individu lain.

Adapun penjelasan yang dimaksud dari bahasa isyarat oleh Maulida merupakan bahasa yang mengutamakan komunikasi manual, bahasa tubuh dan gerak bibir, bukannya suara untuk berkomunikasi di mana kaum tunarungu sebagai pengguna utama dan komunikasi non verbal termasuk bagian dari bahasa isyarat yang berbentuk kombinasi antara orientasi gerak tangan, lengan, tubuh serta ekspresi wajah yang mengungkapkan isi pikiran (Maulida, 2017). Dapat disimpulkan bahwa yang dimaksud dengan bahasa isyarat adalah suatu bahasa isyarat yang digunakan oleh orang yang memiliki berkebutuhan khusus untuk berkomunikasi dengan cara menyampaikan bentuk kombinasi berupa gerak tangan, lengan, tubuh dan ekspresi wajah yang mengungkapkan isi pikiran dari individu ke individu lain.

Adapun belum adanya penetapan bahasa isyarat internasional yang disepakati karena terdapat perbedaan baik dimasing-masing daerah atau negara, sehingga bahasa isyarat biasanya berkembang sesuai dengan lingkungan dan budaya setempat yang memiliki perbedaan namun tetap mempunyai makna yang sama, salah satunya di Amerika Serikat (Ali et al., 2013), Inggris, serta bahasa lainnya.

Bahkan, para penderita tuna rungu dan tuna wicara di Indonesia berkomunikasi menggunakan bahasa isyarat yang mengacu pada dua sistem yaitu, SIBI (Sistem Isyarat Bahasa Indonesia) yang merupakan hasil rekayasa orang normal yang sama dengan sistem isyarat Amerika yaitu ASL (American Sign Language) (Martin, Frehadthomo, & Putri, 2015) dan BISINDO (Bahasa Isyarat Indonesia) yang dikembangkan oleh difabel rungu sendiri melalui komunitas GERKATIN (Gerakan Kesejahteraan Tunarungu Indonesia).

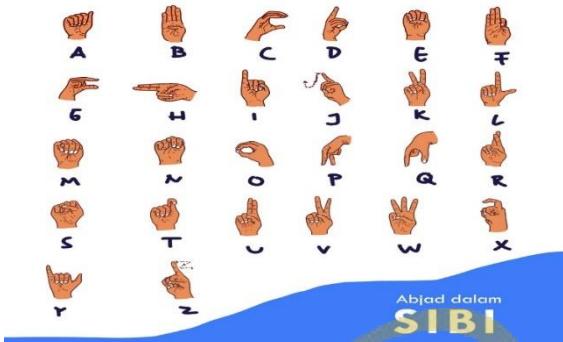
### 2.9.3 Jenis Bahasa Isyarat

#### 1. Sistem Isyarat Bahasa Indonesia (SIBI)

Menurut penjelasan dari Wasita (2012), Sistem Isyarat Bahasa Indonesia (SIBI) adalah suatu media yang membantu interaksi komunikasi sesam kaum tuna rungu didalam masyarakat yang lebih luas yang mengacu pada sistem isyarat struktural bukan sistem isyarat konseptual. Definisi lain dari SIBI adalah suatu sistem isyarat bahasa yang dibakukan sebagai salah satu media yang membantu komunikasi sesama tunarungu ataupun komunikasi penyandang tunarungu di dalam masyarakat yang lebih luas dengan tata makna yang terdiri atas (Hakim & Samino, 2008):

- a. Kata-kata yang memiliki makna yang sama atau sinonim diisyaratkan dengan tempat arah frekuensi **yang sama tapi dengan** penampilan yang berbeda.
- b. Kata yang sama dengan makna yang berbeda (tergolong polisemi) dilambangkan dengan **isyarat yang sama**
- c. Beberapa kata yang memiliki makna berlawanan (antonim) yang diisyaratkan dengan penampilan dan tempat yang sama tapi arah gerakan berbeda.

Dari penjelasan tentang pengertian Sistem Isyarat Bahasa Indonesia (SIBI) oleh beberapa tokoh sebelumnya dapat disimpulkan bahwa SIBI adalah suatu sistem bahasa isyarat yang menjadi media dalam berkomunikasi sesama tunarungu atau masyarakat luas dengan mengacu kepada sistem isyarat struktural sebagaimana contoh pada huruf abjad SIBI pada Gambar 2.31.



**Gambar 2.31** Huruf Abjad Sistem Isyarat Bahasa Indonesia (SIBI)

## 2. Bahasa Isyarat Indonesia (BISINDO)

Pengertian yang dimaksud BISINDO, menurut Dewan Pengurus Daerah Gerakan untuk kesejahteraan tunarungu Indonesia (DPD Gerkatin DKI Jakarta) adalah suatu sistem komunikasi yang bersifat praktis dan efektif untuk penyandang tunarungu Indonesia dikembangkan oleh tunarungu Indonesia yang digunakan sebagai komunikasi antar orang yang mendengar (Gerkatin, 2010), di mana penggunaan BISINDO sendiri menyesuaikan dengan pemahaman bahasa tunarungu dari berbagai latar belakang tanpa memberikan struktur imbuhan bahasa Indonesia. Sedangkan, penjelasan lain tentang BISINDO adalah bahasa isyarat yang dipelajari secara alami oleh Tuli sehingga BISINDO seperti halnya bahasa daerah dan memiliki keunikan di tiap daerah, sehingga kecepatan dan kepraktisannya membuat tuli lebih mudah memahami meski tidak mengikuti aturan bahasa Indonesia sebagaimana jenis bahasa isyarat lain yaitu, SIBI (Gumelar, Hafiar, & Subekti, 2018).

Jadi dapat disimpulkan bahwa definisi dari BISINDO adalah sebuah sistem komunikasi bahasa isyarat alami yang dikembangkan oleh kaum tunarungu di mana bersifat mudah dipahami, praktis dan efektif meski tanpa mengikuti aturan struktur imbuhan berbahasa Indonesia sebagaimana pada Gambar 2.32.



## 2.10 Penelitian Sejenis

Penelitian sejenis atau literatur sejenis adalah tinjauan pustaka di mana penulis mengumpulkan berbagai penelitian terdahulu yang berkaitan atau sejenisnya dengan topik yang sedang peneliti lakukan, kemudian melakukan perbandingan antara peneliti dengan penelitian sebelumnya serta sebagai referensi dalam penelitian.

Literatur sejenis bersumber dari berbagai skripsi, dan jurnal *deep learning* dengan menggunakan model CNN yang bervariasi, maupun penggunaan CNN pada bidang-bidang tertentu. Berikut ini adalah kumpulan literatur sejenis dalam penerapan *deep learning* dengan model CNN pada berbagai bidang pada Tabel 2.1 dan penerapan klasifikasi dan deteksi objek gestur bahasa isyarat pada Tabel 2.2.

**Tabel 2.1** Perbandingan Penelitian Sejenis *Deep Learning*

No	Penulis	Metode	Tools	Hasil	Variabel	Kinerja	Kontribusi	Kelebihan	Kekurangan
1	(Arfian, 2018)	CNN	Bahasa Program: R Studio; <i>Image Downloader</i> : Fatkun Batch Library: Keras, EBImages	Data latih - Andong: 7 salah dari 32 citra - Becak: 4 salah dari 32 citra - Pedati: 6 salah dari 32 citra - Hasil Prediksi: 82 %  Data uji - Andong: 2 salah dari 8 citra - Becak: 3 salah dari 8 citra - Pedati: 1 salah dari 8 citra - Hasil Prediksi: 75 %	Transportasi tradisional: Andong, Becak dan Pedati	- Jumlah data: 120 (96 data latih + 24 data uji) - Ukuran piksel: 32×32, RGB - Fungsi aktivasi: ReLU - Epoch: 30 - Learning rate: 0.01	Penerapan metode CNN untuk klasifikasi pada 3 kategori citra transportasi tradisional	- Memiliki hasil prediksi yang cukup baik lebih dari 75 % ( $> 75\%$ ) - Menggunakan format citra RGB	- Memiliki jumlah data citra yang sedikit - Menghasilkan akurasi data uji yang tergolong kecil untuk kategori yang sedikit - Tidak dijelaskan spesifikasi waktu dan hardware dalam <i>preprocessing data</i> - Tidak adanya UI pada platform tertentu

2.	(Putri, 2018)	CNN	Bahasa Program: R  Software: Fatkun Batch Image Downloader  Library: -	<b>Data latih ke-1</b> - Bulan putih: 5 salah dari 50 citra - Dendrobium: 9 salah dari 43 citra - Ekor tupai: 13 salah dari 47 citra - Hasil Prediksi: 81 %  <b>Data uji ke-1</b> - Bulan putih: 2 salah dari 10 citra - Dendrobium: 1 salah dari 10 citra - Ekor tupai: 2 salah dari 10 citra - Hasil Prediksi: 83 %  <b>Data latih ke-2</b> - Bulan putih: 3 salah dari 30 citra - Dendrobium: 3 salah dari 30 citra - Ekor tupai: 6 salah dari 30 citra - Hasil Prediksi: 87 %  <b>Data uji ke-2</b>	Jenis bunga anggrek: Bulan putih, Anggrek dendrobium, dan Anggrek ekor tupai	- Jumlah data ke-1: 170 (140 data latih + 30 data uji) - Jumlah data ke-2: 120 (90 data latih + 30 data uji) - Ukuran piksel: 32×32, RGB - Fungsi aktivasi: ReLU - Epoch: 500 - Learning rate: 0.001	Penerapan CNN untuk klasifikasi pada 3 kategori citra bunga anggrek dengan jumlah data latih yang berbeda	- Menggunakan perbandingan hasil prediksi dengan jumlah data yang berbeda - Melakukan iterasi yang sangat banyak - Menggunakan format citra RGB	- Tidak dijelaskan library yang digunakan - Terdapat perbedaan jumlah komposisi pada citra data training - Tidak dijelaskan spesifikasi waktu dan hardware dalam preprocessing data - Tidak adanya UI pada platform tertentu
----	------------------	-----	---	---	--	---	---	---	---

				<ul style="list-style-type: none"> <li>- Bulan putih: 2 salah dari 10 citra</li> <li>- Dendrobiun: 1 salah dari 10 citra</li> <li>- Ekor tupai: 2 salah dari 10 citra</li> <li>- Hasil Prediksi: 83 %</li> </ul>					
3	(Kusumani ngrum, 2018)	CNN	<p>Bahasa Program: RStudio versi 1.1.383</p> <p>Software: Fatkun Batch Image Downloader</p> <p>Library: Keras, EBImages</p>	<p><b>Data latih</b></p> <ul style="list-style-type: none"> <li>- Jamur kuping: 0 salah dari 80 citra</li> <li>- Jamur merang: 0 salah dari 80 citra</li> <li>- Jamur tiram: 0 salah dari 80 citra</li> <li>- Hasil Prediksi: 100 %</li> </ul> <p><b>Data uji</b></p> <ul style="list-style-type: none"> <li>- Jamur kuping: 1 salah dari 20 citra</li> <li>- Jamur merang: 2 salah dari 20 citra</li> <li>- Jamur tiram: 8 salah dari 20 citra</li> <li>- Hasil Prediksi : 81,667 %</li> </ul>	<p>Jenis jamur: Jamur kuping, Jamur merang, dan Jamur tiram</p>	<ul style="list-style-type: none"> <li>- Jumlah data: 300 (240 data latih + 60 data uji)</li> <li>- Ukuran piksel: 32×32, RGB</li> <li>- Fungsi aktivasi: ReLU</li> <li>- Epoch: 60</li> <li>- Learning rate: 0.01</li> </ul>	<p>Penerapan CNN untuk klasifikasi pada 3 kategori citra jamur konsumsi</p>	<ul style="list-style-type: none"> <li>- Memiliki nilai akurasi yang baik</li> <li>- Memiliki jumlah data yang lebih banyak</li> <li>- Menggunakan format citra RGB</li> </ul>	<ul style="list-style-type: none"> <li>- Tidak ada keterangan spesifikasi waktu pemrosesan data dan <i>hardware</i> dalam <i>preprocessing data</i></li> <li>- Tidak adanya UI pada <i>platform</i> tertentu</li> </ul>

4	(Salsabila, 2018)	CNN	Bahasa Program: RStudio  Software: Fatkun Batch Image Downloader, Microsoft Excel  Library: Keras, EBIImages, Mxnet	<b>Data latih ke-3</b> - Semar: 0 salah dari 270 citra - Gareng: 0 salah dari 270 citra - Petruk: 0 salah dari 270 citra - Bagong: 0 salah dari 270 - Hasil Prediksi : 98.7 %  <b>Data uji ke-3</b> - Semar: 4 salah dari 30 citra - Gareng: 2 salah dari 30 citra - Petruk: 2 salah dari 30 citra - Bagong: 2 salah dari 30 citra - Hasil Prediksi: 91.65 %	Tokoh citra wayang punakawan: Semar, Gareng, Petruk, dan Bagong	- Jumlah data ke-1: 1200 (840 data latih + 360 data uji) atau (70 : 30) - Jumlah komposisi data ke-2: (960 data latih + 240 data uji) atau (80 : 20) - Jumlah komposisi <b>data ke-3:</b> (1080 data latih + 120 data uji) atau (90 : 10) - Ukuran piksel: 46×46, <i>Greyscale</i> - Fungsi aktivasi: tanh - Epoch: 25, 50, 100, 150, 200, 250 - Learning rate: adam	Penerapan CNN untuk klasifikasi pada 4 kategori citra jamur konsumsi dengan ukuran piksel yang lebih besar, format citra <i>greyscale</i> dan melakukan perbandingan 3 komposisi data, filter serta <i>epoch</i> yang berbeda	- Memiliki tingkat akurasi yang sangat baik dengan hasil lebih dari 90 % (> 90 %) - Jumlah data yang digunakan tergolong banyak yaitu 1200 data - Terdapat skenario perbandingan dengan komposisi jumlah data, filter dan epoch yang berbeda	- Tidak adanya keterangan masing-masing data <i>training</i> dan <i>testing</i> pada hasil prediksi tiap skenario - Tidak ada keterangan spesifikasi waktu pemrosesan data - Kurangnya informasi spesifikasi <i>hardware</i> dalam <i>preprocessing data</i> - Tidak adanya UI pada <i>platform</i> tertentu
---	----------------------	-----	--	--	---	--	---	--	---

5	(Nurhikmat & Purwaningsih, 2018)	CNN	Bahasa Program: Python versi 3.6.2  Software: -  Library: jQuery (Javascript), CV2, Keras, NumPy  Spesifikasi Hardware: Intel core i7-6700HQ, RAM 16 GB, GPU: NVIDIA GeForce GTX 960, OS Windows	<b>Data latih ke-2</b> - Cepot: 0 salah dari 80 citra - Gatotkaca: 0 salah dari 80 citra - Semar: 0 salah dari 80 citra - Hasil Prediksi: 95 %  <b>Data uji ke-2</b> - Cepot: 0 salah dari 20 citra - Gatotkaca: 1 salah dari 20 citra - Semar: 3 salah dari 17 citra - Hasil Prediksi: 90 %  Hasil prediksi dari berbagai skenario dengan parameter tertentu = - Nilai epoch (100) : 97% - Jumlah layer konvolusi (3 layer) : 96%	Tokoh Citra Wayang Golek: Cepot, Gatotkaca dan Semar	- Jumlah komposisi data ke-1: 300 (210 data latih + 90 data uji) - Jumlah komposisi <b>data ke-2:</b> (240 data latih + 60 data uji) - Jumlah komposisi data ke-3: (270 data latih + 30 data uji) - Ukuran piksel: <b>64×64</b> dan 150x150, RGB - Fungsi aktivasi: ReLU - Epoch: <b>20, 30, 50, 100</b> - Learning rate: 0.01, <b>0.001, 0.0001</b> - Waktu pelatihan: 2 menit	Penerapan CNN untuk klasifikasi pada 3 kategori/kelas citra wayang golek dengan melakukan 8 skenario perbandingan sesuai parameter yang berbeda	- Dijelaskan spesifikasi <i>hardware</i> yang digunakan untuk data <i>preprocessing</i> - Setiap skenario dijelaskan rentang waktu pemrosesan data ( <i>data preprocessing</i> ) - Terdapat beberapa keterangan dari pengaruh setiap parameter yang dilakukan	- Mayoritas pada skenario perbandingan tidak diketahui jumlah data <i>training</i> dan data <i>testing</i> yang telah digunakan - Tidak adanya UI pada <i>platform</i> tertentu
---	----------------------------------	-----	---	--	--	--	---	---	--

				<ul style="list-style-type: none"> <li>- Metode <i>pooling layer</i> = <i>max pooling</i>, 95%</li> <li>- Input shape image (64x64 dan 150x150) = 97%</li> <li>- Jumlah data <i>training</i> (300) = 95%</li> <li>- Perbandingan jumlah data <i>training</i> dan data <i>testing</i> (90:10 atau 270 : 30) = 100%</li> <li>- Ukuran kernel (3x3) = 97%</li> <li>- Nilai <i>learning rate</i> (0.001) = 97%</li> </ul>					
6	(Shafira, 2018)	CNN	Bahasa Program: RStudio versi 1.1.383  Software: Fatkun Batch Image Downloader	<p><b>Data latih</b></p> <ul style="list-style-type: none"> <li>- Tomat layak: 1 salah dari 40 citra</li> <li>- Tomat tidak layak: 1 salah dari 40 citra</li> <li>- Hasil Prediksi: 97.5 %</li> </ul> <p><b>Data uji</b></p> <ul style="list-style-type: none"> <li>- Tomat layak: 1 salah dari 10 citra</li> <li>- Tomat tidak layak: 1 salah dari 10 citra</li> </ul>	<p>Varietas Tomat Merah: Tomat Layak dan Tomat Tidak Layak</p>	<ul style="list-style-type: none"> <li>- Jumlah data: 100 (80 data latih + 20 data uji)</li> <li>- Ukuran piksel: 32x32, RGB</li> <li>- Fungsi aktivasi: ReLU</li> <li>- Epoch: 50</li> <li>- Learning rate: 0.01</li> </ul>	<p>Penerapan CNN untuk klasifikasi pada 2 kategori citra tomat dengan 4 skenario perbandingan sesuai parameter yang berbeda</p>	<ul style="list-style-type: none"> <li>- Menghasilkan nilai prediksi yang sangat baik yaitu diatas 90% (&gt; 90%)</li> <li>- Terdapat uji coba klasifikasi dengan data baru dengan hasil yang seluruhnya benar</li> </ul>	<ul style="list-style-type: none"> <li>- Tidak adanya spesifikasi waktu ketika melakukan pemrosesan data</li> <li>- Tidak adanya spesifikasi <i>hardware</i> yang digunakan</li> <li>- Tidak adanya UI pada</li> </ul>

		<p><i>Library:</i> Keras, EBImages</p> <ul style="list-style-type: none"><li>- Hasil Prediksi: 90 %</li><li>Hasil prediksi dari berbagai skenario dengan parameter tertentu =</li><li>- Jumlah data <i>training</i> (100)= 90%</li><li>- Ukuran kernel (3*3)= 90%</li><li>- Perbandingan jumlah data <i>training</i> dan data <i>testing</i> (80:20)= 90%</li><li>- Jumlah neuron (256)= 90%</li></ul>		<p><i>platform</i> tertentu</p>
--	--	--	--	-------------------------------------

7	(Dzulqarnain et al., 2019)	CNN	Bahasa Program: - Software: - Library: - Spesifikasi hardware:	Hasil Prediksi berdasarkan 4 kategori= - 4 Kelas klasifikasi= 93.42 % - 2 Kelas klasifikasi= 94.74 % - Dataset bersifat vertikal= 93.86 % - Dataset bersifat horizontal= 94.76 %	Kategori 4 kelas: Sangat Bagus (AA), Bagus (AB), Kurang Bagus (C), Tidak Bagus (TB)  Kategori 2 kelas: Layak ekspor, Tidak layak ekspor	- Jumlah data: 756 (605 data latih + 151 data uji) - Ukuran piksel: 32x32, <i>Grayscale</i> - Fungsi aktivasi: ReLU - Epoch: 200 - Learning rate: 0.0001, 0.001, 0.01	Peningkatkan akurasi model CNN dengan penghilangan <i>noise</i> pada citra, penambahan parameter seperti <i>stride</i> , <i>zero padding</i> dan <i>adam optimizer</i>	- Berhasil meningkatkan nilai akurasi dari penelitian sebelumnya hingga 13 - 22 % - Menghilangkan <i>noise</i> pada citra dengan metode <i>Gaussian blur</i> dan morfologi terbuka - Adanya perbandingan hasil optimasi dengan <i>learning rate</i> yang berbeda	- Tidak adanya penjelasan masing-masing data latih dan data uji yang digunakan pada setiap kelas - Tidak adanya spesifikasi <i>tools</i> yang digunakan seperti bahasa program, <i>library</i> , <i>hardware</i> dan perangkat yang digunakan - Tidak ada durasi waktu yang dibutuhkan dalam <i>data preprocessing</i> dan <i>data validation</i> - Tidak adanya UI pada <i>platform</i> tertentu
---	----------------------------	-----	---	--	---	---	--	--	--

8	(Dewa et al., 2018)	CNN dan MLP (1 & 2 hidden layer)	Bahasa Program: Python (Framework Django)  Software: <ul style="list-style-type: none"><li>-</li></ul> Library: Theano, OpenCV  Spesifikasi Hardware: Intel core i5-5200U, GPU : NVIDIA GT 940M	Hasil prediksi dari 3 skenario dengan membandingkan 2 model, jumlah layer dan waktu yang berbeda, yaitu:  MLP (1 hidden layer) <ul style="list-style-type: none"><li>- Testing I (00:12:46)= 53%</li><li>- Testing II (00:12:51)= 56%</li><li>- Testing III (00:12:53)= 60%</li><li>- Testing IV (00:12:48)= 62%</li><li>- Testing V (00:12:46)= 52%</li></ul> MLP (2 hidden layer) <ul style="list-style-type: none"><li>- Testing I (00:34:51)= 44%</li><li>- Testing II (00:34:49)= 49%</li><li>- Testing III (00:34:41)= 52%</li><li>- Testing IV (00:34:56)= 59%</li><li>- Testing V (00:34:57)= 47%</li></ul>	Tulisan Tangan Aksara Jawa: Ha, Na, Ca, Ra, Ka, Da, Ta, Sa, Wa, La, Pa, Dha, Ja, Ya, Nya, Ma, Ga, Ba, Tha, Nga	- Jumlah data: 2000 (1600 data latih + 400 data uji) - Ukuran piksel: 28x28, Grayscale - Fungsi aktivasi : ReLU - Epoch : 10000 - Learning rate : -	Perbandingan metode pada penerapan model CNN dan MLP dengan jumlah layer berbeda dengan tujuan membandingkan hasil dari tingkat akurasi yang paling terbaik	- Menghasilkan perbandingan akurasi pada 2 metode yang dijelaskan secara informatif - Setiap metode menjelaskan model yang digunakan - Menjelaskan spesifikasi hardware untuk memproses data - Terdapat UI pada platform berbasis web - Adanya konsep object detection untuk membuat segmentasi pada citra	- Tidak ada
---	---------------------	----------------------------------	---	--	--	---	---	--	-------------

				CNN - Testing I (01:48:30)= 82% - Testing II (01:49:37)= 85% - Testing III (01:50:19)= 89% - Testing IV (01:49:59)= 89% - Testing V (00:50:10)= 78%					
9	(Michele et al., 2019)	CNN dan SVM	Bahasa Program: - Software: - Library: Keras, Scikit-learn Spesifikasi Hardware: -	Hasil Prediksi = - CNN (MobileNet V2) = 99.5 % - CNN (MobileNet V2) + SVM = 100 %	Palmprint: 500 jenis cetakan telapak tangan yang berbeda	- Jumlah data: 6000 (4500 data latih + 1500 data uji) - Ukuran piksel: 128×128, Grayscale - Fungsi aktivasi: ReLU - Epoch: 100 - Learning rate: - Total waktu preprocessing = ± 60 menit (CNN), ± 40 menit (CNN+SVM)	Perbandingan hasil akurasi pada model CNN dengan arsitektur MobileNet V2 dan CNN digabung dengan metode SVM	- Menggunakan salah satu arsitektur dari CNN, MobileNet yang dapat digunakan pada aplikasi berbasis mobile - Melakukan perpaduan 2 metode yang berbeda - Mendapatkan hasil prediksi yang sangat baik dengan kelas yang banyak yaitu lebih dari 99 % (> 99%)	- Tidak diketahui bahasa program yang digunakan - Tidak disajikannya informasi dari spesifikasi hardware yang digunakan untuk preprocessing - Tidak adanya penggunaan UI pada platform tertentu

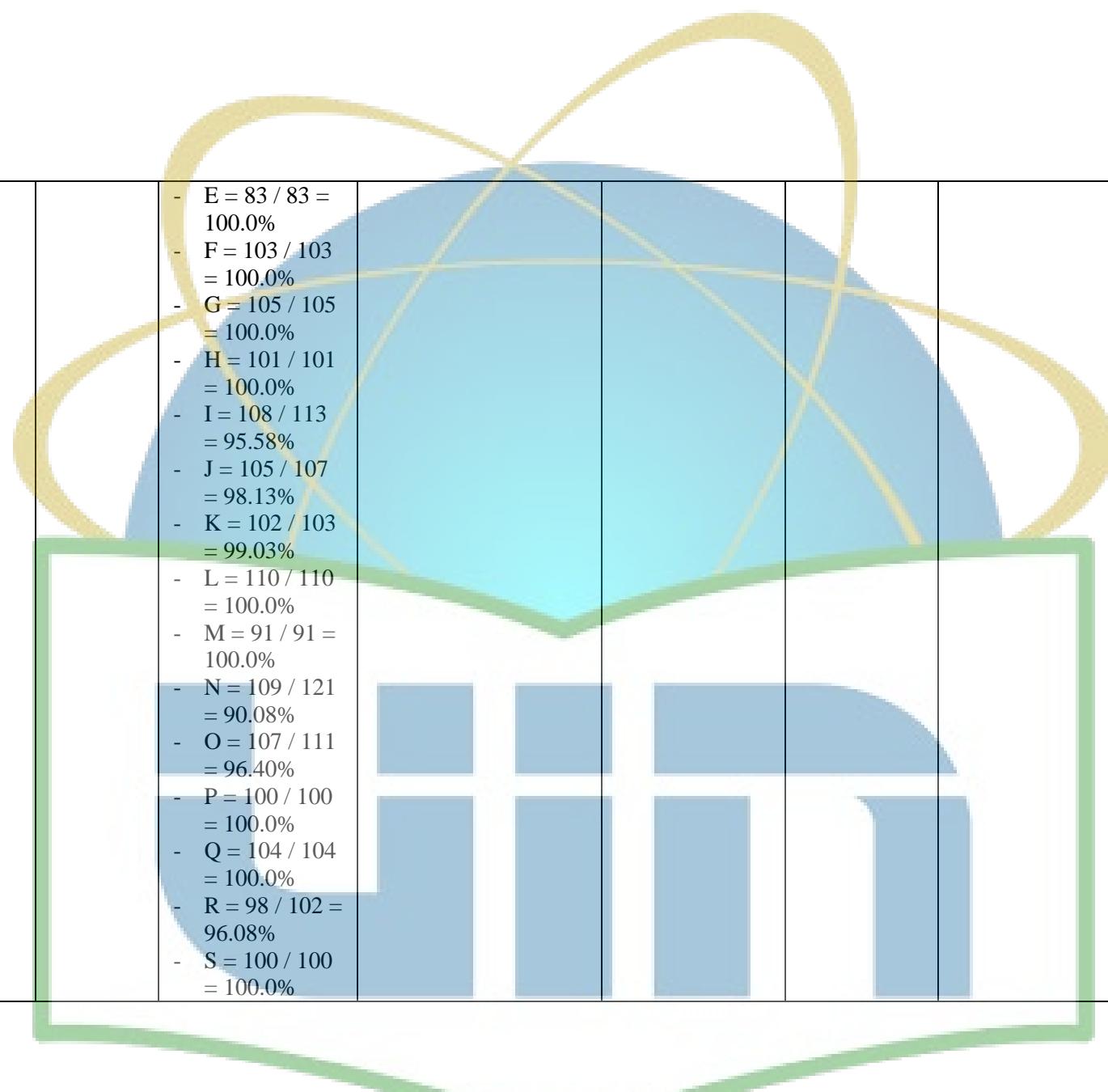
**Tabel 2.2** Perbandingan Penelitian Sejenis Klasifikasi Gestur Bahasa Isyarat

No	Penulis	Metode	Tools	Hasil	Variabel	Kinerja	Kontribusi	Kelebihan	Kekurangan
10	(Borman <i>et al.</i> , 2017)	KNN dan Haar <i>Classifier</i>	-	<p>Hasil prediksi pada 24 kelas =</p> <ul style="list-style-type: none"> <li>- A = 100%</li> <li>- B = 80%</li> <li>- C = 100%</li> <li>- D = 80%</li> <li>- E = 100%</li> <li>- F = 90%</li> <li>- G = 100%</li> <li>- H = 100%</li> <li>- I = 100%</li> <li>- J = -</li> <li>- K = 83%</li> <li>- L = 100%</li> <li>- M = 70%</li> <li>- N = 50%</li> <li>- O = 100%</li> <li>- P = 75%</li> <li>- Q = 100%</li> <li>- R = -</li> <li>- S = 100%</li> <li>- T = 100%</li> <li>- U = 90%</li> <li>- V = 90%</li> <li>- W = 100%</li> <li>- X = 100%</li> <li>- Y = 100%</li> </ul>	Huruf Abjad BISINDO: A – Z, kecuali J dan R	<ul style="list-style-type: none"> <li>- Jumlah kelas: 24</li> <li>- Jumlah data: 50/Kelas</li> <li>- Total data: 1200 data citra</li> <li>- Ukuran piksel: 104×104</li> <li>- Perbandingan jarak deteksi: 30cm, 50cm, 70cm, 90cm, 110cm</li> </ul>	Penerapan objek deteksi dan klasifikasi dengan 24 kelas yang berjumlah 50 citra per kelas dari <i>input</i> video (non-real time)	<ul style="list-style-type: none"> <li>- Memiliki nilai akurasi yang cukup tinggi diatas 90%</li> <li>- Menggunakan perbandingan jarak untuk deteksi objek</li> </ul>	<ul style="list-style-type: none"> <li>- Tidak adanya informasi <i>tools</i> yang digunakan dalam penelitian</li> <li>- Tidak adanya variasi data untuk meningkatkan pembelajaran data</li> <li>- Implementasi menggunakan <i>input</i> video dan tidak berbasis <i>real-time</i></li> </ul>

				<ul style="list-style-type: none"> <li>- Z = 100%</li> <li>Rata-rata akurasi = 92,08%</li> </ul>					
11	(Rakhman et al., 2010)	Haar Classifier dan KNN	OpenCV	<p>Hasil prediksi 19 kelas =</p> <ul style="list-style-type: none"> <li>- A = 97%</li> <li>- B = 100%</li> <li>- C = 85%</li> <li>- D = 95%</li> <li>- E = -</li> <li>- F = 100%</li> <li>- G = 96%</li> <li>- H = 75%</li> <li>- I = 100%</li> <li>- J = -</li> <li>- K = 80%</li> <li>- L = 100%</li> <li>- M = -</li> <li>- N = -</li> <li>- O = 92%</li> <li>- P = 65%</li> <li>- Q = 83%</li> <li>- R = 70%</li> <li>- S = -</li> <li>- T = -</li> <li>- U = 68%</li> <li>- V = 94%</li> </ul>	<p>Huruf Abjad SIBI: A – Z, kecuali E, J, M, N, S, T, dan Z</p>	<ul style="list-style-type: none"> <li>- Jumlah kelas: 19</li> <li>- Jumlah data: 100/kelas</li> <li>- Total data: 1900 data citra</li> <li>- Ukuran piksel: 20×25, 30×30</li> </ul>	<p>Penerapan objek deteksi dan klasifikasi berbasis <i>real-time</i> menggunakan kamera <i>webcam</i></p>	<ul style="list-style-type: none"> <li>- Memiliki nilai akurasi yang cukup akurat</li> <li>- Pengujian sudah berbasis <i>real-time</i> menggunakan kamera <i>webcam</i></li> <li>- Menyajikan nilai perbandingan jarak uji</li> </ul>	<ul style="list-style-type: none"> <li>- Informasi sebatas grafik dan tidak diketahuinya nilai akurasi setiap kelas</li> <li>- Kurang efektifnya mentranslasikan bahasa isyarat menggunakan kamera <i>webcam</i></li> </ul>

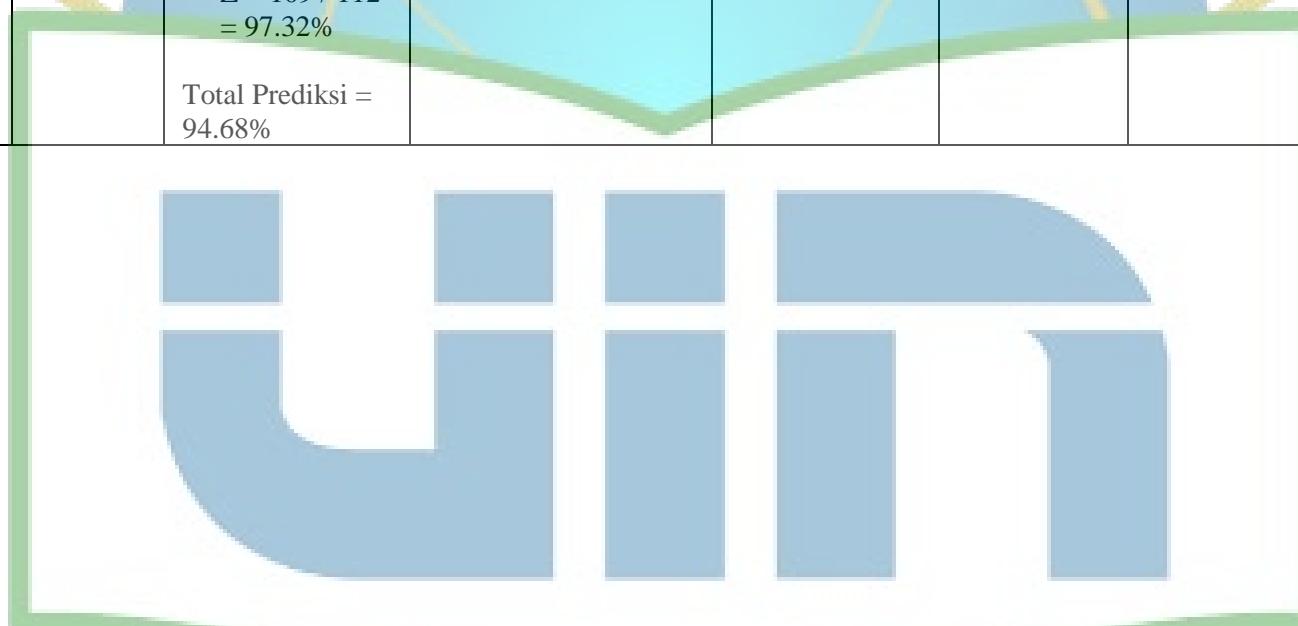
				<ul style="list-style-type: none"> <li>- W = 93%</li> <li>- X = 85%</li> <li>- Y = 100%</li> <li>- Z = -</li> </ul> <p>Rata-rata akurasi = 89,68%</p>					
12	(Raheja et al., 2016)	SVM	Matlab	<p>Hasil prediksi rata-rata = 97,5%</p>	<p>Huruf Abjad dan Kata ISL: A, B, C dan Hello</p>	<ul style="list-style-type: none"> <li>- Jumlah kelas: 4</li> <li>- Jumlah data: 20/kelas</li> <li>- Total data: 80</li> <li>- Ukuran piksel: -</li> </ul>	<p>Penerapan deteksi objek bahasa isyarat ISL pada huruf dan kosakata dengan perangkat Kamera Microsoft Kinect</p>	<ul style="list-style-type: none"> <li>- Memiliki nilai akurasi yang tinggi dengan jumlah data sedikit</li> <li>- Menggunakan teknik filter untuk segmentasi dan pengurangan noise</li> <li>- Deteksi berbasis <i>real-time</i></li> </ul>	<ul style="list-style-type: none"> <li>- Tidak diketahui resolusi citra yang digunakan</li> <li>- Kurangnya jumlah kelas yang diprediksi</li> <li>- Tidak ada variasi data untuk pengujian</li> </ul>
13	(Bheda & Radpour, 2017)	CNN	Python	<p>Hasil prediksi rata-rata =</p> <ul style="list-style-type: none"> <li>- Huruf abjad = 82,5%</li> <li>- Angka = 97%</li> </ul>	<p>Huruf Abjad dan Angka ASL: A – Z, 0 - 10</p>	<ul style="list-style-type: none"> <li>- Jumlah kelas: 37</li> <li>- Jumlah data: 125/kelas</li> <li>- Total data: 4625</li> <li>- Ukuran piksel: 200×200</li> </ul>	<p>Penerapan klasifikasi gambar pada gestur isyarat huruf abjad dan angka BISINDO</p>	<ul style="list-style-type: none"> <li>- Memiliki nilai akurasi yang cukup tinggi</li> <li>- Menggunakan teknik augmentasi data untuk memperbanyak variasi dataset</li> </ul>	<ul style="list-style-type: none"> <li>- Tidak menyajikan nilai akurasi pada setiap kelas</li> <li>- Klasifikasi berbasis input gambar dan belum <i>real-time</i></li> </ul>

14	(Masood <i>et al.</i> , 2018)	CNN	Bahasa Program: - Hasil prediksi pada 36 kategori=	$0 = 59 / 109 = 54,13\%$ $1 = 87 / 91 = 95,60\%$ $2 = 91 / 101 = 90,10\%$ $3 = 105 / 106 = 99,06\%$ $4 = 96 / 106 = 90,57\%$ $5 = 104 / 107 = 97,20\%$ $6 = 93 / 101 = 92,08\%$ $7 = 99 / 99 = 100,0\%$ $8 = 99 / 101 = 98,02\%$ $9 = 95 / 96 = 98,96\%$ $A = 106 / 107 = 99,07\%$ $B = 111 / 111 = 100,0\%$ $C = 105 / 105 = 100,0\%$ $D = 92 / 93 = 98,92\%$	Huruf Abjad ASL: A – Z, Angka Abjad ASL: 0 – 9	- Jumlah data: 14781 (11085 data latih + 3696 data uji) - Ukuran piksel: $224 \times 224$ , RGB - Fungsi aktivasi: ReLU - Epoch: 4 - Learning rate: 0.001	Penerapan model CNN dengan 36 kategori terdiri atas 70 gambar per kategori kemudian gambar diperbanyak sehingga berjumlah total 14.781 data citra dengan model 16 <i>hidden layer</i> (VGG16)	- Mendapat hasil akurasi yang sangat baik yaitu lebih dari 94 % ( $> 94\%$ ) - Menggunakan data yang cukup banyak hingga 14000 data citra - Menyajikan perbandingan nilai prediksi yang benar dan salah dari keseluruhan jumlah data uji	- Tidak adanya penjelasan spesifikasi <i>tools</i> seperti bahasa program, <i>library</i> , dan <i>hardware</i> yang digunakan - Tidak ada penyajian durasi waktu yang dibutuhkan dalam <i>data preprocessing</i> - Tidak adanya penggunaan UI pada <i>platform</i> tertentu - Penyebaran data setiap kategori tidak merata sehingga mempengaruhi nilai prediksi
----	-------------------------------	-----	---	---	---	---	---	--	---



			<ul style="list-style-type: none"><li>- E = <math>83 / 83 = 100.0\%</math></li><li>- F = <math>103 / 103 = 100.0\%</math></li><li>- G = <math>105 / 105 = 100.0\%</math></li><li>- H = <math>101 / 101 = 100.0\%</math></li><li>- I = <math>108 / 113 = 95.58\%</math></li><li>- J = <math>105 / 107 = 98.13\%</math></li><li>- K = <math>102 / 103 = 99.03\%</math></li><li>- L = <math>110 / 110 = 100.0\%</math></li><li>- M = <math>91 / 91 = 100.0\%</math></li><li>- N = <math>109 / 121 = 90.08\%</math></li><li>- O = <math>107 / 111 = 96.40\%</math></li><li>- P = <math>100 / 100 = 100.0\%</math></li><li>- Q = <math>104 / 104 = 100.0\%</math></li><li>- R = <math>98 / 102 = 96.08\%</math></li><li>- S = <math>100 / 100 = 100.0\%</math></li></ul>							

			<ul style="list-style-type: none"><li>- <math>T = 83 / 84 = 98.81\%</math></li><li>- <math>U = 98 / 99 = 98.99\%</math></li><li>- <math>V = 103 / 110 = 93.64\%</math></li><li>- <math>W = 66 / 97 = 68.04\%</math></li><li>- <math>X = 90 / 93 = 96.77\%</math></li><li>- <math>Y = 114 / 114 = 100.0\%</math></li><li>- <math>Z = 109 / 112 = 97.32\%</math></li></ul> <p>Total Prediksi = 94.68%</p>							

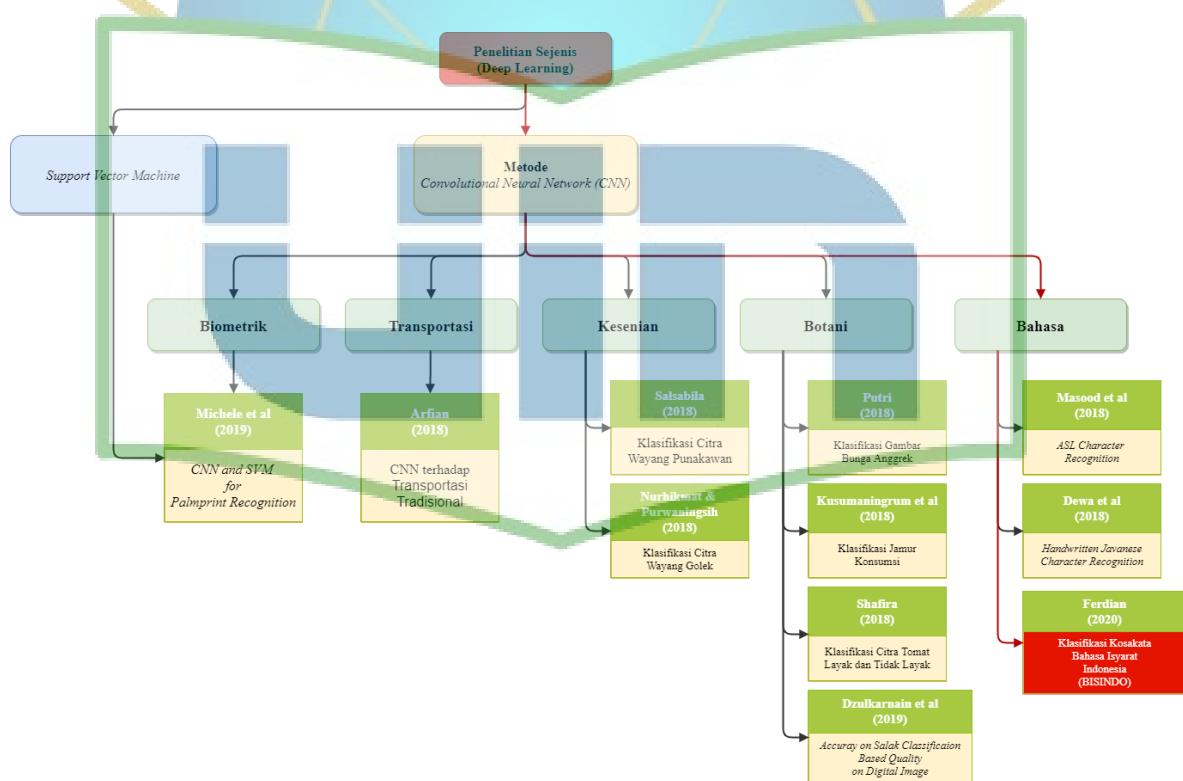


Berdasarkan Tabel 2.1 dan Tabel 2.2 dengan topik penelitian yang penulis lakukan mendekati pada penelitian sejenis no. 9, 13 dan 14, namun terdapat kelebihan pada penelitian yang penulis lakukan. Nilai pembeda penelitian yang ada tersebut dengan penelitian ini adalah:

1. Objek deteksi dan klasifikasi citra bahasa isyarat hanya mengangkat kasus pada kosakata Bahasa Isyarat Indonesia (BISINDO).
2. Metode klasifikasi menggunakan *Convolutional Neural Network* (CNN), alasan penulis menggunakan metode ini karena CNN mempunyai kapabilitas lebih baik dalam mengklasifikasikan daripada KNN dan SVM *Network* dengan nilai akurasi atau prediksi yang sangat baik dan mendalam di antara metode lainnya.
3. Salah satu arsitektur CNN yang digunakan adalah *MobileNet* yang berdasarkan pada penggunaan *platform* berbasis *mobile* yaitu Android dengan konsep *real-time*.

## 2.11 Ranah Penelitian

Pada tahap ini menggambarkan ranah penelitian sejenis yang dilakukan oleh penulis bahwasanya terdapat 9 literatur yang penulis bandingkan berdasarkan topik penelitian mengenai *deep learning* dengan sub-bidang kasus berjumlah 5 bidang serta menggunakan metode utama *Convolutional Neural Network* (CNN) dan di bidang tertentu membandingkan dengan metode lain, di antara bidangnya tersebut adalah biometrik, transportasi, kesenian, botani, dan bahasa. Berdasarkan penelitian sejenis yang identik sesuai dengan ranah penelitian sebelumnya maka ranah penelitian yang penulis ambil adalah berkaitan dengan bidang bahasa khususnya tentang klasifikasi gestur kosakata pada bahasa isyarat dengan menggunakan metode CNN berbasis *mobile*. Gambar 2.33 adalah ilustrasi dari ranah penelitian.



Gambar 2.33 Ranah Penelitian



## **BAB 3**

### **METODOLOGI PENELITIAN**

#### **3.1 Metode Pengumpulan Data**

Dalam melakukan pengumpulan data untuk sampel penelitian, penulis menggunakan 4 metode pengumpulan data yaitu sebagai berikut:

##### **3.1.1 Studi Pustaka**

Penulis menggunakan studi pustaka yang berguna untuk menghimpun data seperti teori, fakta, kasus, dan masalah yang berkaitan dengan penelitian ini. Penulis mencari referensi yang berhubungan dengan obyek penelitian. Penulis melakukan studi pustaka di perpustakaan, toko buku, atau melalui internet secara *online*. Penulis telah mengumpulkan 2 buku, 7 e-book, 7 skripsi, 14 jurnal dari hasil studi pustaka.

##### **3.1.2 Wawancara**

Penulis melakukan mewawancara kepada 2 narasumber dari Departemen Lembaga Riset Bahasa Isyarat Fakultas Ilmu Bahasa Universitas Indonesia (LRBI FIB UI) dan Kopi Tuli yaitu Ibu Silva Tenrisara Putri dan Kak Aldilla. Ibu Silva merupakan penanggung jawab dari LRBI FIB UI, adapun Kak Aldilla merupakan salah satu anggota komunitas PUSBISINDO yang memiliki keterbatasan pendengaran (teman tuli) dan juga sebagai pegawai di Kopi Tuli, Kalibata.

Wawancara dilakukan seputar pertanyaan tentang jenis-jenis Bahasa Isyarat di Indonesia, pengenalan arti gestur tiap huruf, dan berbagai bentuk gestur dari kosakata yang banyak tidak diketahui oleh masyarakat pada umumnya.

Menurut Ibu Silva, Bahasa Isyarat Indonesia memiliki 2 jenis bahasa isyarat umum yang diketahui oleh teman tuli adalah Sistem Isyarat Bahasa Indonesia (SIBI) dan Bahasa Isyarat Indonesia (BISINDO). Kedua ciri ini mempunyai ciri khusus yang membedakan antara SIBI dengan BISINDO. Berbagai bentuk gestur isyarat dapat membedakan arti kosakata yang dimaksud berdasarkan gerak badan atau bentuk polanya. Gerak isyarat yang biasanya menggunakan anggota tangan juga menentukan arti dari kosakata.

Sedangkan Kak Aldilla menjelaskan bahwa jenis Bahasa isyarat yang sering dikenal dan mudah dipakai sehari-hari ialah BISINDO. Adapun jenis BISINDO berdasarkan fungsinya yang sesuai dengan representasi dari kosakata yang dimaksud dan bersifat fleksibel dibandingkan SIBI. Namun, pemerintah lebih mendukung penggunaan SIBI sebagai bahan ajaran di sekolah padahal untuk kehidupan sehari-hari sangat sulit untuk bisa diterapkan secara generik. Kemudian, Kak Aldilla menambahkan dengan penggunaan jenis BISINDO dapat mempermudah interaksi antara sesama teman tuli dan orang awam, biasanya penggunaan gestur isyarat kosakata berdasarkan objek visual yang lebih mudah dipahami serta tanpa tambahan kata imbuhan tertentu seperti me-an, di-, -kan, -kah ataupun kata sambung seperti ke, di, yang, dan lainnya.

### 3.1.3 Observasi

#### 1. Waktu dan Tempat Penelitian

Penulis menentukan waktu dan tempat penelitian untuk melakukan beberapa tahapan yang harus dikerjakan, diantara rinciannya sebagai berikut:

**Tabel 3.1** Waktu dan Tempat Penelitian

Tahapan	Waktu	Tempat / Tools
Wawancara	4 November 2019 & 7 Maret 2020	Fakultas Ilmu Bahasa UI dan Kopi Tuli
Pengumpulan Data Citra	2 – 20 Februari 2020	Rumah Penulis
<i>Data Preprocessing</i>	1 – 10 Maret 2020	Jupyter Notebook dan LabelImg
<i>Training</i>	14 – 20 Juni 2020	Server Google Colabs

#### 2. Perangkat Penelitian

Data citra gestur bahasa isyarat diambil dengan menggunakan kamera utama *smartphone* Lenovo A7770 dengan resolusi 8 *Megapixel* menghasilkan citra yang bagus sebagaimana resolusi yang sama digunakan oleh Tiara (2018) ketika melakukan pengambilan data citra untuk penelitiannya menggunakan kamera utama *smartphone* VIVO Y55. Gambar 3.1 adalah perangkat *smartphone* yang penulis gunakan untuk mengambil data citra:



**Gambar 3.1** Kamera *Smartphone* Lenovo A7700

Kemudian, Tabel 3.2 adalah spesifikasi *hardware* yang penulis gunakan:

**Tabel 3. 2 Spesifikasi Hardware**

Tipe	Hardware / Tools	Spesifikasi
<i>Smartphone</i>	Lenovo A7700	OS Android 6.0 <i>Marshmallow</i> CPU @ 1 GHz MediaTek MT6735P RAM 2 GB
<i>Cloud Server</i>	Google Colaboratory	CPU: <i>Xeon Processor</i> CPU @ 2.2 GHz, RAM 13 GB GPU: 1×Tesla K80, CUDA 10.0, 12 GB DDR5 VRAM

Sedangkan, Tabel 3.3 adalah spesifikasi *software* yang digunakan:

**Tabel 3. 3 Spesifikasi Software dan Library**

Software	Programming Language	Library
Jupyter Notebook 4.4.0	Python 3.7.1	OpenCV 4.1.0.25 Augmentor 0.2.8
LabelImg 1.8.2		Pandas 0.24.2 Tensorflow 1.15.2 Tensorboard 1.14.0
Android Studio 3.4.2	Java 11.0.7	-

### 3. Pengumpulan Dataset

Pengumpulan *dataset* tentang jenis gestur kosakata BISINDO, penulis juga melaksanakan observasi secara langsung dan tidak langsung, beberapa diantaranya dari buku kamus, video, aplikasi, dan wawancara kepada narasumber teman tuli yang berasal dari Kopi Tuli. Tujuan dilakukan

observasi ini untuk melihat, mengenali dan melakukan verifikasi gestur isyarat secara langsung berdasarkan informasi yang penulis dapatkan.

Pada Tabel 3.5 merupakan hasil observasi penulis mengenai jenis gestur kosakata dari gestur BISINDO berdasarkan kosakata lirik lagu “Bidadari Tak Bersayap” sebagai bahan pelatihan dan pengujian.

Dengan kosakata yang berjumlah 32 gestur, kemudian penulis jadikan setiap gestur kosakata mewakili satu kelas, sehingga menjadi 32 kelas gestur. Adapun jumlah setiap kelas gestur sebesar 500 data citra untuk data *training* dan 50 data untuk data *testing*, dengan pembagian 90:10 yang berjumlah 16000 dan 1600 *dataset*. Rincian pembagiannya sebagai berikut:

**Tabel 3.4** Tabel Pembagian *Dataset*

No	Dataset	Split	Jumlah
1	Data <i>training</i>	90%	16000
2	Data <i>testing</i>	10%	1600
Total		100%	17600

**Tabel 3.5** Dataset Gestur Kosakata BISINDO

Kosakata	Gestur Isyarat	Kosakata	Gestur Isyarat	Kosakata	Gestur Isyarat
Bidadari		Tuhan		Tenang	
Tak Bersayap		Dalam		Bersama	
Datang		Wajah		Sederhana	
Aku/Ku		Kamu/Kau		Indah	
Dikirim		Sungguh		Mencintai	

Sampai		Nyawa		Sayang Sekali	
Habis		Satu		Wanita	
Umur/Usia		Dihati		Ini	
Mau		Teristimewa		Katakan	
Teman		Diam-diam		Yes	
Hidup		Memandangi			

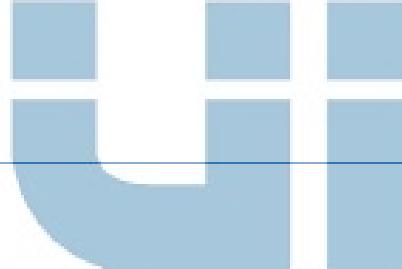
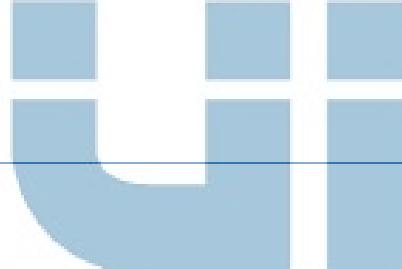
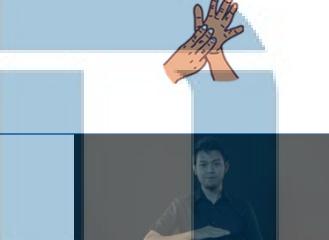
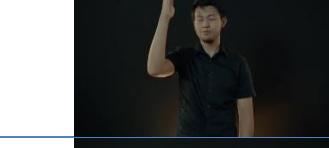
### 3.1.4 Kuesioner

Penulis menyebarkan kuesioner secara acak kepada 165 responden dengan menggunakan fitur yang berasal dari *Google Form*. Dalam kuesioner ini, penulis ingin mengetahui seberapa banyak masyarakat yang memiliki pengetahuan dan pengenalan tentang Bahasa isyarat. Tabel 3.6 merupakan daftar beberapa pertanyaan yang ada dalam kuesioner:

**Tabel 3.6** Daftar Pertanyaan Kuesioner tentang Pengetahuan

No.	Pertanyaan
1.	Apakah kamu tahu jenis-jenis bahasa isyarat yang umum digunakan di Indonesia?
2.	Seberapa besar pengetahuan kamu mengenai bahasa isyarat di Indonesia?
3.	Ada berapakah jenis macam bahasa isyarat di Indonesia?
4.	Apakah kamu pernah membaca artikel atau buku yang membahas tentang jenis-jenis bahasa isyarat di Indonesia?
5.	Apakah kamu tertarik untuk mempelajari dan mengenal jenis-jenis huruf / kosakata dalam Bahasa Isyarat Indonesia?
6.	Bagaimana tanggapan kamu jika ada aplikasi untuk dapat mengenali jenis kosakata Bahasa Isyarat Indonesia?
7.	Menurut kamu seberapa pentingkah kehadiran aplikasi tersebut untuk mengenali jenis gestur kosakata pada BISINDO?

**Tabel 3.7** Daftar Pertanyaan Kuesioner tentang Pengenalan

No.	Pertanyaan	Gestur Isyarat
1.		
2.		
3.		
4.	Apa arti dari huruf ini?	
5.		
6.		
7.		
8.	Apa maksud gestur kosakata di bawah ini ?	
9.		
10.		

Pertanyaan tersebut disebar dengan tujuan agar dapat membuktikan mengenai pengetahuan masyarakat umum yang minim tentang Bahasa isyarat dan dengan adanya aplikasi alternatif yang mampu mengenali gestur Bahasa isyarat tersebut.

**Tabel 3.8** Responden Mengetahui Ragam Jenis Bahasa Isyarat Indonesia

No.	Mengetahui jenis-jenis bahasa isyarat Indonesia	Responden	
		N	%
1.	Ya, Tahu	48 orang	29,1%
2.	Tidak Tahu	117 orang	70,9%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

Pada Tabel 3.8 dapat disimpulkan bahwa 48 dari 165 jumlah responden mengetahui berbagai macam gestur bahasa isyarat di Indonesia. Namun, sisa yang berjumlah 117 dari 165 responden tidak mengetahui berbagai macam gestur bahasa isyarat di Indonesia.

**Tabel 3.9** Tingkat Pengetahuan dari Jumlah Responden Mengenai Bahasa Isyarat Indonesia

No.	Tingkat pengetahuan mengenai Bahasa Isyarat Indonesia	Responden	
		N	%
1.	Sangat Kecil	76 orang	46,1%
2.	Kecil	58 orang	35,2%
3.	Sedang	27 orang	16,4
4.	Besar	2 orang	1,2%
5.	Sangat Besar	2 orang	1,2%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

**Tabel 3.10** Jumlah Responden Menjawab Benar dari Jenis Bahasa Isyarat di Indonesia

No.	Jawaban jumlah jenis Bahasa isyarat Indonesia	Responden	
		N	%
1.	Benar (Jawaban : 2 jenis)	24 orang	14,5%
2.	Salah	17 orang	10,3%
3.	Tidak Tahu	124 orang	75,2%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

**Tabel 3.11** Jumlah Responden Pernah Membaca Artikel tentang Ragam Jenis Bahasa Isyarat

No.	Membaca artikel atau buku yang membahas tentang jenis-jenis Bahasa Isyarat	Responden	
		N	%
1.	Ya, pernah	27 orang	16,4%
2.	Tidak, tidak pernah	138 orang	83,6%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

Dari ketiga Tabel 3.9, 3.10, dan 3.11 dapat dilihat bahwa pengetahuan sebagian besar responden mengenai Bahasa Isyarat masih pada tingkat ‘sangat kecil’, perihal ini dinyatakan pada Tabel 3.9 dengan jumlah responden 76 orang atau 46,1%, kemudian tingkat ‘kecil’ dengan jumlah responden 58 orang atau 35,2%, dan sisanya hanya berjumlah 31 orang atau 18,8% yang memiliki tingkat pengetahuan ‘sedang’ hingga ‘sangat besar’ mengenai bahasa isyarat.

Lalu, Tabel 3.10 sebanyak 24 orang atau 14,5% menjawab dengan benar jumlah dari jenis-jenis Bahasa Isyarat yaitu 2 jenis bahasa isyarat, sedangkan terdapat 17 orang atau 10,3% responden yang menjawab salah serta 124 atau 75,2% responden yang menjawab tidak tahu.

Kemudian, Tabel 3.11 menunjukkan bahwa hanya 27 orang atau 16,4% responden yang menjawab pernah membaca buku atau artikel yang membahas tentang jenis-jenis Bahasa Isyarat, sedangkan sisanya 138 orang atau 83,6% responden yang menjawab tidak pernah membaca segala hal yang berkaitan Bahasa Isyarat.

**Tabel 3.12** Jumlah Penilaian Responden Menjawab Benar dan Salah pada Gestur Huruf Bahasa Isyarat Indonesia

No.	Penilaian tentang pengetahuan gestur huruf Bahasa Isyarat	Responden (Benar)		Responden (Salah)	
		N	%	N	%
1.	Huruf – B	55 orang	33,3%	110 orang	66,7%
2.	Huruf – D	90 orang	54,5%	75 orang	45,4%
3.	Huruf – P	61 orang	37%	104 orang	63%
4.	Huruf – I	135 orang	81,8%	30 orang	18,2%
5.	Huruf – R	84 orang	50,9%	81 orang	49,1%
6.	Huruf – M	47 orang	28,5%	118 orang	71,5%
<b>Jumlah (Rata-Rata)</b>		472 benar	47,7%	518 salah	52,3%

**Tabel 3.13** Jumlah Penilaian Responden Menjawab Benar dan Salah pada Gestur Kosakata Bahasa Isyarat Indonesia

No.	Penilaian tentang pengetahuan gestur kosakata Bahasa Isyarat	Responden (Benar)		Responden (Salah)	
		N	%	N	%
1.	Kosakata “Kirim”	4 orang	2,4%	161 orang	97,6%
2.	Kosakata “Wanita”	63 orang	38,2%	102 orang	61,8%
3.	Kosakata “Tuhan”	37 orang	22,4%	128 orang	77,5 %
4.	Kosakata “Mau”	63 orang	38,2%	102 orang	61,9%
<b>Jumlah (Rata-Rata)</b>		167 benar	25,3%	493 salah	74,7%

Kemudian, Tabel 3.12 menjelaskan bahwa penilaian mengenai pemahaman masyarakat umum kepada bahasa isyarat masih sangatlah minim, perihal tersebut dinyatakan dari 6 pertanyaan tentang pengenalan gestur huruf pada bahasa isyarat, rata-rata hanya 472 responden atau 47,7% yang menjawab benar dan 518 responden atau 52,3% responden yang menjawab salah.

Adapun Tabel 3.13 melihatkan hal yang hampir serupa bahwasanya diantara 4 pertanyaan tentang gestur kosakata pada bahasa isyarat, rata-rata hanya 167 responden atau 25,3% yang menjawab benar dan 493 responden atau 74,7% yang menjawab keliru atau salah. Oleh karena itu, dapat dikatakan masih banyaknya pengetahuan masyarakat umum yang perlu ditingkatkan kembali untuk dapat mengenali gestur bahasa isyarat yang biasa digunakan oleh penyandang tunarungu atau teman tuli.

**Tabel 3.14** Responden Tertarik untuk Mempelajari dan Mengenal Ragam Jenis Gestur Bahasa Isyarat Indonesia

No.	Tertarik untuk mempelajari dan mengenal jenis-jenis gestur Bahasa Isyarat	Responden	
		N	%
1.	Ya	146 orang	88,5%
2.	Tidak	19 orang	11,5%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

**Tabel 3.15** Tanggapan Responden Mengenai Aplikasi untuk Mengenali Ragam Jenis Gestur Bahasa Isyarat Indonesia

No.	Tanggapan mengenai aplikasi untuk mengenali jenis-jenis gestur Bahasa Isyarat	Responden	
		N	%
1.	Setuju	162 orang	98,2%
2.	Tidak Setuju	3 orang	1,8%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

**Tabel 3.16** Tanggapan Responden Mengenai Pentingnya Aplikasi

No.	Tanggapan mengenai seberapa pentingnya aplikasi tersebut	Responden	
		N	%
1.	Sangat Penting	87 orang	52,7%
2.	Penting	60 orang	36,4%
3.	Biasa	17 orang	10,3%
4.	Tidak Penting	1 orang	0,6%
5.	Sangat Tidak Penting	0 orang	0%
<b>Jumlah</b>		<b>165 orang</b>	<b>100%</b>

Dari ketiga Tabel 3.14 – 3.16 dapat digambarkan bahwasanya sebagian besar responden yang berjumlah 146 orang atau 88,5% tertarik untuk mempelajari dan mengenal jenis-jenis gestur Bahasa Isyarat, sedangkan 19 orang atau 11,5% sisanya tidak tertarik.

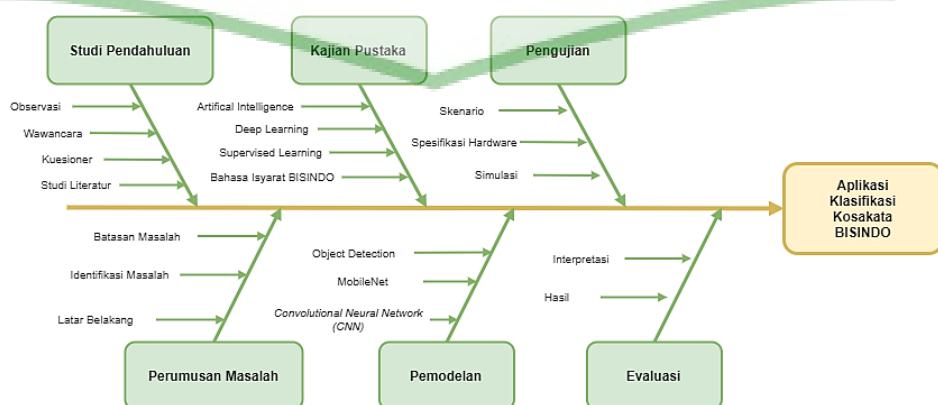
Kemudian, 162 responden atau 98,2% setuju dengan adanya aplikasi yang dapat mengenali jenis-jenis gestur Bahasa Isyarat, sedangkan 3 orang atau 1,8% sisanya memilih tidak setuju. Bahkan tanggapan responden terhadap akan pentingnya aplikasi tersebut memiliki persentase bernilai 87 orang atau 52,7% orang yang menjawab ‘sangat penting’, 60 orang atau 36,4% orang menjawab ‘penting’, 17 orang atau 10,3% menjawab ‘biasa’, dan sisanya menjawab ‘tidak penting’. Oleh karena itu, kesimpulannya bahwa tanggapan sebagian besar responden memiliki respon positif terhadap adanya aplikasi yang dapat mengenali jenis-jenis gestur Bahasa Isyarat tersebut.

### 3.2 Kerangka Penelitian

Kerangka pemikiran menjelaskan tahapan yang dilakukan dalam penelitian yang dilakukan untuk menghasilkan tujuan yang dimaksud. Pada tahap ini menggunakan kerangka penelitian *fishbone diagram* atau *diagram sebab-akibat* (*cause and effect diagram*).

Adapun dalam penelitian ini, penulis mengurutkan serta menghubungkan beberapa faktor yaitu sebagai berikut: (1) Tahap studi pendahuluan, berfungsi untuk mengumpulkan data-data serta informasi dari permasalahan yang berkaitan tentang bahasa isyarat dengan melakukan metode observasi, wawancara, kuesioner, dan studi literatur; (2) tahap perumusan masalah berfungsi untuk mengidentifikasi

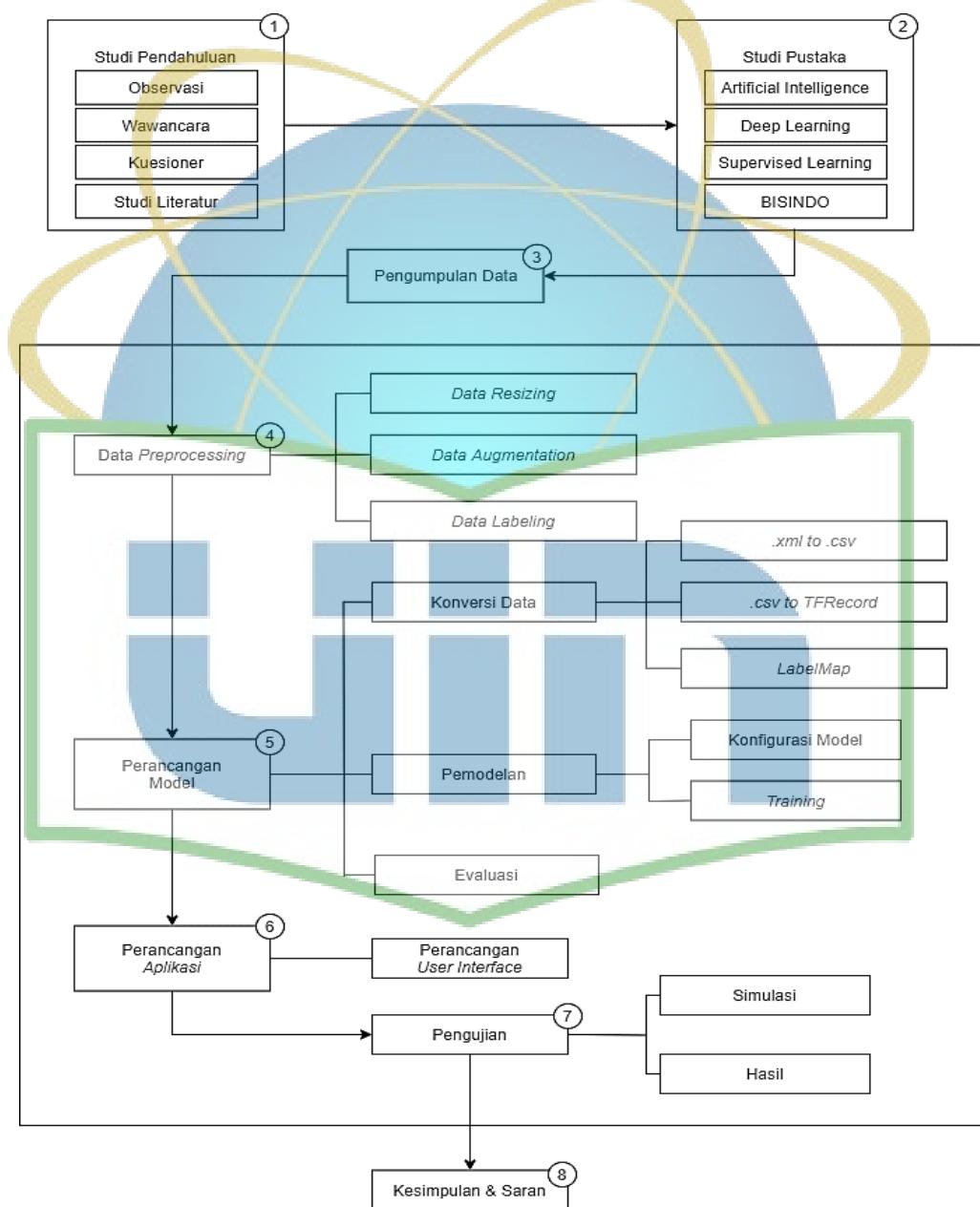
masalah dari informasi yang didapatkan pada tahap sebelumnya, lalu merumuskan batasan dari permasalahan yang ada untuk menjadi pedoman bagi penulis untuk mencapai tujuan dan solusi dari penelitian yang dilakukan; (3) tahap kajian pustaka berfungsi mendapatkan landasan teori dari pembahasan yang dipermasalahkan, agar tergambaran secara jelas oleh penulis definsi dari istilah tersebut seperti *Artificial Intelligence*, *Deep Learning*, *Supervised Learning*, BISINDO, dan lain-lain; (4) tahap pemodelan berguna untuk membangun model *deep learning* untuk melakukan proses pelatihan data (*training data*) sehingga data dapat klasifikasi dan deteksi oleh mesin dengan metode CNN berbasis arsitektur SSD *MobileNet*, (5) tahap pengujian berguna untuk mengetahui hasil kinerja dari model klasifikasi yang telah dibuat pada tahap sebelumnya dengan melakukan skenario, memberikan informasi spesifikasi *hardware* yang digunakan, sehingga simulasi ujian mendapat nilai dari hasil klasifikasi yang telah dibentuk; (6) tahap evaluasi digunakan untuk menggabungkan hasil dari analisis bermacam pertanyaan, kriteria maupun standar tertentu yang berguna untuk mendapatkan interpretasi perbandingan dari pemodelan yang menghasilkan nilai prediksi yang terbaik. Pada Gambar 3.2 merupakan gambaran kerangka penelitian yang penulis lakukan:



Gambar 3.2 Kerangka Penelitian

### 3.3 Tahapan Penelitian

Pada langkah ini penelitian dijelaskan tahapan dari konsep perencanaan penelitian yang akan dilakukan agar memudahkan penulis dalam memulai hingga selesai dan telah mendapatkan hasil yang diharapkan. Adapun penulis menyusun tahapan penelitian sebagaimana pada Gambar 3.3.



Gambar 3.3 Tahapan Penelitian

Berdasarkan langkah-langkah pada tahapan penelitian diatas dijelaskan bahwa:

1. Tahapan pertama diawali dengan studi pendahuluan yang didapatkan dari observasi, wawancara, kuesioner, studi literatur.
2. Kemudian mendapatkan rumusan masalah yang dimulai tahapan penyusunan untuk studi pustaka berdasarkan beberapa landasan teori dari permasalahan yang akan diteliti seperti AI, *deep learning*, *supervised learning*, dan BISINDO.
3. Tahapan berikutnya penulis mengumpulkan data citra yang menjadi objek penelitian yaitu gestur pada kosakata bahasa isyarat Indonesia;
4. Selanjutnya, melakukan augmentasi data untuk memperbanyak variasi data citra dengan menggunakan teknik data *resizing* dan data *augmentation* serta melakukan labelisasi data citra;
5. Pada tahap kelima, penulis melakukan konversi data citra dari .jpg menjadi .csv, lalu TFRecord dan mengkonfigurasi model dengan SSD *MobileNet*, kemudian melakukan proses *training* untuk melakukan pembelajaran pada dataset lalu mendapat nilai evaluasi dari mempelajari objek;
6. Lalu, penulis merancang aplikasi dan desain *user interface* (UI) untuk mengimplementasikan model pada *platform* berbasis *mobile* (Android);
7. Kemudian melakukan pengujian berdasarkan skenario dan disimulasikan sehingga mendapatkan hasil yang diharapkan;
8. Terakhir, penulis mengambil kesimpulan dan saran dari penelitian yang telah dilakukan.



## BAB 4

### HASIL DAN PEMBAHASAN

#### 4.1 Tahapan Data *Preprocessing*

Pada tahap ini merupakan pemrosesan gambar pada masing-masing kosakata dengan menggunakan bermacam *tools* seperti opencv, labelimage, dan augmentor. Hal ini berfungsi memperbanyak variasi data citra, mengubah ukuran gambar, serta membuat labelisasi gambar dengan *bounding box* untuk dapat mendeteksi objek. Berikut adalah tahapannya.

##### 4.1.1 Data *Resizing*

Pengubahan ukuran citra atau *resizing* dilakukan agar mempermudah proses pengolahan citra pada tahap selanjutnya. Tujuan *data resizing* untuk memudahkan dalam pelatihan (*training*) gambar serta dapat mengurangi waktu pemrosesan klasifikasi gambar. Pada proses ini, citra akan diubah dari berbagai ukuran piksel menjadi kisaran  $640 \times 480$  piksel dengan bantuan *library Augmentor* namun dengan acaknya ukuran tidak akan mempengaruhi kinerja saat melakukan *training*. Berikut adalah *code* yang penulis gunakan untuk mengubah ukuran citra.

```
import Augmentor  
  
#To Resize Pixel Of Image in Dataset  
bidadari.resize(probability=1.0, width=640, height=360)
```

##### 4.1.2 Data *Augmentation*

Pada tahap ini, penulis memerlukan variasi data yang beragam untuk memperbanyak jumlah data dengan tujuan dapat menghasilkan nilai validasi dan

*loss* menjadi jauh lebih baik. Penulis menggunakan cara otomatis dengan bantuan dari *library Augmentor*, berikut adalah *code* yang penulis gunakan untuk memperbanyak variasi citra:

```
#To rotate data image without crop
x.rotate_without_crop(probability=1.0, max_left_rotation=10, max_right_rotation=10, expand=True)

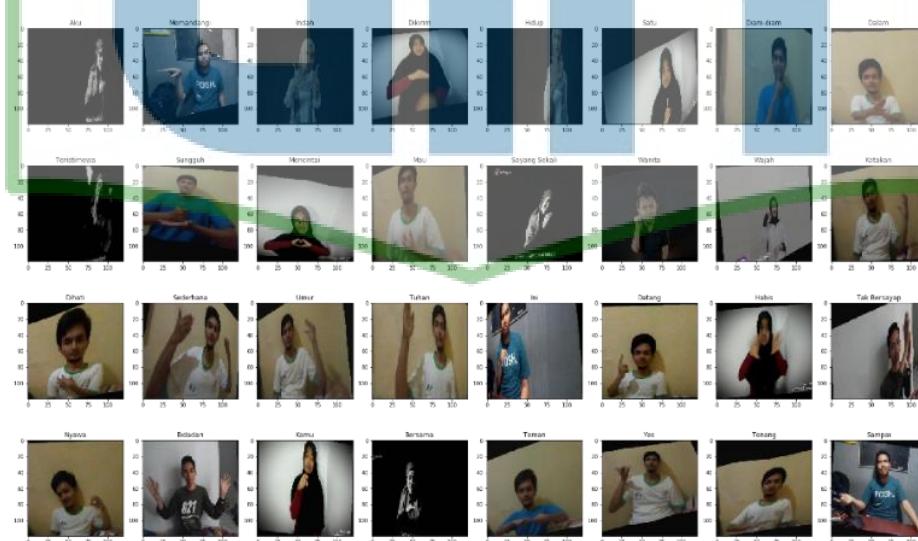
#To Shear the image by a specified number of degrees.
x.shear(probability=1.0, max_shear_left=5, max_shear_right=5)

#To Skew an image towards one corner, randomly by a random magnitude.
x.skew(probability=1.0, magnitude=0.8)

#To Performs a random, elastic distortion on an image.
x.random_distortion(probability=1.0, grid_width=4, grid_height=4, magnitude=10)

#To setting color, brightness, and contrast on data image
x.random_color(probability=1.0, min_factor=0.8, max_factor=1)
x.random_brightness(probability=1.0, min_factor=0.8, max_factor=1)
x.random_contrast(probability=1.0, min_factor=0.8, max_factor=1)
```

Dari *code* tersebut akan menghasilkan beberapa variasi citra gambar pada masing-masing label sebagaimana yang ditunjukkan pada Gambar 4.1.



**Gambar 4.1** Hasil Augmentasi Data Citra

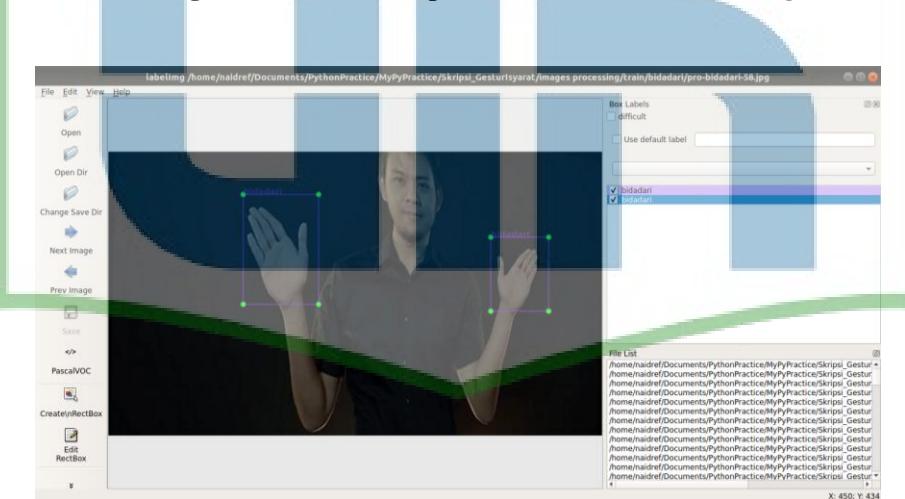
### 4.1.3 Data *Labeling* atau Annotasi

Berikutnya, tahap ini merupakan tahap awal untuk dapat melakukan deteksi suatu objek dengan melakukan annotasi untuk mendapatkan ciri khusus yang mewakili setiap data objek dalam gambar kemudian diberikan label. Hal tersebut akan dijadikan bahan *learning* pada saat melakukan proses *training*.

Annotasi dilakukan untuk mendapatkan beberapa fitur yang dianggap penting pada dataset dimana fitur yang diambil diantaranya adalah nama folder (kelas), direktori penyimpanan dataset, tinggi *layer*, lebar *layer*, kedalaman *layer*, tinggi objek dan lebar objek.

*Tools* yang dimanfaatkan dalam proses ini adalah *labeling*, memberikan tanda kotak pembatas atau *bounding box* pada area objek yang dianggap penting, kemudian disimpan mendapatkan hasil *file* dengan ekstensi dari .jpg menjadi .xml.

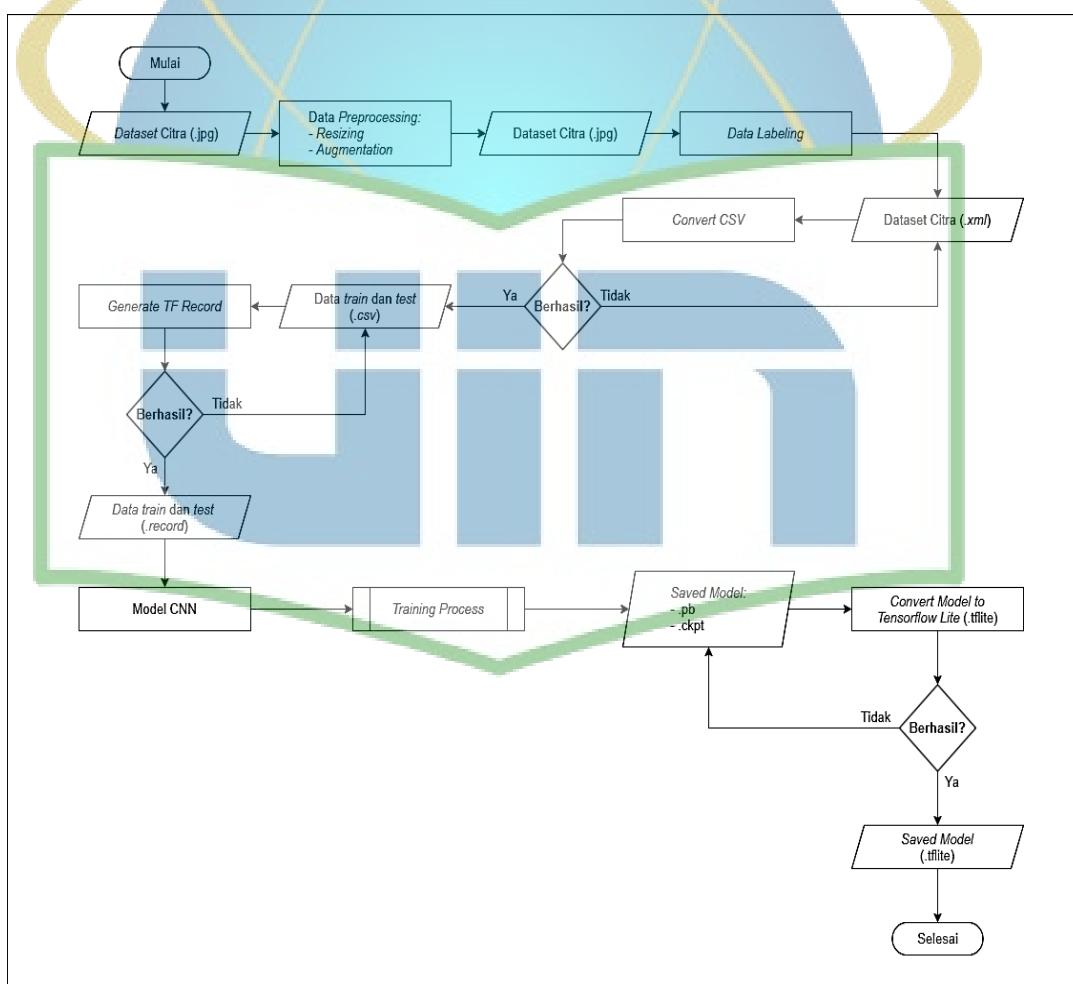
Gambar 4.2 merupakan contoh dari proses annotasi atau *labeling*.



**Gambar 4.2** Proses Annotasi Data (*Labeling*)

## 4.2 Perancangan Model

Pada tahap ini perancangan model yang terdiri atas proses model objek deteksi pada gestur isyarat dengan memerlukan data input citra gestur isyarat sebagai sumber data utama pembelajaran mesin. Lalu, penulis menggunakan model *deep learning* yaitu *SSD MobileNet*, dan dilakukannya proses *training* untuk mendapatkan pemodelan objek deteksi gambar pada gestur isyarat BISINDO. Adapun proses pembuatan model dari awal hingga model dikonversi menjadi ekstensi .tflite yang dapat digunakan pada *platform* Android sebagaimana pada Gambar 4.3.



Gambar 4.3 Flowchart Proses Model

#### 4.2.1 Konversi Data

Pada tahap berikutnya melakukan konversi semua data yang sebelumnya bersifat annotasi dengan ekstensi *file .xml* menjadi *.csv*, kemudian dikonversi kembali menjadi *file TFRecord (Tensorflow Record)*. Berikut *source code* yang digunakan untuk mengkonversi *file .xml* menjadi *.csv*.

```
import os
import re
import glob
import pandas as pd
import xml.etree.ElementTree as ET

def xml_to_csv(path):
    xml_list = []
    for xml_file in glob.glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):
            value = (root.find('filename').text,
                     int(root.find('size')[0].text),
                     int(root.find('size')[1].text),
                     member[0].text,
                     int(member[4][0].text),
                     int(member[4][1].text),
                     int(member[4][2].text),
                     int(member[4][3].text)
                     )
            xml_list.append(value)
    column_name = ['filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax']
    xml_df = pd.DataFrame(xml_list, columns=column_name)
    return xml_df
```

```

def main():
    for directory in ['train', 'test']:
        image_path = os.path.join(os.getcwd(), 'annotations/{}'.format(directory))
        xml_df = xml_to_csv(image_path)
        xml_df.to_csv('data/{}_labels.csv'.format(directory), index=None)
        print('Successfully converted xml to csv.')
main()

```

Konversi xml ke csv dibutuhkan untuk tahapan berikutnya menjadi *feeding data* atau sebagai bahan pembelajaran sistem pada saat melakukan proses *training*, kemudian alasan utama *dataset* diubah menjadi format *TFRecord* dikarenakan merubah format penyimpanan *dataset* yang begitu banyak dari format gambar (*image*) menjadi format biner sehingga *dataset* tidak memakan banyak ruang di dalam *disk*, dan lebih cepat membaca *dataset* ketika *training* berlangsung. Berikut merupakan *source code* yang digunakan untuk konversi file .csv menjadi *TFRecord*.

```

def main(_):
    writer = tf.python_io.TFRecordWriter(FLAGS.output_path)
    path = os.path.join(os.getcwd(), 'images processing/{}'.format(FLAGS.type))
    examples = pd.read_csv(FLAGS.csv_input)
    grouped = split(examples, 'filename')
    for group in grouped:
        tf_example = create_tf_example(group, path)
        writer.write(tf_example.SerializeToString())
    writer.close()
    output_path = os.path.join(os.getcwd(), FLAGS.output_path)
    print('Successfully created the TFRecords: {}'.format(output_path))

```

Pada source code *TFRecord* terdapat fungsi yang mendefinisikan setiap kelas-kelas ke dalam tipe data *integer* yang berguna sebagai ID per masing-masing kelas, susunan ID tersebut disesuaikan dengan susunan pada proses annotasi *dataset*

sebelumnya yang berjumlah 32 kelas dari kosakata gestur isyarat yaitu ‘bersama’, ‘bidadari’, ‘dalam’, dst. Berikut adalah *source code* dari konversi setiap kelas dari format *string* menjadi *integer*.

```
def class_text_to_int(row_label):
    if row_label == 'bersama':
        return 1
    elif row_label == 'bidadari':
        return 2
    elif row_label == 'dalam':
        return 3
    ...
    ...
    elif row_label == 'yes':
        return 32
    else:
        None
```

Sebelum melakukan pemodelan, untuk memberikan informasi pelatihan apa yang perlu dilakukan pada masing-masing objek dengan mendefinisikan pemetaan nama kelas ke nomor ID maka dibutuhkan *file label map* dengan ekstensi .pbtxt sebelum melakukan proses *training*. Susunan nomor ID tersebut harus disesuaikan dengan definisi kelas dalam *file TFRecord* sebelumnya, berikut adalah isi dari *file label map* .pbtxt.

```
item {
  id: 1
  name: 'bersama'
}
item {
  id: 2
  name: 'bidadari'
}
```

#### 4.2.2 Pemodelan dan Pelatihan

##### a. Konfigurasi model

Sebelum melakukan *training data* dibutuhkannya model yang terdefinisi sesuai dengan parameter apa yang akan digunakan. Adapun dalam penelitian ini menggunakan API objek deteksi *tensorflow* yang menggunakan ekstensi *protobuf* untuk konfigurasi proses *training* dan *evaluasi*. Berikut konfigurasi model yang digunakan secara detail.

1. Model terdiri dengan beberapa konfigurasi seperti tipe *feature extractor* ‘SSD MobileNet’, 6 *layers*, *activation RELU*, dan *score converter* menggunakan *RELU*. Berikut adalah konfigurasi model yang dipakai.

```
model {
  ssd {
    num_classes: 32
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
    matcher {
      argmax_matcher {
        matched_threshold: 0.5
        unmatched_threshold: 0.5
        ignore_thresholds: false
        negatives_lower_than_unmatched: true
        force_match_for_each_row: true
      }
    }
    similarity_calculator {
      iou_similarity {
      }
    }
    anchor_generator {
      ssd_anchor_generator {
        num_layers: 6
        min_scale: 0.2
        max_scale: 0.95
        aspect_ratios: 1.0
        aspect_ratios: 2.0
        aspect_ratios: 0.5
      }
    }
  }
}
```

```
        aspect_ratios: 3.0
        aspect_ratios: 0.3333
    }
}
image_resizer {
    fixed_shape_resizer {
        height: 300
        width: 300
    }
}
box_predictor {
    convolutional_box_predictor {
        min_depth: 0
        max_depth: 0
        num_layers_before_predictor: 0
        use_dropout: true
        dropout_keep_probability: 0.8
        kernel_size: 1
        box_code_size: 4
        apply_sigmoid_to_scores: false
        conv_hyperparams {
            activation: RELU_6,
            regularizer {
                l2_regularizer {
                    weight: 0.00004
                }
            }
            initializer {
                truncated_normal_initializer {
                    stddev: 0.03
                    mean: 0.0
                }
            }
            batch_norm {
                train: true,
                scale: true,
                center: true,
                decay: 0.9997,
                epsilon: 0.001,
            }
        }
    }
}
feature_extractor {
    type: 'ssd_mobilenet_v2'
    min_depth: 16
    depth_multiplier: 1.0
    conv_hyperparams {
        activation: RELU_6,
        regularizer {
            l2_regularizer {
                weight: 0.00004
            }
        }
        ..
    }
}
```

```

    }
    loss {
        ..
        hard_example_miner {
            num_hard_examples: 3000
            iou_threshold: 0.99
            loss_type: CLASSIFICATION
            max_negatives_per_positive: 3
            min_negatives_per_image: 0
        }
        classification_weight: 1.0
        localization_weight: 1.0
    }
    normalize_loss_by_num_matches: true
    post_processing {
        batch_non_max_suppression {
            score_threshold: 1e-8
            iou_threshold: 0.6
            max_detections_per_class: 100
            max_total_detections: 100
        }
        score_converter: SIGMOID
    }
}
}

```

2. *Train Config* terdiri atas parameter yang digunakan saat proses *training* berlangsung, berikut adalah konfigurasinya.

```

train_config: {
    batch_size: 24
    optimizer {
        rms_prop_optimizer: {
            learning_rate: {
                exponential_decay_learning_rate {
                    initial_learning_rate: 0.001
                    decay_steps: 10000
                    decay_factor: 0.75
                }
            }
            momentum_optimizer_value: 0.9
            decay: 0.9
            epsilon: 1.0
        }
    }
    fine_tune_checkpoint:
        "models/ssdlite_mobilenet_v2_coco_2018_05_09/model.ckpt"
    fine_tune_checkpoint_type: "detection"
    num_steps: 200000
    data_augmentation_options {
        random_horizontal_flip {
        }
    }
    data_augmentation_options {

```

```
ssd_random_crop {}
```

3. *Train Input Config* berfungsi menentukan lokasi *dataset* yang digunakan untuk proses pelatihan, berikut konfigurasinya.

```
train_input_reader: {
    tf_record_input_reader {
        input_path:
        "model_gestur_isyarat/500data/bias/bias-train.record"
    }
    label_map_path:
    "model_gestur_isyarat/500data/bias/gesturisyarat_map.pbtxt"
}
```

4. *Eval Config* berfungsi untuk memberi keputusan metrik pengukuran dari proses *training* ke evaluasi, berikut ialah konfigurasinya.

```
eval_config: {
    num_examples: 180
    max_evals: 10
    num_visualizations: 180
}
```

5. *Eval Input Config* berfungsi untuk menentukan lokasi *dataset* yang dievaluasi dengan model, berikut konfigurasinya.

```
eval_input_reader: {
    tf_record_input_reader {
        input_path: "model_gestur_isyarat/500data/bias/bias-
test.record"
    }
    label_map_path:
    "model_gestur_isyarat/500data/bias/gesturisyarat_map.pbtxt"
    shuffle: true
    num_readers: 1
}
```

Kemudian, langkah berikutnya melakukan proses *training* dengan jumlah iterasi (*epoch*) sebanyak 200.000 *step* dimana setiap *step* tersebut terdapat variabel hasil pembelajaran atau *learning* yang dilakukan oleh sistem, variabel yang muncul biasanya disebut sebagai nilai *loss*. Nilai *loss* berfungsi sebagai tolak ukur apakah

model dapat dikatakan bagus atau tidak, jika nilai *loss* konsisten semakin kecil maka semakin baik model begitu juga sebaliknya.

Setelah itu, proses *training* yang berjalan akan terotomatisasi oleh sistem untuk menghasilkan *checkpoint* yang berupa *graph tensor* dan tujuan dari *checkpoint* tersebut untuk menyimpan informasi pembelajaran *learning* yang terbaru pada *step* tertentu. Penulis telah melakukan proses *training* selama 5 hari pada server Google Colaboratory untuk mencari beberapa model terbaik, adapun penyebab dari proses *training* yang lama dikarenakan jaringan internet yang kurang stabil sehingga terputus-putus.

Apabila proses *training* selesai, tahap terakhir dilakukan konversi pada *checkpoint* terbaru ataupun merubah *graph tensor* menjadi model yang berekstensi *file protobuf ‘.pb’*. Berikut adalah *source code* yang penulis gunakan.

```
# Exporting the model for Evaluation
from google.protobuf import text_format
from object_detection import exporter
from object_detection.protos import pipeline_pb2

slim = tf.contrib.slim
tf.reset_default_graph()

config_override=''
input_shape=None
trained_checkpoint_prefix = os.path.join(TRAIN_DIR, 'model.ckpt-200000') # EDIT: model.ckpt-<iteration>
output_directory='trained_models/Compare Model/model-4_kelas-bias'

# Type of input node. Can be one of :
# ['image_tensor', 'encoded_image_string_tensor', `tf_example`]
input_type='image_tensor'

pipeline_config = pipeline_pb2.TrainEvalPipelineConfig()
```

```

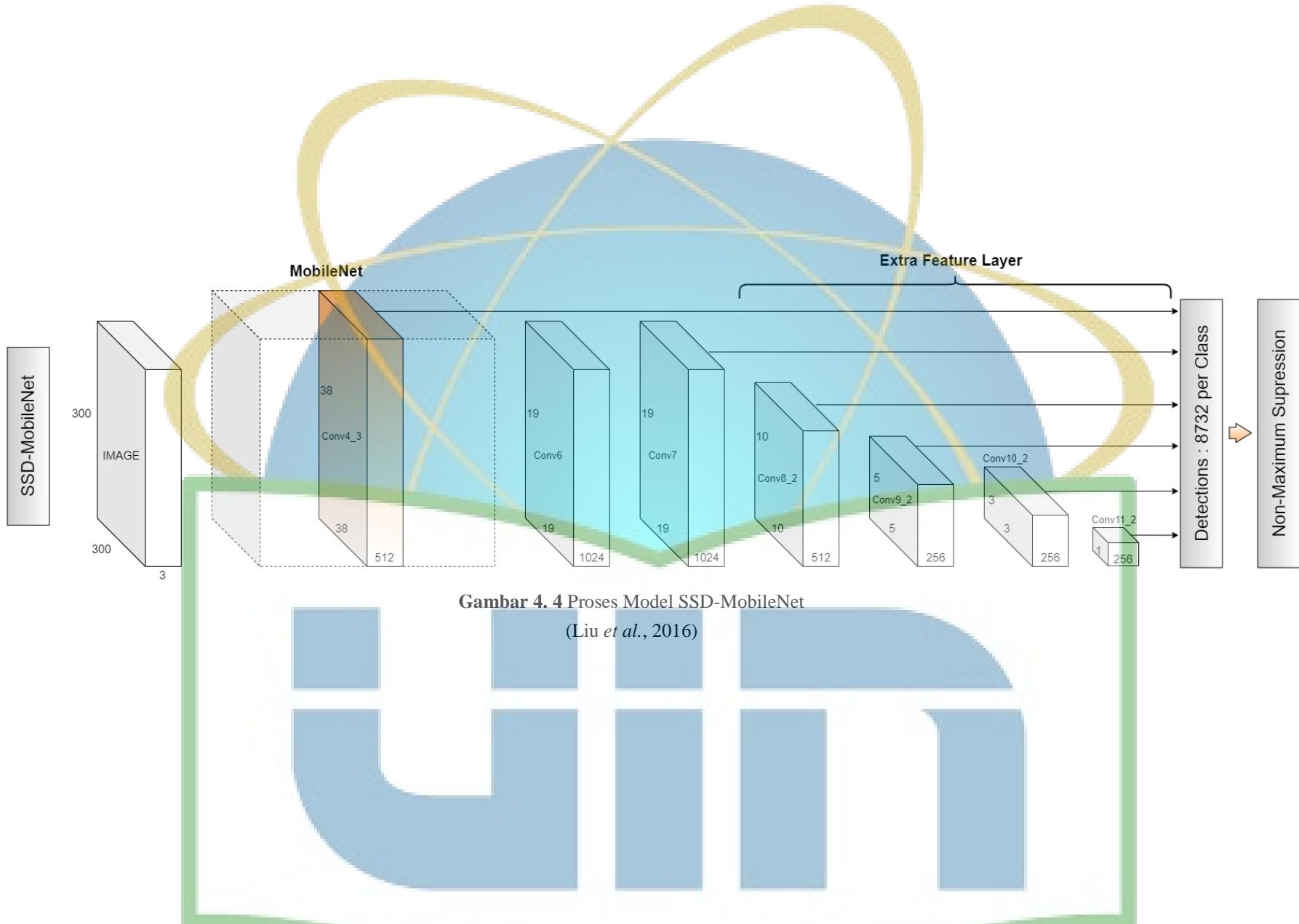
with tf.gfile.GFile(PIPELINE_CONFIG_PATH, 'r') as f:
    text_format.Merge(f.read(), pipeline_config)
text_format.Merge(config_override, pipeline_config)
if input_shape:
    input_shape = [
        int(dim) if dim != '-1' else None
        for dim in input_shape.split(',')
    ]
else:
    input_shape = None
exporter.export_inference_graph(input_type, pipeline_config,
                                 trained_checkpoint_prefix,
                                 output_directory, input_sha
pe)

```

### b. Proses Pelatihan

Pada proses pelatihan ini dijelaskan bagaimana arsitektur dari SSD MobileNet bekerja, pertama salah satu keutamaan dari penggunaan model *mobilenet* ialah mengatasi *computing resource* yang berlebih pada *hardware*. Adapun perbedaan mendasar pada arsitektur ini dengan arsitektur CNN umumnya adalah penggunaan lapisan atau *layer* konvolusi dengan ketebalan *filter* yang sesuai dengan ketebalan dari *input image* dengan salah satunya melakukan pembagian konvolusi menjadi *depthwise convolution* dan *pointwise convolution*.

Sedangkan, SSD atau *Single Shot Detector* ialah algoritma berbasis deteksi objek *real-time* yang didasarkan pada *input image* dan *ground truth box* untuk tiap objek pada pelatihan. Model SSD-MobileNet yang penulis gunakan beserta rincian arsitekturnya ditunjukkan pada Gambar 4.4.



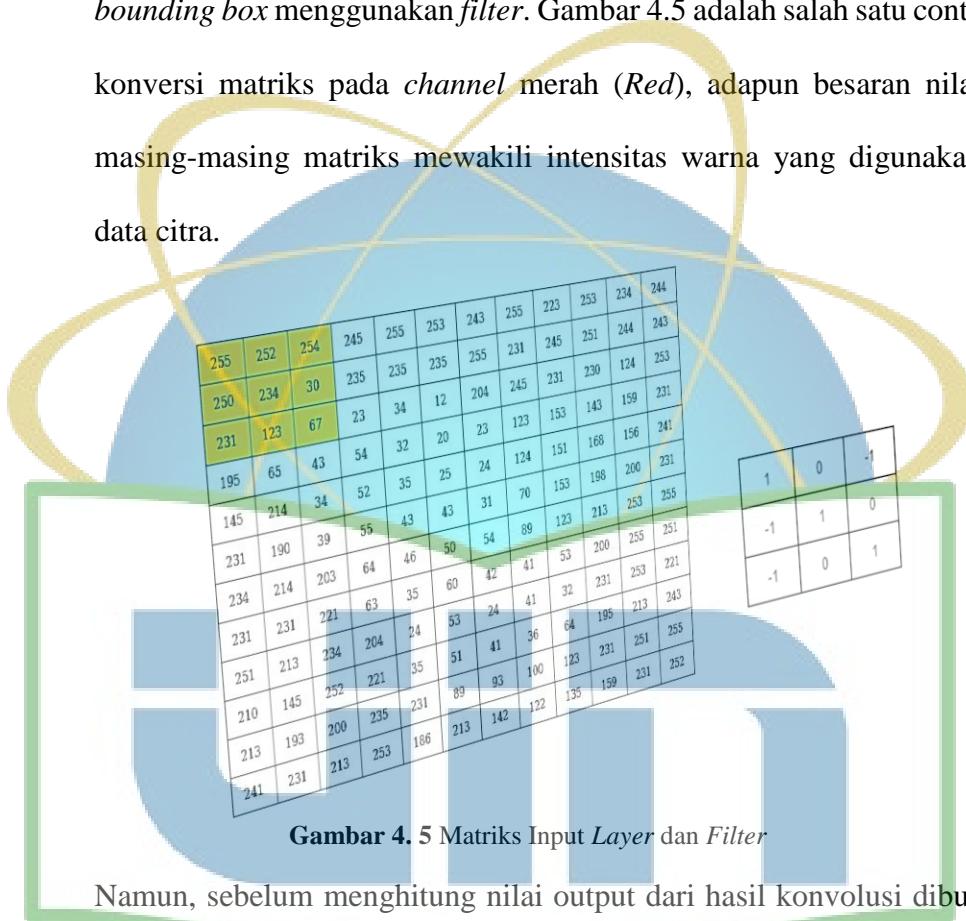
**Tabel 4.1** Struktur Bagan Arsitektur dari MobileNet  
 (Howard *et al.*, 2017)

Type / Stride	Filter Shape	Input Size	
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$	
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$	
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$	
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$	
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$	
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$	
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$	
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$	
Conv / s1	$1 \times 1 \times 128 \times 256$	$56 \times 56 \times 128$	
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$	
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$	
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$	
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$	
5 ×	Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
	Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
	Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
	Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
	Avg Pool / s1	Pool $7 \times 7$	$7 \times 7 \times 1024$
	FC / s1	$1024 \times 1000$	$1 \times 1 \times 1024$
	Softmax / s1	Classifier	$1 \times 1 \times 1000$

Dari Gambar 4.4 terdapat perpaduan antara penggunaan model klasifikasi dari MobileNet dan objek deteksi pada Single Shot Detection (SSD). Adapun pelatihannya pada model yang digunakan, terdiri sebagai berikut:

1. Input *Image*: Pertama melakukan input data citra pada model yang dikonversi menjadi matriks 3 dimensi dengan ukuran  $300 \times 300 \times 3$  channel RGB (*Red, Green, Blue*).

2. *MobileNet – Convolutional Layer:* Setelah melakukan *input image*, model akan langsung diproses pada arsitektur MobileNet dimana awal fase melakukan konvolusi. Lapisan ini berguna untuk mengestraksi fitur-fitur atau karakteristik dari gambar tertutama yang telah diberi garis kotak atau *bounding box* menggunakan *filter*. Gambar 4.5 adalah salah satu contoh dari konversi matriks pada *channel* merah (*Red*), adapun besaran nilai pada masing-masing matriks mewakili intensitas warna yang digunakan pada data citra.

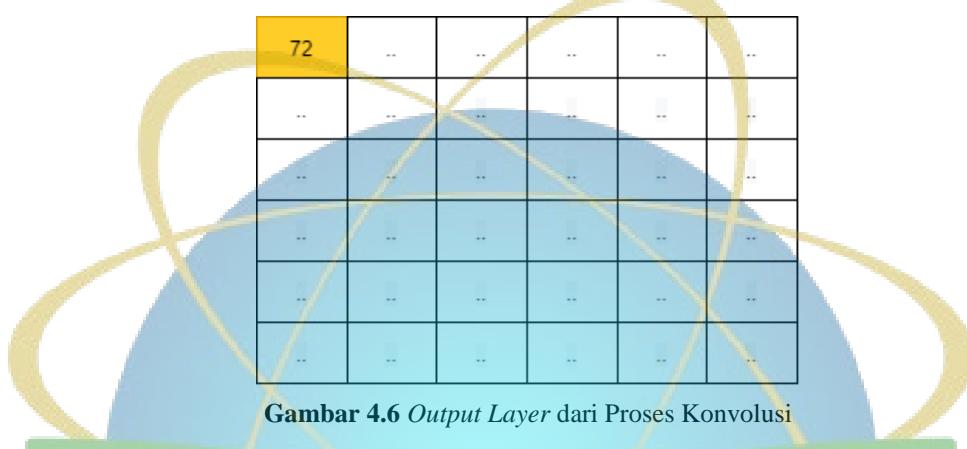


Namun, sebelum menghitung nilai output dari hasil konvolusi dibutuhkan perhitungan untuk ukuran spasial pada citra output yang disesuaikan dengan menggunakan formula berikut berdasarkan arsitektur MobileNet pada lapisan pertama,dst.

$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1$$

$$W_2 = \frac{(224 - 3 + 2(1))}{2} + 1 = 111,5 = 112$$

Setelah itu, fitur pada masing-masing citra memerlukan proses konvolusi dengan *filter* dimana mempunyai nilai bobot atau *weight* secara acak, sebagaimana pada Gambar 4.5 yang terdapat contoh nilai filter. Adapun perhitungan untuk proses konvolusi dari input dengan filter beserta Gambar 4.6 adalah hasil output dari konvolusi layar pertama, dst.



$$y_{m,n} = g \left( \sum_j \sum_k x[j,k] W[m-n, n-k] + b \right)$$

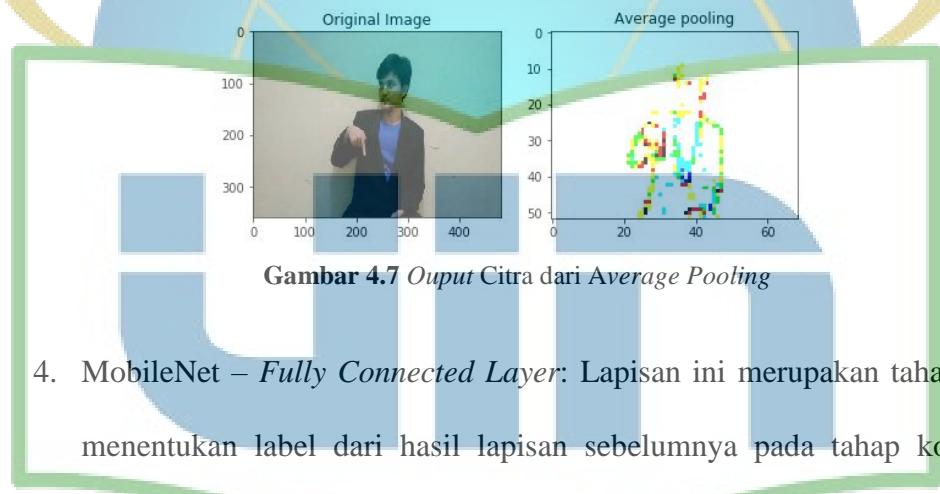
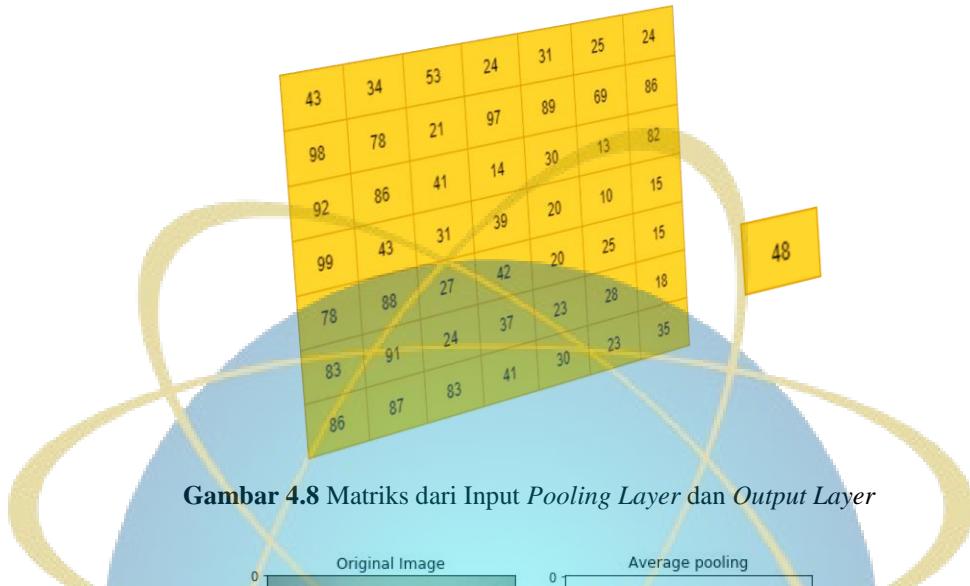
$$(w \times x)_{m,n} = 255 \times 1 + 252 \times 0 + 254 \times (-1) + 250 \times (-1) + 234 \times 1 + 30 \times 0 + 231 \times (-1) + 123 \times 0 + 67 \times 1 = 71$$

$$y_{m,n} = g(\sum_j w_j x_j + b)$$

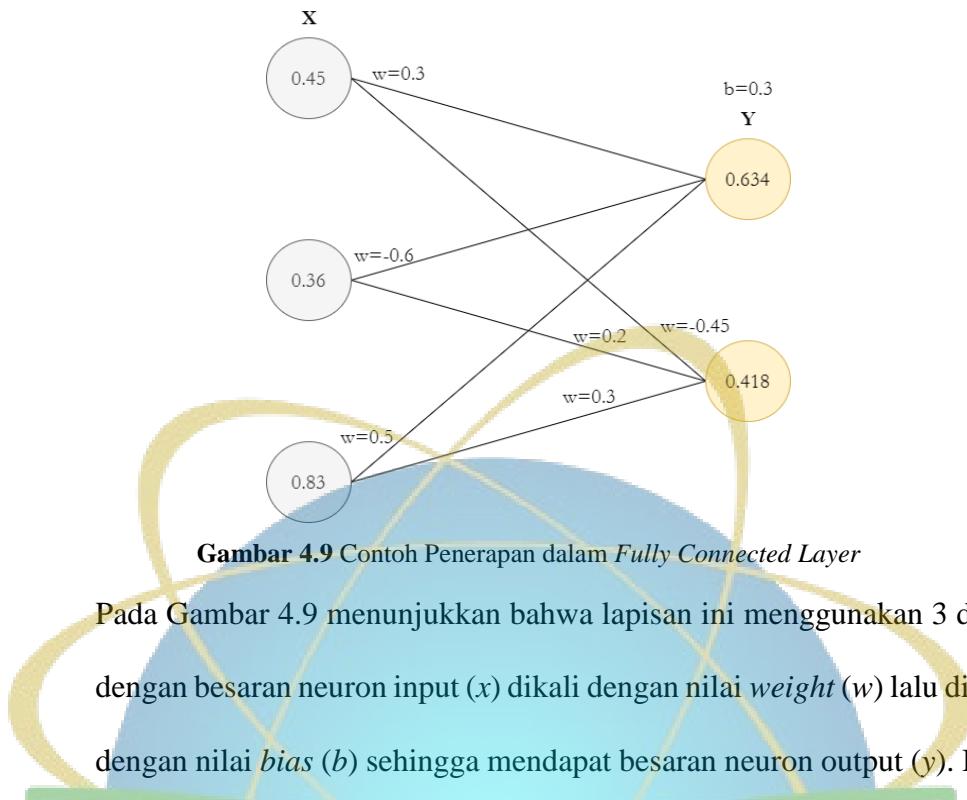
$$y_{0,1} = \max(71 + 1, 0) = 72$$

3. MobileNet – *Average Pooling Layer*: Kemudian, setelah semua lapisan konvolusi menghasilkan fitur map dengan *output* yang baru maka akan diproses kembali pada lapisan *pooling*. Lapisan ini berguna untuk mereduksi ukuran spasial dari citra yang telah dikonvolusi serta mengurangi besaran parameter dan menghaluskan piksel yang ada di citra sehingga

dapat dikenali lebih mudah. Dalam model penelitian ini hanya menggunakan *average pooling* dari ukuran  $7 \times 7$  menjadi  $1 \times 1$  dan stride 1. Berikut adalah contoh matriks dan gambaran dari proses *average pooling*.



4. MobileNet – *Fully Connected Layer*: Lapisan ini merupakan tahap untuk menentukan label dari hasil lapisan sebelumnya pada tahap konvolusi menjadi 1 dimensi vektor (*flatten*) yang memiliki kecenderungan pada label tertentu. Gambar 4.9 adalah salah satu contoh dalam penerapan *Fully Connected Layers*.



$$y = g\left(\sum_{i=1}^n x_i w_i + b\right)$$

$$y = g(0.45 \times 0.3 + 0.36 \times (-0.6) + 0.83 \times 0.5 + 0.3) = 0.634$$

$$y = g(0.45 \times (-0.45) + 0.36 \times 0.2 + 0.83 \times 0.3 + 0.3) = 0.418$$

5. MobileNet – *Softmax*: Tahap terakhir dari arsitektur ini ialah *softmax* dimana fungsinya untuk memproduksi nilai probabilitas dari prediksi klasifikasi. Nilai pada neuron sebelumnya diaplikasikan dengan fungsi aktivasi *softmax*, berikut adalah tabel contoh output dari nilai *softmax*.

**Tabel 4.2** Contoh Penerapan Neuron Terakhir pada *Softmax Classifier*

<i>e</i>	<i>s</i>	Label		
$3.1351e-03$	0.04214	0	0	Bersama
$1.2445e-04$	0.23414	1	0	Bidadari
$4.8527e-06$	0.23511	2	0	Dalam
$2.4214e-04$	0.32352	3	0	Datang
$6.3452e-01$	0.17219	4	1	Mau

6. *Loss Function*: Untuk mengukur dan mengenali seberapa bagus model yang sedang melakukan pembelajaran atau *training*, maka diperlukan variabel antara prediksi dengan aktual agar mengetahui nilai *error* yang didapatkan, jika semakin kecil nilai *loss function* maka semakin bagus model yang digunakan.

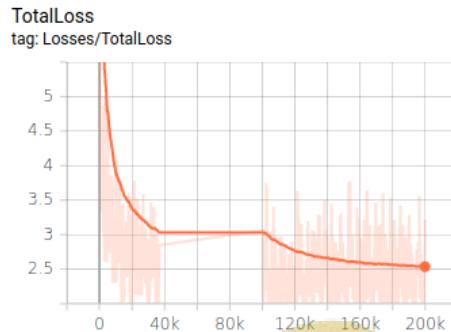
```

INFO:tensorflow:global step 199989: loss = 2.1218 (0.598 sec/step)
INFO:tensorflow:global step 199990: loss = 2.1218 (0.598 sec/step)
INFO:tensorflow:global step 199991: loss = 2.0899 (0.499 sec/step)
INFO:tensorflow:global step 199992: loss = 2.0899 (0.499 sec/step)
INFO:tensorflow:global step 199993: loss = 2.0526 (0.462 sec/step)
INFO:tensorflow:global step 199994: loss = 2.0526 (0.462 sec/step)
INFO:tensorflow:global step 199995: loss = 2.0526 (0.462 sec/step)
INFO:tensorflow:global step 199996: loss = 2.3198 (0.527 sec/step)
INFO:tensorflow:global step 199997: loss = 2.3198 (0.527 sec/step)
INFO:tensorflow:global step 199998: loss = 1.9838 (0.459 sec/step)
INFO:tensorflow:global step 199999: loss = 1.9838 (0.469 sec/step)
INFO:tensorflow:global step 1999990: loss = 2.9857 (0.457 sec/step)
INFO:tensorflow:global step 1999991: loss = 2.9857 (0.457 sec/step)
INFO:tensorflow:global step 1999992: loss = 2.5869 (0.466 sec/step)
INFO:tensorflow:global step 1999993: loss = 2.5869 (0.466 sec/step)
INFO:tensorflow:global step 1999994: loss = 2.5869 (0.466 sec/step)
INFO:tensorflow:global step 1999995: loss = 2.5869 (0.466 sec/step)
INFO:tensorflow:global step 1999996: loss = 2.2585 (0.486 sec/step)
INFO:tensorflow:global step 1999997: loss = 2.2585 (0.486 sec/step)
INFO:tensorflow:global step 1999998: loss = 1.5988 (0.457 sec/step)
INFO:tensorflow:global step 1999999: loss = 1.5988 (0.457 sec/step)
INFO:tensorflow:global step 2000000: loss = 1.5225 (0.457 sec/step)
INFO:tensorflow:Saving model to disk.
INFO:tensorflow:Saving model to disk.
/tensorflow-1.15.2/python3.6/tensorflow/core/python/summary/writer.py:386: UserWarning: Attempting to use a closed FileWriter. The operation warnings.warn("Attempting to use a closed FileWriter."

```

**Gambar 4.10** *Loss function* ketika Proses *Training*

Ketika melakukan proses *training* akan didapat beragam nilai *loss function* sebagaimana Gambar 4.10, model di-*training* dengan *epoch* 200.000 *step*. Jika nilai *loss function* menjadi semakin menurun dan menuju titik terendah dengan nilai konstan maka nilai klasifikasi semakin baik. Adapun Gambar 4.11 adalah grafik model yang penulis gunakan dengan *library tensorflow* dari hasil proses *training*.



**Gambar 4.11 Hasil Total Loss Function dari Proses Training**

Dalam grafik pada Gambar 4.11 menunjukkan terdapat total *loss function*, maksudnya jika grafik mengambarkan terjadinya penurunan nilai *loss/error* dari gabungan *loss classification* dan *loss localization* dengan nilai menurun secara *konvergen*, Hal ini menunjukkan bahwa hasil prediksi yang dilakukan oleh model cukup baik untuk mengklasifikasi dan mendeteksi sebuah objek

7. SSD – Non Maximum Supression: Setelah melakukan proses *training* pada tahap arsitektur MobileNet maka nilai *output* akan diproses kembali pada SSD dengan melakukan deteksi pada lokasi gambar. Setiap gambar akan memiliki beberapa jumlah *box* kebenaran (*ground truth box*) serta nilai *confidence*, hal ini bertujuan untuk memvalidasi sebuah objek dengan label tertentu yang memiliki nilai ambang batas (*threshold*). Jika sebuah model dapat mendeteksi objek dengan lokasi yang benar sesuai pada tahap labeling sebelumnya maka akan terdapat 4 box dalam satu objek dan 1 kelas, seperti pada Gambar 4.12.

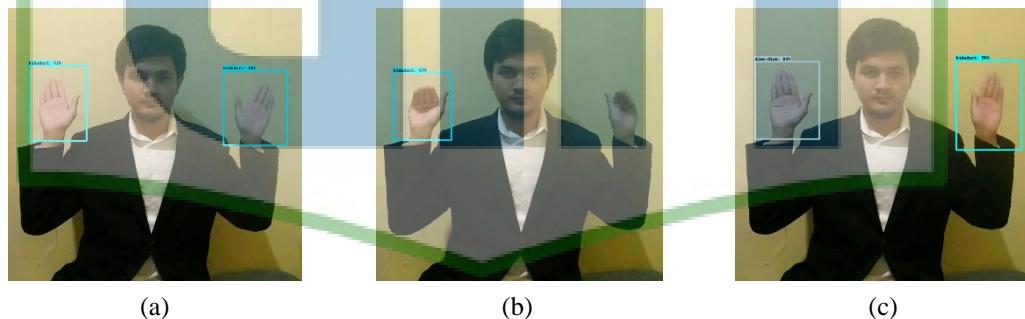


**Gambar 4.12** (a) Sebelum dan (b) Sesudah *Non Maximum Supression*

#### 4.2.3 Evaluasi

Setelah melakukan serangkaian proses *training*, model dievaluasi untuk mengukur kinerja yang dilakukan pada data *testing*. Kemudian hasil dari proses evaluasi *training-testing* akan memiliki 3 variabel penentu untuk mengukur kinerja model yaitu *loss classification*, *loss localization* dan *Average Precision* (AP) untuk setiap kelas atau *Mean Average Precision* (mAP) untuk nilai rata-rata keseluruhan.

Namun, dikarenakan jumlah evaluasi data uji yang banyak dan dilakukan secara *black-box* oleh model, maka penulis melakukan ilustrasi tahapan evaluasi dari satu kelas yang mendekati metode tersebut, sebagaimana pada Gambar 4.13.



**Gambar 4.13** Ilustrasi Evaluasi Gambar Uji

Dari Gambar 4.13 menunjukkan bahwa identifikasi hasil evaluasi gambar berdasarkan (a)*True Positive*, (b)*False Positive*, dan (c)*False Negative*. Setelah diketahui identifikasi *confusion matrix* pada setiap gambar pada kelas ‘Bidadari’, penulis membuat keterangan dari 8 gambar sebagaimana pada Tabel 4.3.

**Tabel 4.3** Identifikasi *Confusion Matrix*

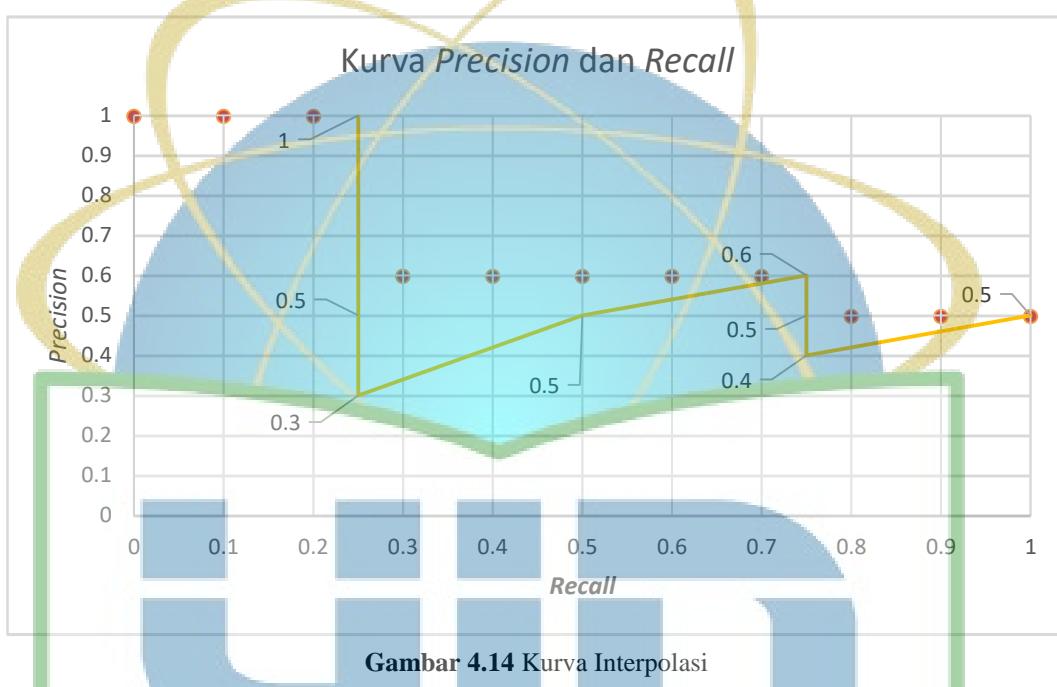
Gambar	Deteksi	Akurasi	TP / FP
Gambar 1	Bidadari	96%	TP
Gambar 2	Bidadari	49%	FP
Gambar 3	Datang	85%	FP
Gambar 4	Bidadari	90%	TP
Gambar 5	Bidadari	91%	TP
Gambar 6	Bidadari	34%	FP
Gambar 7	Bidadari	40%	FP
Gambar 8	Bidadari	92%	TP

Setelah diketahui hasil *confusion matrix* pada Tabel 4.3, maka dihitung nilai *precision* dan *recall* menggunakan persamaan 2.10 dan 2.11. Kemudian, penulis membuat plot dari akumulasi tiap gambar yang terdeteksi TP atau FP serta menentukan nilai *precision interpolation* dengan mengambil nilai *precision* maksimum dan nilai *recall* tertinggi sebagaimana pada Tabel 4.4.

**Tabel 4.4** Akumulasi dari *Precision* dan *Recall*

No.	TP / FP	Precision	Recall	Precision_inter
1	TP	1/1=1	1/4=0,25	1
2	FP	1/2=0,5	1/4=0,25	1
3	FP	1/3=0,3	1/4=0,25	1
4	TP	2/4=0,5	2/4=0,5	0,5
5	TP	3/5=0,6	3/4=0,75	0,6
6	FP	3/6=0,5	3/4=0,75	0,6
7	FP	3/7=0,4	3/4=0,75	0,6
8	TP	4/8=0,5	4/4=1	0,5

Berdasarkan Tabel 4.4 menjelaskan bahwa terdapat nilai *precision*, *recall*, dan *precision interpolation* yang berfungsi untuk mengurangi dampak ‘wiggles’ yang disebabkan oleh variasi kecil dalam peringkat deteksi (Everingham *et al.*, 2010). Kemudian sebelum menghitung *Average Precision* (AP) menggunakan persamaan 2.12, penulis menggambarkan plot kurva dari hasil interpolasi nilai *precision* dan *recall* sebagaimana pada Gambar 4.14.



Setelah itu, AP dihitung dengan mengambil nilai maksimal pada kurva PR yang ditindai dengan titik merah lalu mengelompokkan *recall* secara merata menjadi 11 bagian: {0,0;0,1;0,2;0,3;...;0,9;1,0}. Dengan hasil perhitungan sebagai berikut:

$$AP = \frac{1}{11} \sum_{r \in \{0,0,1,..,0,9,1\}} \rho_{interp(r)} \quad (2.16)$$

$$AP = \frac{1}{11} (3 \times 1,0 + 5 \times 0,6 + 3 \times 0,5)$$

$$AP = 0,6818 = 68,18 \%$$

Adapun penulis telah melatih beberapa model dan membaginya menjadi 2 skenario alternatif untuk mendapatkan hasil prediksi terbaik yang diharapkan, yaitu:

1. **Skenario 1:** Penulis menggunakan model yang dilatih dengan jumlah keseluruhan label 32 kelas, dengan beberapa rincian sebagai berikut:

**Tabel 4.5 Detail Skenario Pertama**

<b>Jumlah kelas</b>	32
<b>Resolusi gambar</b>	$640 \times 480$
<b>Data training</b>	16000
<b>Data testing</b>	1600

Dari hasil proses *training* dengan model sesuai pada tahap konfigurasi sebelumnya, menghasilkan nilai total *loss* yang cukup tinggi dan mengalami kenaikan pada beberapa iterasi terakhir. Hal tersebut ditunjukkan pada

**Gambar 4.15.**



**Gambar 4.15 Hasil Grafik Loss Skenario 1**

Pada kedua Gambar 4.15 menunjukkan bahwa model yang dibangun kurang baik dan tepat untuk melakukan klasifikasi dan deteksi dikarenakan jumlah nilai *loss* sangat tinggi dan berlainan dengan kaidah pada *machine learning* “*jika semakin rendah nilai loss, semakin tinggi nilai akurasi*”. Adapun pada Gambar 4.14 menunjukkan nilai evaluasi rata-rata dari data *testing* (uji).

```

    □ WARNING:root:image 0 does not have groundtruth difficult flag specified
    {'Losses/Loss/classification_loss': 8.035435,
     'Losses/Loss/localization_loss': 1.4505482,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/bersama': 0.03919533138265066,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/bidadari': 0.41328361491685467,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/dalam': 0.9797663495122246,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/datang': 0.000942199992955514,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/diam-diam': 0.04414236583734076,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/dihati': 0.03452130849881557,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/dikirim': 0.275,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/habis': 0.129777660391745,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/hidup': 0.21651766613604193,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/indah': 0.22359047659276998,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/ini': 1.0,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/kamu': 0.08824238593898054,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/katakan': 0.041359447004608293,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/ku': 0.000621180124223602,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/mau': 0.09826802778764573,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/mendangi': 0.35727150669679403,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/mencintai': 0.5881758314114208,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/nyawa': 0.01103122901390494,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sampai': 0.27843162790860787,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/satu': 0.2839852875391421,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sayang sekali': 0.058663924690809874,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sederhana': 0.7390983011168872,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sungguh': 0.3667630923370168,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/tak bersayap': 0.17783577766415298,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/teman': 0.5483749055177626,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/tenang': 3.4685489325540656e-05,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/teristikimewa': 0.2245686853751304,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/tuhan': 0.17650118873632872,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/umur': 0.652713519818783,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/wajah': 0.0032818913748738575,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/wanita': 0.04251140862573453,
     'PascalBoxes_PerformanceByCategory/AP@0.5IOU/yes': 0.17102104603286253,
     'PascalBoxes_Precision/mAP@0.5IOU': 0.258296182661305}

```

**Gambar 4.16 Hasil Evaluasi Skenario 1**

Dari Gambar 4.16 menjelaskan bahwa hasil evaluasi pada setiap kelas menunjukkan nilai klasifikasi dan deteksi yang tertinggi berada pada kelas gestur kosakata ‘**ini**’ dengan nilai AP sebesar 1.0, diikuti gestur kosakata ‘**dalam**’ dengan nilai AP sebesar 0,97. Sedangkan, nilai yang terendah berada pada kelas ‘**tenang**’ dengan nilai sebesar  $3,46 \times 10^{-5}$ , dan diikuti kelas ‘**ku**’ dengan nilai sebesar  $6,21 \times 10^{-4}$ . Kemudian, semua nilai *evaluasi* setiap kelas dijumlahkan dan diambil rata-rata keseluruhannya yang diwakili dengan variabel mAP maka didapatkan nilai sebesar 0,25.

Kesimpulan dari skenario pertama menjelaskan bahwa model yang di-*training* memiliki nilai akurasi yang rendah yaitu 0,25, hal ini dikarenakan beberapa objek gestur/kelas memiliki persamaan yang hampir sama dengan gestur/kelas lain dan beberapa gestur juga memiliki 2 gerakan yang berbeda sehingga ketika melakukan proses *training* mengalami bias data dengan

data lainnya. Oleh karena itu, model yang digunakan sulit untuk melakukan pembelajaran sehingga hasil klasifikasi dan deteksi yang rendah.

2. **Skenario 2:** Disebabkan pada Skenario 1 menghasilkan nilai akurasi yang rendah, maka penulis membagi data membagi model menjadi 2 bagian, yaitu: Model Bias dan Model Non-Bias. Model bias merupakan model yang terdiri atas 8 kelas dengan berisikan objek gerakan isyarat yang hampir mirip dengan kelas lain dan mempunyai banyak gerakan yang berbeda dalam satu kelas. Sedangkan, Model non-bias adalah model yang memiliki siswa kelas dari sebelumnya berjumlah 24 kelas dan memiliki tingkat perbedaan yang lebih jauh antar kelas lain. Rinciannya adalah sebagai berikut:

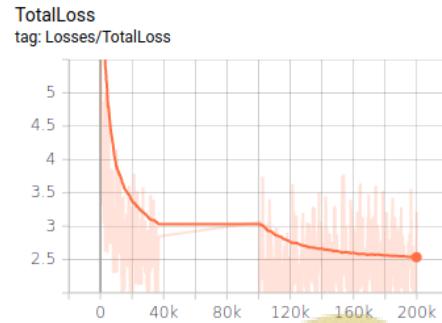
**Tabel 4.6** Detail Skenario Kedua

	Model Bias	Model Non-Bias
Jumlah kelas	8	24
Resolusi gambar	$640 \times 480$	$640 \times 480$
Data training	4000	12000
Data testing	400	1200

Kemudian setelah kedua model dilatih berdasarkan parameter yang sama, maka penulis mendapatkan perbedaan dari hasil kedua model tersebut diantaranya adalah:

- a. Model Bias

Pada model ini menghasilkan *loss classification* dan *loss localization* yang jauh berbeda dari skenario pertama. Hal ini ditunjukkan sebagaimana pada Gambar 4.15



**Gambar 4.18** Hasil Grafik *Loss* Skenario 2 Model Bias

Gambar 4.17 menunjukkan bahwa terjadi penurunan nilai *loss* dengan konstan, dimana nilai total *loss* mencapai angka 2,5. Adapun hasil evaluasi akhirnya sebagaimana pada Gambar 4.16.

```
WARNING:root:image 0 does not have groundtruth difficult flag specified
{'Losses/Loss/classification_loss': 4.8615675,
 'Losses/Loss/localization_loss': 0.8603257,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/datang': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/habis': 0.9485884485884484,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/katakan': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sayang sekali': 0.831875,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sungguh': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/tak bersayap': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/tenang': 0.7005379283386025,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/wanita': 0.7164906895676826,
 'PascalBoxes_Precision/mAP@0.5IOU': 0.8996865083118417}
```

**Gambar 4.17** Hasil Evaluasi Skenario 2 Model Bias

Dari Gambar 4.18 menjelaskan bahwa hasil evaluasi dari setiap kelas yang memiliki nilai tertinggi pada kelas '**katakan**', '**datang**', '**sungguh**' dan '**tak bersayap**' dengan nilai AP sebesar 1,0, sedangkan nilai terendah berada pada kelas '**tenang**' dengan nilai sebesar 0,085.

Selanjutnya, nilai dari setiap evaluasi kelas dirata-rata dengan nilai mAP sebesar 0,899.

#### b. Model Non-bias

Sedangkan untuk model ini menghasilkan nilai *loss* yang lebih tinggi dibanding model bias sebelumnya, sebagaimana hal tersebut ditunjukkan pada Gambar 4.19.



**Gambar 4.19** Hasil Grafik *Loss* pada Skenario 2 Model Non Bias

Pada Gambar 4.19 menunjukkan penurunan nilai *loss* yang terjadi pada kisaran angka 3,15. Namun dalam penilaian hasil evaluasi model bernilai cukup tinggi sebagaimana pada Gambar 4.20.

```
WARNING:root:image 0 does not have groundtruth difficult flag specified
{'Losses/Loss/classification_loss': 1.9136157,
 'Losses/Loss/localization_loss': 0.2972517,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/bersama': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/bidadari': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/dalam': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/diam-diam': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/dihati': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/dikirim': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/hidup': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/indah': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/inti': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/kamu': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/ku': 0.9999999999999999,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/mau': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/memandangi': 0.9999999999999999,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/mencintai': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/nyawa': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sampai': 0.9999999999999999,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/satu': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/sederhana': 0.9693877551020407,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/teman': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/teristimewa': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/tuhan': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/umur': 1.0,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/wajah': 0.9744306418219463,
 'PascalBoxes_PerformanceByCategory/AP@0.5IOU/yes': 1.0,
 'PascalBoxes_PerformanceByCategory/mAP@0.5IOU': 0.9976590998718328}
```

**Gambar 4.20** Hasil Evaluasi Skenario 2 Model Non Bias

Pada Gambar 4.20 menerangkan bahwa nilai evaluasi pada masing-masing kelas memiliki besaran nilai baik, dengan nilai IOU tertinggi hampir pada seluruh kelas sebesar 1,00 dan 0,99. Sedangkan nilai IOU terendah berada pada kelas ‘**sederhana**’ sebesar 0,96. Kemudian dari masing-masing nilai tersebut dievaluasi rata-rata mendapatkan nilai mAP sebesar 0,997.

#### 4.2.4 Perbandingan Evaluasi

Setelah itu, penulis mencoba melakukan perbandingan model dari berbagai hasil evaluasi pengukuran yang terdiri atas beberapa jumlah parameter berbeda sebagaimana pada Tabel 4.7.

**Tabel 4.7** Tabel Perbandingan Evaluasi Model

Jumlah Kelas	Batch Size	Learning Rate	Epoch	Loss	mAP
4				2,4	0,74
8				2,9	0,76
12				3,3	0,57
16				3,2	0,47
20	8	0,004	200.000	3,8	0,53
24				4,0	0,38
28				4,21	0,27
32				4,23	0,25
32	20	0,002	200.000	2,8	0,23
8	24	0,001	200.000	2,5 3,2	0,89 0,99
24					

Berdasarkan Tabel 4.7 menjelaskan bahwa model yang memiliki jumlah kelas antara 4 – 32 kelas dengan parameter nilai *batch size* sebesar 8 dan nilai *learning rate* sebesar 0,004 menghasilkan nilai *loss* antara 2,4 – 4,23 dan nilai mAP antara 0,74 – 0,25. Sedangkan pada model yang memiliki jumlah sebesar 32 kelas dengan parameter nilai *batch size* 20 dan nilai *learning rate* sebesar 0,002 menghasilkan nilai *loss* sebesar 2,8 dan nilai mAP sebesar 0,23.

Adapun kedua model yang dipisah antara 8 kelas (bias) dan 24 kelas (non bias), terdapat hasil yang berbeda dan memiliki perbedaan yang cukup jauh antar masing-masing model dengan parameter yang berbeda dibanding model sebelumnya. Pada model bias dengan jumlah data yang memiliki sifat bias atau variasi gestur gerakan yang dinamis dalam satu kelas, terdapat nilai *loss* dan evaluasi mAP yang memiliki besaran nilai 2,5 dan 0,899. Sedangkan, pada model unbias dengan jumlah data yang bersifat non bias atau statis dalam satu kelas, menghasilkan nilai *loss* dan evaluasi yang lebih tinggi daripada model bias yaitu 3,2 dan 0,997.

Oleh karena itu, penulis menyimpulkan bahwa sifat dataset yang ditentukan dalam tiap kelasnya berdasarkan gestur kosakata dan jumlah kelas mempengaruhi pembelajaran data ketika proses pelatihan berjalan sehingga akan menghasilkan nilai akurasi yang cenderung kecil dan sulit untuk mengklasifikasikan masing-masing label.

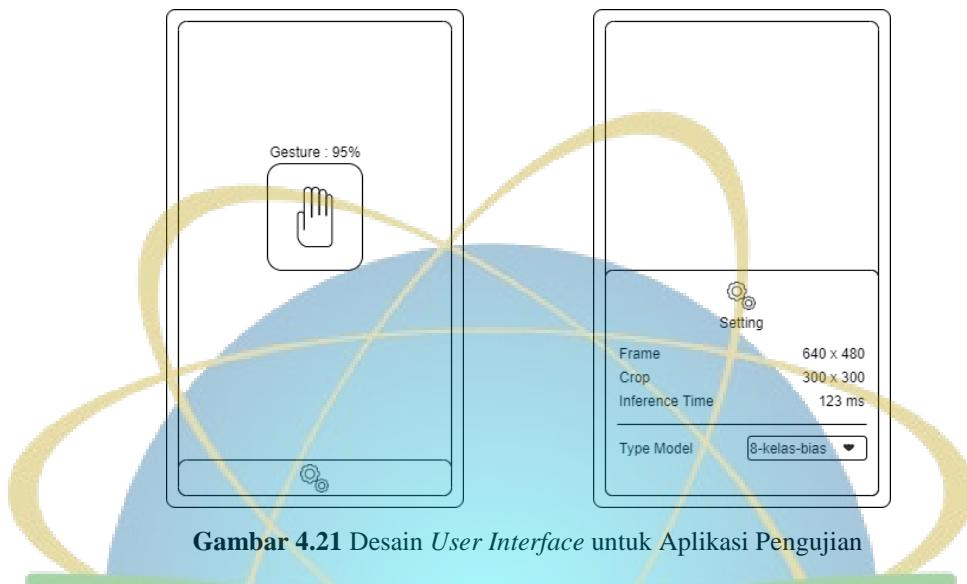
### 4.3 Perancangan Aplikasi

#### 4.3.1 Perancangan User Interface

Setelah melakukan evaluasi kinerja model dengan data *testing*, penulis menguji model CNN dengan mengimplementasikan pada aplikasi berbasis android.

Salah dua alasan utama dari penggunaan *platform* android dikarenakan secara mobilitas lebih mendukung untuk menerjemahkan gerakan isyarat jika dipakai sehari-hari, dan arsitektur dari SSD MobileNet dapat dikonversi menjadi ukuran lebih ringan atau *lite* agar mempermudah kinerja *processor* pada *smartphone*.

Adapun penulis merancang desain *user interface* aplikasi serta menentukan penggunaan fitur yang dimiliki. Hal tersebut diterangkan sebagaimana pada Gambar 4.21.

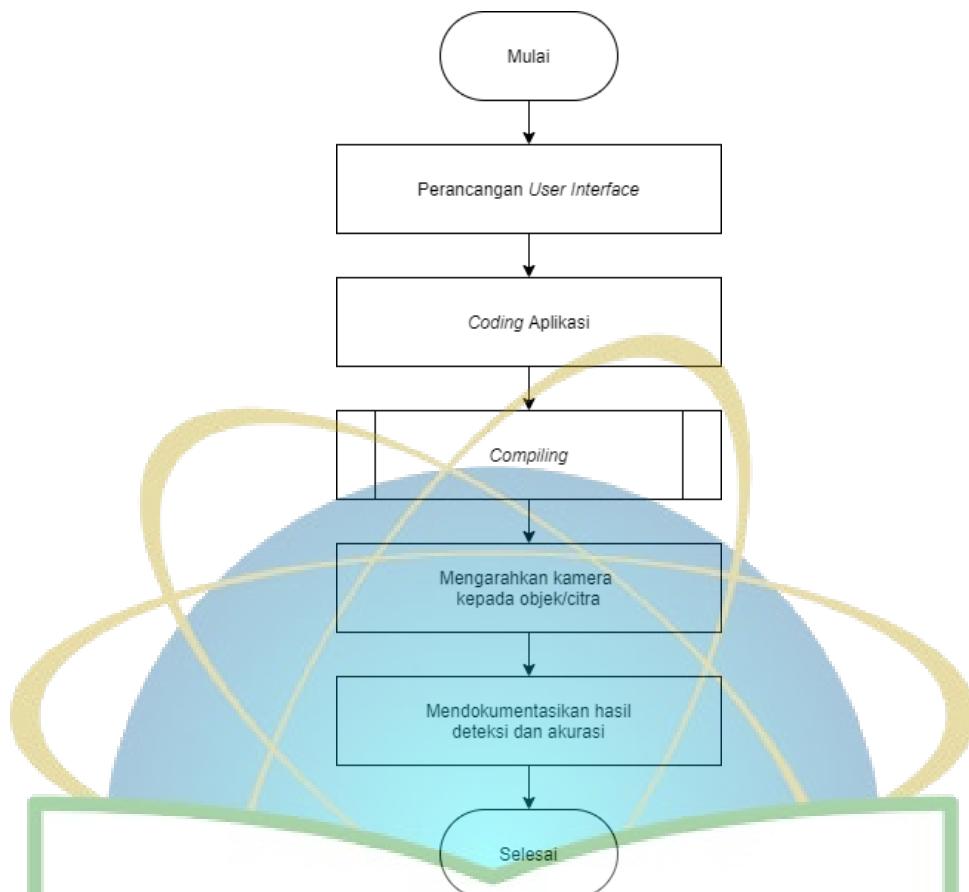


**Gambar 4.21** Desain *User Interface* untuk Aplikasi Pengujian

Pada Gambar 4.21 (a) menggambarkan *layout* dari kamera android serta diarahkan ke objek, kemudian subjek tersebut melakukan gerakan isyarat berdasarkan kelas yang dilatih maka kamera menampilkan sebuah *single box/multi box* yang dapat mendeteksi objek gerakan, sedangkan (b) menggambarkan *layout* dari pengaturan untuk memilih model objek deteksi yang dilatih berdasarkan skenario sebelumnya yaitu model-bias-8-kelas dan model-unbias-24-kelas.

### 4.3.2 Perencanaan Pengujian

Pada tahap ini melakukan perencanaan *testing* dengan implementasi kamera *smartphone* dengan *platform* android. Adapun proses implementasi dan pengujian sebagaimana pada Gambar 4.22.



Gambar 4.22 Proses Implementasi dan Pengujian di Aplikasi

Kemudian, berikut adalah langkah-langkah pada implementasi dan pengujian berdasarkan pada Gambar 4.22.

1. Merancang *user interface*
2. Merancang aplikasi kamera di Android Studio dengan melampirkan model yang sudah dilatih lalu generate menjadi ekstensi ‘.tflite’, dan menyatukan file label gestur dengan ekstensi ‘.txt’.
3. Kemudian, meng-*compile* program pada *smartphone* Galaxy Lenovo A7700.
4. Objek dari pengujian adalah penulis sendiri dengan melakukan 32 label gestur isyarat, agar objek pengujian beragam maka penulis menggunakan 2

jenis pakaian yaitu seragam kemeja putih dan seragam tuxedo (jas hitam dan kemeja putih) dengan latar belakang berwarna kuning.

5. Setelah ditentukan objek pengujian, proses pengujian akan mengambil data citra gestur setiap kelas yang berjumlah 2 sehingga totalnya menjadi kelipatan 2 per model citra yaitu 16 citra pada model 8 bias dan 48 citra pada model 24 non bias.
6. Lalu, penulis menghitung akurasi dari setiap hasil deteksi untuk mengukur kinerja model yang dibangun.

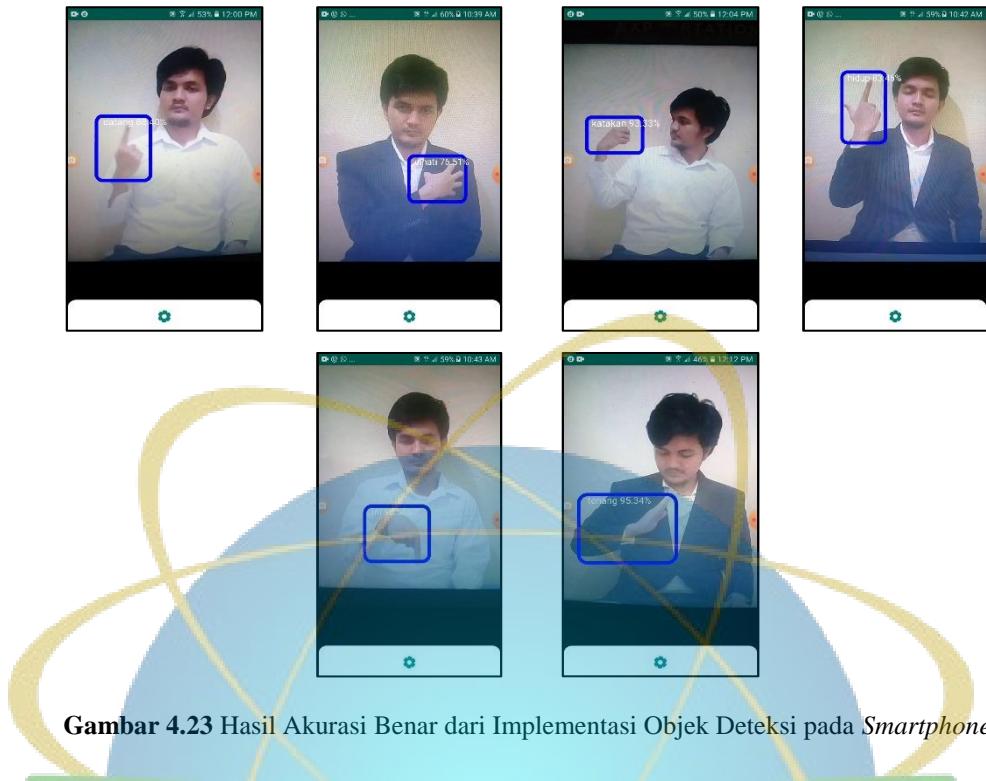
#### **4.3.3 Simulasi dan Hasil**

Penulis menjalankan aplikasi langsung dari *smartphone*, kemudian mengarahkan kamera kepada objek pengujian yaitu penulis sendiri dengan menggunakan pakaian kemeja putih dan tuxedo (jas hitam-kemeja putih). Lalu, melakukan gestur isyarat sesuai dengan label pada masing-masing data citra.

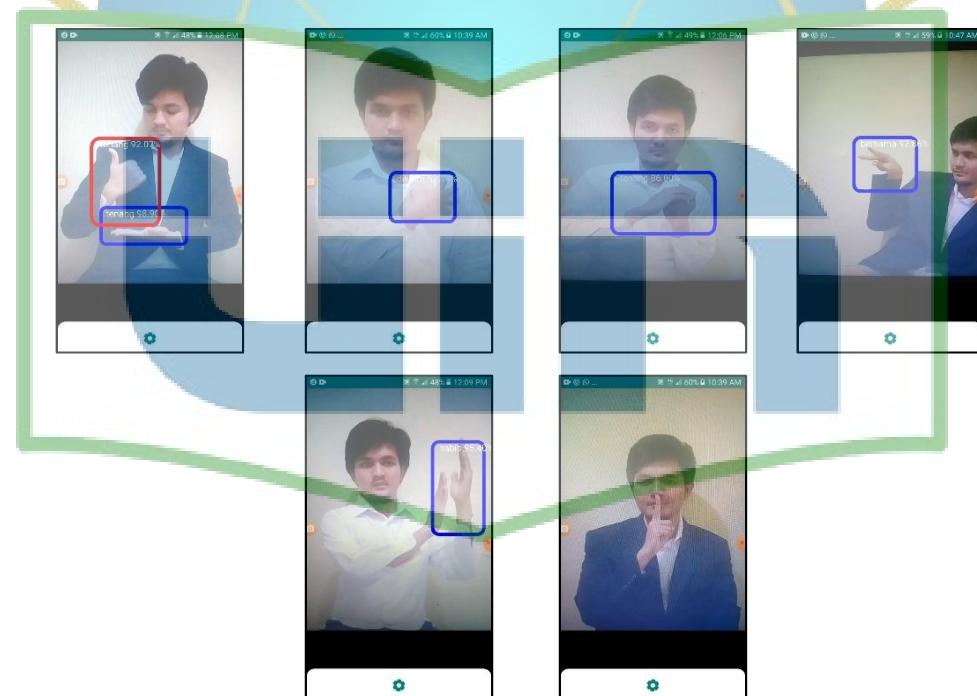
Setiap akurasi dan objek yang terdeteksi dicatat dan didokumentasikan, setiap objek yang terdeteksi dengan benar pada Gambar 4.23 dan setiap objek yang terdeteksi dengan salah pada Gambar 4.24. Setelah itu, penulis mendokumentasikan hasil dari setiap nilai akurasi dan deteksi dikumpulkan dan dihitung besaran nilainya pada masing-masing model berdasarkan 2 gambar per kelas/gestur.

Adapun formula perhitungan akurasi sebagaimana persamaan berikut.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \quad 2.9$$



Gambar 4.23 Hasil Akurasi Benar dari Implementasi Objek Deteksi pada *Smartphone*

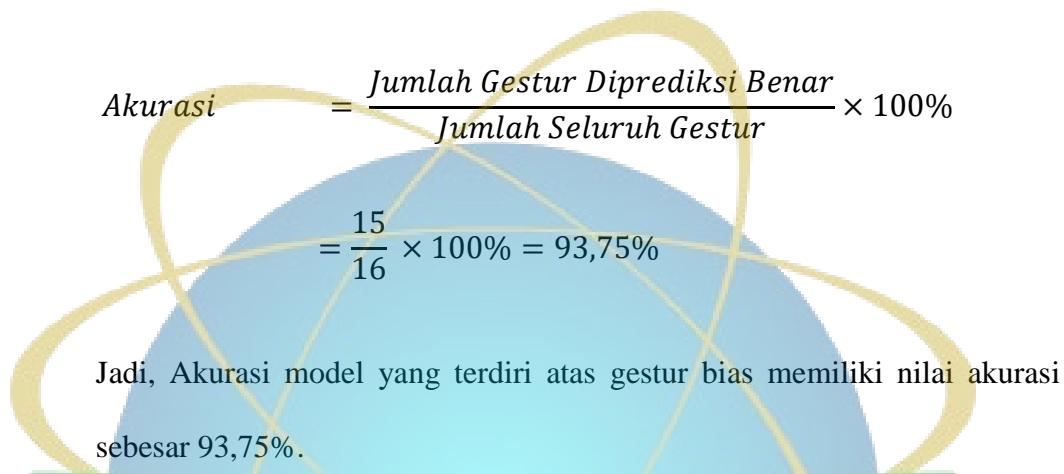


Gambar 4.24 Hasil Akurasi Salah dari Implementasi Objek Deteksi pada *Smartphone*

### 1. Model Bias (8 Kelas)

**Tabel 4.8** Hasil Pengujian Model Bias dengan 8 kelas

Prediksi	
Benar	15
Salah	1



### 2. Model Non Bias (24 Kelas)

**Tabel 4.9** Hasil Pengujian Non Bias dengan 24 Kelas

		Prediksi
Benar		36
Salah		12



Jadi, Akurasi model yang terdiri atas gestur non bias memiliki nilai akurasi sebesar 75%.

#### 4.3.4 Interpretasi

Berdasarkan dari proses pengujian objek deteksi dan klasifikasi menggunakan metode *Convolutional Neural Network* pada 32 jenis gestur kosakata bahasa isyarat, penulis mendapatkan hasil yang dapat diperbandingkan dengan objek penelitian gestur huruf, angka atau kosakata menggunakan metode SVM, *Haar Classifier-KNN*, dan CNN pada Tabel 4.10.

Pada penelitian ini, penulis melakukan perbandingan klasifikasi dan objek deteksi kosakata gestur isyarat dengan metode CNN berbasis arsitektur SSD *MobileNet* yang mendapatkan nilai akurasi pengujian (*real-time*) pada *mobile* untuk 8 kelas dan 24 kelas sebesar 93,75% dan 75%.

Adapun pada penelitian lain, Raheja *et al.* (2016) mengklasifikasikan dan mendeteksi objek pada 4 kosakata ISL dengan metode SVM yang menghasilkan nilai akurasi pengujian 97,5% pada *desktop* (*real-time*) dengan bantuan kamera *Microsfot Kinect*. Kemudian, Borman *et al.* (2017) dan Rakhman *et al.* (2010) melakukan klasifikasi dan objek deteksi pada huruf BISINDO dan SIBI menggunakan metode *Haar Classifier-KNN* mendapat nilai akurasi pengujian di *desktop* sebesar 91,8% dengan 24 kelas (non *real-time*) dan 89,68% dengan 19 kelas (*real-time*). Selain itu, penelitian Bheda & Radpour (2017) dan Masood *et al.* (2018) yang hanya melakukan klasifikasi pada huruf dan angka ASL menggunakan metode CNN berbasis arsitektur model VGG-16 dan GoogleNet yang mendapatkan nilai akurasi (non *real-time*) sebesar 97% dengan 11 kelas, 82,5% dengan 26 kelas, dan 94,68% dengan 36 kelas pada *desktop*.

Model klasifikasi dan objek deteksi pada gestur kosakata isyarat yang menggunakan metode CNN berbasis *SSD MobileNet* mendapatkan nilai akurasi yang sedikit lebih rendah dibanding metode SVM namun memiliki keunggulan dalam jumlah kelas lebih banyak daripada metode tersebut. Disamping itu, metode CNN berbasis *SSD MobileNet* memiliki jumlah nilai akurasi jauh lebih rendah dibandingkan metode *Haar Classifier-KNN* dengan jumlah kelas yang sama namun memiliki perbedaan objek yaitu huruf.

Hal tersebut dikarenakan bantuan *device* yang digunakan, jika kemampuan pada *desktop* dengan bantuan kamera *webcam* atau *Microsoft Kinect* dapat menghasilkan proses komputasi gambar jauh lebih tinggi dibandingkan komputasi pada *mobile* sehingga terjadinya penurunan nilai akurasi. Ditambah lagi, objek gestur isyarat pada huruf **dan angka bersifat statis** dibandingkan gestur gerakan pada kosakata yang bersifat dinamis sehingga sangat mempengaruhi tingkat akurasi, bahkan jika **semakin banyak jumlah kelas maka semakin mempengaruhi nilai akurasi.**

Hal ini dapat dibuktikan pada penelitian yang dilakukan penulis bahwa metode CNN pada model bias dengan 8 kelas memiliki akurasi 93,75% dan model non bias dengan 24 kelas **memiliki akurasi 75%** pada *mobile*. Sedangkan, metode SVM dengan objek kosakata berjumlah 4 kelas memiliki akurasi 97,5%, lalu metode *Haar Classifier-KNN* dengan objek huruf berjumlah 19 kelas dan 24 kelas memiliki akurasi 89,68% dan 92,08%.

Tabel 4.10 Hasil Perbandingan Metode

Jumlah Kelas	Metode			BISINDO
	SVM ISL (Raheja <i>et al.</i> , 2016)	Haar Classifier - KNN SIBI (Rakhman <i>et al.</i> , 2010)	CNN – SSD MobileNet	
4 Kelas	97,5%	-	-	-
8 kelas	-	-	-	93,75%
19 kelas	-	89,68%	-	-
24 kelas	-	-	91,8%	75%

#### 4.4 Implikasi Hasil

Berdasarkan interpretasi diatas maka implikasi dari penelitian ini adalah:

1. Berdasarkan hasil perbandingan evaluasi pada model dari penelitian ini didapatkan bahwa:
  - a. Hasil evaluasi menggunakan model yang berjumlah antara 4 – 32 kelas dengan parameter *batch size* = 8 dan *learning rate* = 0,004 menghasilkan nilai *loss* terendah pada 4 kelas sebesar 2,4 dan nilai *mAP* tertinggi pada 8 kelas sebesar 0,76. Hal ini menjelaskan bahwa semakin banyak jumlah kelas pada model maka nilai *loss* semakin tinggi dan nilai evaluasi semakin rendah.
  - b. Perbandingan evaluasi model 32 kelas dengan parameter *batch size* = 20, *learning rate* = 0,002 menghasilkan nilai *loss* lebih rendah (2,8) namun menghasilkan nilai *mAP* lebih rendah sedikit (0,23) dibanding parameter *batch size* = 8 dan *learning rate* = 0,004. Hal ini menandakan bahwa semakin besar nilai *batch size* dan semakin kecil nilai *learning rate* mempengaruhi hasil evaluasi menjadi lebih baik.
  - c. Berdasarkan perbandingan evaluasi dengan parameter dan sifat gestur yang berbeda, model 8 kelas (**bias**) dan 24 kelas (**non bias**) menggunakan parameter *batch size* lebih besar yaitu 24 dan *learning rate* lebih kecil yaitu 0,001. Dari hasil yang diperoleh, dapat dilihat bahwasanya model bias dan model non bias memiliki nilai *loss* dan *mAP* lebih baik dibanding model sebelumnya.

- d. Berdasarkan nilai *loss* dan mAP lebih baik pada model bias dan non bias maka penulis menyimpulkan model tersebut dapat diimplementasikan pada *smartphone* untuk tahap selanjutnya yaitu tahap pengujian.
2. Hasil pengujian akurasi yang diperoleh pada model bias (8 kelas) dan non bias (24 kelas) dengan menggunakan *smartphone*, kemudian penulis melakukan perbandingan metode dengan literatur sebelumnya bahwa
- Pada Tabel 4.8 dan 4.9, model bias (8 kelas) memiliki nilai akurasi lebih tinggi sebesar 93,75% terhadap model non bias (24 kelas) yang bernilai 75%. Hal ini menandakan bahwasanya CNN dapat mengklasifikasikan beberapa gestur dalam satu kelas, namun memiliki kekurangan apabila jumlah kelas **lebih banyak** maka nilai akurasi menjadi turun.
  - Pada Tabel 4.10 menghasilkan perbandingan akurasi dengan metode lain dimana CNN memiliki **keunggulan jumlah klasifikasi lebih banyak** sebesar 8 kelas dibandingkan SVM berjumlah 4 kelas, namun nilai akurasi menjadi sedikit lebih rendah menjadi 93,75%. Sedangkan, untuk jumlah kelas yang sama berjumlah 24 kelas; metode *Haar Classifier KNN* memiliki nilai akurasi lebih baik sebesar 91,8% dibandingkan CNN yang hanya berjumlah 75%.
  - Hal ini dikarenakan perbedaan proses klasifikasi pada *device*, sehingga menyebabkan proses klasifikasi CNN pada *smartphone* menjadi lebih rendah dibanding metode lain yang memanfaatkan *device* berbasis *desktop*.

Dari hasil interpretasi di atas, dapat diketahui implikasi bahwa untuk orang awam mengklasifikasikan gestur isyarat menjadi lebih mudah dan cepat menggunakan kamera *smartphone* berbasis *realtime* dengan metode CNN. Hal ini berdasarkan hasil penelitian yang diuji pada perangkat kamera *smartphone* berbasis *realtime* pada model 8 kelas yang berjumlah 15 dari 16 prediksi benar dan model 24 kelas yang berjumlah 36 dari 48 prediksi benar.

#### 4.5 Limitation Of Study

Berdasarkan hasil analisis yang didapatkan maka penelitian ini terbatas pada hal-hal berikut, yaitu:

1. Pengumpulan data menggunakan data citra pribadi dari penulis dengan memperbanyak variasi data citra pada teknik *preprocessing* seperti merubah ukuran menjadi 300 piksel, rotasi, distorsi cahaya, warna, dan kontras sehingga berjumlah 16000 data untuk *training* dan 1600 data untuk *testing* dengan proporsi 90% : 10%. Proporsi data untuk *training-testing* masih terbilang cukup tinggi dan rendah untuk variasi data dikarenakan keterbatasan waktu saat proses sebelum dan pelatihan. Oleh karena itu, untuk penelitian berikutnya dapat menambah variasi dan ukuran citra lebih besar menjadi 512 piksel, serta memanfaatkan proporsi data sebesar 80% : 20% dan 70% : 30% untuk dapat menghasilkan klasifikasi dan akurasi yang lebih akurat.
2. Pada penelitian ini hanya menggunakan satu model klasifikasi dan objek deteksi arsitektur SSD *MobileNet* yang diimplementasikan pada *smartphone* dan membandingkan metode lain dengan *device* yang berbeda seperti *desktop*, sehingga menghasilkan nilai yang terbilang rendah dikarenakan

keterbatasan jumlah data citra dan waktu *training* yang digunakan oleh penulis. Oleh karena itu, untuk penelitian berikutnya dapat memanfaatkan algoritma lain yang memiliki akurasi lebih baik dengan *device* yang sama seperti *Inception*, *Faster R-CNN* dan *YOLO* untuk mengetahui seberapa besar pengaruh teknik arsitektur yang dimiliki dalam lingkup metode CNN.

3. Penelitian ini hanya memanfaatkan metode klasifikasi dan objek deteksi secara bersamaan pada sebuah gambar menjadi sebuah kosakata sehingga dapat mengenali gestur secara langsung. Oleh karena itu, diharapkan pada penelitian selanjutnya dapat meningkatkan kualitas aplikasi dengan metode algoritma seperti *insert sorting*, *quick sorting*, dan lain-lain untuk menyusun sebuah pola kalimat secara langsung dari beberapa kosakata.





## BAB 5

### PENUTUP

#### 5.1 Kesimpulan

Penelitian ini telah berhasil mengetahui melakukan penerapan model klasifikasi pada gestur isyarat, mendapatkan hasil klasifikasi dan akurasi terhadap gestur isyarat, mengetahui faktor-faktor yang mempengaruhi hasil akurasi dan perbandingan metode lain pada klasifikasi.

Berdasarkan hasil pembahasan klasifikasi gestur isyarat dengan penerapan model CNN serta pengujinya pada penilitian ini maka dapat diambil kesimpulan bahwa:

1. Penelitian ini telah berhasil mengklasifikasikan gestur isyarat dengan menggunakan 2 model yaitu model 8 kelas yang berjumlah sebesar 15 dari 16 prediksi yang benar dan model 24 kelas yang berjumlah sebesar 35 dari 48 prediksi yang benar.
2. Hubungan antara jumlah kelas dan sifat gestur memiliki pengaruh yang signifikan terhadap akurasi pada pengujian *real time*, sehingga menjadikan 2 model dengan jumlah dan sifat yang berbeda seperti pada model 8 kelas yang bersifat bias menghasilkan nilai akurasi sebesar 93,75% dan model 24 kelas yang bersifat non bias menghasilkan nilai akurasi sebesar 75%.
3. Pada metode CNN dengan SSD *MobileNet* yang diujikan *real-time* berbasis *mobile* memiliki kelebihan mengklasifikasi lebih baik dengan jumlah kelas lebih banyak dibanding metode SVM, namun memiliki kelemahan nilai

akurasi menjadi lebih rendah dibanding metode *Haar Classifier-KNN* dengan jumlah kelas yang sama.

Berdasarkan hasil kesimpulan tersebut, penulis menganggap penelitian ini dapat memberikan manfaat dan kontribusi dalam beberapa hal yang menjadi bahan referensi bagi orang yang tidak mengerti atau mengenali bahasa isyarat (awam), menjadi referensi bagi penelitian-penelitian yang terkait dengan klasifikasi dibidang gambar serta mendorong pemanfaatan metode CNN.

## 5.2 Saran

Berdasarkan hasil dari penelitian yang telah dilakukan, penulis memiliki beberapa saran yang perlu dipertimbangkan baik untuk penelitian selanjutnya maupun untuk lembaga yang terkait, yaitu sebagai berikut:

1. Mengeksplorasi model SSD *MobileNet* dengan menambahkan jumlah *dataset*, variasi data, iterasi dan meningkatkan resolusi data citra serta membandingkan arsitektur CNN yang lain seperti *Inception*, *Faster R-CNN* dan *YOLO*.
2. Membandingkan metode CNN dengan metode lainnya seperti *Scale-Invariant Feature Transform* (SURF), *Speed Up Robust Feature*, dan *Features from Accelerated Segment Test* (FAST).
3. Meningkatkan fitur aplikasi dengan menyusun kosakata menjadi sebuah kalimat bahasa isyarat dengan bantuan algoritma pengurutan seperti *insert sorting*, *quick sorting*, dan lain-lain.



## DAFTAR PUSTAKA

- A'la, F. Y. (2016). *Deteksi Retak Permukaan Jalan Raya Berbasis Pengolahan Citra Menggunakan Metode Ekstraksi Ciri Wavelet*. Yogyakarta: Universitas Muhammadiyah Yogyakarta.
- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... Zheng, X. (2016). Tensorflow : A System For Large-Scale Machine Learning. *Operating Systems Design and Implementation (OSDI)*, 265–283. [https://doi.org/10.1016/0076-6879\(83\)01039-3](https://doi.org/10.1016/0076-6879(83)01039-3)
- Ali, I., Hatta, Z. A., Islamic, I., Damansara, J., Heights, D., Moham, A., ... Mohamed, R. E. K. (2013). Malay Sign Language Courseware for Hearing-Impaired Children in Malaysia. *2nd International Conference on Computer Research and Development, ICCRD 2010*. <https://doi.org/10.1109/ISM.Workshops.2007.87>
- Amrizal, V., & Aini, Q. (2013). *Kecerdasan Buatan* (Q. Aini, Ed.). Jakarta: Halaman Moeka Publishing.
- Android Developers. (2018). Meet Android Studio. <https://doi.org/10.1016/j.catena.2014.01.010>
- Arfian. (2018). *Implementasi Convolutional Neural Network Terhadap Transportasi Tradisional Menggunakan Keras*. Yogyakarta: Universitas Islam Indonesia.
- Ariel. (2019). *Perbandingan Model InceptionV2 dan MobileNet pada CNN untuk Implementasi Algoritma SSD dalam Pencarian Korban Longsor*. Bogor: Insitut Pertanian Bogor.
- Bheda, V., & Radpour, D. (2017). *Using Deep Convolutional Networks for Gesture Recognition in American Sign Language*. Retrieved from <http://arxiv.org/abs/1710.06836>
- Borman, R. I., Prioprudono, B., & Syah, A. R. (2017). *Klasifikasi Objek Kode Tangan pada Pengenalan Isyarat Alphabet Bahasa Isyarat Indonesia (Bisindo)*. (September), 1–4.
- Brownlee, J. (2016). Master Machine Learning Algorithms. Discover How They Work and Implement Them From Scratch. In *Machine Learning Mastery With Python*.
- Budiharto, W. (2018). AI for Beginner. In *AI for Beginner*.
- Danakusumo, K. P. (2017). Implementasi Deep Learning Menggunakan Convolutional Neural Network untuk Klasifikasi Citra Candi Berbasis GPU.
- Demas, P. (2017). Perancangan Informasi Bahasa Isyarat Bisindo Pada Penyandang Tuna Rungu Melalui Media Interaktif Aplikasi Android. In *Undergraduate Theses* (pp. 1–4). Retrieved from [https://elib.unikom.ac.id/files/disk1/765/jbptunikompp-gdl-demasparda-38223-4-unikom\\_d-1.pdf](https://elib.unikom.ac.id/files/disk1/765/jbptunikompp-gdl-demasparda-38223-4-unikom_d-1.pdf)
- Deng, L., & Yu, D. (2013). Deep Learning: Methods and Applications Foundations and Trends R in Signal Processing. *Signal Processing*. <https://doi.org/10.1561/2000000039>

- Dewa, C. K., Fadhilah, A. L., & Afiahayati, A. (2018). Convolutional Neural Networks for Handwritten Javanese Character Recognition. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*. <https://doi.org/10.22146/ijccs.31144>
- Dzulqarnain, M. F., Suprapto, S., & Makhrus, F. (2019). Improvement of Convolutional Neural Network Accuracy on Salak Classification Based Quality on Digital Image. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*. <https://doi.org/10.22146/ijccs.42036>
- Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2), 303–338. <https://doi.org/10.1007/s11263-009-0275-4>
- Gerkatin, D. (2010). *Berkenalan Dengan BISINDO*. Jakarta: DPD GERKATIN, WQA.
- Gong, S., Liu, C., Ji, Y., Zhong, B., Li, Y., & Dong, H. (2019). Visual object recognition. In *Modeling and Optimization in Science and Technologies*. [https://doi.org/10.1007/978-3-319-77223-3\\_10](https://doi.org/10.1007/978-3-319-77223-3_10)
- Goodfellow, I. (2016). Deeplearning. In *MIT press*. <https://doi.org/10.1016/B978-0-12-801775-3.00001-9>
- Gumelar, G., Hafiar, H., & Subekti, P. (2018). KONSTRUKSI MAKNA BISINDO SEBAGAI BUDAYA TULI BAGI ANGGOTA GERKATIN. *INFORMASI*. <https://doi.org/10.21831/informasi.v48i1.17727>
- Hakim, L., & Samino, D. (2008). *Kamus Sistem Isyarat Bahasa Indonesia* (5th ed.). Jakarta: Direktorat Pembinaan Sekolah Luar Biasa.
- Han, J., & Kamber, M. (2006). Data Mining, Southeast Asia Edition: Concepts and Techniques. *Morgan Kaufmann*.
- Hardjana, A. M. (2003). *Komunikasi Intrapersonal dan Interpersonal*. Yogyakarta: Kanisius.
- Hermawan, A. (2006). *Jaringan Syaraf Tiruan dan Aplikasinya*. Yogyakarta: Andi.
- Hermawati, F. A. (2013). *Data Mining*. Yogyakarta: ANDI.
- Hidayatullah, P. (2017). *Pengolahan Citra Digital: Teori Dan Aplikasi Nyata*. Bandung: Informatika.
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. Retrieved from <http://arxiv.org/abs/1704.04861>
- Indrawan, R., & Yaniawati, P. (2014). *Metodologi Penelitian Kuantitatif, Kualitatif, dan Campuran untuk Manajemen, Pembangunan, dan Pendidikan*. Bandung: Refika Aditama.
- Kementerian Kesehatan. (2014). *Situasi Penyandang Disabilitas di Indonesia*. Retrieved from <http://www.pusdatin.kemkes.go.id/resources/download/pusdatin/buletin/buletin-in-disabilitas.pdf>
- Kiki, S. K. (2003). Analisis JST dg BP untuk Mendeteksi Gangguan Psikolog. In *Islam Zeitschrift Für Geschichte Und Kultur Des Islamischen Orients*.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information*

- Processing Systems.*
- Kusumadewi, S. (2003). Artificial Intelligence. *Artificial Intelligence*.
- Kusumaningrum, T. F. (2018). *Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Jamur Konsumsi Di Indonesia Menggunakan Keras*. Yogyakarta: Universitas Islam Indonesia.
- LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4), 541–551. <https://doi.org/10.1162/neco.1989.1.4.541>
- LeCun, Yann, Kavukcuoglu, K., & Farabet, C. (2010). Convolutional networks and applications in vision. *ISCAS 2010 - 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*. <https://doi.org/10.1109/ISCAS.2010.5537907>
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9905 LNCS, 21–37. [https://doi.org/10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2)
- Martin, L. F. N. P., Frehadthomo, K., & Putri, R. N. D. (2015). *SO-Ice (Sign To Voice) Aplikasi Alat Bantu Untuk Tuna Rungu Wicara*. Bandung: Universitas Telkom.
- Masood, S., Thuwal, H. C., & Srivastava, A. (2018). American sign language character recognition using convolution neural network. *Smart Innovation, Systems and Technologies*. [https://doi.org/10.1007/978-981-10-5547-8\\_42](https://doi.org/10.1007/978-981-10-5547-8_42)
- Maulida, D. K. (2017). *Bahasa Isyarat Indonesia di Komunitas Gerakan untuk Kesejahteraan Tunarungu Indonesia*.
- McKinney, W., & Team, P. D. (2015). Pandas - Powerful Python Data Analysis Toolkit. In *Pandas - Powerful Python Data Analysis Toolkit*.
- Michele, A., Colin, V., & Santika, D. D. (2019). MobileNet Convolutional Neural Networks and Support Vector Machines for Palmprint Recognition. *Procedia Computer Science*, 157, 110–117. <https://doi.org/10.1016/j.procs.2019.08.147>
- Michelluci, U., & Moolayil, J. (2019). *Advanced Applied Deep Learning*.
- Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). Foundation of Machine Learning. In *MIT Press*. [https://doi.org/10.1007/978-3-642-34106-9\\_15](https://doi.org/10.1007/978-3-642-34106-9_15)
- Mukherjea, S., & Foley, J. D. (1996). Requirements and Architecture of an Information Visualization Tool. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. [https://doi.org/10.1007/3-540-62221-7\\_6](https://doi.org/10.1007/3-540-62221-7_6)
- Mulyana, D. (2015). Ilmu Komunikasi: Suatu Pengantar. *Biomass Chem Eng*.
- Mulyana, E. (2012). *App Inventor : Ciptakan Sendiri Aplikasi Androidmu* (1st ed.). Yogyakarta: Andi Publisher.
- Nurhikmat, T., & Purwaningsih, T. (2018). Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma Convolutional Neural Network (CNN) Pada Citra Wayang Golek. *ResearchGate*, (June), 1–113. <https://doi.org/10.13140/RG.2.2.10880.53768>
- Nursiyono, J. A. (2015). *Kompas Teknik Pengambilan Sampel*. Bogor: In Media.

- Padilla, R., Netto, S. L., & Da Silva, E. A. B. (2020). Survey on Performance Metrics for Object-Detection Algorithms. *International Conference on Systems, Signals and Image Processing (IWSSIP)*, 6.
- Putra, S. J., Sugiarti, Y., Dimas, G., Nur Gunawan, M., Sutabri, T., & Suryatno, A. (2019). Document Classification using Naïve Bayes for Indonesian Translation of the Quran. *2019 7th International Conference on Cyber and IT Service Management, CITSM 2019*, 5–8. <https://doi.org/10.1109/CITSM47753.2019.8965390>
- Putra, W. S. E. (2016). Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101. *Jurnal Teknik ITS*. <https://doi.org/10.12962/j23373539.v5i1.15696>
- Putri, R. K. S. C. (2018). Implementasi Deep Learning Menggunakan Metode Convolutional Neural Network Untuk Klasifikasi Gambar. In *Universitas Islam Indonesia Yogyakarta*. Yogyakarta: Universitas Islam Indonesia.
- Raheja, J. L., Mishra, A., & Chaudhary, A. (2016). Indian sign language recognition using SVM. *Pattern Recognition and Image Analysis*, 26(2), 434–441. <https://doi.org/10.1134/S1054661816020164>
- Rahmatullah, H. R., Amrizal, V., & Rozy, N. F. (2018). *Klasifikasi Jenis Golok Betawi dengan Naive Bayes Classifier*.
- Rakhman, J. P., Ramadijanti, N., & Satriyanto, E. (2010). Translasi Bahasa Isyarat. *Jurnal Library ITS*. Retrieved from <http://digilib.its.ac.id/ITS-Undergraduate-3100010039056/9843>
- Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. 1–14. Retrieved from <http://arxiv.org/abs/1609.04747>
- Safaat, N. (2012). Android : Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android (Edisi Revisi). In *Android*.
- Salsabila. (2018). *Penerapan Deep Learning Menggunakan Convolutional Neural Network Untuk Klasifikasi Citra Wayang Punakawan*. Yogyakarta: Universitas Islam Indonesia.
- Shafira, T. (2018). *Implementasi Convolutional Neural Network Untuk Klasifikasi Citra Tomat Menggunakan Keras*. Yogyakarta: Universitas Islam Indonesia.
- Sobur, A. (2006). Analisis Teks Media: Suatu Pengantar untuk Analisis Wacana. In *Analisis Semiotik dan Analisis Framing*. <https://doi.org/10.1177/1524838007302594>
- Sugiyono. (2013). *Metode Penelitian Kuantitatif, Kualitatif dan R&D*. Bandung: Alfabeta.
- Sunarwinadi, I. (1993). *Komunikasi Antar Budaya*. Jakarta: UI Pers.
- Tolba, M. F., & Elons, A. S. (2013). Recent Developments in Sign Language Recognition Systems. *Proceedings - 2013 8th International Conference on Computer Engineering and Systems, ICCES 2013*. <https://doi.org/10.1109/ICCES.2013.6707157>
- Tsoi, A. C., & Pearson, R. A. (1991). Comparison of three classification techniques, CART, C4.5 and Multi-Layer Perceptrons. *Advances in Neural Information Processing Systems*.
- Vasilev, I., Slater, D., Spacagna, G., Roelants, P., & Zocca, V. (2019). *Python Deep Learning: Exploring Deep Learning Techniques and Neural Network*

- Architectures with PyTorch, Keras, and TensorFlow* (2 edition). Packt Publishing.
- Wahyono, T. (2018). *Fundamental Of Python For Machine Learning* (Turi, Ed.). Yogyakarta: Penerbit Gava Media.
- Wasita, A. (2012). *Seluk Beluk Tunarungu dan Tunawicara Serta Strategi Pembelajarannya*. Yogyakarta: Javalutera.
- Wehle, H. (2017). Machine Learning, Deep Learning, and AI: What's the Difference?
- Wolfram, S. (2002). New Kind of Science. Wolfram Media Inc.
- Yunanda, A. B., Mandita, F., & Armin, A. P. (2018). Pengenalan Bahasa Isyarat Indonesia (BISINDO) Untuk Karakter Huruf Dengan Menggunakan Microsoft Kinect. *Fountain of Informatics Journal*. <https://doi.org/10.21111/fij.v3i2.2469>









**LAMPIRAN 1:**  
**NASKAH WAWANCARA**

**Nama Narasumber :** Silva Tenrisara Isma Putri

**Peneliti :** Ferdian Rachardi

**Hari/Tanggal :** Senin, 4 November 2019

**Lokasi Wawancara :** Fakultasi Ilmu Budaya – Universitas Indonesia, Depok

---

Peneliti : *Assalaamu'alaikum*, Sebelumnya terimakasih bu telah berkenan untuk meyempatkan waktunya untuk mewancarai ibu Silva selaku kepala Lembaga Riset Bahasa Isyarat FIB UI. Adapun saya ingin menanyakan beberapa hal mengenai bahasa isyarat terutama yang berkembang di Indonesia.

Bu Silva : *Wa'alaikumussalaam*, baik silakan jika ada yang ingin ditanyakan.

Peneliti : Dari beberapa sumber yang saya dapatkan bahwasanya bahasa isyarat yang umum di Indonesia ada 2 yaitu SIBI dan BISINDO. Apa perbedaan dari kedua bahasa isyarat tersebut ya Bu?

Bu Silva : Iya, terdapat 2 jenis bahasa isyarat yang familiar digunakan oleh teman tuli yaitu Sistem Isyarat Bahasa Indonesia (SIBI) dan Bahasa Isyarat Indonesia (BISINDO). Jika SIBI merupakan suatu bahasa yang tumbuh berasal dari teman dengar dengan membentuk sebuah sistem bahasa terstruktur yang kaku dengan diikuti kata imbuhan sedangkan BISINDO ialah bahasa yang tumbuh langsung berasal dari teman tuli dimana dasar bahasa tersebut diutamakan berbasis visual sesuai dengan objek kata yang digunakan tanpa mementingkan sistem bahasa itu sendiri.

- Peneliti : Lalu, apakah itu berarti kegunaan BISINDO jauh lebih mudah dibandingkan dengan SIBI ?
- Bu Silva : Iya secara fungsional penyampaian lebih mudah dikarenakan gerakan/gestur berdasarkan objek visual yang ingin disampaikan sehingga teman tuli menjadi lebih mudah dipahami dan digunakan dibanding SIBI. Namun, SIBI lebih dahulu diresmikan sebagai standar bahasa yang digunakan pada dunia pendidikan sedangkan teman tuli lebih banyak menggunakan BISINDO sebagai bahasa kesehariannya.
- Peneliti : Lalu, Apakah ada permasalahan yang biasanya dialami oleh teman tuli ketika berinteraksi baik ke orang normal ataupun sebaliknya?
- Bu Silva : Jelas ada, **permasalahan yang sering** dialami ketika orang normal yang belum dapat mengetahui bahasa isyarat yang digunakan oleh teman tuli sehingga menyulitkan interaksi secara langsung dan masih banyaknya teman tuli yang canggung ketika bertemu dengan orang normal untuk berkomunikasi.
- Peneliti : Bagaimana cara komunikasi antara dua pihak tersebut?
- Bu Silva : Biasanya **untuk** dapat berinteraksi hanya memperhatikan gerakan mulut dari orang normal atau menggunakan alat tulis agar teman tuli dapat memahami perkataan orang normal begitu juga sebaliknya komunikasi teman tuli kepada orang normal jika menemui orang awam yang belum mengerti tentang gestur bahasa isyarat.

**Nama Narasumber :** Aldilla

**Peneliti :** Ferdian Rachardi

**Hari/Tanggal :** Sabtu, 7 Maret 2020

**Lokasi Wawancara :** Kopi Tuli – Kalibata, Jakarta Selatan

---

Peneliti : *Assalaamu'alaikum*, sebelumnya terima kasih ka atas kesempatan waktunya untuk bersedia diwawancara oleh saya dalam rangka penelitian skripsi mengenai bahasa isyarat Indonesia (BISINDO). Saya ingin menanyakan dan mengkonfirmasi beberapa hal terkait topik tersebut.

Ka Aldilla : *Wa'alaikumussalaam*, Baik silakan.

Peneliti : Dari sumber yang saya dapati, bahwasanya BISINDO jauh lebih umum digunakan keseharian oleh teman tuli dibanding SIBI. Lalu apa perbedaan yang mendasari kedua jenis bahasa tersebut ya ka?

Ka Aldilla : BISINDO itu dibuat teman tuli dan menjadi bahasa alami isyaratnya yang bisa digunakan untuk teman tuli dan teman dengan berkomunikasi alami juga, serta memiliki ciri khas budaya tuli sesuai dengan identitasnya. Sedangkan, SIBI itu bahasa baku yang biasa dipakai untuk sekolah SLB ditambah penggunaan sistem SPOK berlaku layaknya bahasa formal.

Peneliti : Lalu apakah dalam bahasa BISINDO memiliki kata sambung, hubung dan lainnya ya ka?

Ka Aldilla : Setau ku BISINDO tidak ada kata sambung atau hubung karna penggunaannya langsung berdasarkan objek yang ingin divisualisasikan, berbeda dengan SIBI yang punya kata imbuhan me-an, di-, -kan, -kah ataupun kata sambung seperti ke, di, yang, dan lainnya.

Peneliti : Kemudian, apakah ada standar tertentu yang digunakan pada bahasa isyarat seperti di dalam negeri atau luar negeri?

Ka Aldilla : Belum ada standar tertentu, dikarenakan bahasa isyarat menyesuaikan dengan identitas dari budaya atau lingkungan tertentu sehingga punya beragam makna tergantung daerahnya masing-masing yang kemungkinan bisa sama gestur dan arti kosakata ataupun sebaliknya.

Peneliti : Lalu, saya ingin mengkonfirmasi apakah ini sudah sesuai dengan gestur kosakata BISINDO pada lagu ‘Bidadari Tak Bersayap’ – Anji

Ka Aldilla : (BAB 3)



### Model Bias (8 kelas)

Kelas	Gambar	Kelas	Gambar	Kelas	Gambar	Kelas	Gambar
Datang-1		Katakan-5		Sungguh-9		Tenang-13	
Datang-2		Katakan-6		Sungguh-10		Tenang-14	
Habis-3		Sayang Sekali-7		Tak Bersayap-11		Wanita-15	

Habis-4		Sayang Sekali-8		Tak Bersayap-12		Wanita-16	
---------	---	--------------------	---	-----------------	---	-----------	---



**Model Non Bias (24 kelas)**

Kelas	Gambar	Kelas	Gambar	Kelas	Gambar	Kelas	Gambar
Bersama-1		Hidup-13		Memandangi-25		Teman-37	
Bersama-2		Hidup-14		Memandangi-26		Teman-38	
Bidadari-3		Indah-15		Mencintai-27		Teristimewa-39	

Bidadari-4		Indah-16		Mencintai-28		Teristimewa-40	
Dalam-5		Ini-17		Nyawa-29		Tuhan-41	
Dalam-6		Ini-18		Nyawa-30		Tuhan-42	

Diam-7		Kamu-19		Sampai-31		Umur-43	
Diam-8		Kamu-20		Sampai-32		Umur-44	
Dihati-9		Ku-21		Satu-33		Wajah-45	

Dihati-10		Ku-22		Satu-34		Wajah-46	
Dikirim-11		Mau-23		Sederhana-35		Yes-47	
Dikirim-12		Mau-24		Sederhana-36		Yes-48	



## Surat Dosen Pembimbing Skripsi

**KEMENTERIAN AGAMA  
UNIVERSITAS ISLAM NEGERI (UIN)  
SYARIF HIDAYATULLAH JAKARTA  
FAKULTAS SAINS DAN TEKNOLOGI**

Jl. Ir. H. Juanda No. 95 Ciputat 15412 Indonesia  
Telp. (62-21) 7493806 7493547 Fax. (62-21) 7493315

Email: [fasmt@uinjkt.ac.id](mailto:fasmt@uinjkt.ac.id)  
Website: [fasmt.uinjkt.ac.id](http://fasmt.uinjkt.ac.id)

---

Nomor : B- 6296 / F9 / KM.01 / 02 / 2020  
Lampiran : -  
Perihal : Bimbingan Skripsi

Jakarta, 20 Februari 2020

**Kepada Yth.**  
**1. Dr. Qurrotul Aini, MT**  
**2. Zainul Arham, M. Si**

*Assalamu'alaikum Wr.Wb.*

Dengan ini diharapkan kesediaan Saudara untuk menjadi pembimbing I/II/ (Materi/Teknis)\* penulisan skripsi mahasiswa:

Nama : Ferdian Rachardi
Nim : 11150930000033
Program Studi : Sistem Informasi
Judul Skripsi : "Implementasi Deep Learning untuk Image Clasification Kosakata Bahasa Isyarat Indonesia (BISINDO) dengan Convolutional Neural Network (CNN)"

Judul tersebut telah disetujui oleh Program Studi bersangkutan pada tanggal dengan outline, abstraksi dan daftar pustaka terlampir. Bimbingan skripsi ini diharapkan selesai dalam waktu 6 (enam) bulan setelah ditandatanganinya surat penunjukan pembimbing skripsi.

Apabila terjadi perubahan terkait dengan skripsi tersebut selama proses pembimbingan, harap segera melaporkan kepada Program Studi bersangkutan.

Demikian atas kesediaan Saudara, kami ucapan terima kasih.

*Wassalamu'alaikum Wr.Wb.*



a.n Dekan  
Wadek Bidang Akademik

Nasirul Hakiem, S. Si., M.T., Ph.D  
NIP. 19710608 200501 1 005

Tembusan:  
Dekan (sebagai laporan)

## **Surat Permohonan Wawancara**



Jl. Ir. H. Juanda No. 95 Ciputat 15412 Indonesia  
Telp. (62-21) 7493606 7493547 Fax. (62-21) 7493315

KEMENTERIAN AGAMA  
UNIVERSITAS ISLAM NEGERI (UIN)  
SYARIF HIDAYATULLAH JAKARTA  
FAKULTAS SAINS DAN TEKNOLOGI

Email: [fatigasample@msn.com](mailto:fatigasample@msn.com)  
Website: [fatigasample.com](http://fatigasample.com)

Nomor : B-698E/E.9/TL-00/3/2020

Jakarta, 10 Maret 2020

## Lampiran :

Hal : Permohonan Data/Wawancara

Kepada  
Yth. Pimpinan Kopi Tuli (KOPTUL)  
-- Pilih Instansi --

Tempat

Assalamu'alaikum Wr. Wb.  
Dekan Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta  
mengucapkan selamat

**Menjelaskan Bantuan :**

Nama : Ferdian Rachardi  
Tempat/Tanggal : Kota Jakarta Selatan DKI Jakarta / 28 Februari 1997  
NIM : 1115093000033  
**Semester** : 10  
Program Studi : Sistem Informasi  
Alamat : Jl. Bojong Molek II Blok F/20 Rt.03 Rw.14 Kel. Bojong Rawalumbu Kec. Rawalumbu, Kota Bekasi, Jawa Barat  
Telp/Hp : 085692701554

adalah benar yang bersangkutan mahasiswa Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta yang sedang menyusun skripsi dengan judul:

Implementasi Deep Learning pada Kosakata Bahasa Isyarat Indonesia (BISINDO) dengan Convolutional Neural Network

Untuk melengkapi bahan penulisan skripsi, dimohon kiranya Bapak/Ibu dapat menerima yang bersangkutan untuk wawancara serta memperoleh data guna penulisan skripsi dimaksud.

Atas kerjasama dan bantuannya, kami ucapkan terima kasih.  
*Wassalamu'alaikum Wr Wb*

a.n. Dekan  
Wakil Dekan Bidang Kemahasiswaan

Dr. Fahma Wijayanti, M.Si  
NIP.19690317 200312 2 001

Tembusan  
1 Dekan Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta  
2 Kaprodi Sekprodi Sistem Informasi





