

***DEEP LEARNING* UNTUK DETEKSI WAJAH YANG BERHIJAB
MENGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL*
NETWORK (CNN) DENGAN TENSORFLOW**

SKRIPSI

Diajukan Oleh:

WULAN ANGGRAINI

NIM. 160212019

Mahasiswa Fakultas Tarbiyah dan Keguruan (FTK)

Prodi Pendidikan Teknologi Informasi



**FAKULTAS TARBIYAH DAN KEGURUAN
UNIVERSITAS ISLAM NEGERI AR-RANIRY
DARUSSALAM-BANDA ACEH**

2020 M/1441 H

**DEEP LEARNING UNTUK DETEKSI WAJAH YANG BERHIJAB
MENGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL
NETWORK (CNN) DENGAN TENSORFLOW**

SKRIPSI

Diajukan Kepada Fakultas Tarbiyah dan Keguruan (FTK)
Universitas Islam Negeri Ar-Raniry Darussalam Banda
Aceh Sebagai Beban Studi Untuk Memperoleh Gelar Sarjana
dalam Ilmu Pendidikan Islam

Oleh

WULAN ANGGRAINI

NIM. 160212019

Mahasiswa Fakultas Tarbiyah dan Keguruan
Prodi Pendidikan Teknologi Informasi

Disetujui Oleh:

Pembimbing I,

Bustami, M.Sc

Nip. 198604082014031001

Pembimbing II,

Zuhra Sofyan, M.Sc

Nip. 198403092018011001

**Deep Learning untuk Deteksi Wajah yang Berhijab Menggunakan
Algoritma Convolutional Neural Network (CNN) dengan
Tensorflow**

SKRIPSI


**Telah Diuji Oleh Panitia Ujian Munaqasyah Skripsi
Fakultas Tarbiyah dan Keguruan UIN Ar-Raniry dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Beban Studi Program Sarjana (S-1)
dalam Ilmu Pendidikan Teknologi Informasi**

Pada Har/Tanggal:

Rabu, 08 Juli 2020
17 Dzulkaidah 1441 H

Panitia Ujian Munaqasyah Skripsi

Ketua,



Bustami, M.Sc
NIP. 198604082014031001

Sekretaris,



Nurul Fajri, S.Pd

Penguji I,



Zuhra Sofyan, M.Sc
NIP. 19840309201811001

Penguji II,



Mira Maisura, M.sc
NIP.198605272019032011

Mengetahui,
Dekan Fakultas Tarbiyah dan Keguruan
Darussalam - Banda Aceh



Dr. Muslim Khalil, S.H., M.Ag
NIP. 195903091989031001



LEMBAR PERNYATAAN KEASLIAN KARYA ILMIAH

Yang bertanda tangan dibawah ini, saya:

Nama : Wulan Anggraini

NIM : 160212019

Program Studi : Pendidikan Teknologi Informasi

Fakultas : Tarbiyah dan Keguruan

Judul Skripsi : Deep Learning Untuk Deteksi Wajah Yang Berhijab
Menggunakan Algoritma Convolutional Neural Network
(CNN) Dengan Tensorflow

Dengan ini menyatakan bahwa dalam penulisan skripsi ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggung jawabkan.
2. Tidak melakukan plagiasi terhadap naskah karya orang lain.
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemiliknya.
4. Tidak memanipulasi dan memalsukan data.
5. Mengerjakan sendiri karya ini dan mampu bertanggung jawab atas karya ini.

Bila dikemudian hari ada tuntutan dari pihak lain atas karya saya, dan telah melalui pembuktian yang dapat dipertanggung jawabkan dan ternyata memang ditemukan bukti bahwa saya telah melanggar persyaratan, maka saya siap dikenai sanksi berdasarkan aturan yang berlaku di Fakultas Tarbiyah dan Keguruan UIN Ar-Raniry Banda Aceh.

Demikian surat pernyataan saya buat dengan sesungguhnya tanpa ada paksaan dari pihak manapun.

Banda Aceh, Juli 2020

Yang Menyatakan,



Wulan Anggraini

NIM. 160212019

ABSTRAK

Dalam beberapa tahun terakhir ini teknologi biometrik banyak digunakan dalam berbagai bidang aspek. Salah satu teknologi biometrik yang digunakan adalah sistem pengenalan wajah. Dalam sistem biometrik untuk pengenalan wajah, terdiri dari dua tahapan yaitu deteksi dan klasifikasi. Kedua tahapan ini begitu cepat dilakukan oleh manusia, tetapi membutuhkan waktu yang lama untuk dilakukan oleh komputer. Kemampuan manusia itulah yang ingin diduplikasi ke dalam sistem komputer, agar komputer dapat melakukan pengenalan wajah dengan waktu yang cepat. Pengenalan wajah akan bermasalah ketika wajah yang menjadi data masukan mengalami perubahan pada atribut wajah, ekspresi dan pencahayaan, yang nantinya akan sangat mempengaruhi tingkat keakurasiannya. Dalam penelitian ini penulis akan memasukkan wajah yang berhijab dengan ekspresi yang berbeda. Penelitian ini akan menggunakan *deep learning* dengan metode CNN (*Convolutional Neural Network*). Implementasi CNN menggunakan *Tensorflow* dengan bahasa pemrograman *Python*. Jumlah dataset yang digunakan ada 300 gambar wajah yang berhijab. Berdasarkan hasil dari pembahasan diperoleh tingkat keakurasian sebesar 92% pada proses *training* dan 87% pada proses *testing*. Sehingga dari penelitian ini dapat disimpulkan bahwa kinerja dari model yang telah dibuat pada penelitian ini dapat dikatakan berjalan dengan optimal dalam mendeteksi gambar wajah yang menggunakan atribut yaitu hijab.

Kata Kunci: *Deep Learning*, CNN (*Convolutional Neural Network*), *training*, *testing*.

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Segala puji dan syukur selalu kita panjatkan atas kehadiran Allah SWT yang atas segala rahmat dan hidayah-Nya kita masih dapat melihat Alam semesta yang indah ini. Tak lupa pula shalawat beriring salam selalu kita panjatkan untuk tuntunan suri tauladan Baginda Rasulullah *Shallauhu'alaihiwasalam* dan beserta keluarga dan sahabat beliau yang senantiasa menjunjung tinggi nilai-nilai keislaman serta menggali ilmu yang tiada habisnya yang sampai saat ini masih bisa dinikmati oleh setiap manusia, sehingga penulis dapat menyelesaikan skripsi yang berjudul **“Deep Learning untuk Deteksi Wajah yang Berhijab Menggunakan Algoritma Convolutional Neural Network (CNN) dengan Tensorflow”**.

Penulisan skripsi ini adalah salah satu syarat untuk mendapatkan gelar sarjana pada Fakultas Tarbiyah dan Keguruan di UIN Ar-Raniry, Banda Aceh. Dalam penyusunan skripsi ini, penulis banyak sekali menghadapi kesulitan dalam teknik penulisan maupun dalam penguasaan bahan. Walaupun demikian, penulis tidak putus asa dalam menghadapi permasalahan, dan dengan adanya dukungan dari berbagai pihak, terutama sekali dosen pembimbing kesulitan yang dihadapi dapat teratasi. Pada kesempatan ini, penulis mengucapkan ribuan terima kasih kepada:

1. Kepada kedua orang tua yang senantiasa memberikan dukungan moril maupun materil serta doa yang tiada hentinya kepada penulis.
2. Segenap keluarga dan sahabat yang selalu menyemangati dan membantu penulis untu menyelesaikan skripsi ini dari awal hingga akhir.

3. Bapak Rektor UIN Ar-Raniry, Prof. Dr. H. Warul Walidin AK. MA yang selalu mendukung dan memberi motivasi untuk kami.
4. Bapak Bustami, M.Sc selaku pembimbing pertama dan Bapak Zuhra Sofyan, M.Sc selaku pembimbing kedua, yang telah meluangkan waktunya dan mencurahkan pemikirannya dalam membimbing penulis untuk menyelesaikan skripsi ini.
5. Ketua Prodi Pendidikan Teknologi Informasi (PTI) Bapak Yusran, M.Pd, Sekretaris Prodi Pendidikan Teknologi Informasi Bapak Hazrullah, M.Pd, serta staf Prodi yang telah ikut membantu proses pelaksanaan penelitian.
6. Bapak/Ibu dosen pengajar Program Studi Pendidikan Teknologi Informasi yang telah membekali penulis dengan berbagai ilmu pengetahuan sehingga dapat menyelesaikan studi ini.
7. Sahabat dan teman-teman mahasiswa Jurusan Pendidikan Teknologi Informasi leting 2016, serta seluruh keluarga PTI yang telah memberikan dukungan dalam penelitian ini.
8. Teman-teman PPKPM Blang Kolak I yang telah berjuang bersama dan saling memberi dukungan dalam menyelesaikan penelitian ini.
9. Dan untuk semuanya yang tidak dapat penulis sebutkan satu persatu.

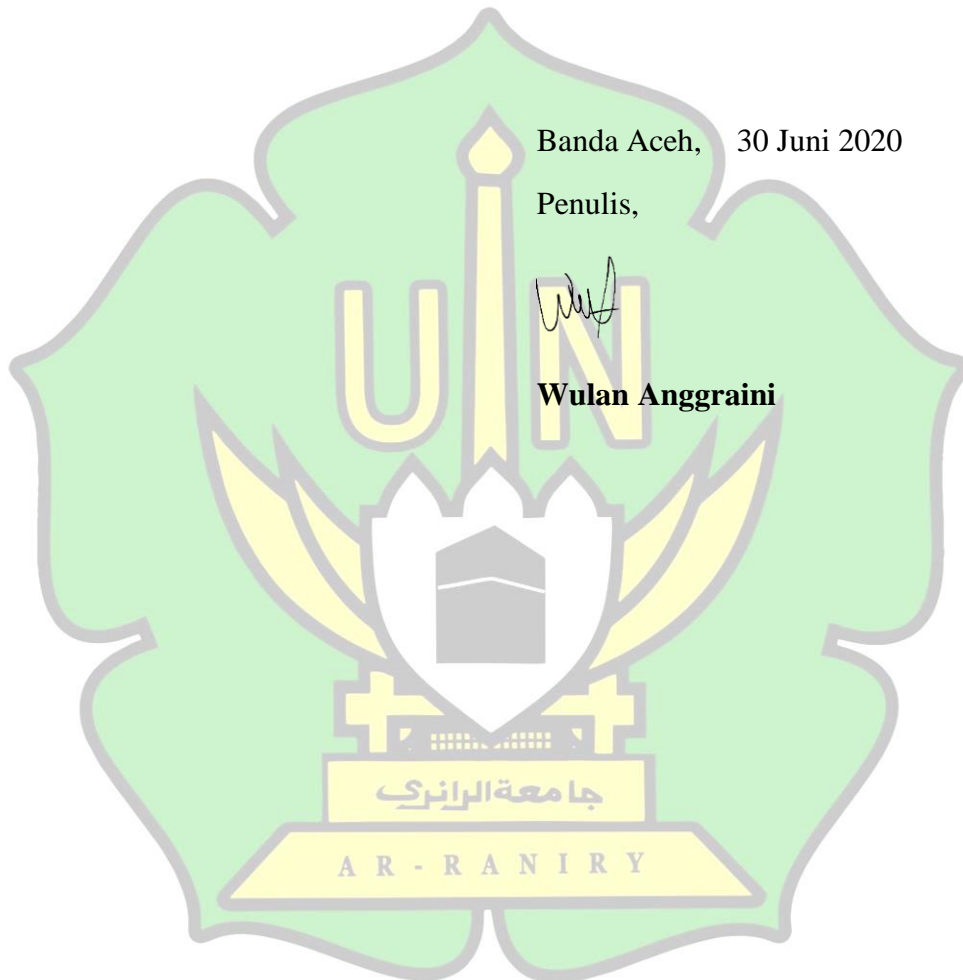
Namun penulis menyadari, dalam penulisan skripsi ini masih banyak kekurangan dan kekhilafan. Oleh karena itu, penulis mengharapkan saran dan kritikan yang membangun guna untuk memperbaiki kesalahan dimasa yang akan datang. Semoga Allah SWT meridhai penulisan ini dan senantiasa memberikan rahmat dan hidayah-Nya kepada kita semua. *Amin ya rabbal' alamin.*

Banda Aceh, 30 Juni 2020

Penulis,



Wulan Anggraini



DAFTAR ISI

LEMBARAN JUDUL	
PENGESAHAN SKRIPSI	
PENGESAHAN SIDANG	
ABSTRAK	v
KATA PENGANTAR.....	vi
DAFTAR GAMBAR.....	xii
DAFTAR TABEL	xiii
DAFTAR GRAFIK.....	xiv
BAB I PENDAHULUAN.....	1
A. Latar Belakang	1
B. Rumusan Masalah.....	3
C. Tujuan Masalah.....	3
D. Batasan Masalah	4
E. Manfaat	4
BAB II LANDASAN TEORI.....	5
A. Penelitian Terdahulu	5
B. Hijab.....	6
C. Wajah.....	7
D. Deteksi Wajah.....	8
E. <i>Biometrical/Biometrik</i>	8
F. Pengolahan Citra.....	9
1. Definisi Citra	9
2. Definisi Citra Digital.....	9
G. <i>Computer Vision</i>	10
H. <i>Artificial Intelligence</i>	10
I. <i>Machine Learning</i>	11
J. <i>Deep Learning</i>	12
K. <i>Convolutional Neural Network (CNN)</i>	13
L. <i>Python</i>	14
M. <i>Tensorflow</i>	14

N. <i>Evaluation Measurement</i> (Pengukuran Evaluasi).....	15
1. <i>Recall</i> dan <i>Precision</i>	15
2. <i>Accuracy</i>	16
BAB III METODOLOGI PENELITIAN	16
A. Tahapan Penelitian.....	16
B. Pengumpulan Data.....	18
C. Metode Analisis Data.....	19
BAB IV PEMBAHASAN DAN HASIL	20
A. <i>Persiapan Software</i>	20
1. <i>Anaconda & Tensorflow</i>	20
B. <i>Preprocessing Image</i>	22
1. Pelabelan Gambar.....	22
2. <i>Konversi Dataset</i>	23
3. <i>Konversi Dataset CSV ke TFRecord</i>	26
C. Pengolahan Image didalam Tensorflow.....	27
1. <i>Konfigurasi Object Detection Pipeline</i>	27
D. <i>Pemodelan Jaringan Convolutional Neural Network (CNN)</i>	28
E. <i>Proses Konvolusi</i>	31
F. <i>Proses Pooling</i>	33
G. <i>Proses Fully Connected</i>	34
H. <i>Proses Classification</i>	34
I. <i>Detection Output</i>	35
J. Hasil.....	35
1. Hasil Model <i>Training</i>	35
2. Hasil Data <i>Test</i>	36
K. <i>Penentuan Parameter Model</i>	38
1. Pengaruh Jumlah Nilai <i>Epoch</i>	39
2. Pengaruh Jumlah Data <i>Training</i>	40
3. Pengaruh Jumlah Nilai <i>Learning Rate</i>	42

BAB V KESIMPULAN	44
A. Kesimpulan	44
B. Saran	45
DAFTAR PUSTAKA.....	46
LAMPIRAN.....	49

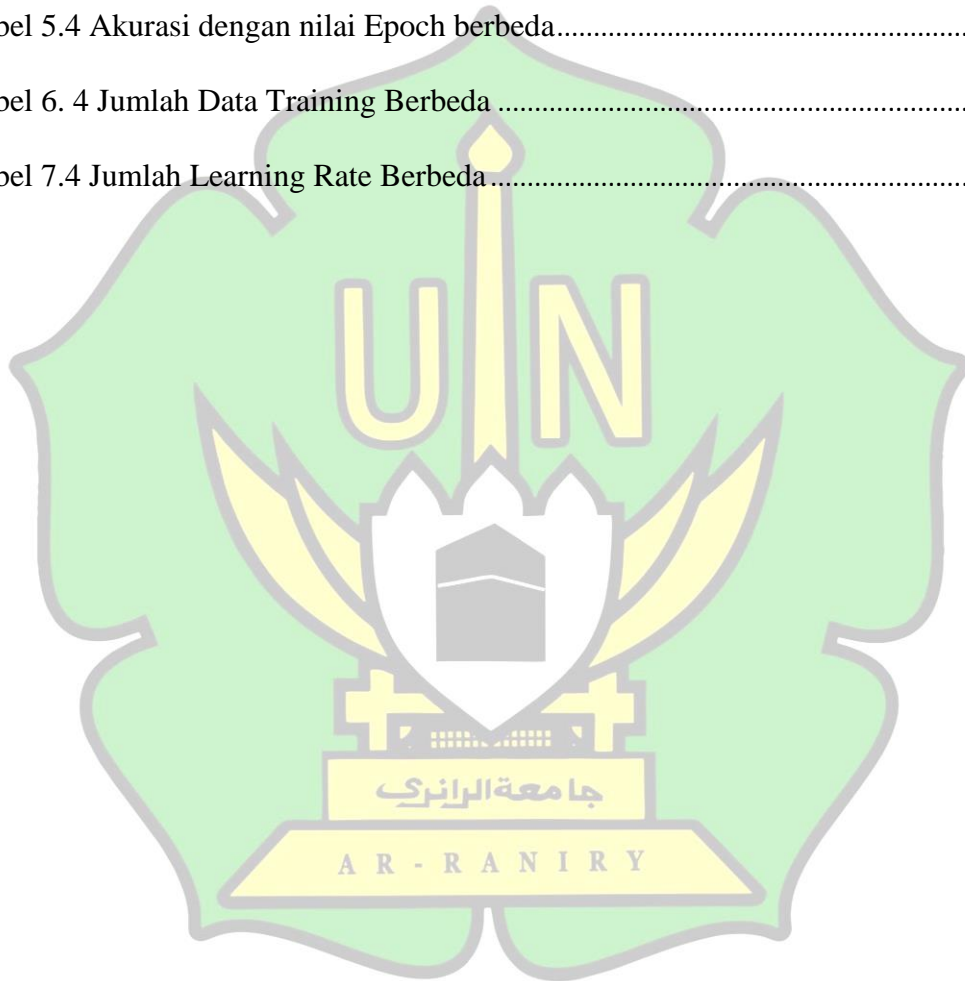


DAFTAR GAMBAR

Gambar 1.2 Konsep Pengerjaan Deep Learning (Sumber: John D. Woodward, Jr., Christopher Horn, Julius Gatune, and Aryn Thomas: “Biometrics A Look at Facial Recognition”).....	12
Gambar 2.2 Tahapan Algoritma Convolutional Neural Network (CNN) (sumber: Jurnal Aditya Santoso “Implementasi Deep Learning Berbasis Keras Untuk Pengenalan Wajah”. Universitas Muhamadiyah, Surakarta.).....	13
Gambar 3.3 Tahapan-tahapan penelitian.....	16
Gambar 4.3 (a) Sebelum diresize (b) Sesudah diresize.....	17
Gambar 5.4 Sebelum Pelabelan image.....	22
Gambar 6.4 Sesudah Pelabelan image.....	23
Gambar 7.4 Data dalam bentuk XML.....	24
Gambar 8.4 Code konversi XML ke CSV.....	25
Gambar 9.4 Code konversi CSV ke Tfrecored.....	26
Gambar 10.4 Model Jaringan.....	28
Gambar 11.4 Proses Konvolusi.....	31
Gambar 12.4 Posisi karnel dalam konvolusi.....	32
Gambar 13.4 Proses Pooling.....	34

DAFTAR TABEL

Tabel 1.2 Confusion matrix dari suatu klasifikasi	15
Tabel 2.4 Model Convolutional Neural Network (CNN)	30
Tabel 3.4 Hasil Loss dan Accuracy	35
Tabel 4.4 confusion matrix, precision dan recall	37
Tabel 5.4 Akurasi dengan nilai Epoch berbeda.....	39
Tabel 6. 4 Jumlah Data Training Berbeda	41
Tabel 7.4 Jumlah Learning Rate Berbeda.....	42



DAFTAR GRAFIK

Grafik 1.4 Hasil Loss dan Accuracy	36
Grafik 2.4 Akurasi dengan nilai Epoch berbeda	40
Grafik 3.4 Jumlah Data Training Berbeda	41
Grafik 4.4 Jumlah Learning Rate Berbeda.....	42



BAB I PENDAHULUAN

A. Latar Belakang

Beberapa tahun terakhir ini teknologi *Biometrik* sudah banyak digunakan dalam segala aspek bidang. Teknologi *Biometrik* (pengenalan wajah, pemindaian iris, retina, pengenalan bunyi/suara, sidik jari, verifikasi penekanan tombol keyboard, gaya berjalan dan lain-lain) walaupun kelihatannya masih belum dikenal secara luas, namun teknologi ini diprediksi akan berkembang dengan pesat di masa yang akan mendatang. Seperti yang disebutkan dalam MIT Technology Review “*Top ten emerging technologies that will change the world*” [1]. Dalam sistem *Biometrik* untuk pengenalan wajah terdiri dari dua tahapan yaitu mendeteksi dan melakukan klasifikasi. Kedua tahapan tersebut dilakukan oleh manusia dengan sangat cepat, tetapi membutuhkan waktu yang cukup lama untuk dilakukan oleh komputer. Kemampuan manusia itulah yang ingin diduplikasi kedalam sistem komputer oleh para peneliti dalam bidang teknologi biometrik, agar komputer dapat melakukan pengenalan wajah dengan waktu yang cepat.

Prinsip-prinsip dalam sistem pengenalan wajah yaitu dengan membandingkan satu gambar wajah dengan database wajah, sehingga menghasilkan pendekatan dan kecocokan gambar wajah. Melihat perkembangannya, masih ada beberapa masalah dalam proses pengenalan wajah. Kondisi gambar wajah yang dijadikan *inputan* kedalam sistem nantinya akan mengalami beberapa permasalahan yang penting ketika mempengaruhi tingkat keakuratan sistem dalam menganalisa wajah, contohnya seperti pencahayaan, ekspresi wajah, dan perubahan atribut wajah (janggut, kumis, kacamata dan

khimar/hijab)[2]. Penelitian ini akan fokus terhadap pendeteksian gambar wajah yang mengenakan khimar/hijab. Khimar/hijab merupakan pakaian tertutup yang biasanya dipakai oleh para wanita muslim. Penggunaan jilbab bagi perempuan muslim merupakan bentuk dari ketaatan mereka pada agamanya, sebagai wanita yang telah balig mereka wajib mengenakan jilbab/hijab[3]. Hijab dikenakan pada area yang berbatasan dengan wajah, dimana untuk melakukan pendeteksian pada wajah yang menggunakan hijab butuh sebuah sistem *machine learning* yang dapat meningkatkan tingkat keakuratan dalam melakukan pendeteksian wajah. Sehingga untuk mengatasi masalah tersebut digunakan sebuah *machine learning* dengan menerapkan model kerja *deep learning*.

Dalam kurun waktu 10 tahun terakhir *deep learning* sedang menjadi topik yang diteliti dalam pengembangan *machine learning*. Alasannya karena penggunaan *deep learning* telah mencapai tingkat keberhasilan yang luar biasa dalam mencapai tujuan komputer[4]. *Deep learning* merupakan *machine learning* yang pembuatannya terinspirasi oleh korteks manusia dengan menerapkan jaringan saraf seperti manusia yang nantinya akan bekerja pada *hidden layer*. *Convolutional Neural Network* (CNN) yaitu salah satu metode yang terdapat dalam *deep learning* yang dirancang untuk menutupi kesalahan dan kelemahan dari metode sebelumnya. Terdapat beberapa kelemahan dalam metode sebelumnya, tetapi dengan menggunakan model ini sejumlah parameter bebas nantinya dapat dikurangi dan perubahan bentuk gambar input seperti, translasi, rotasi, dan skala dapat ditangani[5].

Perkembangan dalam bidang *deep learning* pada saat ini dapat dilakukan dengan mudah karena telah banyaknya *library* dan *Application Program Interface* (API) yang telah tersedia. *Library* yang nantinya akan digunakan dalam implementasi ini adalah Tensorflow dengan menggunakan bahasa pemrograman *python*.

Berdasarkan uraian di atas, maka penelitian ini akan berfokus pada penerapan algoritma *Convolutional Neural Network* (CNN) dalam pendeteksian wajah yang berhijab dengan memanfaatkan *framework* Tensorflow.

B. Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dapat dibuat yang terkait dengan penelitian ini, antara lain:

1. Bagaimana mengimplementasikan *deep learning* dengan menggunakan metode CNN pada *framework* Tensorflow?
2. Bagaimana hasil analisis pengenalan wajah dengan metode CNN pada *framework* Tensorflow untuk mengetahui tingkat keakuratan yang dihasilkan?

C. Tujuan Masalah

Berdasarkan latar belakang dan rumusan masalah di atas maka tujuan dari penelitian ini, antara lain:

1. Mengetahui bagaimana cara mengimplementasikan *deep learning* dengan menggunakan metode CNN pada *framework* Tensorflow.
2. Mengetahui hasil analisis pengenalan wajah dengan metode CNN pada *framework* Tensorflow untuk mengetahui tingkat keakuratan yang dihasilkan.

D. Batasan Masalah

Batasan-batasan dalam penelitian ini yang diambil oleh peneliti adalah:

1. Bahasa pemrograman yang digunakan adalah *Python* dengan menggunakan *framework* Tensorflow.
2. Data yang digunakan dalam penelitian ini adalah data gambar wajah yang mengenakan hijab.
3. Metode yang digunakan yaitu metode CNN.
4. Dataset yang digunakan berjumlah 300 gambar wajah dimana setiap orang memiliki foto gambar wajah dengan dua ekspresi yaitu, flat/datar dan tersenyum.

E. Manfaat

Manfaat dari penelitian ini, antara lain:

1. Dapat memudahkan suatu pihak untuk menerapkan sistem pendeteksian wajah dengan menggunakan bantuan perangkat lain yang mana sistem dapat bekerja terhadap pendeteksian pada wajah dengan menerapkan metode CNN yang telah teruji melalui penelitian ini.
2. Hasil penelitian ini dapat dijadikan sebagai acuan untuk penelitian lebih lanjut terhadap pendeteksian pada objek-objek lain dengan menggunakan metode CNN dan *deep learning* yang berbasis Tensorflow.
3. Menjadi pelopor penelitian pertama di UIN Ar-Raniry, Banda Aceh pada bidang *computer vision*.

BAB II LANDASAN TEORI

A. Penelitian Terdahulu

Terkait dengan penelitian yang dilakukan pada *deep learning*, referensi dari penelitian sebelumnya sangat penting dilakukan untuk menghindari duplikasi atau plagiarisme. Hal tersebut bertujuan agar penelitian dengan tema yang sama akan semakin berkembang dengan kontribusi baru yang dikerjakan oleh penulis. Berikut beberapa ulasan dari penelitian terdahulu yang berkaitan dengan penelitian yang dilakukan.

Penelitian mengenai “Implementasi *Deep Learning* Berbasis Tensorflow untuk Pengenalan Sidik Jari” yang dilakukan oleh Royani Darma Nurfiti[6]. Dataset yang digunakan adalah gambar sidik jari dengan menggunakan metode CNN untuk klasifikasi dan mendeteksi. Dalam penelitian ini ukuran datasetnya 24x24 pixel dengan jumlah gambar sidik jari sebanyak 80 gambar. Tingkat akurasi *training* yang dihasilkan dari penelitian ini yaitu sebesar 100%, yang mana penelitian ini telah berhasil melakukan peningkatan keakurasian pelatihan yang sangat baik.

Penelitian mengenai “*Convolutional Neural Network* (CNN) untuk Pengenalan Wajah secara Real-Time” yang dilakukan oleh Muhammad Zufar dan Budi Setiyono[7]. Penelitian ini menggunakan *framework* Open CV dengan metode CNN. Dataset yang digunakan adalah data gambar yang wajah yang dilakukan secara real-time. Penelitian ini menghasilkan sebuah model CNN dengan kedalaman 7 layer. *Input layer*, *convolutional layer* C1, *pooling layer* P2, *convolutional layer* C3, *pooling layer* P4, *hidden layer* H dan *output layer* F adalah *layer* pembangun

jaringan pada model konvolusi. ketujuh *layer* yang dihasilkan telah berhasil mengklasifikasikan dataset gambar wajah dengan tingkat akurasi lebih dari 87%.

Penelitian mengenai “*Object Recognition with Deep Learning Applied to Fashion Items Detection in Images*” yang dilakukan oleh Helder Filipe de Sousa Russa[8]. Dalam melakukan klasifikasi dan deteksi *item fashion* yang digunakan oleh sebuah gambar maka dalam penelitian ini menggunakan metode Fast R-CNN. Penelitian ini menggunakan data *train* sebanyak 3677 dan data *testing* sebanyak 696. Hasil penelitian ini menunjukkan bahwa penggunaan metode Fast R-CNN dalam mengklasifikasikan dan mendeteksi fashion item yang digunakan oleh seseorang menghasilkan jumlah rata-rata *precision of close* sebesar 78%, untuk *pants* sebesar 65% dan untuk rata-rata aksesoris seperti *glasses* sebesar 57%. Metode Fast R-CNN lebih efisien digunakan untuk lebih memhemat waktu dalam melakukan pelatihan objek.

Berdasarkan penelitian terdahulu yang telah disebutkan, diketahui bahwa belum ada penelitian yang mengenai pendeteksian gambar wajah yang menggunakan hijab dengan menggunakan metode *Convolutional Neural Network* (CNN) yang dibantu oleh Tensorflow. Oleh karena itu, pada penelitian ini akan melakukan pendeteksian pada gambar wajah yang menggunakan hijab dengan metode *Convolutional Neural Network* (CNN) pada Tensorflow.

B. Hijab

Kain yang dikenakan oleh para wanita muslimah yang dijadikan sebagai pakaian tertutup dinamakan hijab/khimar. Pada dasarnya pengertian hijab adalah “penutup”, seperti tirai ataupun kain yang berfungsi untuk menutupi aurat sehingga

terhalang dari pandangan orang lain. Hajaba حجاب yaitu kata hijab yang berasal dari bahasa Arab yang berarti penghalang atau penutup[9]. Penggunaan hijab bagi wanita muslim ada beberapa kriteria, yaitu:

1. Menutup seluruh tubuh wanita, kecuali wajah dan tangan. (HR. Ibnu Jarik at-Thabari).
2. Menjulurkan kain kerudung hingga menutupi dadanya. (QS An Nuur :31, Diponegor).
3. Hijab adalah penutup aurat, bukan seperti perhiasan untuk menarik perhatian orang lain. (Shahab).

C. Wajah

Wajah merupakan bagian dari tubuh manusia yang memiliki peranan sangat penting, baik itu untuk penampilan, ekspresi wajah maupun untuk identitas diri. Tidak ada satu wajahpun seperti halnya kembar identik sekalipun tidak akan memiliki bentuk dan serupa mutlak dengan wajah yang satu dengan yang lainnya. Dibidang keilmuan ada beberapa pengertian dari wajah, yaitu: wajah bagian dari kepala; roman muka; muka, pengertian tersebut sesuai dengan Kamus Besar Bahasa Indonesia (KBBI). Wajah adalah bagian depan dari kepala pada manusia yang meliputi wilayah dahi hingga dagu. Bagian-bagian yang termasuk daerah pada bagian wajah yaitu, dahi, alis, mata, hidung, mulut, pipi, bibir, gigi, kulit, rambut dan dagu[10].

Wajah adalah salah satu bagian tubuh dari manusia yang dapat di kenali dengan sistem *biometrik*. Pada sistem *biometrik* ciri-ciri biologis pada manusia dapat memberikan informasi yang unik terutama pada sisi wajah. Ciri-ciri unik

tersebut dapat berupa karakteristik dari pola wajah pada setiap individu. Karena karakteristik dari pola wajah dapat diukur dan dianalisis untuk proses deteksi atau autentifikasi. Oleh sebab itu itu wajah di gunakan untuk dijadikan indikasi pengenalan seseorang.

D. Deteksi Wajah

Face detection atau pendeteksian wajah merupakan sebuah tahapan awal dilakukan sebelum proses pengenalan wajah (*face recognition*). Bidang-bidang penelitian yang juga berkaitan dengan pemrosesan wajah (*face processing*) adalah autentikasi wajah (*face uthentication*), lokasi wajah (*face localization*), penjajakan wajah (*face tracking*), dan pengenalan ekspresi wajah (*facial expression recognition*)[11].

E. Biometrical/Biometrik

Menurut Dr. Ir. Eko Nugroho, Msi[12], biometrik adalah studi untuk mengenali seseorang secara unik. Didukung faktor harga yang semakin terjangkau dan bisa diterapkan pada banyak sektor, maka teknologi ini akan menggantikan pemakaian kata sandi (*password*) ataupun kartu (misal *credit card*) sebagai alat autentikasi maupun identifikasi. Kemajuan pesat dalam jaringan komunikasi maupun mobilitas alat memang membutuhkan metode yang handal untuk mengidentifikasi seseorang. Cara yang dikembangkan ialah dengan menggunakan *biometrik*, yaitu suatu keadaan fisik tertentu ataupun suatu perilaku unik yang ada pada seseorang. Cara ini juga disebut cara “*what you are*”. Kelebihan *biometrik* ini adalah:

1. Tidak akan hilang (fisik) atau lupa terkecuali karena mengalami trauma.

2. Sangat sulit untuk ditiru walaupun diberikan kepada orang lain.

Biometrik mengharuskan orang yang bersangkutan ada ditempat di mana dilakukannya proses identifikasi[12].

F. Pengolahan Citra

1. Definisi Citra

Citra (*image*) merupakan sebuah bidang dua dimensi pada sebuah gambar yang terdiri dari banyak piksel, yang mana piksel tersebut merupakan bagian terkecil dalam citra. Citra dibentuk dan disusun dari kotak persegi empat yang teratur, sehingga jarak horizontal dan vertikal antara piksel sama. Citra dapat dikelompokkan menjadi dua bagian yaitu citra diam (*still image*) dan citra bergerak (*moving image*). Citra diam yang ditampilkan berurut (*sekuensial*), sehingga memberikan kesan pada mata sebagai gambar yang bergerak. Setiap citra didalam rangkaian tersebut disebut *frame*. Gambar-gambar yang terlihat pada film layar lebar atau televisi pada dasarnya terdiri dari ribuan *frame*[13].

2. Definisi Citra Digital

Elemen piksel merupakan sebuah matriks indeks baris dan kolom yang menyatakan suatu titik pada citra dan elemennya. Tahapan perubahan citra analog menjadi citra digital sering disebut dengan proses digitasi. Digitasi adalah sebuah proses mengubah gambar, teks, dan suara dari benda yang dapat dilihat ke dalam data elektronik dan dapat disimpan serta diproses untuk keperluan lainnya. Citra digital pada komputer disusun dalam bentuk *grid* atau elemen piksel-piksel yang berbentuk matriks 2 dimensi. Untuk mepresentasikan warna-warna dari setiap citra tersebut maka elemen piksel tersebut mempunyai nilainya masing-masing, yang

mana nilai dalam angka pada setiap piksel akan di simpan secara berurutan oleh komputer[13].

G. Computer Vision

Bidang ilmu pengetahuan yang mempelajari dalam pembuatan sistem untuk mengambil keputusan dalam mengenali objek fisik nyata dan sesuai keadaan berdasarkan sebuah citra atau gambar adalah *Computer vision*[14]. *Computer vision* menjadikan sebuah komputer “*act like human sight*”, sehingga komputer mempunyai kemampuan mendekati manusia dalam mendapatkan informasi secara visual. Kelebihan yang dimiliki dalam *computer vision* antara lain adalah:

1. *3D Interface*: Menerjemahkan aksi-aksi 2D dari 3D yang di lihat.
2. *Interpreting*: Menerjemahkan gerakan-gerakan.
3. *Recognition*: Menempatkan label pada objek.
4. *Object Detection*: Mengenali suatu benda yang terdapat pada scane untuk melihat batasannya.
5. *Description*: Menugaskan properti pada objek.

H. Artificial Intelligence

Artificial Intelligence (kecerdasan buatan) adalah ilmu dan teknik untuk pembuatan mesin cerdas. Hal ini berkaitan dengan tugas yang sama dalam penggunaan komputer untuk memahami kecerdasan manusia, akan tetapi Ai tidak harus membatasi dirinya terhadap metode ataupun expository writing yang diamati secara biologis[15]. Menurut pengertian Dobrev[15] yang mengatakan bahwa Ai adalah sebagai pembelajaran bagaimana membuat sebuah komputer dapat melakukan hal-hal yang dimana saat ini masih lebih baik dilakukan oleh manusia.

I. *Machine Learning*

Machine Learning adalah ilmu dari *Artificial Intelligence* yang akan membuat komputer dapat memiliki kemampuan untuk belajar tanpa perlu diprogram lagi[16]. *Machine learning* menggunakan sebuah algoritma yang akan membuat komputer untuk belajar dan melakukan tugasnya tanpa harus adanya intruksi dari pengguna. Algoritma ini bekerja dengan cara membangun sebuah model dari masukan agar dapat menghasilkan suatu prediksi atau pengambilan keputusan berdasarkan data yang ada.

Dalam pengembangan *machine learning* ada tiga hal kategori utama didalamnya, yaitu:

1. *Reinforcement Learning*

Reinforcement learning merupakan pembelajaran terhadap aksi yang dilakukan untuk memperoleh *reward* yang maksimal. Dalam proses pembelajarannya tidak akan diberitahu aksi mana yang diambil, tetapi lebih pada menemukan aksi yang dapat memberikan *reward* maksimal dengan cara menjalankan aksi-aksi tersebut.

2. *Supervised Learning*

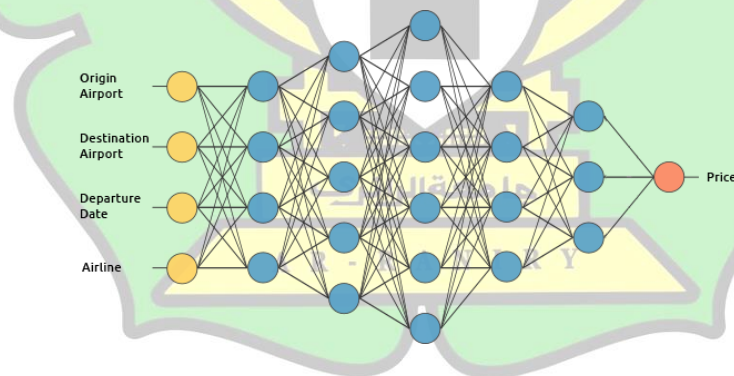
Supervised learning merupakan data yang ada pada *machine learning* dilengkapi dengan kelas/label yang menunjukkan klasifikasi dari data tersebut. Dalam kategori ini model yang dihasilkan adalah model prediksi dari data yang telah diberi kelas/label.

3. *Unsupervised Learning*

Unsupervised learning ini merupakan kebalikan dari *supervised learning*, dimana data yang dimiliki tidak dilengkapi dengan kelas/label.

J. Deep Learning

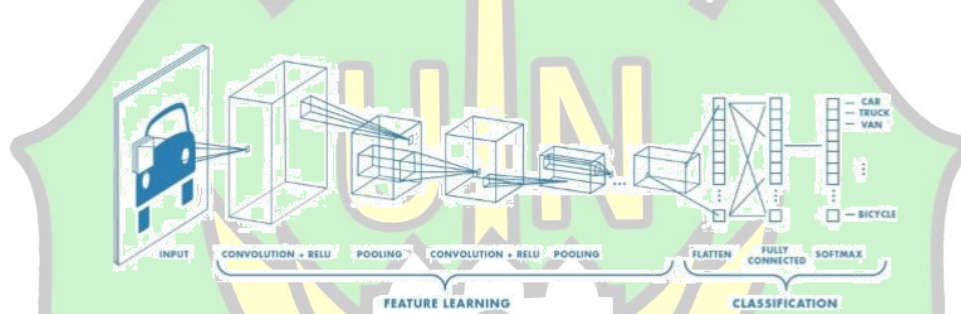
Deep Learning merupakan salah satu bidang *artificial intelligence* yang memanfaatkan banyak *layer* pengolahan informasi yang bekerja nonlinier untuk dapat melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi[17]. Menurut Goodfellow[16], *deep learning* merupakan sebuah pendekatan dalam melakukan penyelesaian masalah pada sistem pembelajaran komputer dengan memakai konsep hierarki. Konsep hierarki ini mampu mempelajari konsep yang kompleks iranian penggabungan konsep-konsep yang lebih sederhana. Jika digambarkan dalam sebuah graf bagaimana konsep tersebut dibangun di atas konsep yang lain, maka graf ini akan dalam dengan banyak *layer* hal ini yang menjadi alasan mengapa disebut sebagai *Deep Learning* (pembelajaran mendalam).



Gambar 1.2 Konsep Pengerjaan *Deep Learning* (Sumber: John D. Woodward, Jr., Christopher Horn, Julius Gatune, and Aryn Thoma: “*Biometrics A Look at Facial Recognition*”)

K. Convolutional Neural Network (CNN)

CNN (*Convolutional Neural Network*) adalah tipe *iranian nervous net* untuk memproses *facts* yang mempunyai topologi jalan (*grid-like topology*). Penggunaan kata CNN mengidentifikasi bahwa dalam jaringan tersebut menggunakan pengoperasian matematika yang dikenal dengan konvolusi. Konvolusi merupakan sebuah operasi one-dimensional Jadi CNN merupakan bagian utama ANN (*Artificial Neural Ntwork*) yang saat ini dinyatakan sebagai modeling terbaik dalam memecahkan masalah *aim identification* dan *detection*[18].



Gambar 2.2 Tahapan Algoritma Convolutional Neural Network (CNN) (sumber: Jurnal Aditya Santoso “Implementasi Deep Learning Berbasis Keras Untuk Pengenalan Wajah”. Universitas Muhamadiyah, Surakarta.)

Algoritma *Convolutional Neural Network* (CNN) termasuk kedalam jenis *deep neural network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra. Secara teknis, *convolutional network* merupakan arsitektur yang bisa *training* dari beberapa tahap. Masukan dan keluaran dari masing-masing tahapan adalah *array* yang disebut *feature map*. Keluaran dari masing-masing tahapan adalah *feature map* hasil pengolahan dari semua lokasi pada masukan.

L. Python

Python merupakan sebuah bahasa pemrograman komputer yang dikembangkan khusus untuk membuat *source code* mudah dibaca. Bahasa pemrograman *python* memiliki *library* yang lengkap sehingga memudahkan seorang *programmer* untuk membuat sebuah aplikasi sesuai dengan keinginan dengan menggunakan *source code* yang terlihat sederhana[19].

M. Tensorflow

Tensorflow adalah *library* perangkat lunak yang dikembangkan oleh Tim Google Brain Mesin Cerdas Google Asosiasi, yang bertujuan untuk melakukan pembelajaran mesin dan jaringan syaraf dalam penelitiannya. *Tensorflow* menggabungkan aljabar komputasi dengan teknik optimasi kompilasi, yang memfasilitasi perhitungan banyak ekspresi matematika[20]. Fitur utama yang terdapat dalam *tensorflow* adalah:

1. Mendefinisikan, mengoptimalkan, dan menghitung secara matematis ekspresi wajah yang melibatkan *array multidimension* (tensors).
2. Pemrograman pendukung jaringan syaraf dalam dan teknik *machine learning*.
3. Pemakaian GPU (*Graphics Processing Unit*) yang efisien, mengotomasi manajemen dan optimalisasi memori yang sama terhadap data yang digunakan. *Tensorflow* mampu menulis kode yang sama dan menjalankannya di CPU atau GPU. Lebih khususnya lagi *tensorflow* dapat mengetahui bagian mana yang harus dipindahkan ke GPU.

4. Skalabilitas komputasi yang tinggi pada keseluruhan mesin terhadap kumpulan data yang besar.

N. *Evaluation Measurement* (Pengukuran Evaluasi)

Evaluation Measurement (pengukuran evaluasi) merupakan sebuah pengujian yang diusulkan, yang kemudian akan ditemukannya perbandingan antara metode yang digunakan. Tahapan evaluasi adalah tahapan terakhir yang dilakukan dalam penelitian ini, dimana hasil atau kesimpulan dapat diketahui. Metode evaluasi yang digunakan dalam penelitian ini adalah *recall*, *precision*, dan *accuracy*[21].

1. *Recall* dan *Precision*

Recall dan *precision* merupakan sebuah rasio prediksi yang dilakukan terhadap kelas positif, yang berfungsi sebagai pengukur seberapa tepat dan lengkap klasifikasi yang telah dilakukan. Berikut ini adalah acuan *confusion matrix* yang memperlihatkan prediksi dan kondisi sebenarnya dari data yang dihasilkan oleh algoritma yang digunakan.

Tabel 1.2 *Confusion matrix* dari suatu klasifikasi

Predicted Values	Actual Values	
	Postive (1)	Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

Keterangan:

TP (*True Positive*) : jumlah prediksi yang benar dalam kelas positif

FP (*False Positive*) : jumlah prediksi yang salah dalam kelas positif

FN (*False Negative*) : jumlah prediksi yang salah dalam kelas negatif

TN (*True Negative*) : jumlah prediksi yang benar dalam kelas positif

Berdasarkan *confusion matrix* diatas yaitu recall (r) dan precision (p) kelas positif dapat dihitung dengan menggunakan rumus berikut:

$$r = \frac{TP}{TP+FN} , \quad p = \frac{TP}{TP+FP} \quad (1)$$

Dari rumus di atas dapat diartikan recall (r) merupakan jumlah prediksi yang benar dalam kelas positif dibagi dengan jumlah positif yang sebenarnya dalam penelitian. Presisi (p) adalah jumlah prediksi yang benar di kelas yang positif dibagi dengan jumlah diklasifikasi sebagai positif.

2. Accuracy

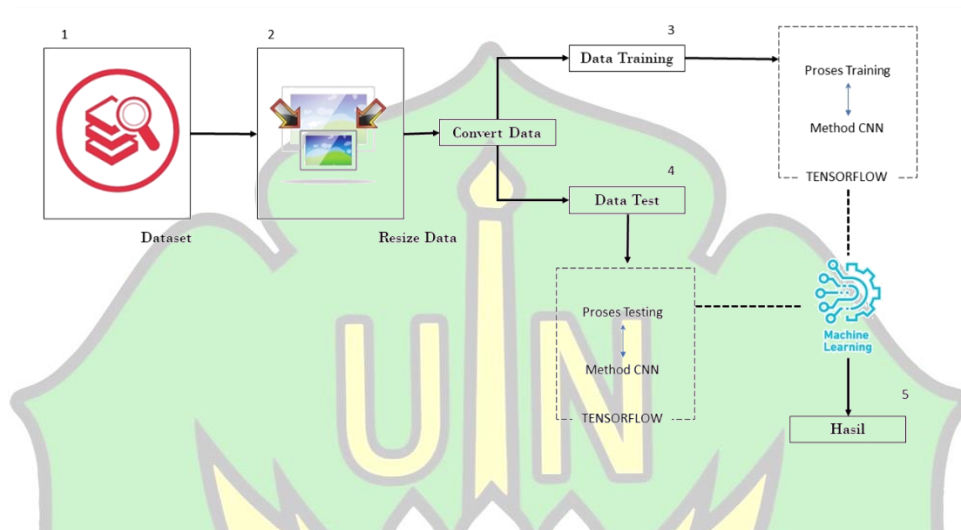
Accuracy adalah rasio prediksi yang benar (positif dan negatif) dengan data keseluruhan. Sehingga akurasi ini dapat menjawab prediksi yang benar dari keseluruhan data yang ada. Rumus yang digunakan dalam *accuracy* adalah sebagai berikut:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (2)$$

BAB III METODOLOGI PENELITIAN

A. Tahapan Penelitian

Tahapan-tahapan yang dilakukan dalam penelitian ini dapat dilihat pada gambar di bawah ini:



Gambar 3.3 Tahapan-tahapan penelitian

Berikut ini adalah penjelasan pada gambar tahapan-tahapan penelitian:

1. Dataset

Dalam melakukan penelitian ini, langkah pertama yang harus dilakukan adalah menyiapkan kumpulan data yang terdiri dari set gambar wajah. Dataset tersebut digunakan untuk masukan yang akan diproses dalam sistem. Dataset gambar wajah didapatkan dari proses pengambilan data yang dilakukan di Fakultas Tarbiyah dan Keguruan UIN Ar-Raniry, Banda Aceh. Ukuran dataset gambar wajah adalah 800x650 pixel resolusi 500dpi yang berjumlah 300 data wajah yang mengenakan hijab dengan setiap orangnya memiliki gambar wajah dengan 2

ekspresi wajah. Dari setiap foto gambar wajah tersebut memiliki tiga sisi foto gambar wajah yaitu, depan, samping kanan dan samping kiri. Yang terdiri dari:

a) *Data training*

Data gambar wajah yang digunakan untuk proses *training* berjumlah 250 data gambar wajah.

b) *Data testing*

Data gambar wajah yang digunakan untuk proses *testing* berjumlah 50 data gambar wajah.

2. *Resize Data*

Ukuran data gambar yang digunakan adalah 800x650 piksel, yang merupakan ukuran yang sangat besar dan terlalu berat dan sistem akan sulit dalam melakukan pengolahan data *training*. Maka sebelum memasuki tahap training data dilakukan *resize data* menjadi 80x80 pixel. Data gambar wajah yang telah di *resize* kemudian datanya akan di convert kedalam XML kemudian dalam format CSV dan terakhir dalam format Tfrecored, agar data dapat dimasukkan kedalam *library tensorflow*.



Gambar 4.3 (a) Sebelum diresize (b) Sesudah diresize

3. Data Training

Pada tahapan ini dataset yang telah siap akan *ditraining* dengan menggunakan metode *convolutional neural network*. Dalam proses training ini metode *convolutional neural network* akan dilatih untuk memperoleh akurasi yang tinggi dari klasifikasi yang dilakukan[19].

4. Data Testing

Data testing disini merupakan bagian dari dataset yang akan dites untuk melihat keakuratannya atau performanya. Data test juga akan menggunakan metode *convolutional neural network*.

5. Hasil

Pada tahapan terakhir akan memperlihatkan hasil kinerja dari tensorflow yang menampilkan tingkat keakuratan dari dataset yang telah dimasukkan.

B. Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah data gambar wajah mahasiswi di Fakultas Tarbiyah dan Keguruan, UIN Ar-Raniry, Banda Aceh. Ukuran data yang digunakan adalah 800x650 pixel resolusi 500dpi yang berjumlah 300 data wajah yang mengenakan hijab dengan setiap orangnya memiliki gambar wajah dengan 2 ekspresi wajah. Data yang telah dikumpulkan selanjutnya akan di *resize* karena ukuran data sebelumnya terlalu besar dan sistem juga akan berat ketika melakukan proses training.

C. Metode Analisis Data

Software yang digunakan dalam penelitian ini yaitu tensorflow dan *Python*

3.6.3. Metode analisis data yang digunakan dalam penelitian ini adalah *Convolutional Neural Network (CNN)*.



BAB IV PEMBAHASAN DAN HASIL

A. Persiapan *Software*

1. *Anaconda & Tensorflow*

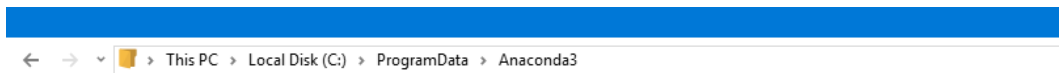
Anaconda merupakan *tools* yang dirancang untuk para *data scientist* yang mana didalamnya terdapat *environment manager* dan *package* yang akan digunakan untuk Python/R. Didalam *anaconda* terdapat berbagai *library* yang dapat diinstall, salah satunya adalah *library* untuk Tensorflow. Berikut adalah langkah-langkah untuk menginstall *Anaconda* dan Tensorflow:

Langkah-langkah yang harus dilakukan untuk penginstalan *Anaconda & Tensorflow* adalah sebagai berikut:

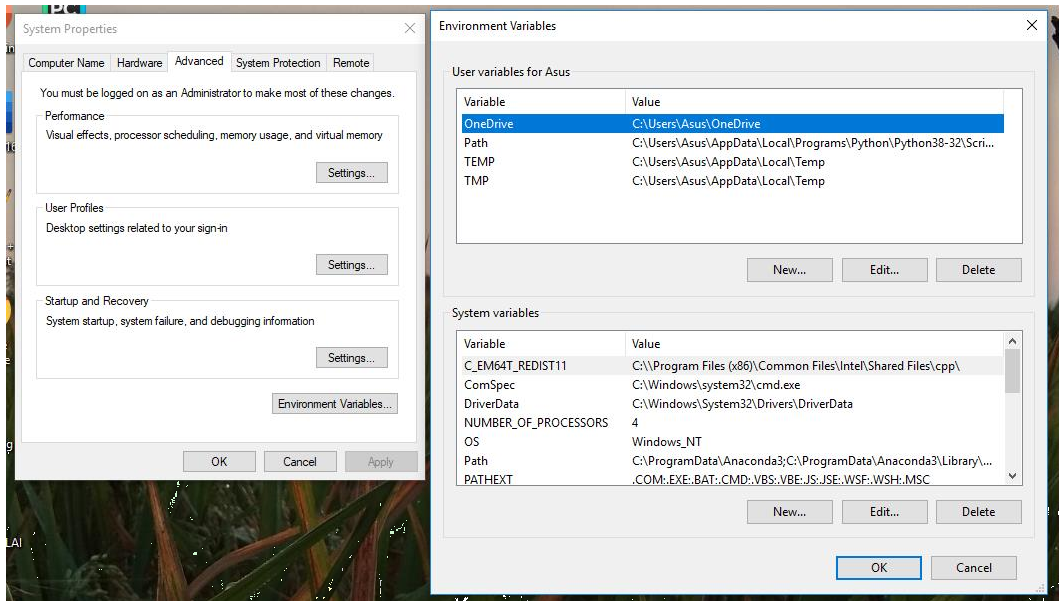
- a. Mengunjungi halaman website untuk mendownload *Anaconda*:
<https://www.anaconda.com/>
- b. Kemudian pilih *download*, kemudian pilih yang versi windows lalu klik yang Python versi 3.7.



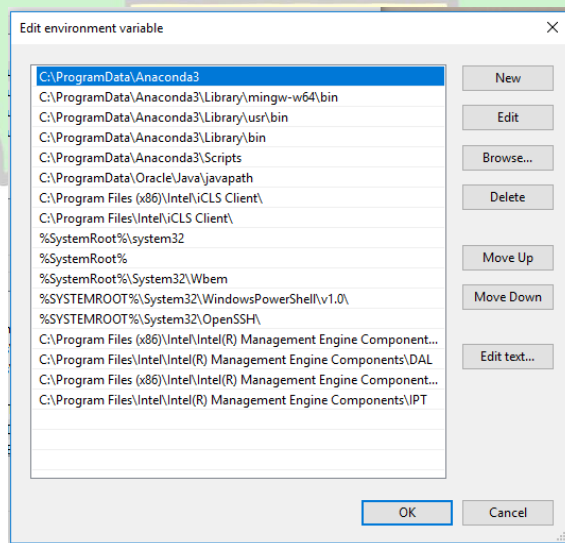
- c. Setelah berhasil di *download* lalu di install, ketika pada proses penginstalan tempatkan pada folder *Anaconda* di *C:\ProgramData\Anaconda3*.



d. Selanjutnya buat pengaturan pada setting path di environment variables, seperti pada gambar dibawah ini.



e. Pada setting *environment* tambahkan *variables* sesuai folder *Anaconda* diatas.



f. Setelah *Anaconda* berhasil di install, kemudian buka *Command prompt*.

g. Lalu ketikkan kode dibawah ini:

- Conda create -n
- Tensorflow_cpu pip python=3.7
- Activate tensorflow_cpu

B. *Preprocessing Image*

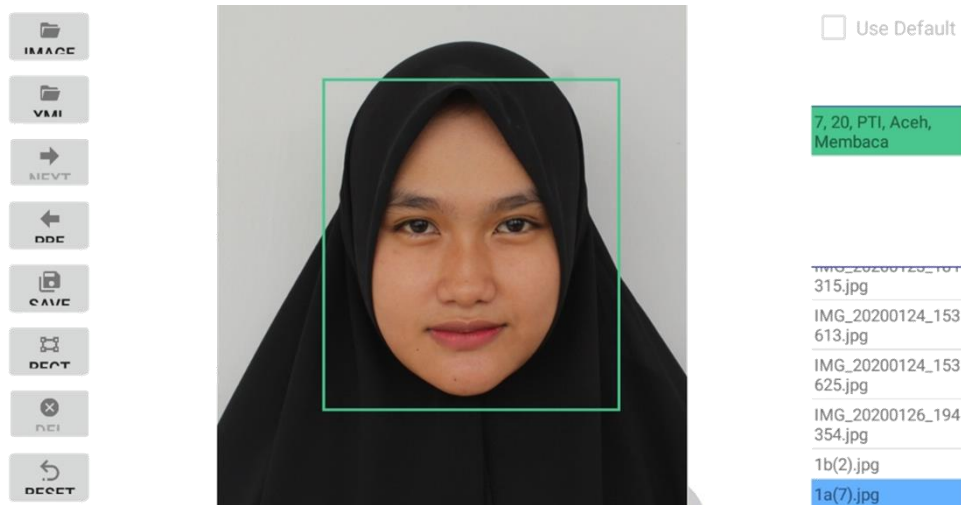
Setelah menyelesaikan tahapan pengumpulan dataset di Fakultas Tarbiyah dan Keguruan di UIN Ar-Raniry, Banda Aceh yang berupa dataset gambar wajah yang menggunakan hijab, maka selanjutnya akan dilakukannya tahapan *preprocessing image*. Tahapan-tahapan yang akan dilakukan adalah sebagai berikut:

1. Pelabelan Gambar

Pelabelan gambar ini dilakukan untuk menyimpan informasi gambar yang nantinya akan disimpan dalam bentuk XML dengan format Pascal VOC. Pelabelan ini akan dilakukan pada tahapan awal untuk dapat menginput dataset. Proses dalam pelabelan akan dilakukan secara manual untuk 300 dataset gambar wajah yang berhijab menggunakan LabelImg.



Gambar 5.4 Sebelum Pelabelan *image*



Gambar 6.4 Sesudah Pelabelan image

2. Konversi Dataset

Konversi data dilakukan untuk mengubah *extension* yang ada pada data setelah pelabelan. Agar data dari *image* yang telah diberi label dapat dimasukkan kedalam Tensorflow. Konversi *dataset* yang akan dilakukan yaitu dari konversi *dataset* XML-CSV dan CSV-TFRecord.

a. XML ke CSV

Setelah tahap pelabelan selesai dilakukan, maka selanjutnya akan dilakukan tahapan konversi dataset dari XML ke CSV. Berikut kode konversi dataset *train* dan *test* dari XML ke CSV.

```
LabelImage.xml x
1 <?xml version="1.0" encoding="utf-8"?>
2 <Company>
3 <Projects>
4 <Project>
5 <Id>1</Id>
6 <Name>ZYNK PROJECT</Name>
7 <Description>Label Image</Description>
8 <StartDate>2020-01-13T00:00:00</StartDate>
9 <EndDate>2020-01-14T00:00:00</EndDate>
10 <Status>ACTIVE</Status>
11 <AccountReference>Label</AccountReference>
12 <ProjectAddress>
13 <ContactName>1</ContactName>
14 <Address1>Aceh</Address1>
15 <Address2>Aceh</Address2>
16 <Age>20</Age>
17 <Hoby>Travelling</Hoby>
18 <Jurusan>PTI</Jurusan>
19 </ProjectAddress>
20 <QuotedPrice>300</QuotedPrice>
21 <Analysis1>ANALYSIS1</Analysis1>
22 <Analysis2>ANALYSIS2</Analysis2>
```

Gambar 7.4 Data dalam bentuk XML

Kemudian data XML yang telah ada akan dikonversikan dalam CSV dengan menggunakan python. Berikut adalah coding dari konversi data XML to CSV:



```

generate_tfrecord.py x xml_to_csv.py x
1 import os
2 import glob
3 import pandas as pd
4 import xml.etree.ElementTree as ET
5 def xml_to_csv(path):
6     xml_list = []
7     for xml_file in glob.glob(path + '/*.xml'):
8         tree = ET.parse(xml_file)
9         root = tree.getroot()
10        for member in root.findall('object'):
11            value = (root.find('filename').text,
12                    int(root.find('size')[0].text),
13                    int(root.find('size')[1].text),
14                    member[0].text,
15                    int(member[4][0].text),
16                    int(member[4][1].text),
17                    int(member[4][2].text),
18                    int(member[4][3].text)
19            )
20            xml_list.append(value)
21        column_name = ['filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax']
22        xml_df = pd.DataFrame(xml_list, columns=column_name)
23        return xml_df
24    def main():
25        for directory in ['train', 'test']:
26            image_path = os.path.join(os.getcwd(), 'annotations/{}'.format(directory))
27            xml_df = xml_to_csv(image_path)
28            xml_df.to_csv('data/{}_labels.csv'.format(directory), index=None)
29            print('Successfully converted xml to csv.')

```

Gambar 8.4 Code konversi XML ke CSV

Berikut adalah pseudocode dari *coding* konversi XML ke CSV seperti

Gambar.7 diatas:

Tahapan-Tahapan atau alur file xml_to_csv.py:

1. Import file xml
2. Import library csv
3. Menentukan array yang akan digunakan
4. Membuat directory untuk train dan test

Setelah menyimpan *code* diatas, maka selanjutnya kita akan mengkonversi

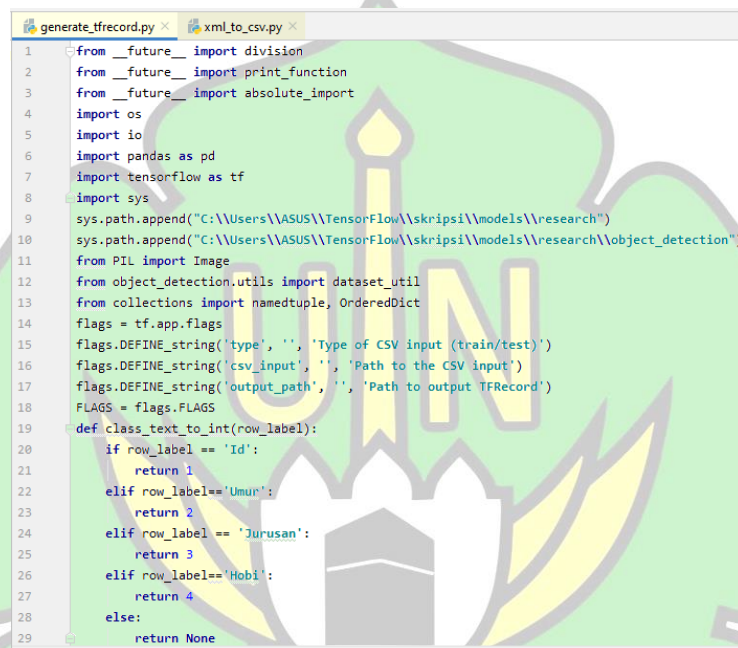
file diatas dengan menjalankan perintah seperti dibawah ini:

Python xml_tocsv.py

Maka setelah kita menjalankan perintah tersebut file csv akan langsung terinput kedalam *library* Tensorflow.

3. Konversi Dataset CSV ke TFRecord

Setelah proses konversi dataset dari XML ke CSV selesai, maka selanjutnya akan dilakukan konversi CSV ke *TFRecord*. Konversi dalam bentuk *TFRecord* akan digunakan untuk *feeding* (pertukaran) data pada proses *training*. Berikut adalah kode untuk membuat *TFRecord*.



```
1 from __future__ import division
2 from __future__ import print_function
3 from __future__ import absolute_import
4 import os
5 import io
6 import pandas as pd
7 import tensorflow as tf
8 import sys
9 sys.path.append("C:\\Users\\ASUS\\TensorFlow\\skripsi\\models\\research")
10 sys.path.append("C:\\Users\\ASUS\\TensorFlow\\skripsi\\models\\research\\object_detection")
11 from PIL import Image
12 from object_detection.utils import dataset_util
13 from collections import namedtuple, OrderedDict
14 flags = tf.app.flags
15 flags.DEFINE_string('type', '', 'Type of CSV input (train/test)')
16 flags.DEFINE_string('csv_input', '', 'Path to the CSV input')
17 flags.DEFINE_string('output_path', '', 'Path to output TFRecord')
18 FLAGS = flags.FLAGS
19 def class_text_to_int(row_label):
20     if row_label == 'Id':
21         return 1
22     elif row_label == 'Umur':
23         return 2
24     elif row_label == 'Jurusan':
25         return 3
26     elif row_label == 'Hobi':
27         return 4
28     else:
29         return None
```

Gambar 9.4 Code konversi CSV ke Tfrecored

Berikut adalah psedocode dari *coding* konversi CSV ke Tfrecored seperti

Gambar.8 diatas:

Tahapan-Tahapan atau alur file generate_tfrecord.py:

1. Import library tensorflow
2. Import path dokumen image atau letak file image
3. Membuat class klasifikasi
4. Membuat pemanggilan format file image (jpg/png)
5. Membuat deklarasi Xmins, Xmaxs, Ymins, Ymaxs

Setelah menyimpan *code* diatas, maka selanjutnya kita akan mengkonversi file diatas dengan menjalankan perintah seperti dibawah ini:

```
python generate_tfrecord.py --type=train --
csv_input=data/train_labels.csv --output_path=data/train.record
# dan
python generate_tfrecord.py --type=test --
csv_input=data/test_labels.csv --output_path=data/test.record
```

Maka setelah kita menjalankan perintah tersebut file tfrecord akan langsung terinput kedalam *library* Tensorflow.

C. Pengolahan Image didalam Tensorflow

1. Konfigurasi *Object Detection Pipeline*

Konfigurasi *object detection pipeline* ini merupakan folder ataupun berkas-berkas file yang akan diinputkan kedalam Tensorflow. Berikut adalah struktur file yang akan diinputkan kedalam Tensorflow dengan menggunakan *python*:

a. *Annotations*

Annotations merupakan nama folder yang didalamnya terdapat subfolder yang filenya berextension.xml. File ini adalah hasil dari *convert image* yang sudah diberi label.

b. Data

Didalam folder data terdapat file dengan *extention* .record dan .csv.

c. *Images*

Folder *images* memiliki 2 subfolder yaitu folder test dan folder train, dimana dalam folder tersebut terdapat file gambar.

d. *Training*

Didalam folder *training* terdapat file *pipeline config* yang nantinya digunakan untuk proses *training object detection*.

e. *Generate_tfrecord.py*

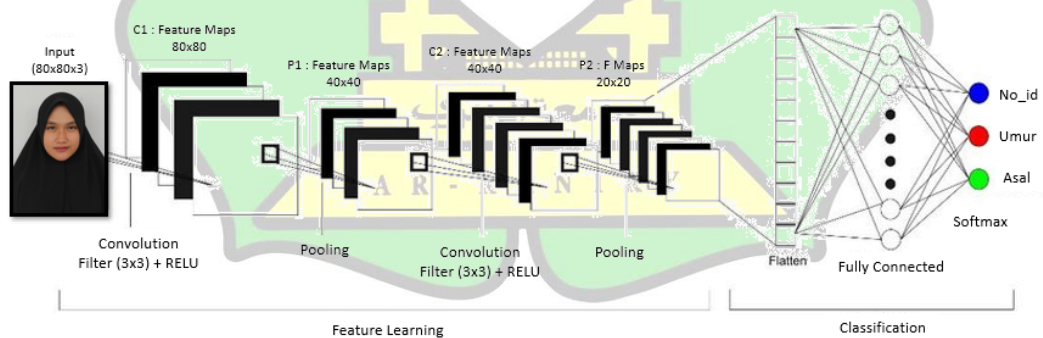
Folder ini adalah sebuah *script* file dengan *extention .py* yang nantinya digunakan untuk *convert* file *.csv* menjadi *.tfrecord*.

f. *Xml_to_csv.py*

Folder ini adalah sebuah *script* file dengan *extention .py* yang nantinya digunakan untuk *convert .xml* menjadi *.csv*.

D. Pemodelan Jaringan *Convolutional Neural Network* (CNN)

Setelah melalui beberapa tahapan, yaitu dari tahapan pelabelan, *convert* data dan membuat folder *object detection pipeline*. Maka selanjutnya data yang telah dikumpulkan akan *training* dengan menggunakan algoritma CNN. Pembentukan model jaringan algoritma CNN sangat mempengaruhi hasil dari akurasi model.



Gambar 10.4 Model Jaringan

Pada gambar di atas memperlihatkan pemodelan jaringan pada proses *training* yang akan menghasilkan model yang sederhana tetapi akurat. Input gambar

yang digunakan dalam penelitian ini berukuran 80x80. Pemodelan jaringan CNN dapat dijelaskan sebagai berikut:

1. Tahap konvolusi yang pertama digunakan karnel 3x3 dengan jumlah filter 80. Tahapan ini adalah proses kombinasi antara dua buah matriks yang berbeda untuk menghasilkan nilai matriks yang baru. Setelah tahap konvolusi selesai, maka akan ditambahkan sebuah aktivasi baru yaitu RELU (*Retrified Linear Unit*). RELU ini berfungsi untuk mengubah nilai negatif menjadi nol (menghilangkan nilai negatif pada matriks hasil konvolusi). Nilai *padding* yang digunakan dalam tahapan konvolusi adalah 0, sehingga hasil ukuran dari konvolusi tetap sama yaitu 80x80.
2. Tahapan *pooling* merupakan sebuah tahapan pengurangan ukuran matriks dengan memakai operasi *pooling*. *Pooling layer* terdiri atas sebuah *filter* yang ukuran nilai matriksnya secara bergantian akan bergeser pada area *feature maps*. Hasil dari tahapan *pooling* akan menghasilkan nilai matriks yang baru, karena dalam penelitian ini akan menggunakan aktivasi *maxpooling*. Cara kerja dari *maxpooling* adalah dengan mengambil nilai maksimum yang berdasarkan pergeseran karnelnya.
3. Tahap konvolusi yang kedua yaitu meneruskan hasil dari tahapan *pooling* pertama, yaitu dengan inputan matriks gambar yang berukuran 40x40 dengan jumlah *filter* 80 yang ukuran karnelnya 3x3. Pada tahapan ini juga menggunakan aktivasi RELU.

4. Tahapan selanjutnya yaitu *pooling* kedua, yang mana proses tahapannya sama dengan *pooling* yang pertama. Perbedaan antara keduanya hanya pada hasil akhir matriksnya. *Pooling* kedua menghasilkan *output* yang berukuran 40x40.
5. Setelah itu masuk ke tahapan *flatten* atau *fully connected*. Tahapan ini hanya terdiri dari satu *hidden layer*. Tahapan *flatten* mengubah *output pooling* layer menjadi *vector*. Lalu selanjutnya tahapan *flatten* akan melakukan proses pendeteksian gambar.
6. Tahapan yang terakhir adalah penggunaan aktivasi fungsi *softmax*. Fungsi ini biasanya digunakan dalam deteksi *multiclass linear discriminant analysis* dan *multinomial logistic regression*.

Jadi pemodelan jaringan CNN di atas digunakan untuk proses *training*. Dari proses *training* didapatkan bentuk model CNN sebagai berikut:

Tabel 2.4 Model Convolutional Neural Network (CNN)

No.	Nama	Size	Parameter
0	<i>Input</i>	80x80x3	0
1	<i>Conv2_1</i>	$(80+(2*1)-(3-1)) = 80x80x40$	$((3*3*3)+1)*40 = 1120$
2	<i>Maxpool_1</i>	40x40x40	0
3	<i>Conv2_2</i>	$(40+(2*1)-(3-1)) = 40x40x80$	$((3*3*40)+1)*80 = 28880$
4	<i>Maxpool_2</i>	20x20x40	0
5	<i>Flatten</i>	32000	0
6	<i>Dense</i>	400	$(3200*400)+400=$ 12800400
7	<i>Output</i>	3	$(400+1)*3 = 1203$

Total	12831603
--------------	-----------------

Tabel di atas adalah tabel yang dihasilkan dari hasil *training* dengan menggunakan pemodelan jaringan CNN seperti pada gambar di atas. Perhitungan *input* kedalam konvosional menggunakan rumus “ $input_size + 2*padding - (filter_size - 1)$ ”. Jadi, total keseluruhan parameter yang dihasilkan dari pemodelan tersebut adalah sebanyak 12831603 *neuron*.

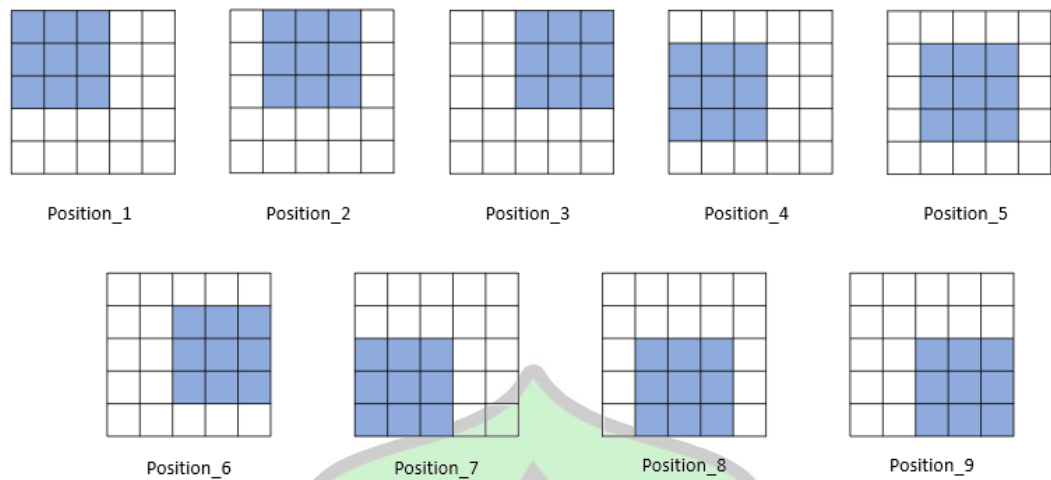
E. Proses Konvolusi

Agar proses konvolusi dapat dipahami, maka peneliti akan mengambil sampel dari sebagian matriks yang ada pada *input image*. Ukuran *input image* yang digunakan sebesar 80x80 *pixel*. Maka dari itu peneliti hanya menggunakan sebagian matriks dari *input image* untuk dijadikan sampel yang akan digunakan pada proses konvolusi.



Gambar 11.4 Proses Konvolusi

Pada gambar 11.4 di atas, *stride* yang digunakan pada karnel adalah 1 dengan ukuran 3x3. *Stride* ini memperlihatkan pergeseran jumlah karnel dalam matriks yang bernilai satu. Visualisasinya adalah sebagai berikut:



Gambar 12.4 Posisi kernel dalam konvolusi

Gambar di atas menunjukkan perhitungan dari *dot product* yang terdapat dalam proses konvolusi dengan ukuran karnelnya 3x3. Nilai *padding* yang digunakan adalah 1, agar terjadinya penambahan nilai 0 dinilai matriks inputan, yang nantinya berguna untuk menghasilkan nilai matriks *input* dan *output* yang sama. Perhitungan dari *dot product* adalah sebagai berikut:

$$\begin{aligned}
 \text{Position}_1 &= (2*1) + (3*(-1)) + (2*1) + (3*(-1)) + (4*1) + (3*(-1)) + (5*1) + \\
 &\quad (4*(-1)) + (6*(-1)) = -6
 \end{aligned}$$

$$\begin{aligned}
 \text{Position}_2 &= (3*1) + (4*(-1)) + (3*1) + (5*(-1)) + (4*1) + (6*(-1)) + (4*1) + \\
 &\quad (3*(-1)) + (3*(-1)) = -7
 \end{aligned}$$

$$\begin{aligned}
 \text{Position}_3 &= (5*1) + (4*(-1)) + (6*1) + (4*(-1)) + (3*1) + (3*(-1)) + (3*1) + \\
 &\quad (2*(-1)) + (5*(-1)) = -1
 \end{aligned}$$

$$\begin{aligned}
 \text{Position}_4 &= (3*1) + (2*(-1)) + (3*1) + (4*(-1)) + (3*1) + (4*(-1)) + (3*1) + \\
 &\quad (2*(-1)) + (5*(-1)) = -6
 \end{aligned}$$

$$\begin{aligned} \text{Position}_5 &= (4*1) + (3*(-1)) + (4*1) + (4*(-1)) + (6*1) + (3*(-1)) + (3*1) + \\ &\quad (3*(-1)) + (3*(-1)) = 1 \end{aligned}$$

$$\begin{aligned} \text{Position}_6 &= (4*1) + (6*(-1)) + (3*1) + (3*(-1)) + (3*1) + (3*(-1)) + (2*1) + \\ &\quad (5*(-1)) + (1*(-1)) = -6 \end{aligned}$$

$$\begin{aligned} \text{Position}_7 &= (2*1) + (3*(-1)) + (4*1) + (3*(-1)) + (4*1) + (5*(-1)) + (6*1) + \\ &\quad (3*(-1)) + (5*(-1)) = -3 \end{aligned}$$

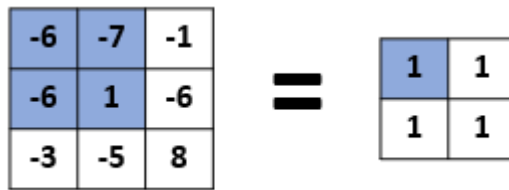
$$\begin{aligned} \text{Position}_8 &= (3*1) + (4*(-1)) + (5*1) + (6*(-1)) + (3*1) + (5*(-1)) + (3*1) + \\ &\quad (3*(-1)) + (1*(-1)) = -5 \end{aligned}$$

$$\begin{aligned} \text{Position}_9 &= (6*1) + (3*(-1)) + (5*1) + (3*(-1)) + (3*1) + (1*(-1)) + (5*1) + \\ &\quad (1*(-1)) + (3*(-1)) = 8 \end{aligned}$$

Setelah proses konvolusi selesai, maka akan dilanjutkan dengan proses *pooling layer*. *Pooling layer* ini digunakan untuk menghilangkan nilai negatif yang ada pada hasil matriks, dengan menambahkan aktivasi fungsi RELU (*Rectified Linear Unit*).

F. Proses *Pooling*

Pooling adalah pengurangan langkah dalam matriks nilai dengan menggunakan proses penggabungan operasi. Dalam proses *pooling* ini biasanya menggunakan metode *maxpooling*. Metode ini biasanya digunakan dalam penelitian yang berkaitan dengan *deep learning*. Gambaran dari proses *pooling* adalah sebagai berikut:



Gambar 13.4 Proses Pooling

Ukuran *pooling* yang dipakai seperti pada gambar di atas berukuran 2x2 dengan penggunaan *stride* 1, dimana pergeseran karnel pada *input* matriks berjumlah satu. Dalam proses *pooling* digunakan *maxpooling*, yang nantinya akan bergeser sesuai dengan *stride* dan ukurannya, yang bertujuan untuk mendapatkan nilai paling maksimum. Seperti pada gambar di atas *output* yang dihasilkan berasal dari nilai maksimum yang diambil dari matrik *feature map* hasil konvolusi. *Output* dari *maxpooling* berukuran 2x2.

G. Proses *Fully Connected*

Proses *fully connected* adalah proses yang hasilnya masih dalam bentuk *multidimensional array*. Karena bentuk hasilnya masih dalam bentuk *array*, maka perlu ditambahkan fungsi *flatten*. Fungsi *flatten* ini akan membuat hasil dalam bentuk *array* menjadi bentuk *vector*, yang nantinya akan dapat digunakan sebagai *input* dari *fully connected layer*. Setelah penggunaan fungsi *flatten* ditambahkan, selanjutnya akan ditambahkan fungsi *dense*. Fungsi ini akan menambahkan *layer* pada *fully connected* yang akan dijadikan sebagai *layer classification*.

H. Proses *Classification*

Pada proses *classification* nantinya akan melakukan penentuan bagian-bagian mana yang akan dipilih pada setiap piksel akan dibentuk menjadi pola

ataupun bentuk wajah yang sesuai dari *input image* yang telah ada. Dalam penelitian ini peneliti menguji gambar wajah yang menggunakan hijab dengan 2 ekspresi (*flat* dan *tersenyum*) dengan sudut *image* yang berbeda, yang nantinya apakah gambar *input image* yang berhijab dapat dideteksi dengan baik.

I. *Detection Output*

Detection output adalah hasil akhir dari deteksi gambar wajah yang menggunakan hijab.

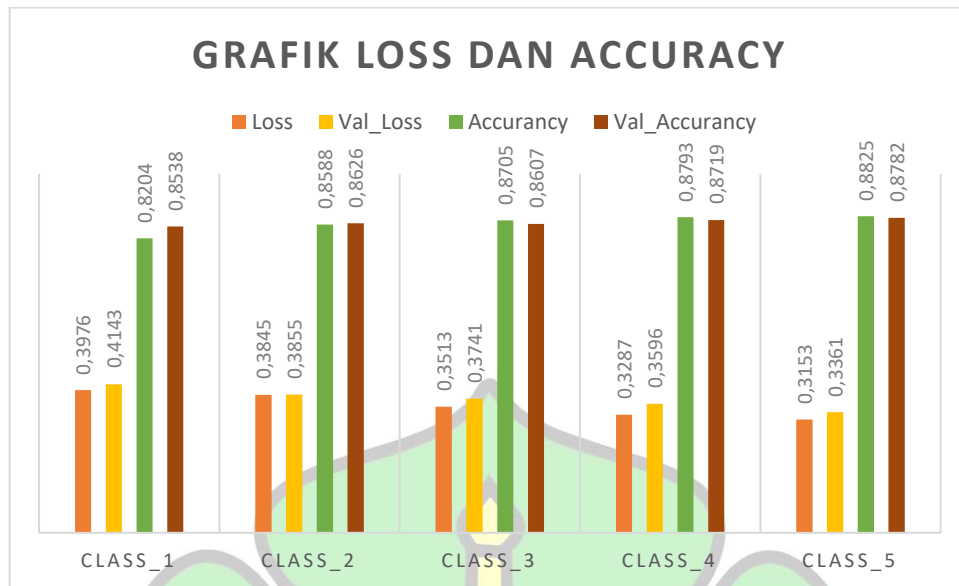
J. Hasil

1. Hasil Model Training

Setelah selesai melalui proses pemodelan jaringan CNN, kemudian model itu akan diuji nilai *loss* dan nilai akurasi. Dengan menggunakan *epoch* sebesar 20 dan nilai *learning rate* 0,001. Nilai *epoch* dan *learning rate* ditentukan oleh peneliti untuk mengurangi waktu kinerja sistem. Berikut adalah hasil *loss* dan akurasi yang telah dijalankan di *tensorflow*.

Tabel 3.4 Hasil Loss dan Accuracy

Class	Loss	Val_Loss	Accuracy	Val_Accuracy
Class_1	0,3976	0,4143	0,8204	0,8538
Class_2	0,3845	0,3855	0,8588	0,8626
Class_3	0,3513	0,3741	0,8705	0,8607
Class_4	0,3287	0,3596	0,8793	0,8719
Class_5	0,3153	0,3361	0,8825	0,8782



Grafik 1.4 Hasil Loss dan Accuracy

Dari hasil gambar di atas dapat disimpulkan bahwa satu *input image* yang dimasukkan dapat mencari pasangan *imagenya* dengan mengecek *image* lain yang dimasukkan dalam *training*. Dalam hasil *training* tersebut menghasilkan 5 *accuracy* dan *loss*, karena *image* inputan membandingkan dengan 5 sisi yang ada pada dataset. Dalam dataset terdapat 6 sisi pada setiap *image*, dimana 1 sisi diambil untuk menjadi *input image*. Waktu pelatihan yang dibutuhkan untuk 20 *epoch* pada *training* ini adalah 1,5 menit. Banyaknya *epoch* yang akan digunakan juga mempengaruhi waktu yang akan dilakukan oleh pemodelan *training*. Karena semakin banyaknya *epoch*, waktu yang akan diperlukan untuk *training* akan semakin lama. Kemudian nilai *accuracy* dari data *validation* adalah sebesar 92% dengan nilai *loss* sebesar 0,4143.

2. Hasil Data Test

Data yang digunakan untuk *testing* adalah sebanyak 120, untuk setiap kelas jenis *image* adalah sebanyak 6 gambar, yang diambil dari sisi dan ekspresi pada

gambar dataset. Dari hasil perhitungan data *test* didapatkan *confusion matrix*, *precision* dan *recall* sebagai berikut:

Tabel 4.4 confusion matrix, precision dan recall

Matriks	Class	Predict_Class	Loss_Class	Precision	Recall	Confidences
Id_1	6	6		1	0,0666	95%
Id_2	6	5	1	0,6666	0,1333	91%
Id_3	6	6		0,5	0,0666	95%
Id_4	6	5	1	0,5	0,1333	88%
Id_5	6	4	2	0,3	0,2	70%
Id_6	6	5	1	0,4	0,1333	84%
Id_7	6	6		1	0,0666	95%
Id_8	6	4	2	0,2222	0,1333	71%
Id_9	6	5	1	0,5	0,1333	88%
Id_10	6	5	1	0,5	0,1333	88%

Dari hasil tabel *confusion matrix* didapatkan hasil prediksi model dari data *testing* adalah sangat baik. Prediksi terhadap gambar wajah yang menggunakan hijab dideteksi berdasarkan *Id* nya, yang mana *Id* menunjukkan *class* utama dataset. Prediksi *Id_1* dideteksi kedalam *Id_1*, yang artinya deteksi terhadap gambar tersebut adalah benar. Prediksi pada *Id_2* dideteksi benar sebagai *Id_2* sebanyak 5 dan *missing data* dari *input image Id_2* sebagai *Id_11* sebanyak 1. Lalu prediksi pada *Id_3* dideteksi kedalam *Id_3* yang mana deteksi terhadap gambar adalah benar. Prediksi pada *Id_4* dideteksi benar sebagai *Id_4* sebanyak 5 dan *missing data* dari *input image Id_4* sebagai *Id_15* sebanyak 1. Sampai seterusnya hingga *Id_50*, yang nilai *confusion matrix* nya terdapat dalam tabel pada lampiran *confusion matrix*. Nilai yang terdapat dikolom *precision* adalah hasil yang diperoleh dari perbandingan jumlah total positif yang diklasifikasikan benar dengan jumlah

total positif yang diprediksi. Dan nilai yang terdapat dikolom *recall* adalah hasil perbandingan antara penjumlahan positif yang diklasifikasikan benar dengan jumlah total positif. Sedangkan nilai *confidences* adalah nilai tingkat keakuratan yang dihasilkan oleh setiap *predict class*. Kemudian perhitungan akurasi dari total matriks yang ada di atas adalah sebagai berikut:

$$\text{Akurasi Klasifikasi} = \frac{\text{Jumlah Prediksi Benar}}{\text{Jumlah Total Prediksi}} \times 100\% \quad (3)$$

$$\text{Akurasi Klasifikasi} = \frac{263}{300} \times 100\%$$

$$\text{Akurasi Klasifikasi} = 87\%$$

Jadi, hasil akurasi yang diperoleh dari model dengan ukuran *image* 80x80 piksel dan nilai *learning rate* 0,001 adalah sebesar 87%, yang mana nilai 87% sudah menunjukkan hasil akurasi yang baik karena *image* yang dideteksi memakai atribut (hijab) dan memiliki ekspresi yang berbeda (datar dan tersenyum) serta posisi *image* yang tidak searah (posisi depan, samping kanan dan samping kiri).

K. Penentuan Parameter Model

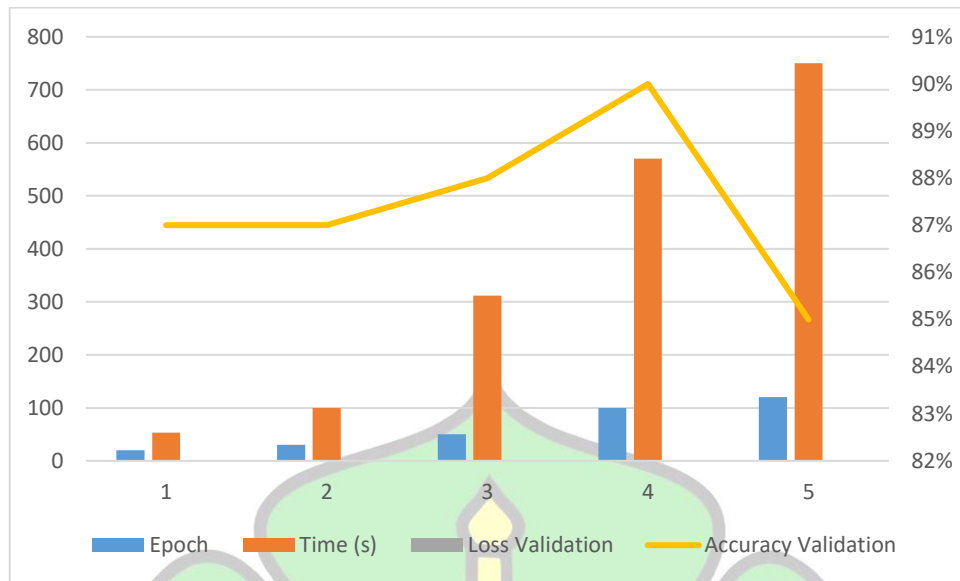
Penentuan ini bertujuan untuk menentukan parameter-parameter terbaik dalam model CNN (*Convolutional Neural Network*). Parameter-parameter yang akan diuji yaitu pengaruh jumlah *epoch*, pengaruh jumlah data *training*, dan pengaruh jumlah nilai *learning rate*. Parameter ini diuji untuk membandingkan model mana yang paling baik dengan tetap memperhatikan nilai dari parameter itu sendiri.

1. Pengaruh Jumlah Nilai *Epoch*

Epoch merupakan suatu proses dimana ketika keseluruhan dataset sudah melalui proses *training* dalam NN (*Neural Network*) sampai proses itu kembali keawal dalam satu kali putaran. Dalam *deep learning* satu *epoch* itu memakan tempat yang besar dalam pelatihan, karena seluruh dataset yang ada akan diikuti dalam tahapan *training* sehingga membutuhkan waktu yang lama. Agar mempermudah dan mempercepat pelatihan, maka dataset akan dibagi menjadi beberapa *batch* (*Batch Size*). Nilai dari *batch size* biasanya ditentukan oleh peneliti dengan tetap melihat jumlah dataset yang ada. Berikut adalah hasil akurasi dengan *epoch* yang berbeda yang dijalankan di *tensorflow*:

Tabel 5.4 Akurasi dengan nilai Epoch berbeda

<i>Epoch</i>	<i>Accuracy Validation</i>	<i>Loss Validation</i>	<i>Time (s)</i>
20	87%	0.4143	53
30	87%	0.3214	100
50	88%	0.1924	312
100	90%	0.1265	570
120	85%	0,1345	750



Grafik 2.4 Akurasi dengan nilai Epoch berbeda

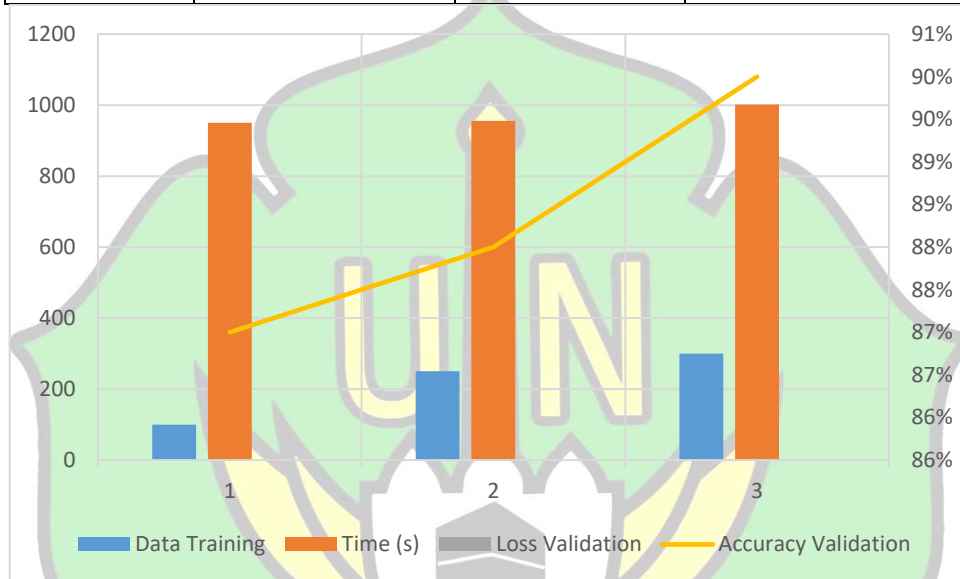
Dari hasil di atas maka dapat dilihat nilai akurasi paling tinggi terdapat pada *epoch* 100 dengan akurasi sebesar 90%. Jadi dapat disimpulkan bahwa semakin menuju ke *epoch* 100, maka tingkat akurasi dari *testing* akan semakin tinggi. Tetapi ketika nilai *epoch* lebih dari 100, maka nilai akurasi yang dihasilkan akan menurun karena disebabkan oleh jumlah *epoch* yang semakin banyak dan tidak sebanding dengan jumlah dataset yang ada.

2. Pengaruh Jumlah Data Training

Untuk menguji pengaruh ini, maka peneliti akan membagi jumlah data *training* menjadi 3 bagian yaitu 100, 250 dan 300. Setiap masing-masing dari data yang akan di *training* akan diambil 20 gambar untuk dijadikan proses validasi. Hasil dari ketiga jumlah data *training* yang telah dibagi adalah sebagai berikut:

Tabel 6. 4 Jumlah Data Training Berbeda

<i>Data Training</i>	<i>Accuracy Validation</i>	<i>Loss Validation</i>	<i>Time (s)</i>
100	87%	0.4143	950
250	88%	0.4651	956
300	90%	0.1924	1002



Grafik 3.4 Jumlah Data Training Berbeda

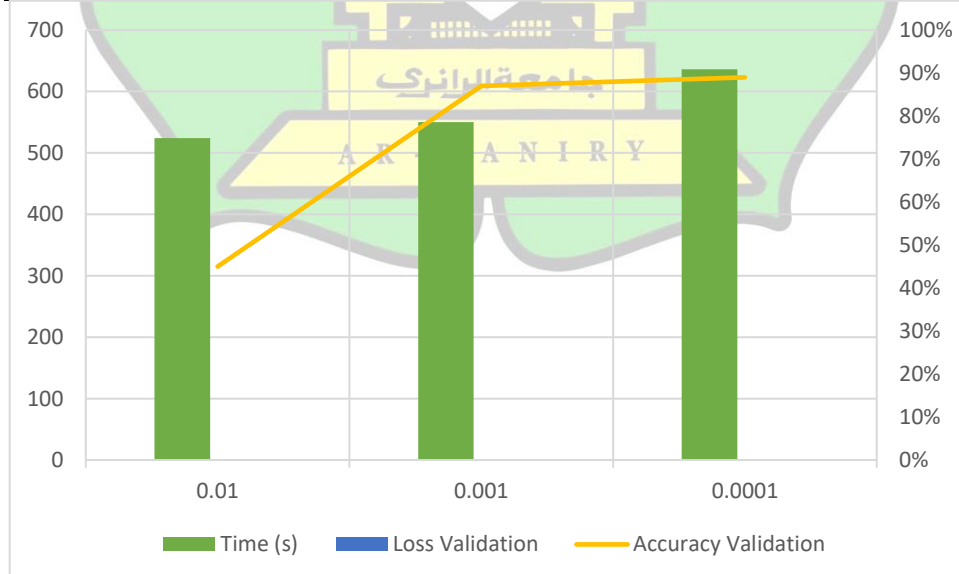
Dari tabel diatas dapat disimpulkan bahwa hasil akurasi yang diperoleh untuk data *training* yang berbeda berkisaran antara 87-90%. Dengan hasil presentase yang didapatkan terbukti bahwa dengan model jaringan CNN (*Convolutional Neural Network*) yang telah dibuat berhasil mendeteksi wajah yang menggunakan hijab dengan baik. Jadi semakin banyak data *training* yang dipakai, maka tingkat hasil akurasinya akan semakin tinggi, karena komputer dapat lebih banyak memahami pola dari gambar yang dimasukkan. Sehingga ketepatan dalam proses pendeteksian akan semakin baik.

3. Pengaruh Jumlah Nilai *Learning Rate*

Learning rate adalah parameter dari *gradient descent*. *Gradient descent* ini merupakan salah satu algoritma yang digunakan untuk mencari nilai minimum lokal yang dapat dihasilkan oleh suatu fungsi parametrik. Dalam penelitian ini peneliti menggunakan nilai *learning rate* 0.01, 0.001, dan 0.0001. Hasil nilai *learning rate* langsung dapat dijalankan di *tensorflow* dengan menambahkan nilai *learning rate* yang telah ditetapkan. Penentuan nilai *learning rate* ini akan sangat berpengaruh terhadap nilai akurasi yang akan dihasilkan. Berikut adalah tabel hasil akurasi dari masing-masing nilai *learning rate* yang telah dimasukkan.

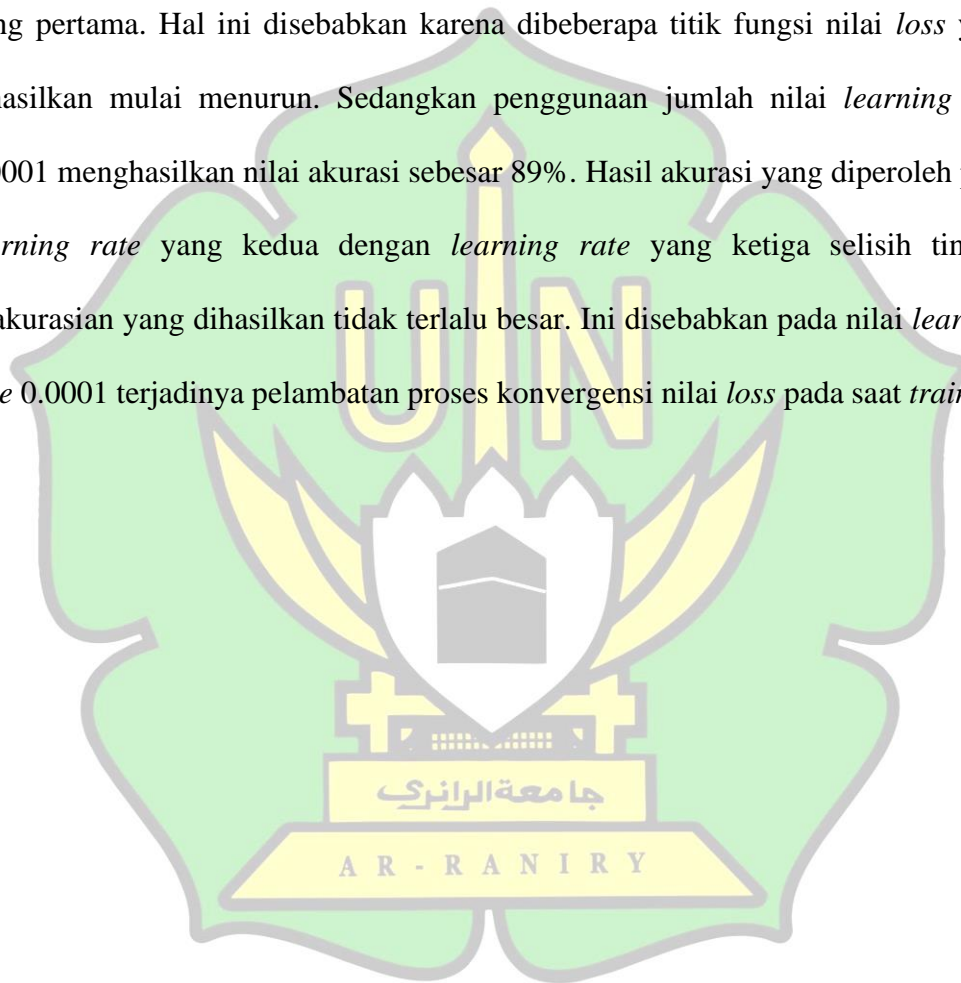
Tabel 7.4 Jumlah Learning Rate Berbeda

<i>Learning Rate</i>	<i>Accuracy Validation</i>	<i>Loss Validation</i>	<i>Time (s)</i>
0.01	45%	1.0423	524
0.001	87%	0.4143	550
0.0001	89%	0.6734	636



Grafik 4.4 Jumlah Learning Rate Berbeda

Dari hasil tabel diatas dapat dilihat pada nilai *learning rate* 0.01 tingkat akurasi yang dihasilkan sebesar 45%, ini dikarenakan semakin besar jumlah nilai *learning rate* maka nilai jumlah *loss* akan semakin meningkat. Lalu pada nilai *learning rate* 0.001 tingkat akurasi yang dihasilkan adalah sebesar 87%. Tingkat akurasi yang dihasilkan lebih meningkat daripada penggunaan nilai *learning rate* yang pertama. Hal ini disebabkan karena di beberapa titik fungsi nilai *loss* yang dihasilkan mulai menurun. Sedangkan penggunaan jumlah nilai *learning rate* 0.0001 menghasilkan nilai akurasi sebesar 89%. Hasil akurasi yang diperoleh pada *learning rate* yang kedua dengan *learning rate* yang ketiga selisih tingkat keakurasian yang dihasilkan tidak terlalu besar. Ini disebabkan pada nilai *learning rate* 0.0001 terjadinya pelambatan proses konvergensi nilai *loss* pada saat *training*.



BAB VI KESIMPULAN

A. Kesimpulan

Dalam penelitian model CNN ini menggunakan *inputan image* yang berukuran 80x80 *pixel*, dengan *filter* 3x3. *Epoch* yang digunakan dalam penelitian ini sebesar 20 dan *learning rate* yang dipakai sebesar 0,001. Data *training* yang digunakan sebanyak 250 dan untuk data *testing* sebanyak 50. Gambar wajah yang menggunakan hijab ada 300 gambar, dengan menggunakan ekspresi wajah *flat*/datar dan tersenyum.

Berdasarkan pembahasan dan hasil analisis yang telah dilakukan dalam penelitian ini, maka kesimpulannya adalah sebagai berikut:

1. Hasil nilai *accuracy* yang dihasilkan dari pemodelan CNN yang telah dibuat yaitu data *training* sebesar 92% dan data *test* sebesar 87%.
2. Tingkat keakurasian yang paling tinggi yaitu dengan menggunakan nilai *epoch* 100 sebesar 90%. Hal ini dikarenakan semakin menuju ke *epoch* 100, maka tingkat akurasi dari *testing* akan semakin tinggi. Tetapi ketika nilai *epoch* lebih dari 100, maka nilai akurasi yang dihasilkan akan menurun karena disebabkan oleh jumlah *epoch* yang semakin banyak dan tidak sebanding dengan jumlah dataset yang ada.
3. Jumlah maksimal dari pengambilan jumlah nilai untuk *training* yaitu sesuai dengan jumlah dataset yang dimiliki. Dalam penelitian ini jumlah dataset untuk *training* yaitu sebesar 300 dengan tingkat keakurasian yang paling tinggi yaitu 90%. Namun sistem membutuhkan waktu lebih lama untuk bisa mendeteksi seluruh *image* yang diinput. Dengan hasil presentase yang didapatkan terbukti

bahwa dengan model jaringan CNN yang telah dibuat berhasil mendeteksi wajah yang menggunakan hijab dengan baik. Jadi semakin banyak data *training* yang dipakai, maka tingkat hasil akurasi akan semakin tinggi, karena komputer dapat lebih banyak memahami pola dari gambar yang dimasukkan. Sehingga ketepatan dalam proses pendeteksian akan semakin baik.

4. Jumlah nilai *learning rate* yang menghasilkan tingkat keakurasian yang tinggi yaitu 0,0001 dengan keakuratan 89%. Hal ini dikarenakan semakin kecil jumlah nilai *learning rate* maka nilai jumlah *loss* akan semakin rendah. Namun semakin kecil jumlah nilai *learning rate* yang digunakan, maka kecepatan waktu yang dibutuhkan juga akan semakin meningkat, karena sistem akan berusaha mengurangi nilai *loss* yang akan terjadi.

B. Saran

Dari hasil yang telah diuji dalam penelitian ini, peneliti memberikan beberapa saran untuk penelitian yang lebih lanjut dalam konteks yang sama, yaitu sebagai berikut:

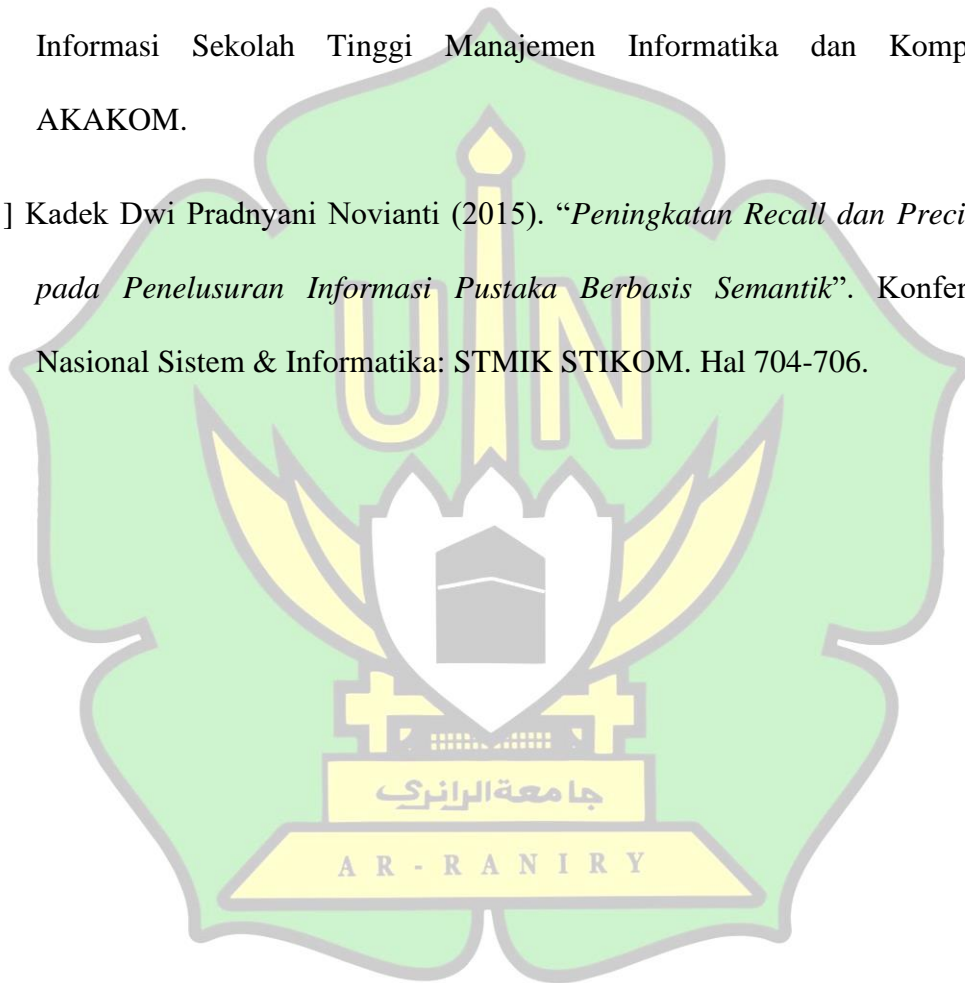
1. Menambahkan jumlah dataset dan menambahkan objek gambar lain untuk melatih model CNN yang ada serta dapat mencapai akurasi yang tinggi.
2. Menambahkan objek gambar dengan memakai atribut pada sekitar area wajah yang berbeda dan dengan menggunakan ekspresi yang lebih banyak.
3. Gambar yang telah berhasil di identifikasikan maka kedepannya harus dapat diimplementasikan kedalam perangkat keras, seperti Webcam dan CCTV.

DAFTAR PUSTAKA

- [1] Woodward, J; Horn; Gatune; Thomas (2003). *Biometrics: A look at facial recognition*, virgina State Crime Commision.
- [2] S. Ravi and D. Ph (2013). “A Study on Face Recognition Technique based on Eigenface” vol. 5, no. 4. Hal 57–62.
- [3] Shihab, M. Quraish (2004). *Jilbab: Pakaian Wanita Muslimah*. Bandung: Mizan Media Utama. Hal 10.
- [4] A. Krizhevsky, I. Sutskever, and G. E. Hinton (2012). “ImageNet Classification with Deep Convolutional Neural Networks” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc. Hal 1097–1105.
- [5] P. Ilmiah, A. Santoso (2018). “Implementasi Deep Learning Berbasis Keras”. P. S. Informatika, F. Komunikasi, D. A. N. Informatika, and U. M. Surakarta.
- [6] Royani Darma Nurfiti, Gunawan Ariyanto (2018). “Implementasi Deep Learning Berbasis Tensorflow Untuk Pengenalan Sidik Jari”. Program Studi Informatika Universitas Muhammadiyah Surakarta (UMS) Surakarta, Indonesia.
- [7] Muhammad Zufar dan Budi Setiyono (2016). “Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time”. *Jurnal SAINS DAN SENI ITS* Vol. 5 No. 2. Institut Teknologi Sepuluh Nopember (ITS).
- [8] Russa, Helder Filipe de Sausa (2017). “Computer Vision: Object Recognition With Deep Learning Applied to Fashion Items Detection in Images”. Tesis. Faculdade de Economia Universidade Do Porto.

- [9] Shahab, Husein (2013). “*Hijab Menurut Al-Quran dan Al-Sunnah*”. Bandung: Mizan Media Utama. Hal 105-110.
- [10] KBBI, 2019. Kamus Besar Bahasa Indonesia (KBBI). [Online] Available at: <https://kbbi.web.id/wajah> [Diakses 1 November 2019].
- [11] S. R. DEWI (2018). “*Deep Learning Object Detection Pada Video*”.
- [12] A. D. L. Tumuli, X. B. N. Najoan, and A. Sambul (2017). “*Implementasi Teknologi Biometrical Identification untuk Login Hotspot*” *J. Tek. Inform.*, vol. 12, no. 1. Hal 1–5.
- [13] D. Putra (2010). “*Pengolahan citra digital*”. Penerbit Andi. Hal 26-27.
- [14] Samuel, Arthur (1959). *Machine Vision*. McGraw-Hill.
- [15] Dobrev. 2004. *Artificial Intelligence*. Binus University.
- [16] Zhou, Shaohua Kevin, Rama Chellappa, and Wenyi Zhao (2006). “*Unconstrained face recognition in Machine Learning*”. Vol. 5. Springer Science & Business Media.
- [17] Deng, L & Yu, D (2014). *Deep Learning: Methods and Application, Foundations and Trends in Signal Processing*. Hal 27-25.
- [18] Nasichuddin, Moch Ari (2017). *Implementasi CNN untuk Klasifikasi Teks Menggunakan Tensorflow*. Diakses 29 Oktober 2019 dari <https://medium.com/@arynas92/implementasi-cnn-untuk-klasifikasi-teks-menggunakan-tensorflow-3a720cc3afbc>

- [19] Perkovic, Ljubomir (2012). *Introduction to Computing Using Python: An Application Development Focus*.
- [20] Taufiq, Imam (2018). “*Deep Learning Untuk Deteksi Tanda Nomor Kendaraan Bermotor Menggunakan Algoritma Convolutional Neural Network Dengan Python Dan Tensorflow*”. Skripsi. Program Studi Sistem Informasi Sekolah Tinggi Manajemen Informatika dan Komputer AKAKOM.
- [21] Kadek Dwi Pradnyani Novianti (2015). “*Peningkatan Recall dan Precision pada Penelusuran Informasi Pustaka Berbasis Semantik*”. Konferensi Nasional Sistem & Informatika: STMIK STIKOM. Hal 704-706.



Lampiran 1: Coding

Lampiran 1: Script LabelImg

```
# Callback functions:
def newShape(self):
    """Pop-up and give focus to the label editor.

    position MUST be in global coordinates.
    """
    if not self.useDefaultLabelCheckbox.isChecked() or not
self.defaultLabelTextLine.text():
        if len(self.labelHist) > 0:
            self.labelDialog = LabelDialog(
                parent=self, listItem=self.labelHist)

        # Sync single class mode from PR#106
        if self.singleClassMode.isChecked() and self.lastLabel:
            text = self.lastLabel
        else:
            text = self.labelDialog.popUp(text=self.prevLabelText)
            self.lastLabel = text
    else:
        text = self.defaultLabelTextLine.text()

    # Add Chris
    self.diffcButton.setChecked(False)
    if text is not None:
        self.prevLabelText = text
        generate_color = generateColorByText(text)
        shape = self.canvas.setLastLabel(text, generate_color,
generate_color)
        self.addLabel(shape)
        if self.beginner(): # Switch to edit mode.
            self.canvas.setEditing(True)
            self.actions.create.setEnabled(True)
        else:
            self.actions.editMode.setEnabled(True)
        self.setDirty()

        if text not in self.labelHist:
            self.labelHist.append(text)
    else:
        # self.canvas.undoLastLine()
        self.canvas.resetAllLines()

def scrollRequest(self, delta, orientation):
    units = - delta / (8 * 15)
    bar = self.scrollBars[orientation]
    bar.setValue(bar.value() + bar.singleStep() * units)

def setZoom(self, value):
    self.actions.fitWidth.setChecked(False)
    self.actions.fitWindow.setChecked(False)
    self.zoomMode = self.MANUAL_ZOOM
```



```

self.zoomWidget.setValue(value)

def addZoom(self, increment=10):
    self.setZoom(self.zoomWidget.value() + increment)

def zoomRequest(self, delta):
    # get the current scrollbar positions
    # calculate the percentages ~ coordinates
    h_bar = self.scrollBars[Qt.Horizontal]
    v_bar = self.scrollBars[Qt.Vertical]

    # get the current maximum, to know the difference after zooming
    h_bar_max = h_bar.maximum()
    v_bar_max = v_bar.maximum()

    # get the cursor position and canvas size
    # calculate the desired movement from 0 to 1
    # where 0 = move left
    #       1 = move right
    # up and down analogous
    cursor = QCursor()
    pos = cursor.pos()
    relative_pos = QWidget.mapFromGlobal(self, pos)

    cursor_x = relative_pos.x()
    cursor_y = relative_pos.y()

    w = self.scrollArea.width()
    h = self.scrollArea.height()

    # the scaling from 0 to 1 has some padding
    # you don't have to hit the very leftmost pixel for a maximum-left
movement
margin = 0.1
move_x = (cursor_x - margin * w) / (w - 2 * margin * w)
move_y = (cursor_y - margin * h) / (h - 2 * margin * h)

# clamp the values from 0 to 1
move_x = min(max(move_x, 0), 1)
move_y = min(max(move_y, 0), 1)

```

Lampiran 2: *Script* Konversi XML ke CSV

```
import os
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()
```

Lampiran 3: *Script* konversi CSV ke TFRecord

```
from __future__ import division
from __future__ import print_function
from __future__ import absolute_import
import os
import io
import pandas as pd
import tensorflow as tf
import sys
sys.path.append("C:\\Users\\ASUS\\TensorFlow\\skripsi\\models\\research"
)
sys.path.append("C:\\Users\\ASUS\\TensorFlow\\skripsi\\models\\research\\
object_detection")
from PIL import Image
from object_detection.utils import dataset_util
from collections import namedtuple, OrderedDict
flags = tf.app.flags
flags.DEFINE_string('type', '', 'Type of CSV input (train/test)')
flags.DEFINE_string('csv_input', '', 'Path to the CSV input')
flags.DEFINE_string('output_path', '', 'Path to output TFRecord')
FLAGS = flags.FLAGS
def class_text_to_int(row_label):
    if row_label == 'Id':
        return 1
    elif row_label=='Umur':
        return 2
    elif row_label == 'Jurusan':
        return 3
    elif row_label=='Hobi':
        return 4
    else:
        return None
def split(df, group):
    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in
zip(gb.groups.keys(), gb.groups)]
def create_tf_example(group, path):
    with tf.gfile.GFile(os.path.join(path, '{}'.format(group.filename)),
'rb') as fid:
        encoded_jpg = fid.read()
        encoded_jpg_io = io.BytesIO(encoded_jpg)
        image = Image.open(encoded_jpg_io)
        width, height = image.size
        filename = group.filename.encode('utf8')
        image_format = b'jpg'
        xmins = []
        xmaxs = []
        ymins = []
        ymaxs = []
        classes_text = []
        classes = []
        for index, row in group.object.iterrows():
            xmins.append(row['xmin'] / width)
```

```

xmaxs.append(row['xmax'] / width)
ymins.append(row['ymin'] / height)
ymaxs.append(row['ymax'] / height)
classes_text.append(row['class'].encode('utf8'))
classes.append(class_text_to_int(row['class']))
tf_example = tf.train.Example(features=tf.train.Features(feature={
    'image/height': dataset_util.int64_feature(height),
    'image/width': dataset_util.int64_feature(width),
    'image/filename': dataset_util.bytes_feature(filename),
    'image/source_id': dataset_util.bytes_feature(filename),
    'image/encoded': dataset_util.bytes_feature(encoded_jpg),
    'image/format': dataset_util.bytes_feature(image_format),
    'image/object/bbox/xmin':
dataset_util.float_list_feature(xmins),
    'image/object/bbox/xmax':
dataset_util.float_list_feature(xmaxs),
    'image/object/bbox/ymin':
dataset_util.float_list_feature(ymins),
    'image/object/bbox/ymax':
dataset_util.float_list_feature(ymaxs),
    'image/object/class/text':
dataset_util.bytes_list_feature(classes_text),
    'image/object/class/label':
dataset_util.int64_list_feature(classes),
}))
return tf_example
def main(_):
writer = tf.python_io.TFRecordWriter(FLAGS.output_path)
path = os.path.join(os.getcwd(), 'images/{}'.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))
if __name__ == '__main__':
tf.app.run()

```

Lampiran 4: Script Pipeline

```
{
  "cell_type": "code",
  "execution_count": 19,
  "metadata": {},
  "outputs": [],
  "source": [
    "ds = ds.apply(
tf.data.experimental.shuffle_and_repeat(buffer_size=len(image_path_list)
))\n",
    "ds =
ds.apply(tf.data.experimental.map_and_batch(map_func=load_and_preprocess
_from_path_label, batch_size=32))\n",
    "ds = ds.prefetch(buffer_size=tf.data.experimental.AUTOTUNE)"
  ]
},
{
  "cell_type": "code",
  "execution_count": 23,
  "metadata": {
    "scrolled": false
  },
  "outputs": [
    {
      "data": {
        "text/plain": [
          "<PrefetchDataset shapes: ((None, 192, 192, 3), (None,)), types:
(tf.float32, tf.int32)>"
        ]
      },
      "execution_count": 23,
      "metadata": {},
      "output_type": "execute_result"
    }
  ],
  "source": [
    "ds\n",
    "# image_label_ds = ds.map(load_and_preprocess_from_path_label,
num_parallel_calls=tf.data.experimental.AUTOTUNE)"
  ]
},
{
  "cell_type": "markdown",
  "metadata": {},
  "source": [
    "Now the data pipeline is ready. We can feed this pipeline to train
the model"
  ]
},
{
  "cell_type": "code",
  "execution_count": 25,
  "metadata": {
    "scrolled": false
  }
}
```

```

    },
    "outputs": [],
    "source": [
        "mobile_net = tf.keras.applications.MobileNetV2(input_shape=(192,
192, 3), include_top=False)\n",
        "mobile_net.trainable=False"
    ]
},
{
    "cell_type": "code",
    "execution_count": 53,
    "metadata": {},
    "outputs": [],
    "source": [
        "image_batch, label_batch = next(iter(ds))"
    ]
},
{
    "cell_type": "code",
    "execution_count": null,
    "metadata": {},
    "outputs": [],
    "source": [
        "feature_map_batch = mobile_net(image_batch)\n",
        "print(feature_map_batch.shape)"
    ]
},
{
    "cell_type": "code",
    "execution_count": 26,
    "metadata": {},
    "outputs": [],
    "source": [
        "model = tf.keras.Sequential([\n",
        "    mobile_net,\n",
        "    tf.keras.layers.GlobalAveragePooling2D(),\n",
        "    tf.keras.layers.Dense(len(label_names))])"
    ]
},
{
    "cell_type": "code",
    "execution_count": 27,
    "metadata": {},
    "outputs": [],
    "source": [
        "model.compile(optimizer=tf.keras.optimizers.Adam(),\n",
        "                loss='sparse_categorical_crossentropy',\n",
        "                metrics=[\"accuracy\"])"
    ]
},
{
    "cell_type": "code",
    "execution_count": 28,
    "metadata": {},

```

```

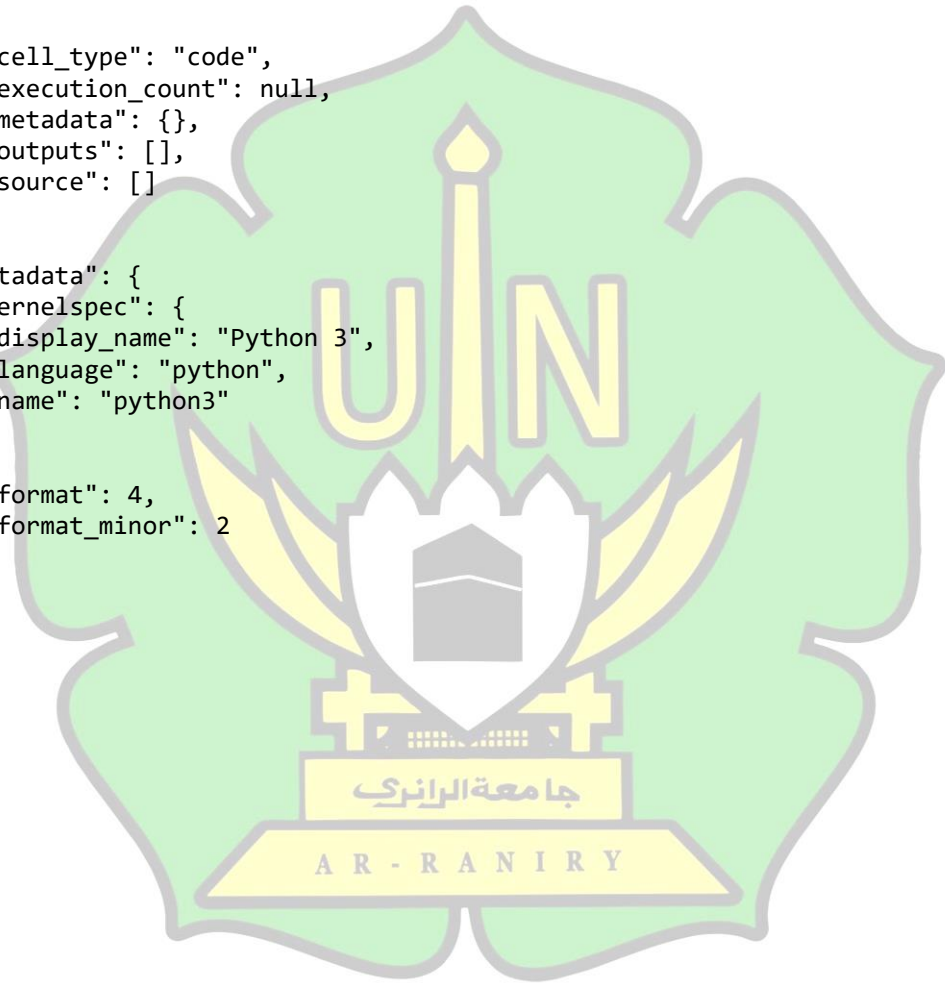
"outputs": [
  {
    "name": "stdout",
    "output_type": "stream",
    "text": [
      "Model: \"sequential_1\"\\n",
      "-----\\n",
      "Layer (type)                Output Shape                Param #
\\n",
      "-----\\n",
      "mobilenetv2_1.00_192 (Model) (None, 6, 6, 1280)        2257984
\\n",
      "-----\\n",
      "global_average_pooling2d_1 ( (None, 1280)                0
\\n",
      "-----\\n",
      "dense_1 (Dense)              (None, 5)                   6405
\\n",
      "-----\\n",
      "Total params: 2,264,389\\n",
      "Trainable params: 6,405\\n",
      "Non-trainable params: 2,257,984\\n",
      "-----\\n",
    ]
  }
],
"source": [
  "model.summary()"
],
},
{
  "cell_type": "code",
  "execution_count": 34,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        " 4/115 [>.....] - ETA: 3:00 - loss:
1.6094 - accuracy: 0.3047"
      ]
    },
    {
      "ename": "KeyboardInterrupt",
      "value": "",
      "output_type": "error",
      "traceback": [

```

```

-----\u001b[0m",
-----\u001b[0m",
-----\u001b[0;31mKeyboardInterrupt\u001b[0m
Traceback (most recent call last)",
]
}
],
"source": [
"model.fit(ds,epochs=1, steps_per_epoch=
tf.math.ceil(len(image_path_list)/32).numpy())"
],
{
"cell_type": "code",
"execution_count": null,
"metadata": {},
"outputs": [],
"source": []
}
],
"metadata": {
"kernel_spec": {
"display_name": "Python 3",
"language": "python",
"name": "python3"
}
},
"nbformat": 4,
"nbformat_minor": 2
}

```



Lampiran 5: Script Training Images

```
import functools
import json
import os
import tensorflow as tf

from object_detection_legacy import trainer
from object_detection.builders import input_reader_builder
from object_detection.builders import model_builder
from object_detection.utils import config_util

tf.logging.set_verbosity(tf.logging.INFO)

flags = tf.app.flags
flags.DEFINE_string('master', '', 'Name of the TensorFlow master to use.')
flags.DEFINE_integer('task', 0, 'task id')
flags.DEFINE_integer('num_clones', 1, 'Number of clones to deploy per worker.')
flags.DEFINE_boolean('clone_on_cpu', False,
                    'Force clones to be deployed on CPU. Note that even if '
                    'set to False (allowing ops to run on gpu), some ops may '
                    'still be run on the CPU if they have no GPU kernel.')
flags.DEFINE_integer('worker_replicas', 1, 'Number of worker+trainer '
                    'replicas.')
flags.DEFINE_integer('ps_tasks', 0,
                    'Number of parameter server tasks. If None, does not use '
                    'a parameter server.')
flags.DEFINE_string('train_dir', '',
                    'Directory to save the checkpoints and training summaries.')
flags.DEFINE_string('pipeline_config_path', '',
                    'Path to a pipeline_pb2.TrainEvalPipelineConfig config '
                    'file. If provided, other configs are ignored')
flags.DEFINE_string('train_config_path', '',
                    'Path to a train_pb2.TrainConfig config file.')
flags.DEFINE_string('input_config_path', '',
                    'Path to an input_reader_pb2.InputReader config file.')
flags.DEFINE_string('model_config_path', '',
                    'Path to a model_pb2.DetectionModel config file.')

FLAGS = flags.FLAGS

def main(_):
    assert FLAGS.train_dir, '`train_dir` is missing.'
```

```

if FLAGS.task == 0: tf.gfile.MakeDirs(FLAGS.train_dir)
if FLAGS.pipeline_config_path:
    configs = config_util.get_configs_from_pipeline_file(
        FLAGS.pipeline_config_path)
    if FLAGS.task == 0:
        tf.gfile.Copy(FLAGS.pipeline_config_path,
            os.path.join(FLAGS.train_dir, 'pipeline.config'),
            overwrite=True)
else:
    configs = config_util.get_configs_from_multiple_files(
        model_config_path=FLAGS.model_config_path,
        train_config_path=FLAGS.train_config_path,
        train_input_config_path=FLAGS.input_config_path)
    if FLAGS.task == 0:
        for name, config in [('model.config', FLAGS.model_config_path),
            ('train.config', FLAGS.train_config_path),
            ('input.config', FLAGS.input_config_path)]:
            tf.gfile.Copy(config, os.path.join(FLAGS.train_dir, name),
                overwrite=True)

model_config = configs['model']
train_config = configs['train_config']
input_config = configs['train_input_config']

model_fn = funtools.partial(
    model_builder.build,
    model_config=model_config,
    is_training=True)

create_input_dict_fn = funtools.partial(
    input_reader_builder.build, input_config)

env = json.loads(os.environ.get('TF_CONFIG', '{}'))
cluster_data = env.get('cluster', None)
cluster = tf.train.ClusterSpec(cluster_data) if cluster_data else None
task_data = env.get('task', None) or {'type': 'master', 'index': 0}
task_info = type('TaskSpec', (object,), task_data)

# Parameters for a single worker.
ps_tasks = 0
worker_replicas = 1
worker_job_name = 'lonely_worker'
task = 0
is_chief = True
master = ''

if cluster_data and 'worker' in cluster_data:
    # Number of total worker replicas include "worker"s and the
    "master".
    worker_replicas = len(cluster_data['worker']) + 1
if cluster_data and 'ps' in cluster_data:
    ps_tasks = len(cluster_data['ps'])

if worker_replicas > 1 and ps_tasks < 1:

```

```

    raise ValueError('At least 1 ps task is needed for distributed
training.')

if worker_replicas >= 1 and ps_tasks > 0:
    # Set up distributed training.
    server = tf.train.Server(tf.train.ClusterSpec(cluster),
protocol='grpc',
                                job_name=task_info.type,
                                task_index=task_info.index)
    if task_info.type == 'ps':
        server.join()
        return

    worker_job_name = '%s/task:%d' % (task_info.type, task_info.index)
    task = task_info.index
    is_chief = (task_info.type == 'master')
    master = server.target

    trainer.train(create_input_dict_fn, model_fn, train_config, master,
task,
                    FLAGS.num_clones, worker_replicas, FLAGS.clone_on_cpu,
ps_tasks,
                    worker_job_name, is_chief, FLAGS.train_dir)

if __name__ == '__main__':
    tf.app.run()

```



Lampiran 6: Script Export Graph Model

```
import tensorflow as tf
from tensorflow.core.protobuf import rewriter_config_pb2
from tensorflow.python import pywrap_tensorflow
from tensorflow.python.client import session
from tensorflow.python.framework import graph_util
from tensorflow.python.platform import gfile
from tensorflow.python.saved_model import signature_constants
from tensorflow.python.training import saver as saver_lib
from object_detection.builders import model_builder
from object_detection.core import standard_fields as fields
from object_detection.data_decoders import tf_example_decoder

slim = tf.contrib.slim

def freeze_graph_with_def_protos(
    input_graph_def,
    input_saver_def,
    input_checkpoint,
    output_node_names,
    clear_devices,
    initializer_nodes,
    optimize_graph=True,
    variable_names_blacklist=''):
    if not saver_lib.checkpoint_exists(input_checkpoint):
        raise ValueError(
            'Input checkpoint "' + input_checkpoint + '" does not exist!')

    if not output_node_names:
        raise ValueError(
            'You must supply the name of a node to --output_node_names.')
    if clear_devices:
        for node in input_graph_def.node:
            node.device = ''

    with tf.Graph().as_default():
        tf.import_graph_def(input_graph_def, name='')

        if optimize_graph:
            logging.info('Graph Rewriter optimizations enabled')
            rewrite_options = rewriter_config_pb2.RewriterConfig(
                layout_optimizer=rewriter_config_pb2.RewriterConfig.ON)
            rewrite_options.optimizers.append('pruning')
            rewrite_options.optimizers.append('constfold')
            rewrite_options.optimizers.append('layout')
            graph_options = tf.GraphOptions(
                rewrite_options=rewrite_options, infer_shapes=True)
        else:
            logging.info('Graph Rewriter optimizations disabled')
            graph_options = tf.GraphOptions()
        config = tf.ConfigProto(graph_options=graph_options)
    with session.Session(config=config) as sess:
        if input_saver_def:
            saver = saver_lib.Saver(saver_def=input_saver_def)
```

```

    saver.restore(sess, input_checkpoint)
else:
    var_list = {}
    reader = pywrap_tensorflow.NewCheckpointReader(input_checkpoint)
    var_to_shape_map = reader.get_variable_to_shape_map()
    for key in var_to_shape_map:
        try:
            tensor = sess.graph.get_tensor_by_name(key + ':0')
        except KeyError:
            continue
        var_list[key] = tensor
    saver = saver_lib.Saver(var_list=var_list)
    saver.restore(sess, input_checkpoint)
    if initializer_nodes:
        sess.run(initializer_nodes)

    variable_names_blacklist = (variable_names_blacklist.split(',') if
                                variable_names_blacklist else None)
    output_graph_def = graph_util.convert_variables_to_constants(
        sess,
        input_graph_def,
        output_node_names.split(','),
        variable_names_blacklist=variable_names_blacklist)

    return output_graph_def

def replace_variable_values_with_moving_averages(graph,
current_checkpoint_file,
new_checkpoint_file):

    with graph.as_default():
        variable_averages = tf.train.ExponentialMovingAverage(0.0)
        ema_variables_to_restore = variable_averages.variables_to_restore()
        with tf.Session() as sess:
            read_saver = tf.train.Saver(ema_variables_to_restore)
            read_saver.restore(sess, current_checkpoint_file)
            write_saver = tf.train.Saver()
            write_saver.save(sess, new_checkpoint_file)

def _image_tensor_input_placeholder(input_shape=None):
    """Returns input placeholder and a 4-D uint8 image tensor."""
    if input_shape is None:
        input_shape = (None, None, None, 3)
    input_tensor = tf.placeholder(
        dtype=tf.uint8, shape=input_shape, name='image_tensor')
    return input_tensor, input_tensor

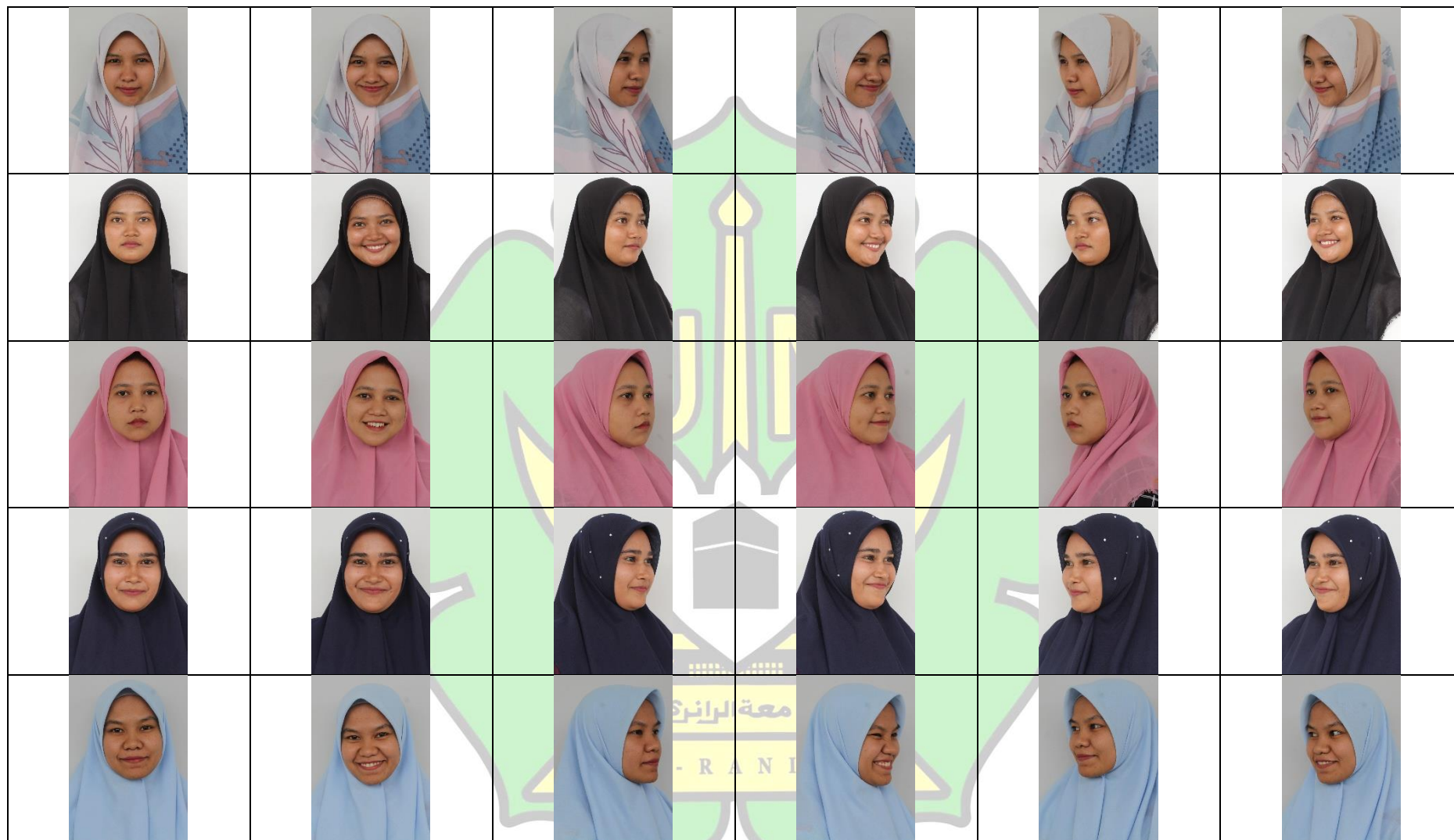
```

Lampiran 2: Dataset

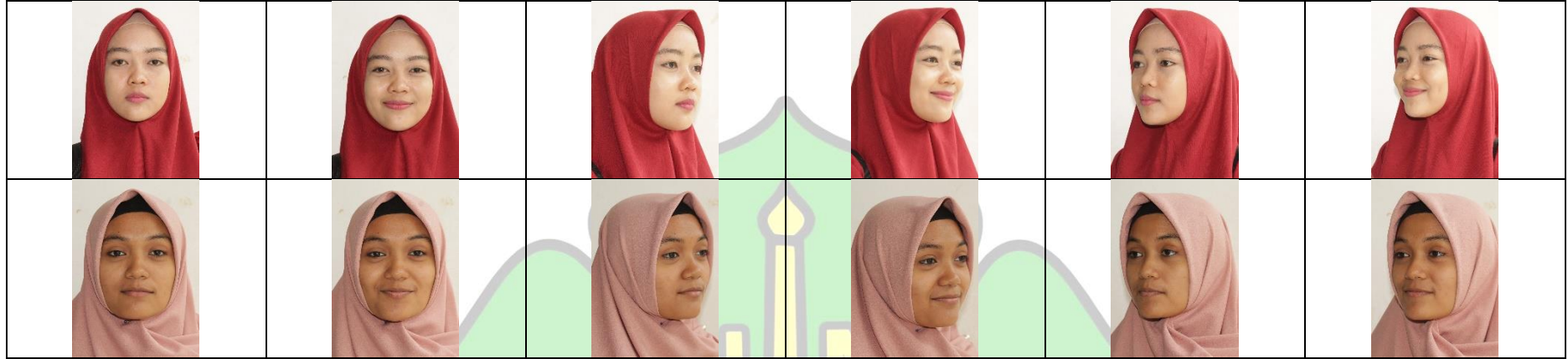
DATASET GAMBAR WAJAH BERHIJAB

Sisi Depan		Sisi Samping Kanan		Sisi Samping Kiri	
Flat/Datar	Senyum	Flat/Datar	Senyum	Flat/Datar	Senyum









Lampiran 3: SK Pengesahan Skripsi

85

**SURAT KEPUTUSAN DEKAN FTK UIN AR-RANIRY BANDA ACEH
NOMOR: B-17506/Un.08/FTK/KP.07.6/12/2019**

**TENTANG:
PENGANGKATAN PEMBIMBING SKRIPSI MAHASISWA FAKULTAS TARBIYAH DAN KEGURUAN
UIN AR-RANIRY BANDA ACEH
DEKAN FTK UIN AR-RANIRY BANDA ACEH**

Menimbang : a. bahwa untuk kelancaran bimbingan skripsi dan ujian munaqasyah mahasiswa pada Fakultas Tarbiyah dan Keguruan UIN Ar-Raniry Banda Aceh maka dipandang perlu menunjuk pembimbing skripsi tersebut yang dituangkan dalam Surat Keputusan Dekan;
b. bahwa saudara yang tersebut namanya dalam surat keputusan ini dipandang cakap dan memenuhi syarat untuk diangkat sebagai pembimbing skripsi.

Mengingat : 1. Undang-Undang Nomor 20 Tahun 2003, tentang Sistem Pendidikan Nasional;
2. Undang-Undang Nomor 14 Tahun 2005, tentang Guru dan Dosen;
3. Undang-Undang Nomor 12 Tahun 2012, tentang Sistem Pendidikan Tinggi;
4. Peraturan Pemerintah No. 74 Tahun 2012 tentang Perubahan atas Peraturan Pemerintah RI Nomor 23 Tahun 2005 tentang Pengelolaan Keuangan Badan Layanan Umum;
5. Peraturan Pemerintah Nomor 4 Tahun 2014, tentang Penyelenggaraan Pendidikan Tinggi dan Pengelolaan Perguruan Tinggi;
6. Peraturan Presiden Nomor 64 Tahun 2013, tentang Perubahan Institut Agama Islam Negeri Ar-Raniry Banda Aceh menjadi Universitas Islam Negeri Ar-Raniry Banda Aceh;
7. Peraturan Menteri Agama RI Nomor 12 Tahun 2014, tentang Organisasi & Tata Kerja UIN Ar-Raniry Banda Aceh;
8. Peraturan Menteri Agama RI Nomor 21 Tahun 2015, tentang Statuta UIN Ar-Raniry Banda Aceh;
9. Keputusan Menteri Agama Nomor 492 Tahun 2003, tentang Pendelegasian Wewenang Pengangkatan, Pemindahan, dan Pemberhentian PNS di Lingkungan Depag. RI;
10. Keputusan Menteri Keuangan Nomor 293/KMK.05/2011 tentang Penetapan Institut Agama Islam Negeri Ar-Raniry Banda Aceh pada Kementerian Agama sebagai Instansi Pemerintah yang Menerapkan Pengelolaan Badan Layanan Umum;
11. Keputusan Rektor UIN Ar-Raniry Nomor 01 Tahun 2015, tentang Pendelegasian Wewenang Kepada Dekan dan Direktur Pascasarjana di Lingkungan UIN Ar-Raniry Banda Aceh;

Memperhatikan : Keputusan Sidang Seminar Proposal Skripsi Prodi Pendidikan Teknologi Informasi tanggal 18 Desember 2019

MEMUTUSKAN

Menetapkan :

PERTAMA

Menunjuk Saudara:

1. Bustami, M.Sc sebagai pembimbing pertama
2. Zuhra Sofyan, M.Sc sebagai pembimbing kedua

Untuk membimbing skripsi :

Nama : Wulan Anggraini
NIM : 160212019
Program Studi : Pendidikan Teknologi Informasi
Judul Skripsi : Deep Learning untuk Deteksi Wajah yang Berhijab Menggunakan Algoritma Convolutional Neural Network (CNN) Dengan Tensorflow

KEDUA : Pembiayaan honorarium pembimbing pertama dan kedua tersebut di atas dibebankan pada DIPA UIN Ar-Raniry Banda Aceh Tahun 2019;

KETIGA : Surat Keputusan ini berlaku sampai akhir semester Ganjil Tahun Akademik 2020/2021

KEEMPAT : Surat Keputusan ini berlaku sejak tanggal ditetapkan dengan ketentuan bahwa segala sesuatu akan dirubah dan diperbaiki kembali sebagaimana mestinya, apabila kemudian hari ternyata terdapat kekeliruan dalam surat keputusan ini.

Ditetapkan di : Banda Aceh
Pada tanggal : 18 Desember 2019

Ah. Rektor
Dekan

Muslim Razali

Tembusan

1. Rektor UIN Ar-Raniry di Banda Aceh;
2. Ketua Prodi Pendidikan Teknologi Informasi;
3. Pembimbing yang bersangkutan untuk dimaklumi dan dilaksanakan;
4. Yang bersangkutan.