p-ISSN 2086-5805
e-ISSN 2808-2540

Klasifikasi Wajah Manusia Pada Gambar 360 Derajat (*Fish Eye*) Dengan Menggunakan *Tensorflow*

(Face Classification Of In 360 Degree Images (Fish Eye) Using Tensorflow)

Muhammad Fajar Sutejo[1], Arief Suryadi Satyawan[2], Sri Desy Siswanti[3]
[1] *Universitas Nurtanio*
E-mail: fajaar7@gmail.com
[2] *Badan Riset dan Inovasi Nasional*
E-mail: arief.suryadi@akane.waseda.jp
[3] *Universitas Nurtanio*
desy0712unnur@gmail.com

Abstract— ***Technology knows no boundaries, in fact it always shows new developments, one of which is classification in pictures. Human face classification is a method used to distinguish the characteristics of a person's facial pattern. The face classification system is an application that can find out a person's face according to the human face image that has been trained and stored in the machine's database. It is hoped that this application system can work well for classifying human faces in 360˚ image formats that have significant distortion. Classification of human faces can be done in various ways, one of which is the Convolutional Neural Network (CNN) method using Tensorflow. This final project is carried out using 5 classifications of human face datasets totaling 6600 images that have been trained with an image size of 180 x 180 using a 360˚ camera and the Python programming language. The classification of human faces in 360˚ (fish eye) images was successfully carried out with a percentage of 65% true detection and 35% false detection from the total 135 images that have been tested. In further research, other deep learning methods can be used to obtain better classification accuracy***

Keywords— ***Image Classification, Face Classification, Deep learning, Convolutional Neural Network, Tensorflow, Camera 360.***

Abstrak— ***Teknologi tidak mengenal batas, bahkan selalu menunjukkan perkembangan baru, salah satunya klasifikasi pada gambar. Klasifikasi wajah manusia merupakan cara yang digunakan untuk membedakan karakteristik pola wajah pada seseorang. Sistem klasifikasi wajah merupakan aplikasi yang dapat mengetahui wajah seseorang sesuai dengan gambar wajah manusia yang telah dilakukan training dan disimpan pada database mesin tersebut. Diharapkan sistem aplikasi ini mampu bekerja dengan baik untuk mengklasifikasi wajah manusia dalam format gambar 360˚ yang memiliki distorsi signifikan. Klasifikasi wajah manusia dapat dilakukan dengan berbagai cara, salah satunya dengan metode Convolutional Neural Network (CNN) menggunakan Tensorflow. Tugas akhir ini dilakukan menggunakan 5 klasifikasi dataset wajah manusia yang berjumlah 6600 gambar yang telah dilakukan training dengan ukuran gambar 180 x 180 menggunakan kamera 360˚ dan bahasa pemograman Python. Klasifikasi wajah manusia pada gambar 360˚ (fish eye) berhasil dilakukan dengan presentase 65% true detection dan 35% false detection dari jumlah 135 Gambar yang telah diujikan. Pada penelitian lebih lanjut dapat digunakan metode deep learning lainnya sehingga diperoleh akurasi klasifikasi yang lebih baik***

Kata Kunci— ***Image Classification, Face Classification, Deep learning, Convolutional Neural Network, Tensorflow, Camera 360.***

Penulis Korespondensi (Muhammad Fajar Sutejo)
E-mail: fajaar7@gmail.com

p-ISSN 2086-5805
e-ISSN 2808-2540

I. PENDAHULUAN

Teknologi merupakan salah satu bidang yang selalu menunjukkan perkembangan baru. Bahkan dapat dikatakan bahwa perkembangan teknologi seperti ini tidak mengenal batas. Terbukti, penemuan teknologi baru dari berbagai jenis muncul dari waktu ke waktu dan menjadi lebih canggih, salah satunya klasifikasi gambar. Klasifikasi gambar merupakan salah satu metode *machine learning* / *artificial intelligence* yang dapat digunakan untuk mendeteksi sebuah gambar dengan cepat. Klasifikasi wajah manusia merupakan cara yang digunakan untuk membedakan karakteristik pola wajah pada seseorang. Klasifikasi wajah manusia dapat dilakukan dengan berbagai cara, salah satunya dengan metode *Convolutional Neural Network* (CNN) menggunakan *tensorflow*.

Pada penelitian ini akan dibuat konstruksi CNN untuk aplikasi klasifikasi wajah manusia pada gambar 360˚ (fish eye) dengan menggunakan *tensorflow*. Model algoritmanya akan diimplementasikan pada komputer dan informasi gambar didapatkan dari kamera 360˚. Diharapkan sistem aplikasi ini mampu bekerja dengan baik untuk mengklasifikasi wajah manusia dalam format gambar 360 ˚ yang memiliki distorsi signifikan

II. LANDASAN TEORI

Convolutional Neural Network adalah salah satu algoritma *Deep Learning* yang merupakan pengembangan dari *Multilayer Perceptron* (MLP) yang dirancang untuk mengolah data dalam bentuk dua dimensi, misalnya gambar atau suara. CNN dibuat dengan prinsip translation invariance yaitu dapat mengenali objek dalam citra pada berbagai macam posisi yang mungkin. Terdapat 2000 citra daun yang diklasifikasi menggunakan *Alexnet*. *Alexnet* merupakan arsitektur CNN milik Krizhevsky yang memiliki delapan layer ekstraksi fitur. Layer tersebut terdiri dari lima layer konvolusi dan tiga *pooling layer*. Dalam layer klasifikasinya, Alexnet mempunyai dua *layer Fully Connected* yang masing-masing mempunyai 4096 neuron. Pada akhir layer terdapat pengklasifikasian kedalam 20 kategori menggunakan aktifasi *softmax*. Rata-rata akurasi dari hasil klasifikasi mencapai 85%. Sedangkan akurasi dari identifikasi berhasil mencapai 90% yang didapatkan dari pengujian 40 citra[1].

Pada penelitian ini, dilakukan pengujian dengan mengklasifikasikan gambar wajah yang menggunakan masker dan tanpa menggunakan masker dengan menggunakan algoritma *Convolutional Neural Network* (CNN). Hasil penelitian ini menunjukkan model yang dibangun dapat melakukan klasifikasi gambar dengan tingkat akurasi mencapai 96%. Pengujian pada gambar wajah yang menggunakan masker memperoleh nilai *precision* 98%, *recall* 94% dan gambar wajah yang tidak menggunakan masker memperoleh nilai *precision* 94%, *recall* 98[2].

A. Artificial Intelligent

Artificial Intelligence (AI) atau kecerdasan buatan adalah ilmu dan rekayasa pembuatan mesin cerdas, melibatkan mekanisme menggunakan komputer untuk menjalankan suatu tugas. Artificial intelligent bekerja berdasarkan dengan algoritma pemrograman yang diberikan dalam proses pembuatannya pada sistem komputer. Algoritma pemrograman kerangka berpikir berdasarkan artificial intelligent dalam memproses berbagai jenis data.

B. Deep Learning

Deep learning merupakan salah satu jenis algoritma jaringan saraf tiruan yang menggunakan metadata sebagai masukan dan memprosesnya menggunakan sejumlah lapisan tersembunyi (hidden layer) transformasi non linier dari data masukan untuk menghitung nilai luaran. Algortima pada deep learning memiliki fitur yang unik yaitu sebuah fitur yang mampu mengekstraksi secara otomatis. Hal ini berarti algoritma yang dimilikinya secara otomatis dapat menangkap fitur yang relevan sebagai keperluan dalam pemecahan suatu masalah. Algortima semacam ini sangat penting dalam sebuah kecerdasan buatan karena dapat mengurangi beban pemrograman dalam memilih fitur yang eksplisit. Algortima ini dapat digunakan untuk

memecahkan permasalahan yang perlu pengawasan (supervised), tanpa pengawasan (unsupervised), dan semi terawasi (semi supervised).

C. Convolutional Neural Network

Pada dasarnya CNN merupakan sebuah arsitektur ANN yang memiliki karakteristik khusus, dan dapat dilatih serta memiliki beberapa tahapan yang terdiri dari masukan (input) dan keluaran (output). Masukan dan keluaran pada setiap tahapan terdiri dari beberapa array yang berfungsi untuk menghasilkan *feature map*. Setiap tahap terdiri dari tiga proses utama yaitu konvolusi, fungsi aktivasi dan *pooling.*

III. METODE/MODEL YANG DIUSULKAN

A. Metodologi Penelitian

Metodologi penelitian ini melakukan penelitian secara sistematis yang dimulai dari tahapan studi literatur sampai pembuatan laporan tugas akhir ini. Tahapan, metode, dan hasil penelitian akan diperlihatkan pada Tabel 1 dibawah ini.

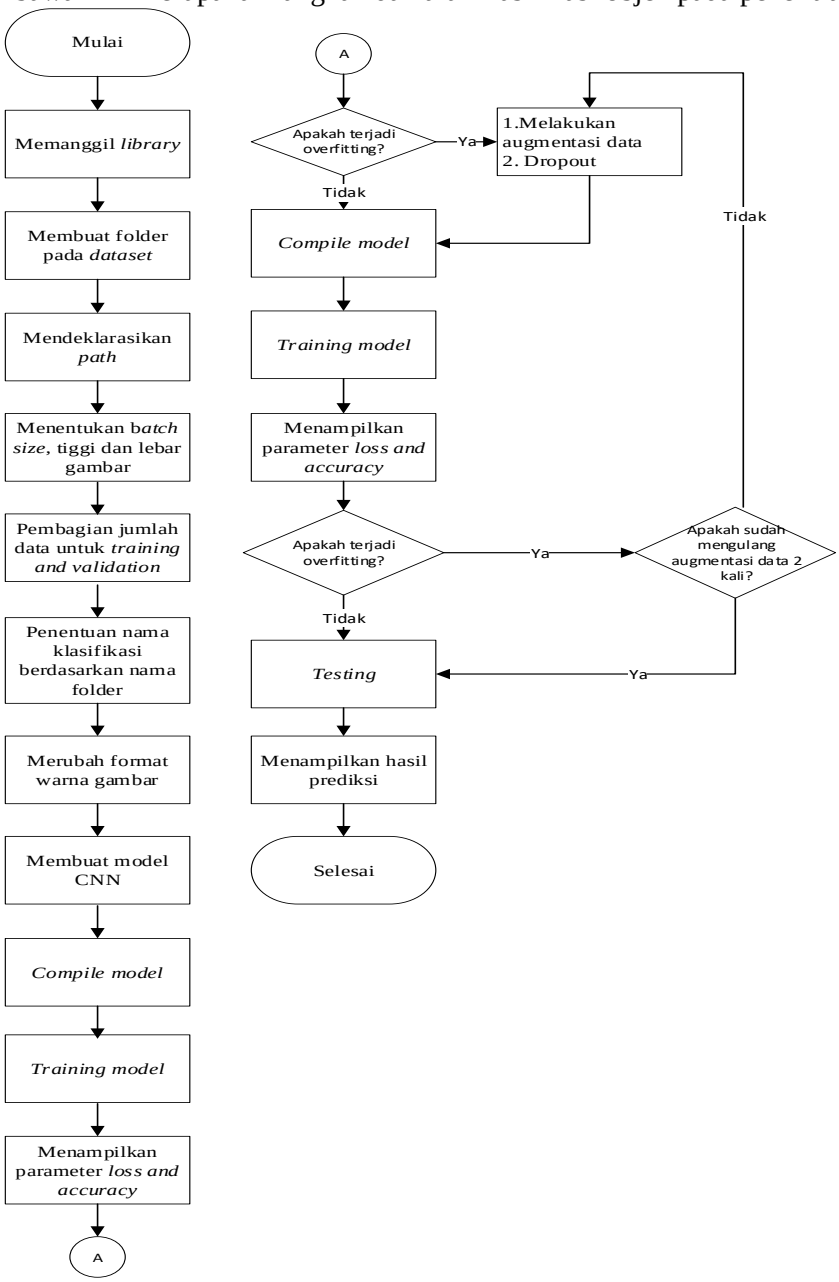
Tabel 1. Tahapan Penelitian

No	Tahapan	Metode	Hasil
1.	Studi kepustakaan	1. Peninjauan internet 2. Studi buku serta jurnal terkait	Pemahaman: 1. Mengenai teknologi *Convolutional Neural Network* (CNN) dan pemrosesannya 2. Pemrograman dengan bahasa *Python* 3. *Hardware* dan *software* yang digunakan
2.	Desain perangkat lunak klasifikasi jenis tas pada gambar 360˚	Menggunakan *software* pemograman *python*	Desain *software* pengolahan data pada CNN
3.	Pengumpulan *dataset*	1. *Dataset* diambil menggunakan kamera 360º 2. Mengubah video ke *frame* 3. Membuat klasifikasi data	Dataset segmentasi dengan jumlah 6600 *frame* siap uji dengan keterangan klasifikasi
4.	Realisasi perangkat lunak pada PC	Memakai *python* sebagai *software* pada pemrogramannya	*Software* menerima dan menyimpan data
5.	Pengukuran kinerja *software* melakukan analisis	1. Pengukuran skala laboratorium 2. Analisis kuantitatif 3. Analisis kualitatif	Kinerja *software* dalam melatih dan menguji data, serta kesimpulan penelitian

p-ISSN 2086-5805
e-ISSN 2808-2540

B. Flowchart

Gambar 1. di bawah ini merupakan langkah dan alur klasifikasi objek pada penelitian ini.

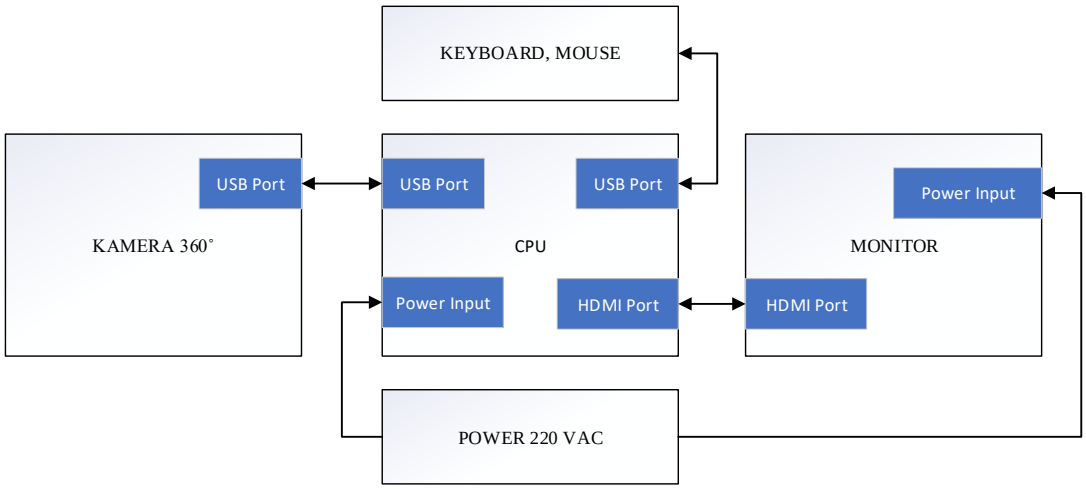


Gambar1. *Flowchart* Sistem Klasifikasi Wajah

IV. HASIL/IMPLEMENTASI MODEL DAN PEMBAHASAN

Bagian keempat ini menjelaskan perihal desain hingga pengaplikasian klasifikasi wajah manusia secara rinci dengan arsitektur CNN menggunakan gambar dari kamera 360^{o}.

A. Desain Model



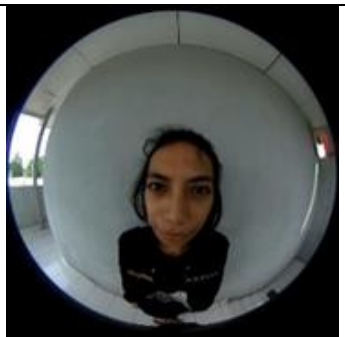
Gambar 2. Konstruksi *Hardware*

Gambar 1 diatas menunjukan desain gambar konstruksi *hardware* yang digunakan pada klasifikasi wajah manusia pada gambar 360^{o} (*fish eye*) dengan menggunakan *Tensorflow*.

B. Pembahasan

Tahapan ini menjelaskan tahapan analisis pada klasifikasi wajah manusia pada gambar 360^{o} (*fish eye*) dengan menggunakan *Tensorflow*. Pada pengujian ini menggunakan 5 klasifikasi dataset dengan total 6600 gambar yang dapat dilihat pada Tabel II.

Tabel 2. Klasifikasi Dataset

No	Nama Klasifikasi	Jumlah Data	Gambar
1.	*Agnes_photos*	1.320	
2	*Aris_photos*	1.350	

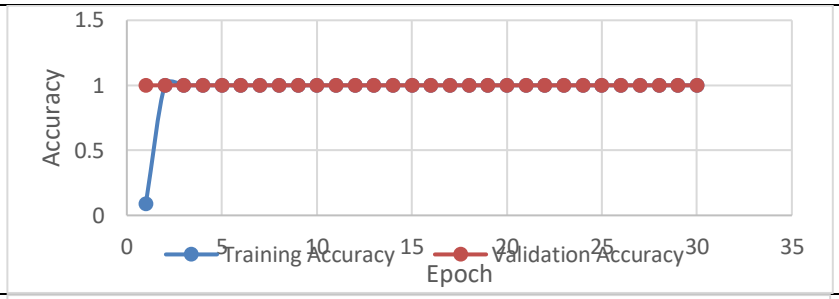
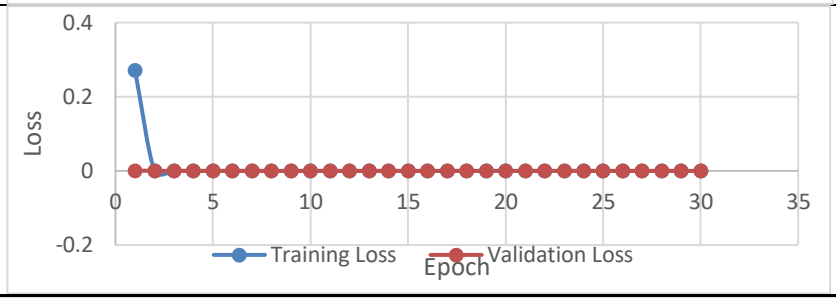
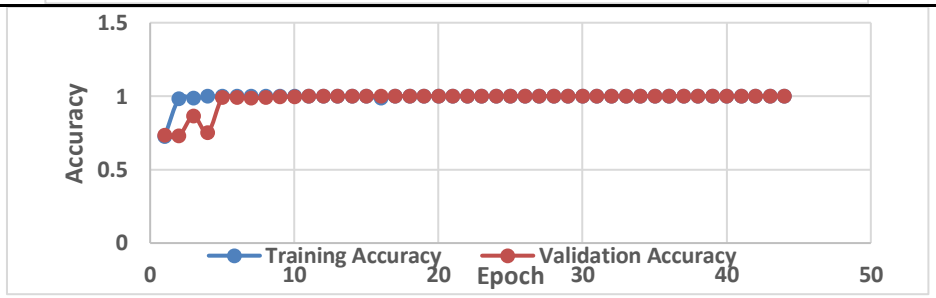
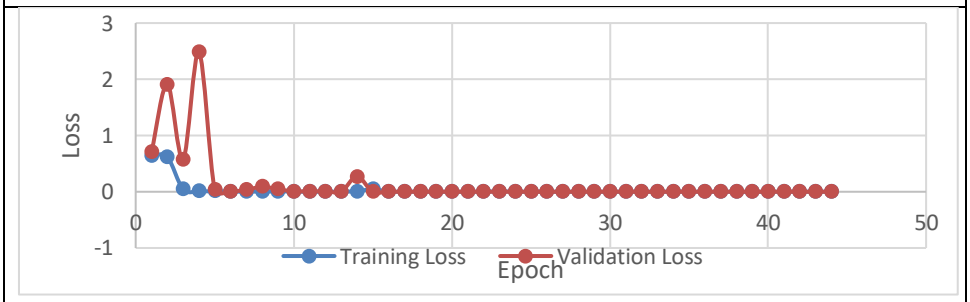
p-ISSN 2086-5805
e-ISSN 2808-2540

No	Nama Klasifikasi	Jumlah Data	Gambar
3.	*Eko_photos*	1.320	
4.	*Tejo_photos*	1.260	
5.	*Yucha_photos*	1.350	

Tabel 2 menunjukan *dataset* yang diambil untuk nantinya akan diuji pada proses *training dataset*. *Output* dari *training dataset* berupa gambar yang telah diuji apakah gambar tersebut tepat atau tidak sesuai hasil *percent confidence*. Dalam hal ini terdapat *training and validation.* Dimana *Training dataset* merupakan himpunan data yang digunakan untuk melatih atau membangun model. Kemudian, *validation dataset* merupakan himpunan data yang digunakan untuk mengoptimasi saat melatih model. Model dilatih menggunakan *training dataset*, kemudian kinerja saat latihan tersebut diuji menggunakan *validation dataset*. Hal ini bertujuan untuk melihat kemampuan model pada saat *training* apakah dapat mengenal pola secara umum. Pengujian ini dibagi menjadi 3, berdasarkan epoch yang digunakan yaitu, epoch (10, 15) (20, 30) (30, 45) serta *training and validation* 0.1 (90% Training, 10% Validation), 0.2 (80% Training, 20% Validation), 0.3 (70% Training, 30% Validation) dan menguji gambar. Pengujian tersebut dipilih satu yang paling tepat dan akan dijelaskan dibawah ini.

Pengujian ini menggunakan epoch (30, 45) dan hasil dari pembagian dataset menggunakan *training and validation* 0.2 (80% Training, 20% Validation), serta menguji 15 gambar, yang ditunjukan Tabel III

Tabel 3. Hasil *Training and Validation* 0.2 epoch (30, 45)

Epoch	Hasil
Pertama (30)	 
Kedua (45)	 

Pada Tabel 3 dengan menggunakan *epoch* yang dinaikkan 3 kali lipat dari default(10,15) menjadi (30,45) dengan *training* and *validation* 0.2 didapatkan hasil pengujian *epoch* pertama(30) tidak terjadi fluktuasi yang signifikan namun pada *epoch* kedua(45) terjadi fluktuasi. Pada pengujian ini menggunakan 15 gambar uji yaitu 6 gambar uji menggunakan dataset baru, 6 gambar uji menggunakan *dataset* yang di-*training* dan 3 gambar uji diluar klasifikasi. Pada Tabel 4 diperlihatkan hasil pengujian tersebut.

p-ISSN 2086-5805
e-ISSN 2808-2540

Tabel 4. Hasil Uji Prediksi

No	Gambar yang diujikan	Hasil prediksi	Ket

1.		`photos_path = "C:/Users/lipi/.keras/datasets/pengujian/new/agnes_out_cut_63.jpg"`
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		`1/1 [==============================] - 0s 15ms/step`
		`This image most likely belongs to agnes_photos with a 100.00 percent confidence.`
			True

2.		`photos_path = "C:/file_tejo/pengujian/new/aris_in_cut_65.jpg"`
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		`1/1 [==============================] - 0s 14ms/step`
		`This image most likely belongs to aris_photos with a 100.00 percent confidence.`
			True

3		`photos_path = "C:/file_tejo/pengujian/new/eko_out_cut_45.jpg"`
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		`1/1 [==============================] - 0s 13ms/step`
		`This image most likely belongs to aris_photos with a 100.00 percent confidence.`
			False

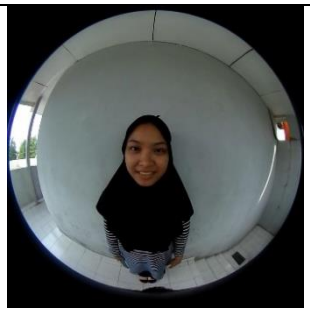
4		`photos_path = "C:/file_tejo/pengujian/new/tejo_out_cut_25.jpg"`
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		`1/1 [==============================] - 0s 15ms/step`
		`This image most likely belongs to tejo_photos with a 100.00 percent confidence.`
			True

5		`photos_path = "C:/file_tejo/pengujian/new/tejo_out_cut_45.jpg"`
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		`1/1 [==============================] - 0s 13ms/step`
		`This image most likely belongs to tejo_photos with a 100.00 percent confidence.`
			True

6.		photos_path = "C:/file_tejo/pengujian/new/yucha_out_cut_64.jpg" img = tf.keras.utils.load_img(photos_path, target_size=(img_height, img_width)) img_array = tf.keras.utils.img_to_array(img) img_array = tf.expand_dims(img_array, 0) # Create a batch predictions = model.predict(img_array) score = tf.nn.softmax(predictions[0]) print("This image most likely belongs to {} with a {:.2f} percent confidence." .format(class_names[np.argmax(score)], 100* np.max(score))) 1/1 [==============================] - 0s 13ms/step This image most likely belongs to aris_photos with a 100.00 percent confidence.	*False*
7		photos_path = "C:/file_tejo/pengujian/old/agnes_cut_364.jpg" img = tf.keras.utils.load_img(photos_path, target_size=(img_height, img_width)) img_array = tf.keras.utils.img_to_array(img) img_array = tf.expand_dims(img_array, 0) # Create a batch predictions = model.predict(img_array) score = tf.nn.softmax(predictions[0]) print("This image most likely belongs to {} with a {:.2f} percent confidence." .format(class_names[np.argmax(score)], 100* np.max(score))) 1/1 [==============================] - 0s 14ms/step This image most likely belongs to agnes_photos with a 100.00 percent confidence.	*True*
8		photos_path = "C:/file_tejo/pengujian/old/aris_cut_123.jpg" img = tf.keras.utils.load_img(photos_path, target_size=(img_height, img_width)) img_array = tf.keras.utils.img_to_array(img) img_array = tf.expand_dims(img_array, 0) # Create a batch predictions = model.predict(img_array) score = tf.nn.softmax(predictions[0]) print("This image most likely belongs to {} with a {:.2f} percent confidence." .format(class_names[np.argmax(score)], 100* np.max(score))) 1/1 [==============================] - 0s 13ms/step This image most likely belongs to aris_photos with a 100.00 percent confidence.	*True*
9		photos_path = "C:/file_tejo/pengujian/old/aris_cut_672.jpg" img = tf.keras.utils.load_img(photos_path, target_size=(img_height, img_width)) img_array = tf.keras.utils.img_to_array(img) img_array = tf.expand_dims(img_array, 0) # Create a batch predictions = model.predict(img_array) score = tf.nn.softmax(predictions[0]) print("This image most likely belongs to {} with a {:.2f} percent confidence." .format(class_names[np.argmax(score)], 100* np.max(score))) 1/1 [==============================] - 0s 13ms/step This image most likely belongs to aris_photos with a 100.00 percent confidence.	*True*
10		photos_path = "C:/file_tejo/pengujian/old/eko_cut_73.jpg" img = tf.keras.utils.load_img(photos_path, target_size=(img_height, img_width)) img_array = tf.keras.utils.img_to_array(img) img_array = tf.expand_dims(img_array, 0) # Create a batch predictions = model.predict(img_array) score = tf.nn.softmax(predictions[0]) print("This image most likely belongs to {} with a {:.2f} percent confidence." .format(class_names[np.argmax(score)], 100* np.max(score))) 1/1 [==============================] - 0s 13ms/step This image most likely belongs to eko_photos with a 100.00 percent confidence.	*True*

p-ISSN 2086-5805
e-ISSN 2808-2540

11		`photos_path = "C:/file_tejo/pengujian/old/tejo_cut_606.jpg"`
		
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		
		`1/1 [==============================] - 0s 13ms/step`
		`This image most likely belongs to tejo_photos with a 100.00 percent confidence.`
			True

12		`photos_path = "C:/file_tejo/pengujian/old/yucha_cut_768.jpg"`
		
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		
		`1/1 [==============================] - 0s 13ms/step`
		`This image most likely belongs to yucha_photos with a 100.00 percent confidence.`
			True

13		`photos_path = "C:/file_tejo/pengujian/other/aldam_in_cut_42.jpg"`
		
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		
		`1/1 [==============================] - 0s 13ms/step`
		`This image most likely belongs to aris_photos with a 100.00 percent confidence.`
			False

14		`photos_path = "C:/file_tejo/pengujian/other/tri_out_cut_37.jpg"`
		
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		
		`1/1 [==============================] - 0s 13ms/step`
		`This image most likely belongs to aris_photos with a 65.40 percent confidence.`
			False

15		`photos_path = "C:/file_tejo/pengujian/other/tri_out_cut_75.jpg"`
		
		`img = tf.keras.utils.load_img(`
		`  photos_path, target_size=(img_height, img_width)`
		`)`
		`img_array = tf.keras.utils.img_to_array(img)`
		`img_array = tf.expand_dims(img_array, 0) # Create a batch`
		
		`predictions = model.predict(img_array)`
		`score = tf.nn.softmax(predictions[0])`
		
		`print(`
		`  "This image most likely belongs to {} with a {:.2f} percent confidence."`
		`  .format(class_names[np.argmax(score)], 100* np.max(score))`
		`)`
		
		`1/1 [==============================] - 0s 14ms/step`
		`This image most likely belongs to aris_photos with a 100.00 percent confidence.`
			False

Pada Tabel 4 di atas hasil pengujian dengan menggunakan *epoch* (30, 45) dengan *training and validation* 0.2 didapatkan hasil pengujian *epoch* pertama (30) tidak terjadi fluktuasi yang signifikan namun pada *epoch* kedua (45) terjadi fluktuasi. Pengujian prediksi gambar pada

training and validation 0.2 dari 15 gambar yang diuji terdapat 10 gambar yang menunjukkan klasifikasi tepat dan 5 gambar menunjukkan klasifikasi tidak tepat.

V. KESIMPULAN

Bagian kelima ini adalah kesimpulan dari klasifikasi wajah manusia pada gambar 360^o (*fish eye*) dengan menggunakan *tensorflow*. Adapun hasilnya dapat disimpulkan sebagai berikut :

1. Klasifikasi wajah berhasil dilakukan dengan presentase 65% *true detection* dan 35% *false detection* dari jumlah 135 Gambar yang telah diujikan.
2. Jumlah *dataset* yang lebih banyak, lingkungan dalam pengambilan dataset dan parameter deep learning dapat meningkatkan keakurasian dalam proses klasifikasi.
3. Proses dengan jumlah *epoch* yang sedikit tidak membutuhkan waktu yang lama, sementara proses dengan jumlah *epoch* yang banyak membutuhkan waktu yang cukup lama dan menghasilkan klasifikasi yang lebih baik.

UCAPAN TERIMA KASIH

Ucapan terima kasih kami sampaikan kepada pihak pihak yang telah berkontribusi dalam penelitian ini hingga makalah ini dapat di publikasikan pada Seminar Nasional Sains Teknologi dan Inovasi Indonesia 2022, diantaranya kepada:

1. LPDP (Lembaga Pengelola Dana Pendidikan) sebagai sponsor penelitian.
2. BRIN (Badan Riset dan Inovasi Nasional) sebagai penyedia prasarana penelitian.
3. Universitas Nurtanio sebagai penyedia prasarana penelitian.
4. AAU (Akademi Angkatan Udara) sebagai penyelenggara konferensi dan prosiding.
5. Rekan rekan –rekan peneliti

REFERENSI

[1] Ilahiyah, S., & Nilogiri, A. (2018). Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. JUSTINDO (Jurnal Sistem Dan Teknologi Informasi Indonesia), 3(2), 49-56.

[2] Rahim, A., Kusrini, K., & Luthfi, E. T. (2020). Convolutional Neural Network untuk Kalasifikasi Penggunaan Masker. Inspiration: Jurnal Teknologi Informasi dan Komunikasi, 10(2), 109-115.

[3] Salawazo, V. M. P., Gea, D. P. J., Gea, R. F., & Azmi, F. (2019). Implementasi Metode Convolutional Neural Network (CNN) Pada Peneganalan Objek Video Cctv. Jurnal Mantik Penusa, 3(1.1).

[4] Fadlia, N., & Kosasih, R. (2020). Klasifikasi Jenis Kendaraan Menggunakan Metode Convolutional Neural Network (Cnn). Jurnal Ilmiah Teknologi dan Rekayasa, 24(3), 207-215.

[5] [5] Yusuf, A., Wihandika, R. C., & Dewi, C. (2019). Klasifikasi Emosi Berdasarkan Ciri Wajah Menggunakan Convolutional Neural Network. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN, 2548, 964X.