BAB III
METODE PENELITIAN

3.1 ALAT YANG DIGUNKAN

Pada penelitian ini alat yang digunakan yaitu berupa perangkat keras dan perangkat lunak yang dapat mendukung dan memudahkan proses penelitian.

3.1.1 Hardware

1. *Processor AMD Ryzen 3 3250U with Radeon Graphich 2.60Hz*
2. *Operating system Windows 10*
3. *RAM 8.00 GB*
4. *System type 64-bit*
5. *Lux meter*

3.1.2 Software

1. *Google Colaboratory*

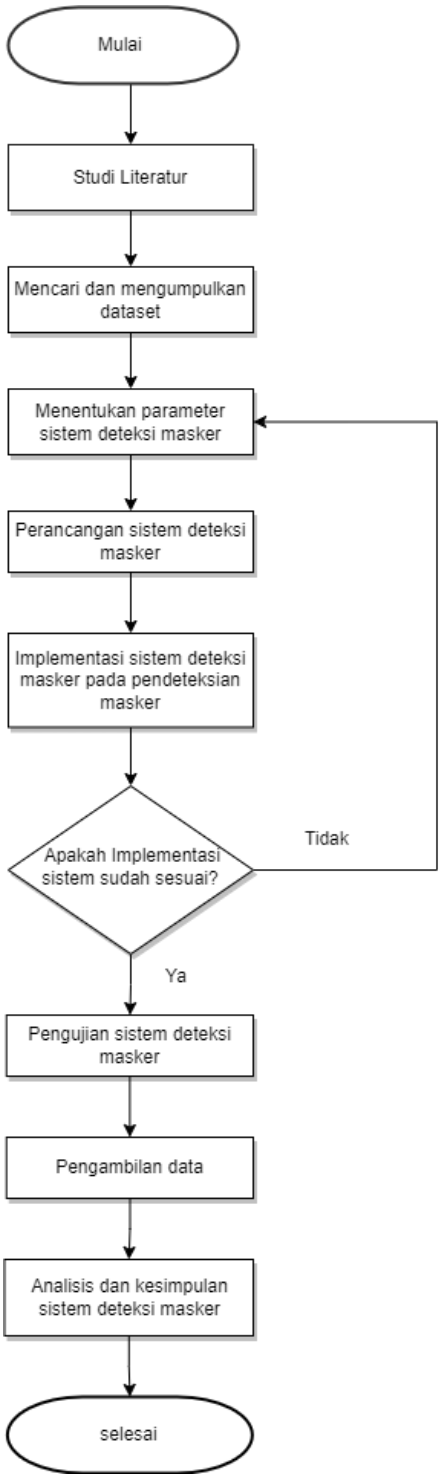
Google colaboratory atau biasa disebut *google colab* merupakan perangkat komputasi yang dibuat oleh *google* berbasis *Jupyter notebook* atau *iPython* (*interactive python*) yang termasuk kedalam kategori *high-level programming language*. Pemograman bahasa *python* mudah untuk diaplikasikan, bersifat *extendible* (dapat dikembangkan) serta dilengkapi beberapa modul bawaan atau *library* seperti *Numpy*, *OpenCV*, *Tensorflow* dan *keras*. *Google colab* menyediakan tiga jenis prosesor untuk keperluan pembelajaran mesin seperti *Central Processing Unit* (CPU), *Graphic Processing Unit* (GPU) dan *Tensor Processing Unit* (TPU).

2. *Google Drive*

Google Drive merupakan media penyimpanan *online* (*cloud storage*) yang dimiliki *Google* dengan menyediakan ruang penyimpanan hingga 15GB secara gratis. Penggunaan *Google drive* sebagai *backup* data, *upload* data dan sinkronisasi dengan perangkat lain serta *Google drive* akan saling terhubung atau terintegrasi dengan *Google colab* untuk penyimpanan *source code* yang kemudian dapat memanggil dataset yang tersimpan pada *Google drive*.

3.2 ALUR PENELITIAN

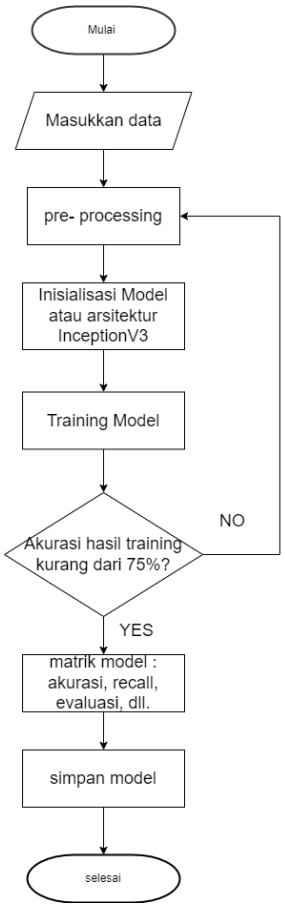
Penelitian ini akan dilakukan dalam beberapa tahap, berikut *flowchart* dari penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Flowchart penelitian

3.3 ALUR SIMULASI

Alur simulasi merupakan penjelasan mengenai prosedur yang akan dilakukan dalam pembuatan sistem yang akan diuji dalam penelitian. Berikut merupakan alur simulasi dari penelitian dapat dilihat pada Gambar 3.2.

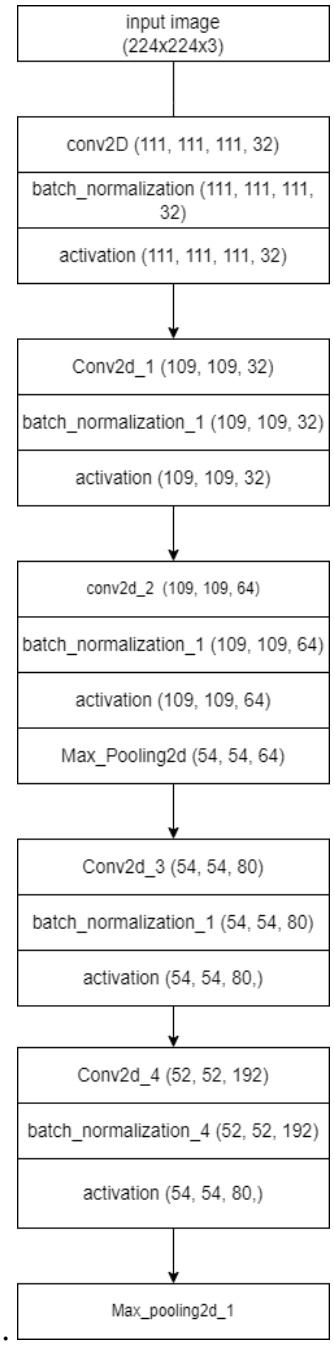


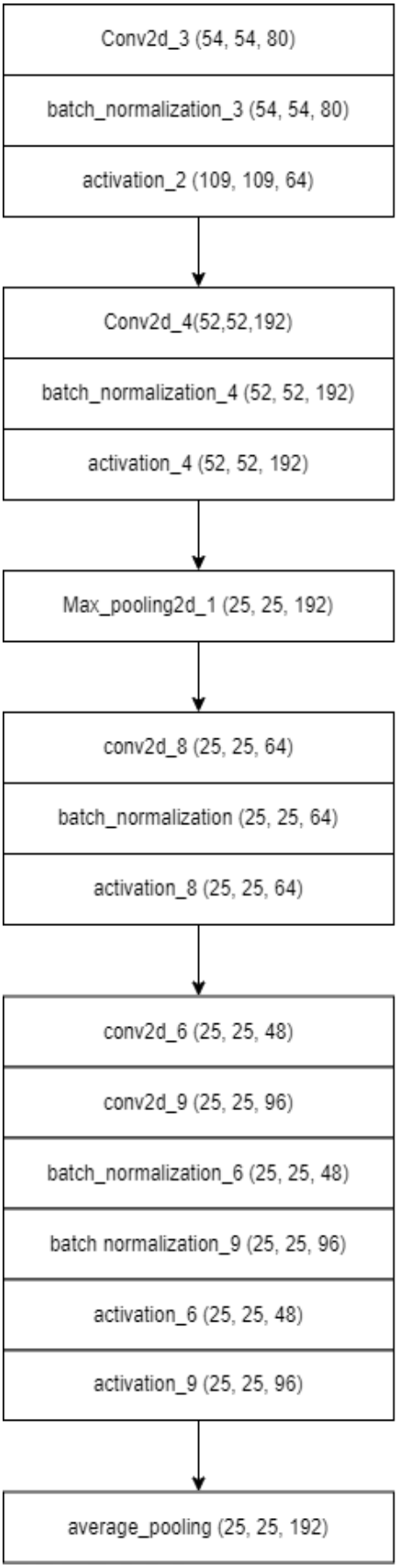
Gambar 3.2 Alur simulasi

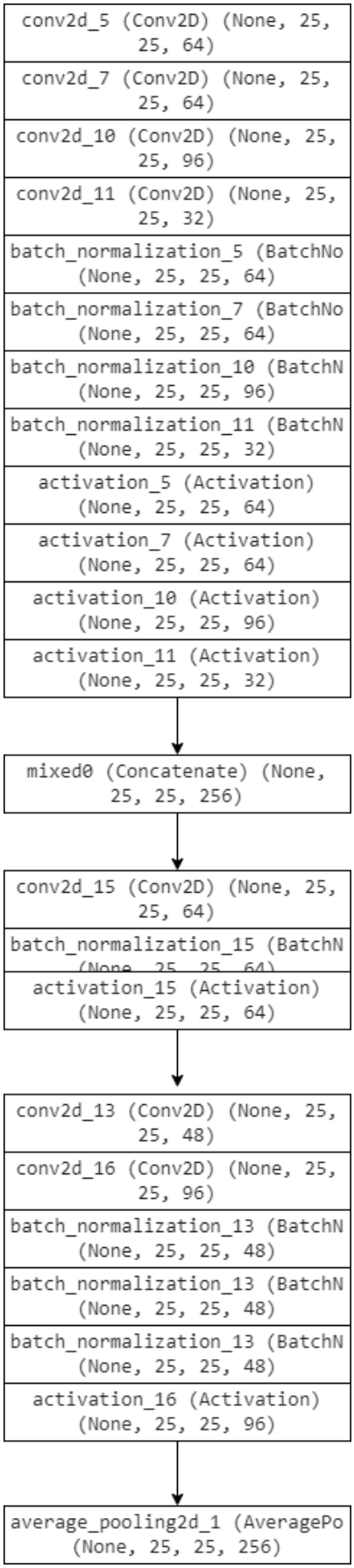
Alur simulasi dilakukan dengan beberapa tahap, tahap pertama yaitu mulai kemudian pada tahap kedua yaitu mencari dan mengumpulkan dataset yang diambil dari basis data terbuka dengan dua kelas yaitu seseorang menggunakan masker dan tidak menggunakan masker yang kemudian *upload* ke *google darive* untuk nantinya diolah pada sistem yang dibuat. Dataset yang digunakan mencakup 3860 gambar terbagi atas 1930 dataset bermasker dan tidak bermasker yang akan diolah menjadi data *training* dan validasi. Ketiga, *pre-processing* dataset proses mengubah data mentah kedalam bentuk yang dapat dipahami serta menentukan parameter yang akan digunakan pada penerapan sistem dengan menyamakan seluruh gambar dataset dengan ukuran gambar 224x224 dengan *channel Red, Green, Blue* (R,G,B). Kemudian juga memasukkan nilai *Epoch*,

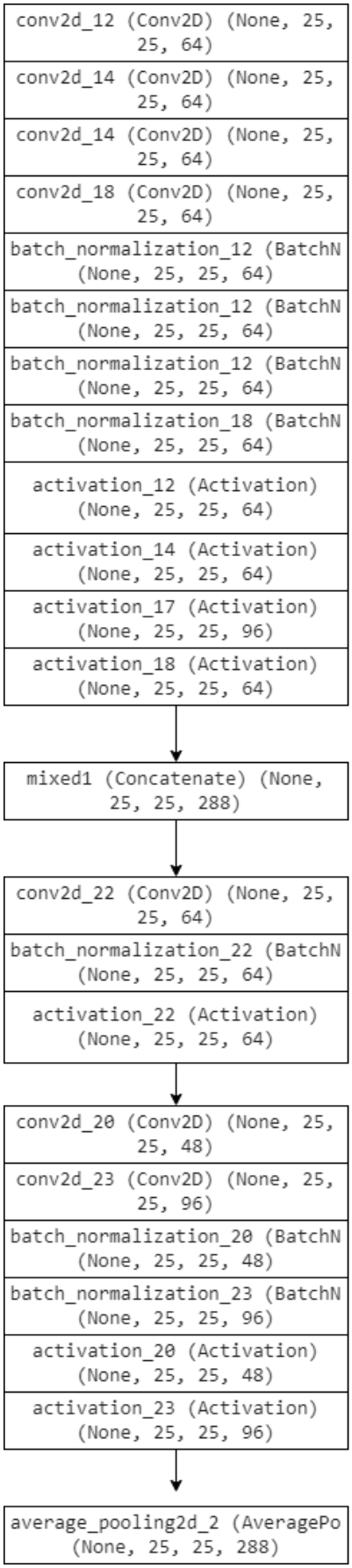
Batch Size dan *Learning rate* serta Implementasi model sistem yang telah dibuat akan langsung diaplikasikan pada sistem deteksi masker secara *realtime*.

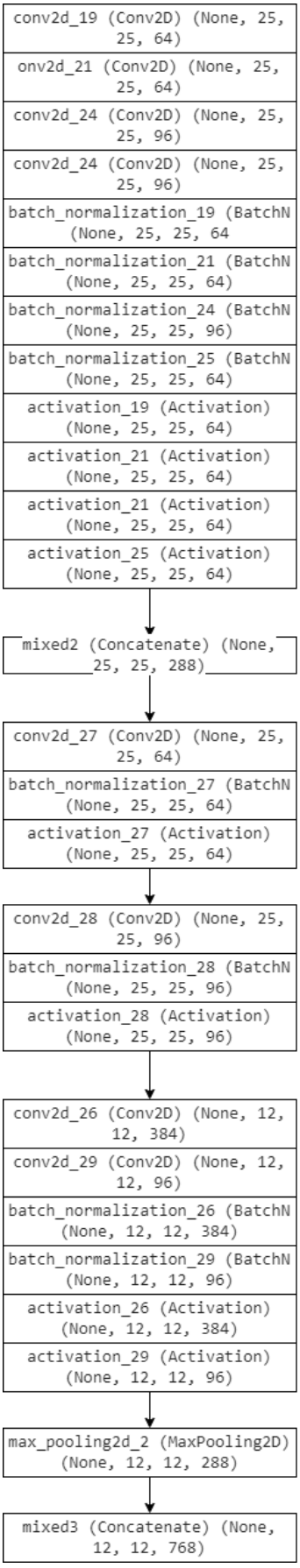
Tahap keempat membuat model sistem deteksi masker menggunakan penerapan arsitektur *InceptionV3* dengan bantuan *Google Colaboratory* serta menggunakan berbagai *libray* yang ada seperti *Tensorflow*, *Keras*, *Numpy* dan Pandas.

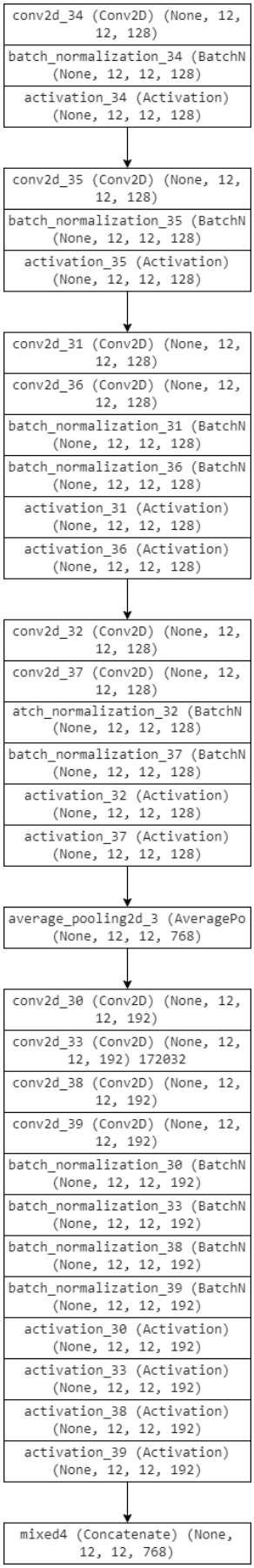


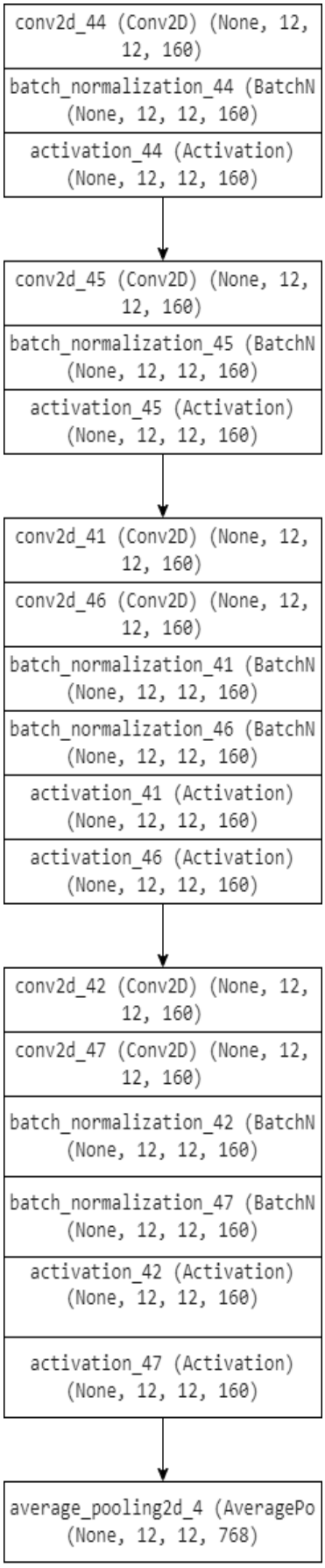

Conv2d_3 (54, 54, 80)
batch_normalization_3 (54, 54, 80)
activation_2 (109, 109, 64)
Conv2d_4(52,52,192)
batch_normalization_4 (52, 52, 192)
activation_4 (52, 52, 192)
Max_pooling2d_1 (25, 25, 192)
conv2d_8 (25, 25, 64)
batch_normalization (25, 25, 64)
activation_8 (25, 25, 64)
conv2d_6 (25, 25, 48)
conv2d_9 (25, 25, 96)
batch_normalization_6 (25, 25, 48)
batch normalization_9 (25, 25, 96)
activation_6 (25, 25, 48)
activation_9 (25, 25, 96)
average_pooling (25, 25, 192)

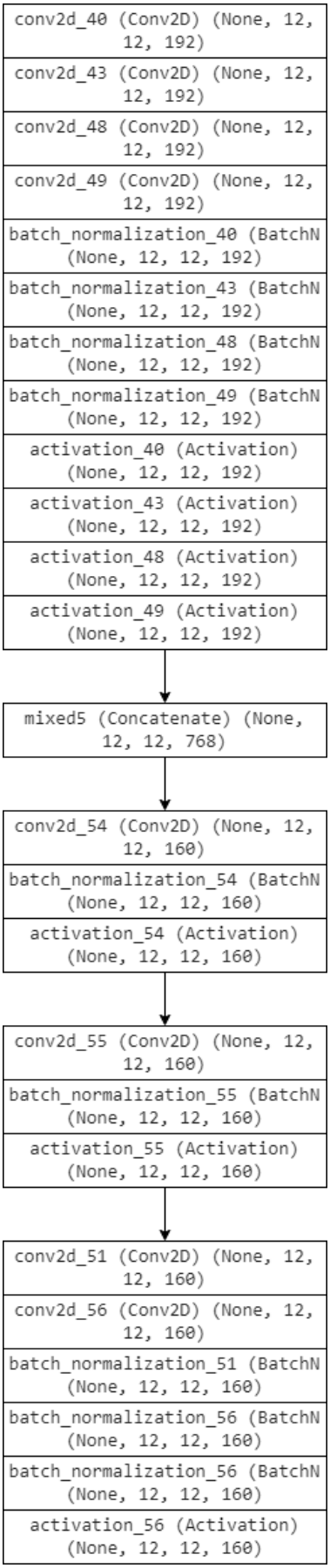

conv2d_5 (Conv2D) (None, 25, 25, 64)
conv2d_7 (Conv2D) (None, 25, 25, 64)
conv2d_10 (Conv2D) (None, 25, 25, 96)
conv2d_11 (Conv2D) (None, 25, 25, 32)
batch_normalization_5 (BatchNo (None, 25, 25, 64)
batch_normalization_7 (BatchNo (None, 25, 25, 64)
batch_normalization_10 (BatchN (None, 25, 25, 96)
batch_normalization_11 (BatchN (None, 25, 25, 32)
activation_5 (Activation) (None, 25, 25, 64)
activation_7 (Activation) (None, 25, 25, 64)
activation_10 (Activation) (None, 25, 25, 96)
activation_11 (Activation) (None, 25, 25, 32)
mixed0 (Concatenate) (None, 25, 25, 256)
conv2d_15 (Conv2D) (None, 25, 25, 64)
batch_normalization_15 (BatchN
activation_15 (Activation) (None, 25, 25, 64)
conv2d_13 (Conv2D) (None, 25, 25, 48)
conv2d_16 (Conv2D) (None, 25, 25, 96)
batch_normalization_13 (BatchN (None, 25, 25, 48)
batch_normalization_13 (BatchN (None, 25, 25, 48)
batch_normalization_13 (BatchN (None, 25, 25, 48)
activation_16 (Activation) (None, 25, 25, 96)
average_pooling2d_1 (AveragePo (None, 25, 25, 256)

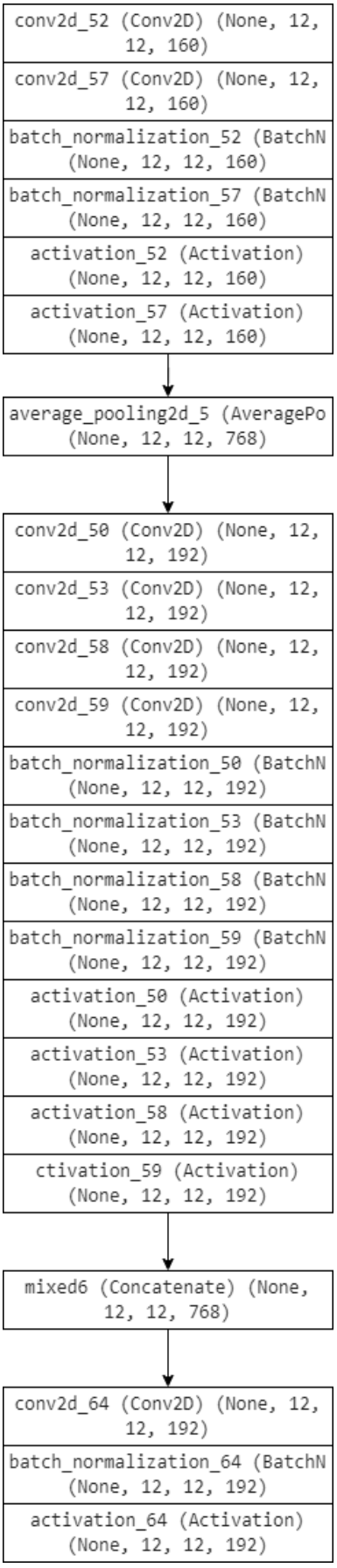

conv2d_12 (Conv2D) (None, 25, 25, 64)
conv2d_14 (Conv2D) (None, 25, 25, 64)
conv2d_14 (Conv2D) (None, 25, 25, 64)
conv2d_18 (Conv2D) (None, 25, 25, 64)
batch_normalization_12 (BatchN (None, 25, 25, 64)
batch_normalization_12 (BatchN (None, 25, 25, 64)
batch_normalization_12 (BatchN (None, 25, 25, 64)
batch_normalization_18 (BatchN (None, 25, 25, 64)
activation_12 (Activation) (None, 25, 25, 64)
activation_14 (Activation) (None, 25, 25, 64)
activation_17 (Activation) (None, 25, 25, 96)
activation_18 (Activation) (None, 25, 25, 64)
mixed1 (Concatenate) (None, 25, 25, 288)
conv2d_22 (Conv2D) (None, 25, 25, 64)
batch_normalization_22 (BatchN (None, 25, 25, 64)
activation_22 (Activation) (None, 25, 25, 64)
conv2d_20 (Conv2D) (None, 25, 25, 48)
conv2d_23 (Conv2D) (None, 25, 25, 96)
batch_normalization_20 (BatchN (None, 25, 25, 48)
batch_normalization_23 (BatchN (None, 25, 25, 96)
activation_20 (Activation) (None, 25, 25, 48)
activation_23 (Activation) (None, 25, 25, 96)
average_pooling2d_2 (AveragePo (None, 25, 25, 288)

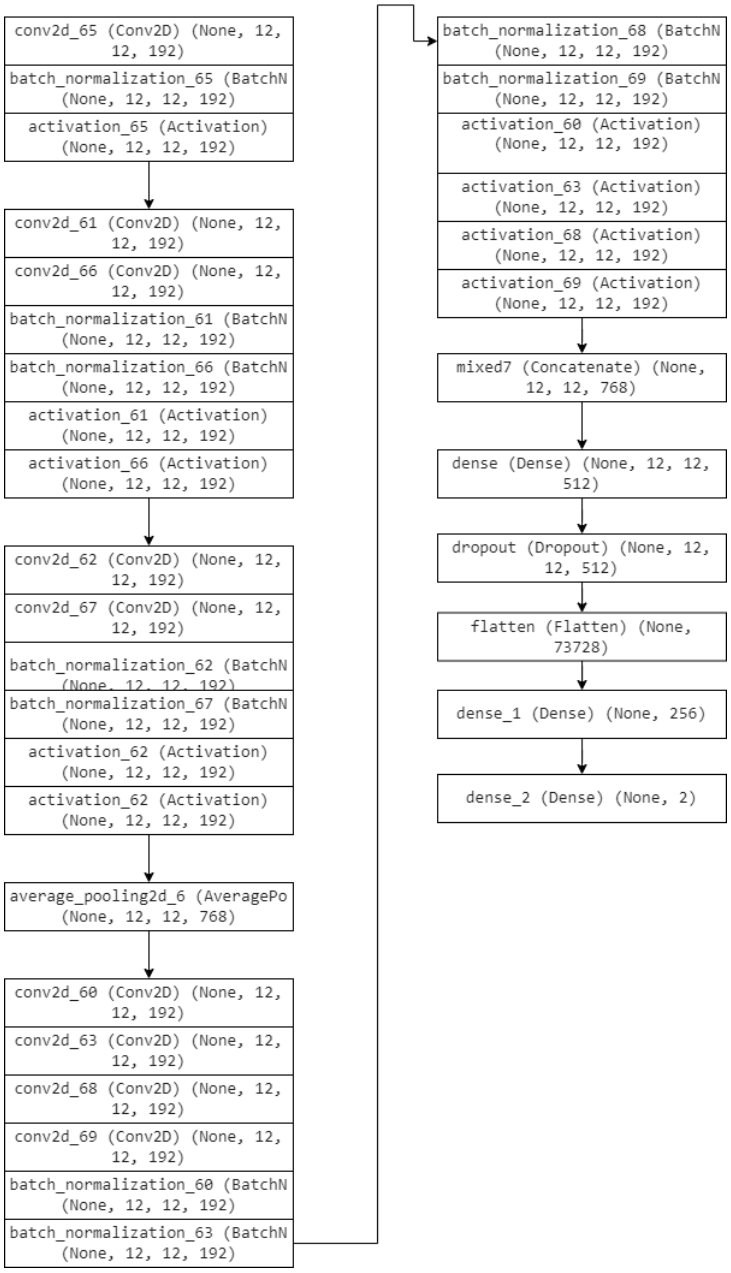

conv2d_19 (Conv2D) (None, 25, 25, 64)
onv2d_21 (Conv2D) (None, 25, 25, 64)
conv2d_24 (Conv2D) (None, 25, 25, 96)
conv2d_24 (Conv2D) (None, 25, 25, 96)
batch_normalization_19 (BatchN (None, 25, 25, 64
batch_normalization_21 (BatchN (None, 25, 25, 64)
batch_normalization_24 (BatchN (None, 25, 25, 96)
batch_normalization_25 (BatchN (None, 25, 25, 64)
activation_19 (Activation) (None, 25, 25, 64)
activation_21 (Activation) (None, 25, 25, 64)
activation_21 (Activation) (None, 25, 25, 64)
activation_25 (Activation) (None, 25, 25, 64)
mixed2 (Concatenate) (None, 25, 25, 288)
conv2d_27 (Conv2D) (None, 25, 25, 64)
batch_normalization_27 (BatchN (None, 25, 25, 64)
activation_27 (Activation) (None, 25, 25, 64)
conv2d_28 (Conv2D) (None, 25, 25, 96)
batch_normalization_28 (BatchN (None, 25, 25, 96)
activation_28 (Activation) (None, 25, 25, 96)
conv2d_26 (Conv2D) (None, 12, 12, 384)
conv2d_29 (Conv2D) (None, 12, 12, 96)
batch_normalization_26 (BatchN (None, 12, 12, 384)
batch_normalization_29 (BatchN (None, 12, 12, 96)
activation_26 (Activation) (None, 12, 12, 384)
activation_29 (Activation) (None, 12, 12, 96)
max_pooling2d_2 (MaxPooling2D) (None, 12, 12, 288)
mixed3 (Concatenate) (None, 12, 12, 768)


conv2d_34 (Conv2D) (None, 12, 12, 128)
batch_normalization_34 (BatchN (None, 12, 12, 128)
activation_34 (Activation) (None, 12, 12, 128)
conv2d_35 (Conv2D) (None, 12, 12, 128)
batch_normalization_35 (BatchN (None, 12, 12, 128)
activation_35 (Activation) (None, 12, 12, 128)
conv2d_31 (Conv2D) (None, 12, 12, 128)
conv2d_36 (Conv2D) (None, 12, 12, 128)
batch_normalization_31 (BatchN (None, 12, 12, 128)
batch_normalization_36 (BatchN (None, 12, 12, 128)
activation_31 (Activation) (None, 12, 12, 128)
activation_36 (Activation) (None, 12, 12, 128)
conv2d_32 (Conv2D) (None, 12, 12, 128)
conv2d_37 (Conv2D) (None, 12, 12, 128)
atch_normalization_32 (BatchN (None, 12, 12, 128)
batch_normalization_37 (BatchN (None, 12, 12, 128)
activation_32 (Activation) (None, 12, 12, 128)
activation_37 (Activation) (None, 12, 12, 128)
average_pooling2d_3 (AveragePo (None, 12, 12, 768)
conv2d_30 (Conv2D) (None, 12, 12, 192)
conv2d_33 (Conv2D) (None, 12, 12, 192) 172032
conv2d_38 (Conv2D) (None, 12, 12, 192)
conv2d_39 (Conv2D) (None, 12, 12, 192)
batch_normalization_30 (BatchN (None, 12, 12, 192)
batch_normalization_33 (BatchN (None, 12, 12, 192)
batch_normalization_38 (BatchN (None, 12, 12, 192)
batch_normalization_39 (BatchN (None, 12, 12, 192)
activation_30 (Activation) (None, 12, 12, 192)
activation_33 (Activation) (None, 12, 12, 192)
activation_38 (Activation) (None, 12, 12, 192)
activation_39 (Activation) (None, 12, 12, 192)
mixed4 (Concatenate) (None, 12, 12, 768)


conv2d_44 (Conv2D) (None, 12, 12, 160)
batch_normalization_44 (BatchN (None, 12, 12, 160)
activation_44 (Activation) (None, 12, 12, 160)
conv2d_45 (Conv2D) (None, 12, 12, 160)
batch_normalization_45 (BatchN (None, 12, 12, 160)
activation_45 (Activation) (None, 12, 12, 160)
conv2d_41 (Conv2D) (None, 12, 12, 160)
conv2d_46 (Conv2D) (None, 12, 12, 160)
batch_normalization_41 (BatchN (None, 12, 12, 160)
batch_normalization_46 (BatchN (None, 12, 12, 160)
activation_41 (Activation) (None, 12, 12, 160)
activation_46 (Activation) (None, 12, 12, 160)
conv2d_42 (Conv2D) (None, 12, 12, 160)
conv2d_47 (Conv2D) (None, 12, 12, 160)
batch_normalization_42 (BatchN (None, 12, 12, 160)
batch_normalization_47 (BatchN (None, 12, 12, 160)
activation_42 (Activation) (None, 12, 12, 160)
activation_47 (Activation) (None, 12, 12, 160)
average_pooling2d_4 (AveragePo (None, 12, 12, 768)


conv2d_40 (Conv2D) (None, 12, 12, 192)
conv2d_43 (Conv2D) (None, 12, 12, 192)
conv2d_48 (Conv2D) (None, 12, 12, 192)
conv2d_49 (Conv2D) (None, 12, 12, 192)
batch_normalization_40 (BatchN (None, 12, 12, 192)
batch_normalization_43 (BatchN (None, 12, 12, 192)
batch_normalization_48 (BatchN (None, 12, 12, 192)
batch_normalization_49 (BatchN (None, 12, 12, 192)
activation_40 (Activation) (None, 12, 12, 192)
activation_43 (Activation) (None, 12, 12, 192)
activation_48 (Activation) (None, 12, 12, 192)
activation_49 (Activation) (None, 12, 12, 192)
mixed5 (Concatenate) (None, 12, 12, 768)
conv2d_54 (Conv2D) (None, 12, 12, 160)
batch_normalization_54 (BatchN (None, 12, 12, 160)
activation_54 (Activation) (None, 12, 12, 160)
conv2d_55 (Conv2D) (None, 12, 12, 160)
batch_normalization_55 (BatchN (None, 12, 12, 160)
activation_55 (Activation) (None, 12, 12, 160)
conv2d_51 (Conv2D) (None, 12, 12, 160)
conv2d_56 (Conv2D) (None, 12, 12, 160)
batch_normalization_51 (BatchN (None, 12, 12, 160)
batch_normalization_56 (BatchN (None, 12, 12, 160)
batch_normalization_56 (BatchN (None, 12, 12, 160)
activation_56 (Activation) (None, 12, 12, 160)


conv2d_52 (Conv2D) (None, 12, 12, 160)
conv2d_57 (Conv2D) (None, 12, 12, 160)
batch_normalization_52 (BatchN (None, 12, 12, 160)
batch_normalization_57 (BatchN (None, 12, 12, 160)
activation_52 (Activation) (None, 12, 12, 160)
activation_57 (Activation) (None, 12, 12, 160)
average_pooling2d_5 (AveragePo (None, 12, 12, 768)
conv2d_50 (Conv2D) (None, 12, 12, 192)
conv2d_53 (Conv2D) (None, 12, 12, 192)
conv2d_58 (Conv2D) (None, 12, 12, 192)
conv2d_59 (Conv2D) (None, 12, 12, 192)
batch_normalization_50 (BatchN (None, 12, 12, 192)
batch_normalization_53 (BatchN (None, 12, 12, 192)
batch_normalization_58 (BatchN (None, 12, 12, 192)
batch_normalization_59 (BatchN (None, 12, 12, 192)
activation_50 (Activation) (None, 12, 12, 192)
activation_53 (Activation) (None, 12, 12, 192)
activation_58 (Activation) (None, 12, 12, 192)
ctivation_59 (Activation) (None, 12, 12, 192)
mixed6 (Concatenate) (None, 12, 12, 768)
conv2d_64 (Conv2D) (None, 12, 12, 192)
batch_normalization_64 (BatchN (None, 12, 12, 192)
activation_64 (Activation) (None, 12, 12, 192)



Gambar 3.3 Struktur Base Model

Pada arsitektur *InceptionV3* basis *transfer learning base* model menggunakan *Transfer learning* untuk meningkatkan *learning* dari satu domain dengan cara *transfer* informasi dari domain yang sama serta menggunakan model yang sudah dilatih (*pre-trained*) dengan data latih yang besar agar digunakan kembali untuk ekstraksi fitur. Jenis *transfer learning* yang digunakan yaitu untuk ekstraksi fitur dimana sebuah *classifier* yang sudah di dilatih dari awal serta representasi fitur dari model *pre-tarined* mengekstraksi fitur secara umum seperti garis tepi sehingga tidak perlu melatih ulang seluruh model.

Tahap kelima, melakukan proses *training* data dengan memberikan alamat *directori* penyimpanan dataset yang sudah di*upload* pada *Google drive* kemudian memberi nilai pada parameter *Epoch, Learning Rate* dan *Batch Size* maka model sistem akan melakukan proses pengolahan data yang akan berlangsung beberapa menit sesuai dengan banyaknya dataset yang digunakan. Pada arsitektur *InceptionV3* menggunakan *base transfer learning* terdiri dari 5 lapisan konvolusi dasar (*stem*) dengan *valid padding* yang terdiri dari *conv2d_0* hingga *conv2d_69* dimana setiap konvolusi diikuti oleh aktivasi *ReLu* dan *BatchNormalization*. *InceptionV3* salah satu versi terbaru dari versi sebelumnya yang melakukan beberapa perbaikikan termasuk label *smoothing*, faktorisasi konvolusi 7x7 dan penyebaran informasi label terhadap jaringan yang lebih rendah atau mengurangi parameter. Pada modul *inception* represetasi fitur multi skala digubungkan serta dimasukkan ke dalam pengklasifikasi tambahan dengan karnel konvolusi yang beragam yaitu 1x1, 1x3, 3x1, 3x3, 3x5, 5x5, 1x7 dan 7x7 digunakan untuk menghasilkan konvergensi.

Tahap keenam setelah proses *training* berhasil maka akan dilakukan implementasi dengan ketentuan parameter kemudian hasil tersebut akan menampilkan grafik yaitu grafik nilai akurasi dan nilai *loss*. Pada tahap ketujuh apabila kondisi grafik tidak sesuai serta terjadi kondisi *underfitting* dan *overfiting* maka model sistem akan melakukan evaluasi serta menampilkan hasil evaluasi seperti akurasi, *F1 score, recall* dan lain-lain kemudian tahap terakhir akan menampilkan fitur *confusion matrix* untuk melihat keakuratan hasil sistem yang sudah dilakukan.

Pada hasil deteksi wajah bermasker dan tidak bermasker nilai persentasi pada kotak wajah deteksi merupakan nilai dari predikat yaitu merupakan dataset wajah bermasker dan tidak bermasker yang sudah di*input* dan melalu proses *training*. Tertulis pada *sourse code* sebagai berikut :

```
label = "Bermasker" if mask > withoutMask else "Tidak Bermasker"
color = (0, 255, 0) if label == "Bermasker" else (255, 0, 0)

label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
```

Gambar 3.4 Source Code Nilai Deteksi

Pada variabel “Label” yang berisikan *String* kemudian {:.2f} disi dengan *function* “.format()”. Variabel pada label dimasukkan dengan variabel yang sudah di cari

pada kondisi if else sebelumnya kemudian pada format diambil nilai prediksi tertinggi antara nilai bermasker dan tidak bermasker.

3.4 SKENARIO PENGUJIAN

Pada penelitian ini *scenario* pengujiannya dimulai dari penentuan nilai parameter *Epoch*, *Batch Size* dan *Larning rate* untuk mengetahu kecepatan serta akurasi proses Model yang dilatih. Selanjutnya dilakukan pengujian sistem deteksi masker dengan ketentuan berdasarkan jarak dan berdasarkan intensitas cahaya menggunakan satu objek yang akan diuji secara *realtime*.

3.4.1 Pengujian sistem berdasarkan jarak dan intensitas cahaya

Pengujian sistem berdasarkan jarak untuk mendeteksi seseorang menggunakan masker atau tidak menggunakan masker yang akan diuji secara *realtime* dengan satu objek mengarah pada webcam. Jarak yang akan digunakan pada pengujian sejauh 50cm, 100cm, 150cm, 200cm dan 250cm kemudian dilakukan analisis untuk mengetahui seberapa keakuratan sistem dapat berkerja untuk mendeteksi dan dilakukan evaluasi serta Pengujian sistem berdasarkan intensitas cahaya akan menggunakan alat ukur intensitas cahaya yaitu *lux* meter untuk didalam ruangan. Pengujian deteksi masker menggunakan objek untuk dideteksi secara *realtime*, Setelah dilakukan pengujian dan pengambilan data maka akan dilakukan analisis sistem dapat mendeteksi masker pada kondisi yang sudah ditetapkan.

3.4.2 Pengujian sistem berdasarkan jenis masker

Pengujian berdasarkan jenis masker menggunakan beberapa jenis masker untuk membuktikan hasil pembacaan deteksi masker bahwa sistem yang dibuat berhasil dalam pendeteksian masker. Jenis masker yang digunakan antara lain yaitu masker duckbill, masker KF 94, masker medis 3 lapis, KN 95, KAI dan Buff.