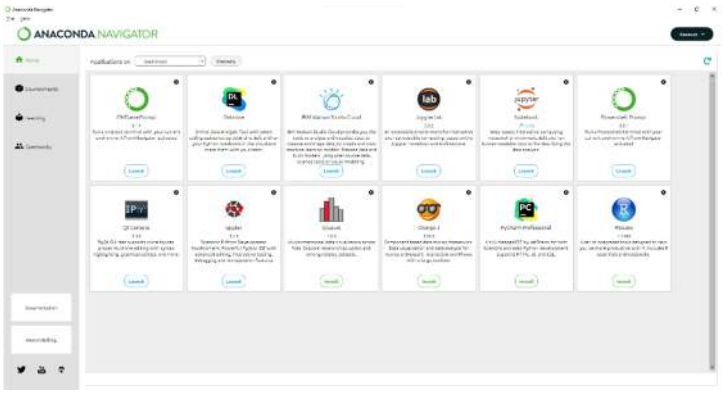
BAB 3
METODE PENELITIAN

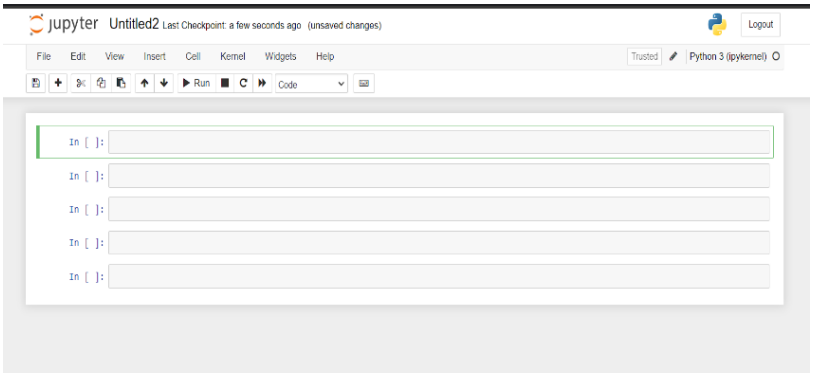
3.1 PEMODELAN SISTEM

Penelitian ini menggunakan suatu pemodelan sistem dalam menganalisis klasifikasi keaslian tanda tangan menggunakan algoritma *Convolutional Neural Network* (CNN) yang dijalankan di *anaconda navigator* dalam *environment jupyter notebook* versi 6.4.8. Pada gambar 3.1 Menunjukan tampilan *Anaconda Navigator* yang didalamnya terdapat bermacam *environments* sesuai dengan fungsi masing-masing. Dalam penelitian ini, *environments* yang digunakan adalah *jupyther notebook* dimana memungkinkan pengguna untuk menjalankan pengeditan dan dokumen *notebook* melalui browser web seperti chrome yang terhubung kedalam internet. Dokumen *notebook* ini merupakan *codingan* seperti *python* yang didalamnya berisi kode program yang nantinya akan dieksekusi dan menampilkan analisis dari hasil program.



Gambar 3.1 Tampilan *Anaconda Navigator*

Gambar 3.2 Tampilan file folder *Jupyter Notebook*



Gambar 3.3 Tampilan *Script Jupyter Notebook*

Didalam *Jupyter noetbook* ini akan digunakan untuk menuliskan *sript coding* model dimana nantinya akan dibuat dua file, yang pertama file algoritma sebagai pembagi dataset kedalam data *train* dan *test*, yang kedua file algoritma sebagai model untuk melakukan *training* dan *testing* data. Alat dan bahan yang digunakan dalam penelitian ini adlah sebagai berikut:

3.2.1 Alat

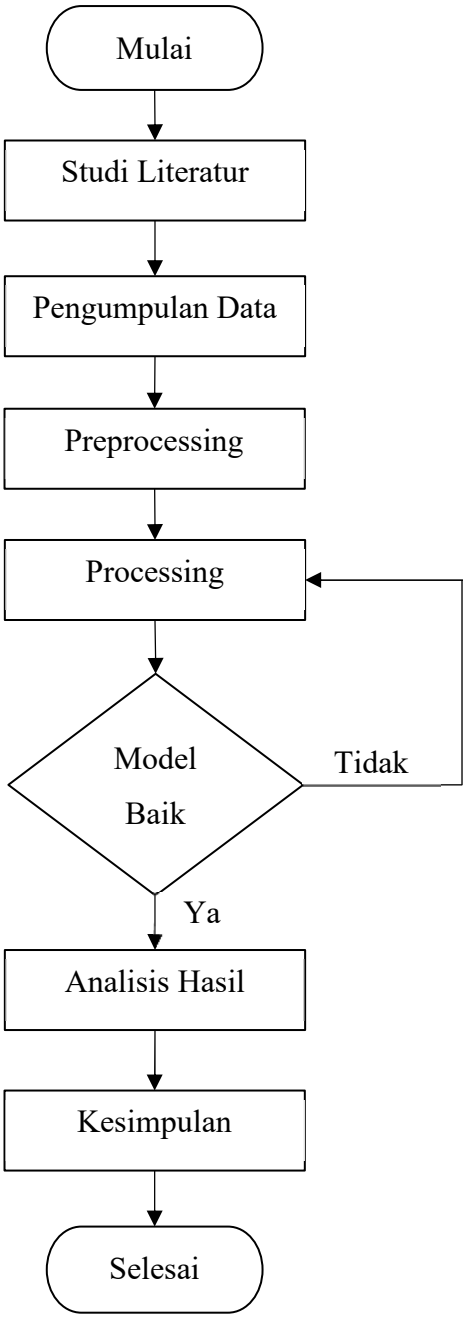
1. Laptop (AMD Ryzen 5500U, 16GB RAM, 512 SSD, Windows 11).
2. *Smartphone* Samsung a50s.
3. Tripod *for smartphone.*

3.2.2 Bahan

1. *Software* (Python 3.9.0, Anaconda, Jupyter Notebook 6.4.8, Photoshop).
2. Citra tanda tangan berjumlah 400 citra yang terdiri dari 4 kelas terbagi menjadi folder (Asli_1, Asli_2, Palsu_1 dan Palsu_2).

Spesifikasi laptop dapat mempengaruhi terhadap lamanya sebuah model *deep learning* berjalan karena dalam pemrosesan data memakan cukup banyak *resource* dari laptop.

3.2 ALUR PENELITIAN



Gambar 3.4 *Flowchart* Alur Penelitian

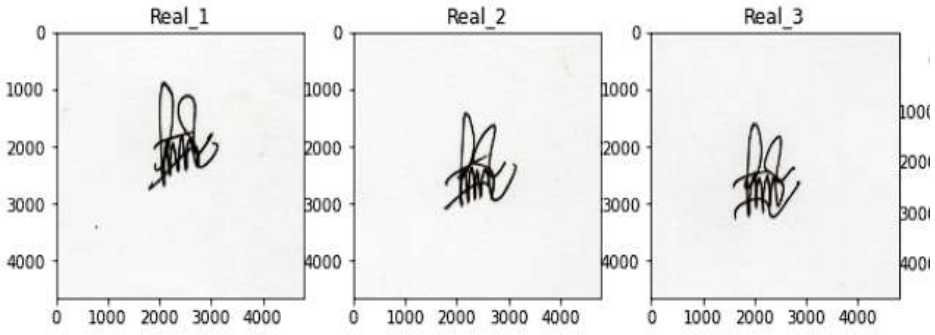
Penelitian klasifikasi keaslian tanda tangan dilakukan dalam beberapa tahap seperti pada gambar 3.3 yaitu tahap studi literatur, pengumpulan data, *preprocessing* data, *processing* data, analisis hasil, dan kesimpulan yang mana akan dijelaskan sebagai berikut:

3.2.1 Studi Literatur

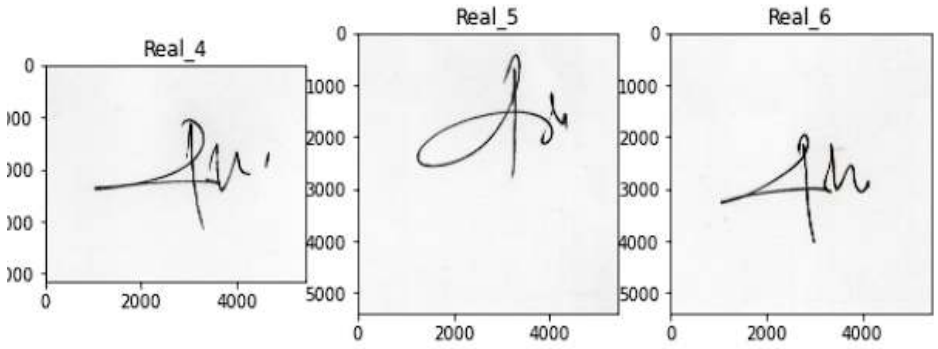
Tahap pertama kali yang dilakukan yaitu studi literatur dengan melakukan pencarian informasi terkait penelitian yang serupa. Pencarian informasi dapat berupa jurnal dari penelitian yang pernah dilakukan sebelumnya dengan berbagai metode atau *dataset* yang berbeda, dan buku elektronik ataupun website yang menjelaskan teori terkait *machine learning*, *deep learning* atau *pengolahan* citra. Dari studi literatur yang dikumpulkan dan dibaca, semuanya mengarah terhadap penelitian klasifikasi citra menggunakan algoritma *machine learning* atau *deep learning.* Selain pencarian informasi terkait *machine learning,* dilakukan juga pencarian informasi terkait pemrograman python yang akan dijadikan sebagai dasar bahasa program.

3.2.2 Pengumpulan Data

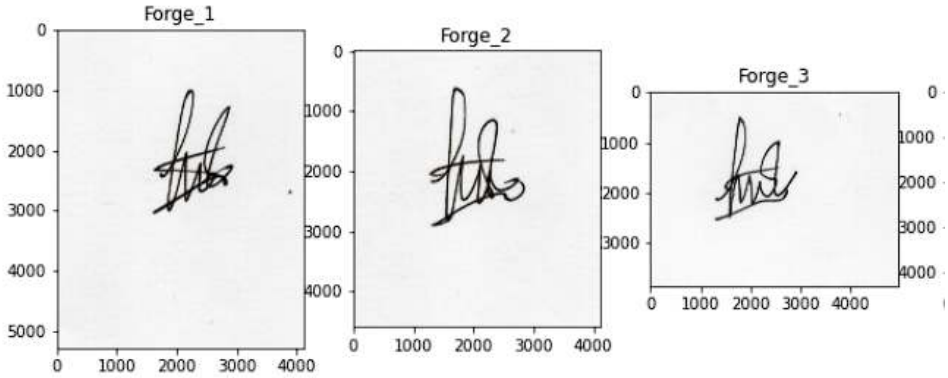
Pengumpulan data dilakukan dengan pembuatan 2 jenis tanda tangan asli masing-masing sebanyak 100 citra dan 2 jenis tanda tangan palsu yang masing-masing juga sebanyak 100 citra, dimana keempat jenis tanda tangan ini dibuat oleh 2 orang. Total banyaknya *dataset* yang diambil adalah sebanyak 400 citra tanda tangan. Tanda tangan ini dibuat diatas kertas hvs putih yang kemudian teknik pengambilan *dataset* dilakukan dengan menggunakan kamera smartphone Samsung a50s resolusi 48mp dan dibantu dengan tripod *smartphone* dengan jarak kamera terhadap *dataset* adalah sejauh 5cm. Waktu pengambilan *dataset* dilakukan pada siang hari dengan kondisi cuaca cerah agar hasil foto yang dihasilkan nampak jelas.



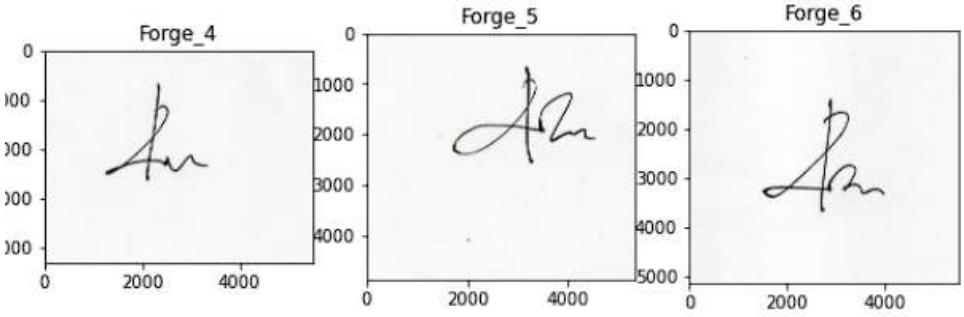
Gambar 3.5 Tanda Tangan Asli 1



Gambar 3.6 Tanda Tangan Asli 2



Gambar 3.7 Tanda Tangan Palsu 1



Gambar 3.8 Tanda Tangan Palsu 2

Setelah citra difoto, langkah selanjutnya citra hasil tanda tangan didalam kertas akan di*crop* menggunakan aplikasi Photoshop agar mempercepat proses *cropping* citra. Teknik *cropping* yang dilakukan menggunakan aplikasi photoshop dan dilakukan secara acak seperti ukuran size yang berbeda dan posisi tanda tangan yang tidak harus simetris ditengah. Hal ini dilakukan untuk menguji model *deep learning* apakah model masih dapat mengenali citra tanda tangan meskipun dari size dan sudut yang berbeda. Pada tahap selanjutnya, dilakukan pengelompokan

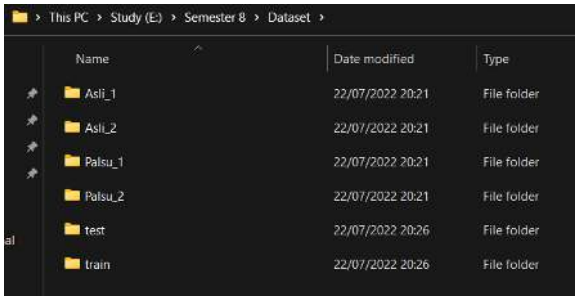
setiap citra kedalam folder yang sudah disiapkan, yaitu Asli_1, Asli_2, Palsu_1 dan Palsu_2. Setiap jenis tanda tangan yang dibuat juga akan memiliki penamaan berbeda agar memudahkan dalam menganalisis prediksi, dimulai dari tanda tangan asli (Asli01_1 dan Asli02_1) serta tanda tangan palsu (Palsu01_1 dan Palsu02_1).

Tabel 3.1 Jumlah Citra

No	Kelas	Jumlah
1	Asli_1	100
2	Asli_2	100
3	Palsu_1	100
4	Palsu_2	100

3.2.3 Preprocessing

Preprocessing dilakukan sebagai langkah awal sebelum *dataset* diproses untuk klasifikasi. Sebelum masuk kedalam *preprocessing, dataset* yang terbagi menjadi 4 folder yaitu Asli_1; Asli_2; Palsu_1; Palsu_2, akan dibagi kembali menjadi *dataset train* dan *test* menggunakan *script coding split dataset* dimana presentase perbandingan data *train* dan *test* adalah sebesar 80% : 20%. Secara umum belum ada ketentuan yang mengharuskan presentase perbandingan data *train* dan *test* harus 80% : 20%, namun dari perbedaan perbandingan tentunya akan mempengaruhi kinerja dari model dalam mempelajari suatu data. Dari total 400 *dataset* yang diambil, maka 80% yang didapatkan adalah sebanyak 320 akan masuk kedalam data *train.* Sementara sisa presentase 20%, didapatkan total citra 80 yang akan digunakan sebagai data *validation.* Hasil folder setelah displit dapat dilihat pada gambar 3.9 dimana terdapat tambahan folder *train* dan *test* dari hasil split keseluruhan citra. Perlu diketahui juga bahwa model akan secara otomatis mengambil citra secara random yang akan dibagi kedalam data *train* dan *test*



Gambar 3.9 Folder hasil pembagian *dataset*

Setelah dataset terbagi menjadi *train* dan *test*, langkah selanjutnya *dataset* yang berbentuk *images* akan *digenerate* dari foler penyimpanan oleh sistem lalu akan merubah bentuk citra kedalam bentuk *array* agar dapat di*augmentasi* sebagai langkah *preproccessing*. Langkah *augmentasi* data dilakukan untuk memaksimalkan model sebelum proses pelatihan sebagai transformasi acak yang dapat membantu model menggeneralisasi dengan baik. Beberapa teknik *augmentasi* atau *preprocessing* data yang digunakan adalah:

1. *Zoom*

   Zoom merupakan metode untuk memperbesar atau memperkecil citra data yang secara acak dengan cara menambahkan beberapa pixel disekitar citra agar dapat diperbesar. Teknik pembesaran ini menggunakan argumen zoom-_range dari kelas ImageDataGenerator. Nilai presentase zoom pada penelitian ini yaitu menggunakan bentuk float dengan nilai zoom_range=0,2. Konteks penentuan nilai float yaitu [1-floatValue, 1+floatValue], yang dalam arti penelitian ini menggunakan nilai zoom [0,8, 1,2]. Nilai 0,8 sama dengan 80% yang artinya citra akan di perbesar 80% dari ukuran asli. Sementara 1,2 berarti citra akan diperkecil sebesar 120% dari ukurn asli.

2. *Flip*

   Flip yang berarti membalikan kolom atau baris *pixel* citra. Teknik flip dalam *deep learning* yang dapat dilakukan yaitu *horizontal_flip* dan *vertical_flip*. Dalam penelitian ini, teknik *flip* yang digunakan hanya *horizontal_flip*. *Augmentasi flip* dapat diaktifkan menggunakan fitur horizontal_flip yang akan di *generate* dengan memberikan keterangan *true* yang artinya mengaktifkan. Keterangan *true* ini merupakan tipe *input*an berjenis *boolean* yang hanya ada *true or false.*

3. *Rescale*

   Rescale atau fungsi normalisasi merupakan fungsi yang digunakan untuk merubah nilai skala *pixel* citra. Setiap citra baik itu jenis citra *grayscale* atau citra RGB memiliki bentuk rentang *pixel* dari 0 sampai 255, dengan 0 berwarna hitam sampai 255 berwarna putih. Namun bagi sebuah model, skala *pixel* 0-255 dikatakan terlalu besar untuk dilakukan sebuah proses pembelajaran model dan dapat memperlambat proses pembelajaran itu sendiri. Pada *augmentasi*

rescale ini, fitur rescalse yang digunakan adalah sebesar 1,0/255,0 yang artinya pixel akan dikonversi dari skala [0,255] menjadi skala [0,1]. Hal ini dilakukan agar model dapat mempercepat proses inputan semua citra sebelum dapat dimasukan kedalam *processing*.

4. *Width shift*

   Width shift digunakan sebagai *augmentasi* untuk menggeser citra ke kiri atau ke kanan (*horizontal*). Nilai *input*an yang dimasukan berjenis *float* dimana akan dikategorikan terhadap bentuk presentase ketika nilai <= 1. Pada penelitian, *widh_shift_range* yang digunakan adalah sebesar 0,10 yang artinya citra akan digeser dalam rentang -10% + 10%. Nilai pergeseran negatif menunjukan bahwa citra akan digeser sebesar 10% ke kiri, sebaliknya nilai pergeseran positif menunjukan citra akan digeser ke kanan sejauh 10%. Selain jenis *float*, terdapat juga inputan jenis integer yang berarti bilangan bulat, namun tidak digunakan dalam penelitian ini. Perbedaannya dengan jenis *float*, integer ini dikategorikan kedalam bentuk *pixel* (px), bukan presentase.

5. *Height shift*

   Height shift merupakan fitur *augmentasi* yang hampir sama dengan *width shift*. Bedanya hanya jika *height shift,* pergeseran yang dilakukan ke arah atas dan bawah (*vertical*). Dalam penelitian ini, besar *height shift* yang digunakan juga sama dengan *widht shift,* yaitu sebesar 0,10.

6. *Shear,*

   Shear digunakan untuk mendistorsi atau menggeseer citra sepanjang sumbu sehingga komputer dapat melihat citra dari sudu pandang yang berbeda. Ukuran nilai *shear* yang digunakan dalam penelitian ini yaitu *shear_range*=0,1 yang artinya citra di distorsi sebesar 10 derajat.

Selain proses *augmentasi,* pada *preprocessing* juga dilakukan *inpuut*an ukuran atau *target_size* sebesar 1000 x 1000 yang artinya citra yang diinputkan akan di*resize* kedalam ukuran tinggi dan lebar 1000 x 1000. Fitur lainnya terdapat juga *class_mode* dimana dalam penelitian ini memakai *class* berjenis *categorical* karena menggunakan 4 *class.* Ketika menggunakan 2 kelas, maka jenis *class* yang digunakan termasuk kedalam kategori *binary*. Kemudian fitur *augmentasi* terakhir yang digunakan dalam data *train* yaitu *batch_size* yang merupakan sebuah

hyperparameter dalam *deep learning*. *Batch size* digunakan sebagai ukuran sample data pelatihan yang memperkirakan kesalahan rata-rata sebelum bobot model diperbaharui. Ukuran *batch size* yang dgunakan adalah 32 dimana artinya setiap epoch dari 320 data *train* akan dibagi oleh *batch size* menjadi 10 sebagai langkah iterasi.

3.2.4 Processing

Processing disini merupakan inti dari model sebagai proses *training* dalam klasifikasi citra tanda tangan menggunakan *Convolutional Neural Network.* Jaringan *Convolutional Neural Network* (CNN) merupakan metode khusus dalam *deep learning* yang dirancang untuk bekerja dengan gambar dua dimensi. Pada penelitian ini menggunakan library keras sebagai conv2D yang merupakan sebuah kernel konvolusi. Untuk model layer yang digunakan adalah berjenis *sequential* dikarenakan jenis model ini merupakan model yang paling digunakan dalam *deep learning* karena memungkinkan untuk menggunakan filter layer demi layer.

Dalam konteks CNN, konvolusi merupakan operasi linier yang melibatkan perkalian dari sekumpulan bobot dengan input, seperti jaringan saraf tradisional. Mengingat bahwa teknik ini dirancang untuk input dua dimensi, perkalian dilakukan antara array data input dan array bobot dua dimensi, yang disebut filter atau kernel. Dalam pemrosesan gambar, kernel adalah matriks atau topeng konvolusi yang dapat digunakan untuk mengaburkan, mempertajam, embossing, deteksi tepi, dan banyak lagi dengan melakukan konvolusi antara kernel dan gambar. Dibawah ini adalah *processing* yang dilakukan dalam penelitian ini:

1. Proses *Conv2D*

   Proses awal dalam *processing* yaitu terdapat parameter *conv2D* yang merupakan jumlah filter yang akan dipelajari oleh lapisan konvolusional. Lapisan di awal arsitektur jaringan yang lebih dekat ke gambar *input* aktual mempelajari lebih sedikit filter konvolusi. Sementara lapisan yang lebih dalam di jaringan atau lapisan yang lebih dekat ke prediksi *output* akan mempelajari lebih banyak filter. Pada penelitian ini, lapisan konvolusi filter yang digunakan adalah 32, 64, dan 128 filter. Sementara untuk parameter ukuran filter atau *kernel size* yang digunakan pada setiap filter berukuran 5 x 5. *Kernel size* ini dilakukan dalam bentuk perkalian dot matriks. Terdapat juga *input size* yang

berukuran (1000, 1000) yang artinya citra yang di*processing* akan memiliki pixel berukuran 1000 x 1000. Setiap lapisan konvolusi menggunakan aktivasi jenis *Rectified Linear Unit* (ReLU) untuk mengubah nilai *input*an. Fungsi aktivasi penting bagi model jaringan saraf untuk mempelajari dan memahami fungsi non-linier yang kompleks. Tanpa fungsi aktivasi, sinyal keluaran hanyalah fungsi linier sederhana. Hasil akhir dari proses *convolutional* ini adalah *feature maps.*

2. Proses *Pooling*

   Prose *pooling* merupakan sebuah proses *downsampling* dengan cara meringkas dimensi dari *feature maps* yang dihasilkan pada proses *convolution.* Dengan *pooling layer,* maka dapat mengurangi jumlah parameter serta mempercepat dikomputasi yang dilakukan jaringan dalam sebuah model. Jenis *pooling* yang digunakan dalam penelitian ini yaitu *max pooling 2D* dengan ukuran kernel (5, 5). *Max pooling* bekerja dengan mengambil nilai tertinggi dari piksel.

3. Proses *Flattening*

   Flattening digunakan sebagai konversi hasil *feature maps* yang telah di *downsampling* dari *array* 2D menjadi *single vetor* untuk dilanjutkan kedalam proses klasifikasi.

4. Proses *Dense*

   Dense dalam penelitian ini dilakukan dengan dua jenis aktivasi, yang pertama aktivasi ReLU dan kedua aktivasi *sigmoid.* Pada *dense* menggunakan aktivasi ReLU ini bekerja sebagai hidden layer sebagai penghubung neuron dari setiap lapisan sebelumnya. Dalam penelitian ini, jumlah *hidden layer* yang digunakan adalah 128. Sementara untuk proses *dense* menggunakan aktivasi *sigmoid* menggunakan ukuran 4 sebagai output akhir dikarenakan penelitian ini menggunakan 4 *class.* Proses *dense* ini sekaligus merupakan *processing* akhir yang mendefinisikan parameter kedalam *output.*

3.2.5 Pengujian Model

Pengujian model merupakan tahap akhir yang dilakukan dengan memasukan *dataset test* kedalam model yang telah dibuat. Pengujian model

dilakukan dengan tujuan untuk melihat apakah model sudah dapat mengklasifikasikan citra dengan baik. Selain itu, dari parameter *epoch* yang diubah akan dilakukan pengujian mulai dari *epoch* 10, 20 dan terakhir 30 untuk mencocokan hasil *validation accuracy* dengan hasil pengujian model. Data citra yang digunakan untuk pengujian yaitu data *test* dimana didapatkan dari hasil *split dataset*. Pengujian yang dilakukan sebanyak 4 kali dimana setiap data *test* citra dari 4 kelas diuji satu persatu menggunakan fungsi *test generator* dan mencocokan klasifikasi kedalam kelas untuk meyakinkan bahwa akurasi yang didapatkan sudah sesuai dengan hasil *validation accuracy*. Hasil *array* akhir yang terbentuk adalah 0, 1, 2, dan 3 yang mendefinisikan sebagai *array output* dalam *testing,* dimana pembagian untuk *output* prediksi citra ketika *output array* 0, data *test* adalah tanda tangan Asli_1, *array* 1 untuk data *test* Asli_2, *array* 2 untuk data *test* tanda tangan Palsu_1 dan *array* 3 untuk data *test* tanda tangan Palsu_2.

Setelah pengujian model menggunakan sistem, pengujian selanjutnya dilakukan perhitungan manual menggunakan *confusion matrix.* Hal ini dilakukan untuk mencocokan hasil besarnya *validation accuracy* akhir dengan perhitungan manual. Proses perhitungan manual dengan *confusion matrix* ini merupakan proses pengujian akhir sebagai tolak ukur kinerja model.