**Michael Christian Handoko**
**J0404231117**

Praktikum Deep Learning menggunakan metode *Convolution Neural Network* (CNN) dilakukan pada Google Colab yang secara otomatis sudah terinstall library yang dibutuhkan (Tensorflow, Keras).

```
1 # Imports
2 from tensorflow.keras.models import Sequential
3 from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
4
5 # Inisiasi CNN
6 MesinKlasifikasi = Sequential()
7
8 # Convolution
9 MesinKlasifikasi.add(Conv2D(32, (3, 3), input_shape=(128,128, 3), activation = 'relu')) # Filters = 32, kernel_size=3,3
10
11 # Pooling
12 MesinKlasifikasi.add(MaxPooling2D(pool_size=(2,2)))
13
```

- Pada line kode 2-3, diimpor library Sequential, COnv2D, MaxPooling2D, Flatten, Dense
    - Sequential menjadi inti dari proyek yang digunakan untuk memulai *neural network*.
    - Conv2D digunakan untuk memulai CNN di tahap Convolution
    - MaxPooling2D digunakan untuk mengambil nilai maksimum setelah tahap *convolution* untuk memperkecil ukuran *feature maps* dengan mencari nilai maksimumnya.
    - Flatten digunakan untuk proses flattening, yaitu
    - Dense digunakan untuk mendefinisikan parameter *neural network*.
- Line 6 didefinisikan objek MesinKlasifikasi yang merupakan objek untuk melakukan klasifikasi dengan metode Sequential.
- Line 9 dilakukan *convolution* dengan parameter sebagai berikut:
    - 32 feature detectors yang kernel_sizenya berukuran 3x3.
    - ukuran gambar yang masuk dalam input layer berukuran 128x128 dan 3 array (berwarna)
    - aktivasi ReLU (Rectifier Linear Unit)
- Line 12 dijalankan proses *maxpooling* dengan pool size 2x2, sehingga memperkecil ukuran *feature maps* dan mempercepat analisis.

```
14 # Menambah Convolutional Layer
15 MesinKlasifikasi.add(Conv2D(32, (3, 3), activation = 'relu'))
16 MesinKlasifikasi.add(MaxPooling2D(pool_size=(2,2)))
17
18 # Flatten
19 MesinKlasifikasi.add(Flatten())
20
21 # Full Connection
22 MesinKlasifikasi.add(Dense(units=128, activation='relu'))
23 MesinKlasifikasi.add(Dense(units=1, activation='sigmoid'))
24
25 # Menjalankan CNN
26 MesinKlasifikasi.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
27
```

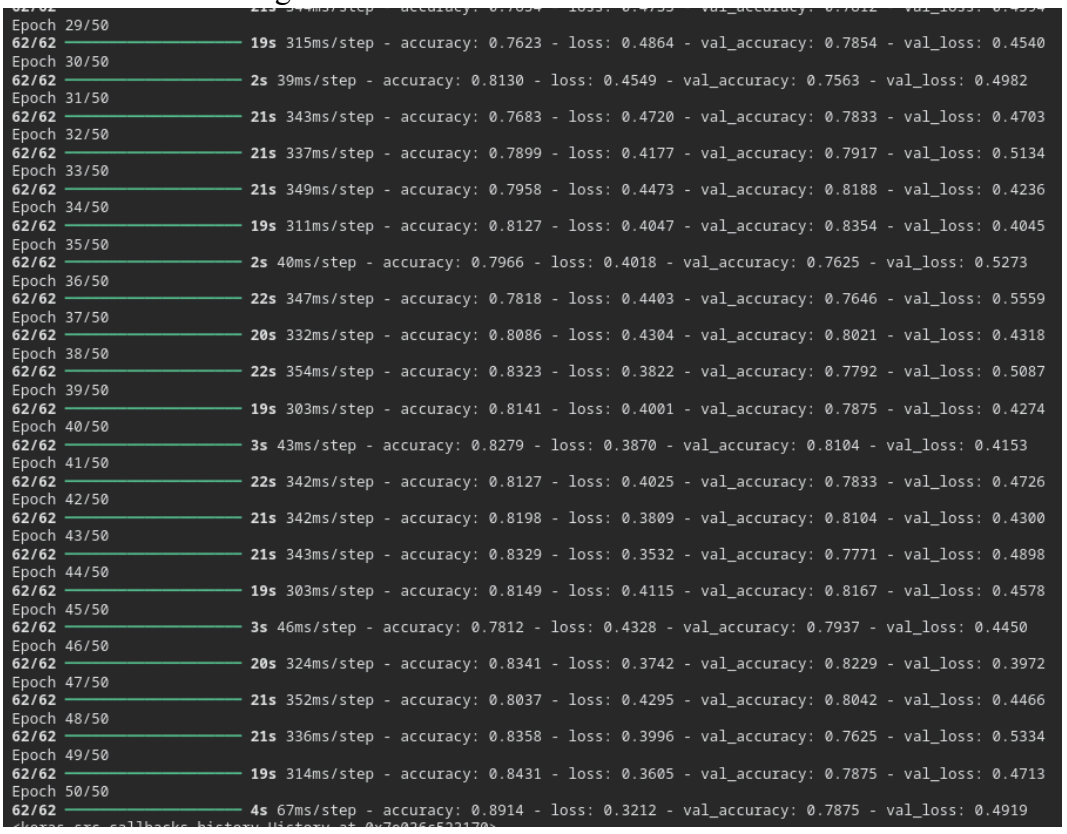
- Line 15-16 ditambahkan lapisan *convolution* + *pooling* untuk mempertajam ekstraksi fitur supaya lebih detail.
- Line 19 diubah seluruh *feature map* 2D/3D menjadi 1 vektor tunggal sebagai input.

- Line 22-23 ditambahkan hidden layer dengan masing-masing 128 dan 1 unit dan aktivasi ReLU dan sigmoid.
    - pada line 23, digunakan 1 unit neuron karena klasifikasi biner (pemilihan anjing atau kucing).
    - Aktivasi Sigmoid digunakan untuk menghasilkan probabilitas yang cocok untuk *binary classification* pada kasus pemilihan anjing atau kucing ini.
- Line 26 dilakukan kompilasi dengan parameter sebagai berikut:
    - optimizer adam digunakan karena lebih stabil dan cepat. Digunakan untuk menghitung arah perbaikan bobot berdasarkan gradien
    - loss binary_crossentropy disesuaikan pada kasus yang diberikan, yaitu kasus probabilitas. Digunakan untuk mengukur jarak probabilitas prediksi model terhadap target sebenarnya
    - metrics digunakan untuk mengevaluasi performa model dengan cara yang mudah dimengerti dengan metrics accuracy. Metrik ini menghitung persentase prediksi yang benar dari seluruh data.

```
27
28 # Bagi Training dan Test set
29 from tensorflow.keras.preprocessing.image import ImageDataGenerator
30
31 train_datagen = ImageDataGenerator(rescale=1./255,
32                                    shear_range=0.2,
33                                    zoom_range=0.2,
34                                    horizontal_flip=True)
35
36 test_datagen = ImageDataGenerator(rescale = 1./255)
37
38 training_set = train_datagen.flow_from_directory('dataset/training_set',
39                                                  target_size = (128, 128),
40                                                  batch_size = 32,
41                                                  class_mode = 'binary')
42
43 test_set = test_datagen.flow_from_directory('dataset/test_set',
44                                             target_size = (128, 128),
45                                             batch_size = 32,
46                                             class_mode = 'binary')
47
48 MesinKlasifikasi.fit_generator(training_set,
49                                steps_per_epoch = 8000/32,
50                                epochs = 50,
51                                validation_data = test_set,
52                                validation_steps = 2000/32)
-INSERT--
```

- Line 29 dilakukan impor library ImageDataGenerator untuk melakukan *image augmentation process.* Membantu mengurangi *overfitting* dengan membuat variasi gambar selama training.
- Line 31 dibuat objek ImageDataGenerator untuk training dengan parameter sebagai berikut:
    - rescale=1/255 berfungsi untuk menormalisasi dataset dengan mengubah nilai pixel dari 0–255 menjadi 0–1
    - shear_range=0.2 → menambah efek miring (shear) pada gambar.
    - zoom_range=0.2 → melakukan zoom in/out kecil.
    - horizontal_flip=True → membalik gambar secara horizontal.
- Line 36 dibuat objek ImageDataGenerator untuk test set

    - Digunakan parameter rescale karena test set tidak boleh di-augmentasi, supaya evaluasi tetap murni
- Line 38 dan 43 membaca gambar dari folder dengan modifikasi parameter sebagai berikut:
    - target_size = (128,128) yang disamakan dengan *convolution* yang dilakukan pada line 9
    - batch_size 32 yang berfungsi membagi 32 gambar ke beberapa batch
    - class_mode = binary yang disesuaikan dengan kasus yang diberikan, yaitu probabilitas kucing atau anjing
- line 48 dijalankan model training dengan parameter:
    - steps_per_epoch merupakan banyaknya batch yang diproses per epoch. Berarti dalam total 8000 training set, terdapat 250 steps yang diproses per epoch. Dalam pengujian, digunakan 2000 training set untuk meringankan beban pada perangkat yang memiliki keterbatasan dalam performa.
    - validation_steps merupakan berapa batch test yang dipakai tiap epoch. Jika terdapat 2000 test set, terdapat 2000/32 steps yang dipakai per epoch. Dalam pengujian, digunakan 500 test set untuk meringankan beban pada perangkat yang memiliki keterbatasan dalam performa.
    - epochs merupakan jumlah seluruh dataset digunakan untuk training.



Dari hasil yang ditunjukan, nilai loss di training set sebesar 0,32. Nilai akurasi cukup tinggi, yaitu sebesar 89%. Val_loss menunjukkan angka 0.49. Nilai val_acc lebih kecil dibanding nilai training, yaitu 78%.

Hasil tersebut menunjukkan adanya overfitting. Meskipun dapat digunakan, model tersebut perlu dikembangkan lagi agar dapat membedakan gambar kucing dan anjing dengan maksimal. Model dapat ditingkatkan kembali dengan cara meningkatkan jumlah dataset, menambah lagi *convolution layer dan max pooling*, atau menambah *fully connected layer* dengan menambahkan perintah Dense seperti di line 24.