LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	: Kecerdasan Artifisial
Kelas	: 3IA19
Praktikum ke-	: 1
Tanggal	: 10/19/2023
Materi	: Pengenalan Kecerdasan Artifisial
NPM	: 50421859
Nama	: Muhamad Ariel Dwi Prayoga
Asisten	: Muhammad Hauzan Dini Fakhri
Paraf Asisten	: -
Nama Asisten	: -
Jumlah Lembar	: 4

LABORATORIUM TEKNIK INFORMATIKA

UNIVERSITAS GUNADARMA

2023

LISTING

1. Berikan Pendapat anda, apakah lebih baik menggunakan Method Callback atau tanpa Callback ? Berikan Alasannya

   Jawab:

   Metode Callback:

   Metode callback sangat berguna dalam operasi yang memerlukan waktu lama untuk menyelesaikan tugasnya, seperti operasi I/O, permintaan jaringan, atau operasi yang berjalan di latar belakang. Dengan menggunakan callback, program dapat melanjutkan eksekusi dan tidak perlu menunggu operasi tersebut selesai.

   Callback juga memungkinkan kita untuk menulis kode yang lebih modular dan dapat digunakan kembali, karena kita dapat mem-pass fungsi sebagai argumen ke fungsi lain.

   Tanpa Callback:

   Di sisi lain, menggunakan metode tanpa callback dapat membuat kode lebih mudah dibaca dan dipahami, terutama untuk orang yang baru belajar pemrograman atau tidak terbiasa dengan konsep callback.

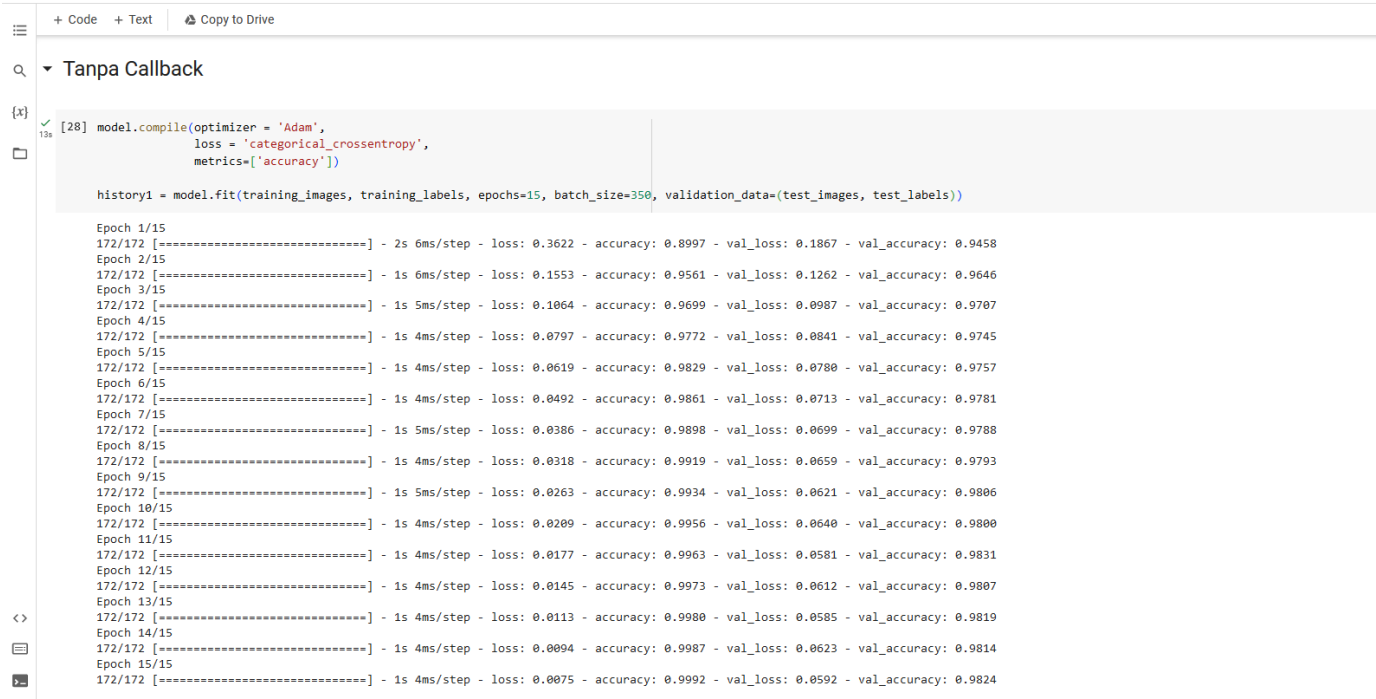
   Kode yang ditulis tanpa callback biasanya mengikuti alur yang lebih linier dan intuitif.

   Namun, penting untuk dicatat bahwa dalam banyak kasus modern, Promise dan async/await (yang merupakan bentuk lain dari callback) digunakan untuk menangani operasi asinkron daripada callback tradisional karena mereka menyediakan sintaks yang lebih bersih dan mudah dibaca.

   Secara keseluruhan, baik metode callback maupun non-callback memiliki kegunaannya masing-masing dan sebaiknya dipilih berdasarkan kebutuhan spesifik proyek atau tugas.

2. Run Kembali blok code Call Back dan tanpa Callback dengan Epochs dan Batch_Size yang kalian inginkan Epochs = 15, Batch_size = 350

   Dan screenshot kode serta outputnya!



```
+ Code  + Text  Copy to Drive

Tanpa Callback

[28] model.compile(optimizer = 'Adam',
                   loss = 'categorical_crossentropy',
                   metrics=['accuracy'])

     history1 = model.fit(training_images, training_labels, epochs=15, batch_size=350, validation_data=(test_images, test_labels))

     Epoch 1/15
     172/172 [==============================] - 2s 6ms/step - loss: 0.3622 - accuracy: 0.8997 - val_loss: 0.1867 - val_accuracy: 0.9458
     Epoch 2/15
     172/172 [==============================] - 1s 6ms/step - loss: 0.1553 - accuracy: 0.9561 - val_loss: 0.1262 - val_accuracy: 0.9646
     Epoch 3/15
     172/172 [==============================] - 1s 5ms/step - loss: 0.1064 - accuracy: 0.9699 - val_loss: 0.0987 - val_accuracy: 0.9707
     Epoch 4/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0797 - accuracy: 0.9772 - val_loss: 0.0841 - val_accuracy: 0.9745
     Epoch 5/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0619 - accuracy: 0.9829 - val_loss: 0.0780 - val_accuracy: 0.9757
     Epoch 6/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0492 - accuracy: 0.9861 - val_loss: 0.0713 - val_accuracy: 0.9781
     Epoch 7/15
     172/172 [==============================] - 1s 5ms/step - loss: 0.0386 - accuracy: 0.9898 - val_loss: 0.0699 - val_accuracy: 0.9788
     Epoch 8/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0318 - accuracy: 0.9919 - val_loss: 0.0659 - val_accuracy: 0.9793
     Epoch 9/15
     172/172 [==============================] - 1s 5ms/step - loss: 0.0263 - accuracy: 0.9934 - val_loss: 0.0621 - val_accuracy: 0.9806
     Epoch 10/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0209 - accuracy: 0.9956 - val_loss: 0.0640 - val_accuracy: 0.9800
     Epoch 11/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0177 - accuracy: 0.9963 - val_loss: 0.0581 - val_accuracy: 0.9831
     Epoch 12/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0145 - accuracy: 0.9973 - val_loss: 0.0612 - val_accuracy: 0.9807
     Epoch 13/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0113 - accuracy: 0.9980 - val_loss: 0.0585 - val_accuracy: 0.9819
     Epoch 14/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0094 - accuracy: 0.9987 - val_loss: 0.0623 - val_accuracy: 0.9814
     Epoch 15/15
     172/172 [==============================] - 1s 4ms/step - loss: 0.0075 - accuracy: 0.9992 - val_loss: 0.0592 - val_accuracy: 0.9824
```

```
Dengan Callback

model.compile(optimizer = 'Adam',
              loss = 'categorical_crossentropy',
              metrics=['accuracy'])

history2 = model.fit(training_images, training_labels, epochs=15, batch_size=350, callbacks=[callbacks], validation_data=(test_images, test_labels))

Epoch 1/15
167/172 [============================>.] - ETA: 0s - loss: 0.0105 - accuracy: 0.9977
Akurasi tinggi, batalkan training!
172/172 [==============================] - 2s 7ms/step - loss: 0.0105 - accuracy: 0.9977 - val_loss: 0.0636 - val_accuracy: 0.9821
```

3. Apa perbedaan antara accuracy dan Val_acurracy dan seberapa pengaruh kedua hal tersebut dalam pembuatan suatu model?

   Dalam konteks pembuatan model machine learning, accuracy dan val_accuracy memiliki perbedaan sebagai berikut:

   Accuracy adalah metrik yang mengukur seberapa baik model Anda dalam memprediksi data latihan. Ini adalah rasio antara jumlah prediksi yang benar dan total jumlah prediksi.

   Val_accuracy (Validation Accuracy) adalah metrik yang mengukur seberapa baik model Anda dalam memprediksi data validasi. Data validasi adalah subset dari data latihan yang tidak digunakan selama proses pelatihan dan hanya digunakan untuk menguji kinerja model.

Kedua metrik ini sangat penting dalam pembuatan model karena mereka memberikan gambaran tentang kinerja model. Namun, mereka memiliki peran yang berbeda:

Jika accuracy tinggi tetapi val_accuracy rendah, ini mungkin menunjukkan bahwa model Anda overfitting, yaitu model belajar terlalu baik pada data latihan tetapi tidak mampu memgeneralisasi dengan baik pada data yang belum pernah dilihat sebelumnya.

Sebaliknya, jika accuracy rendah tetapi val_accuracy tinggi, ini mungkin menunjukkan bahwa model Anda underfitting, yaitu model tidak belajar cukup baik pada data latihan.

Oleh karena itu, tujuan dalam pembuatan model adalah mencapai keseimbangan di mana kedua metrik ini sebanding dan sebesar mungkin21. Ini akan menunjukkan bahwa model Anda memiliki kinerja yang baik dan mampu memgeneralisasi dengan baik pada data baru.