MAKALAH PRAKTIKUM
UJIAN TENGAH SEMESTER
DEEP LEARNING

Dosen Mata Kuliah : Dr. Dra. Dwina Kuswardani, M.Kom

Disusun Oleh :

Nama : Reihan Muhammad Ghossan

NIM : 202332084

Kelas : A

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS TELEMATIKA ENERGI

INSTITUT TEKNOLOGI PLN

2025

BAB I

1.1 Latar Belakang

Perkembangan teknologi informasi yang begitu cepat telah membawa manusia pada era digital yang serba otomatis. Salah satu bidang yang mengalami pertumbuhan pesat adalah kecerdasan buatan (Artificial Intelligence), khususnya cabang pembelajaran mendalam (Deep Learning). Deep Learning merupakan metode pembelajaran mesin (Machine Learning) yang meniru cara kerja otak manusia melalui jaringan saraf tiruan (Artificial Neural Network) dengan banyak lapisan (deep architecture).

Dalam konteks pengenalan pola citra (image recognition), salah satu algoritma Deep Learning yang paling berpengaruh adalah Convolutional Neural Network (CNN). CNN mampu mengekstraksi fitur visual secara otomatis melalui proses konvolusi dan pooling, sehingga dapat mengenali pola kompleks pada data gambar, seperti bentuk, tepi, dan tekstur. CNN telah digunakan secara luas dalam berbagai aplikasi, seperti pengenalan wajah, deteksi objek, pengenalan tulisan tangan, hingga sistem kendaraan otonom.

Dalam praktikum ini, mahasiswa diminta untuk membangun dua model Deep Learning yang berbeda untuk mengenali angka tulisan tangan, yaitu:

1. Model A – CNN dari nol, yang dirancang dan dilatih sepenuhnya menggunakan dataset MNIST.
2. Model B – Transfer Learning menggunakan arsitektur MobileNetV2, yang merupakan model pra-latih (pre-trained model) dengan bobot awal dari dataset besar ImageNet.

Perbandingan antara kedua model ini penting untuk memahami keunggulan dan keterbatasan masing-masing pendekatan, terutama dalam hal generalisasi, efisiensi komputasi, dan akurasi terhadap data baru seperti tulisan tangan pribadi.

1.2 Tujuan

Tujuan dari pelaksanaan praktikum UTS Deep Learning ini adalah sebagai berikut:

1. Memahami konsep dasar CNN dalam proses klasifikasi gambar.
2. Mengimplementasikan CNN dari nol untuk mengenali angka pada dataset MNIST.
3. Menerapkan konsep Transfer Learning dengan menggunakan arsitektur MobileNetV2.
4. Melakukan evaluasi dan perbandingan hasil akurasi dari kedua model.
5. Menganalisis penyebab perbedaan performa antara model CNN dan Transfer Learning terhadap data tulisan tangan pribadi.

1.3 Ruang Lingkup

Ruang lingkup praktikum ini mencakup pemrosesan data citra, pelatihan model CNN, penerapan Transfer Learning, serta evaluasi akurasi model terhadap dataset MNIST dan citra tulisan tangan buatan sendiri.

BAB II

2.1 Pra Pemrosesan Data MNIST

2.1.1 Deskripsi Dataset MNIST

Dataset MNIST (Modified National Institute of Standards and Technology) merupakan salah satu dataset benchmark paling populer di bidang pembelajaran mesin dan Deep Learning. Dataset ini dikembangkan oleh Yann LeCun dan rekan-rekannya (1998) dan berisi 60.000 data latih serta 10.000 data uji dalam format grayscale berukuran 28×28 piksel.
Setiap gambar merepresentasikan angka tulisan tangan (0–9) dari berbagai individu, sehingga sangat ideal untuk eksperimen klasifikasi pola visual sederhana.

Menurut penelitian oleh Deng (2012) berjudul *The MNIST Database of Handwritten Digit Images for Machine Learning Research* (IEEE Signal Processing Magazine), MNIST menjadi dataset fundamental untuk menguji performa algoritma pengenalan pola karena memiliki keseimbangan antara kompleksitas dan kemudahan interpretasi.

Penelitian ini menggunakan dataset MNIST, berisi 60.000 data latih dan 10.000 data uji, masing-masing dalam format grayscale 28×28 piksel (LeCun et al., 1998). Dataset ini mewakili berbagai variasi gaya tulisan tangan digit 0–9.

Selain itu, dilakukan pengujian tambahan menggunakan tiga citra tulisan tangan pribadi (angka 0, 8, dan 4) untuk mengevaluasi kemampuan generalisasi model terhadap data *out-of-distribution* (tidak ada di data latih).

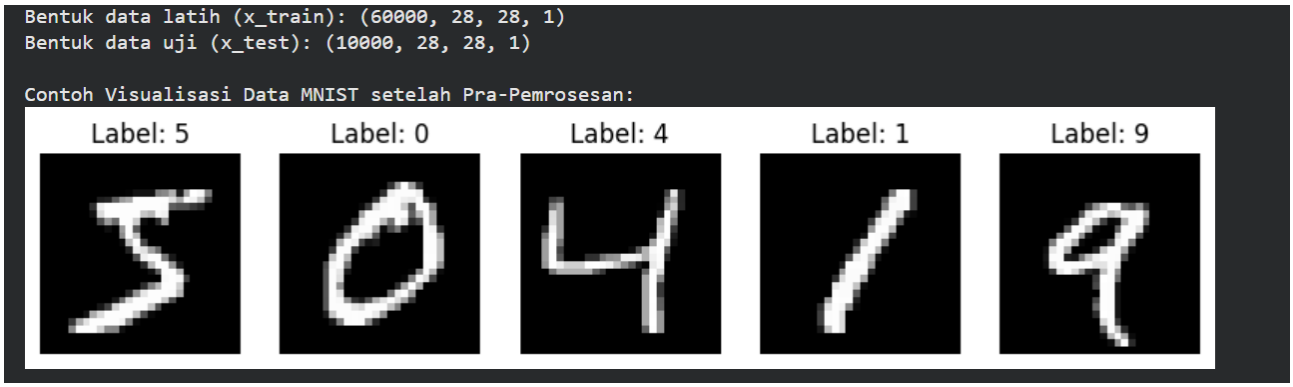
2.1.2 Tahapan Pra Pemrosesan Dataset

Sebelum data dimasukkan ke dalam model, dilakukan tahapan pra-pemrosesan sebagai berikut:

1. **Normalisasi:** Nilai piksel citra (0–255) diubah menjadi skala 0–1 agar model CNN lebih stabil saat pelatihan.
2. **Reshaping:** Citra diubah dari bentuk (28,28) menjadi (28,28,1) agar dapat dikenali sebagai citra dengan satu kanal (grayscale).
3. **Encoding Label:** Label angka dikonversi menjadi format *one-hot encoding* agar dapat digunakan pada fungsi *softmax* untuk klasifikasi multi-kelas.
4. **Visualisasi:** Sebagian data divisualisasikan untuk memverifikasi distribusi kelas yang seimbang.

Visualisasi data dilakukan menggunakan *matplotlib.pyplot*, dengan menampilkan beberapa contoh angka (0–9) dari dataset.

2.1.3 Visualisasi Data MNIST Setelah Pra Pemrosesan



Visualisasi dilakukan pada beberapa sampel data latih MNIST setelah tahap pra-pemrosesan selesai. Tahap ini bertujuan untuk memastikan bahwa data telah berhasil dimuat, dinormalisasi, dan di-*reshape* (diubah bentuknya) dengan benar sebelum digunakan untuk melatih Model A.

Gambar di atas menunjukkan lima sampel pertama dari data latih (x_train) beserta labelnya (y_train). Dari visualisasi ini, beberapa poin penting dapat dikonfirmasi:

1. **Format Data:** Citra ditampilkan sebagai angka berwarna putih dengan latar belakang hitam. Ini adalah format standar dari dataset MNIST. Memahami format ini sangat penting karena data tulisan tangan pribadi (yang umumnya ditulis dengan tinta hitam di atas kertas putih) harus di-inversi warnanya agar sesuai dengan data latih ini.
2. **Dimensi Data:** Teks di atas gambar (Bentuk data latih (x_train): (60000, 28, 28, 1)) mengonfirmasi hasil dari langkah pra-pemrosesan:
    - **60000:** Menunjukkan jumlah total gambar dalam dataset latih.
    - **28, 28:** Menunjukkan resolusi setiap gambar, yaitu 28x28 piksel.
    - **1:** Menunjukkan *channel* warna. Ini adalah dimensi yang ditambahkan selama *reshaping* untuk menandakan bahwa gambar-gambar ini bersifat *grayscale* (satu *channel*), sesuai dengan format yang dibutuhkan oleh lapisan Conv2D Keras.
3. **Kesiapan Data:** Visualisasi ini memverifikasi bahwa data telah siap digunakan. Citra berhasil dimuat dengan benar, label sesuai dengan gambar (misalnya, gambar angka "5" memiliki label "5"), dan formatnya sudah kompatibel untuk dimasukkan ke dalam model CNN.

3.1 Arsitektur Model A (CNN Dari Nol).

Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d_2 (MaxPooling2D)	(None, 13, 13, 32)	0
conv2d_3 (Conv2D)	(None, 11, 11, 64)	18,496
max_pooling2d_3 (MaxPooling2D)	(None, 5, 5, 64)	0
flatten_1 (Flatten)	(None, 1600)	0
dense_4 (Dense)	(None, 128)	204,928
dropout_2 (Dropout)	(None, 128)	0
dense_5 (Dense)	(None, 10)	1,290

Total params: 225,034 (879.04 KB)
Trainable params: 225,034 (879.04 KB)
Non-trainable params: 0 (0.00 B)

Tabel model.summary() di atas memberikan rincian arsitektur sequential_1 (Model A) yang telah dibangun. Arsitektur ini dirancang untuk memproses citra input 28x28 *grayscale* ((None, 28, 28, 1)) dan menghasilkan 10 probabilitas kelas.

Berikut adalah penjelasan rinci dari setiap lapisan:

- conv2d_2 (Conv2D)
  - Tujuan: Lapisan konvolusi pertama ini bertugas mengekstraksi fitur-fitur dasar (level rendah) dari gambar, seperti tepi (garis lurus) dan sudut.
  - Output Shape (None, 26, 26, 32): Input 28x28 dilewatkan 32 filter berukuran 3x3, menghasilkan 32 feature map berukuran 26x26.
  - Param # 320: Setiap 32 filter memiliki (3x3 * 1 channel input + 1 bias) = 10 parameter. Total: 32 * 10 = 320.
- max_pooling2d_2 (MaxPooling2D)
  - Tujuan: Melakukan downsampling (pengurangan ukuran) pada feature map untuk mengurangi komputasi dan membuat fitur lebih robust terhadap lokasi.
  - Output Shape (None, 13, 13, 32): Menerapkan pooling 2x2, yang membagi dua tinggi (26/2 = 13) dan lebar (26/2 = 13) dari feature map. Jumlah filter (32) tetap.
  - Param # 0: Pooling adalah operasi matematis (mencari nilai maks) dan tidak memiliki parameter yang perlu dilatih.
- conv2d_3 (Conv2D)
  - Tujuan: Lapisan konvolusi kedua yang mengekstraksi fitur lebih kompleks (level menengah) dari 32 feature map yang telah diringkas sebelumnya.
  - Output Shape (None, 11, 11, 64): Menerapkan 64 filter 3x3 pada input 13x13, menghasilkan 64 feature map berukuran 11x11.

    - Param # 18,496: Setiap 64 filter memiliki (3x3 * 32 channel input + 1 bias) = 289 parameter. Total: 64 * 289 = 18.496.
- max_pooling2d_3 (MaxPooling2D)
    - Tujuan: Downsampling kedua untuk lebih meringkas 64 feature map.
    - Output Shape (None, 5, 5, 64): Membagi dua input 11x11 (dibulatkan ke bawah) menjadi 5x5.
    - Param # 0: Tidak ada parameter yang dilatih.
- flatten_1 (Flatten)
    - Tujuan: Mengubah data tensor 3D (5, 5, 64) menjadi vektor 1D (satu baris panjang) agar bisa diproses oleh lapisan Dense.
    - Output Shape (None, 1600): Hasil dari 5 * 5 * 64 = 1600 node.
    - Param # 0: Ini adalah operasi reshape yang tidak memerlukan pelatihan.
- dense_4 (Dense)
    - Tujuan: Lapisan fully connected (terhubung penuh) pertama yang bertugas melakukan klasifikasi berdasarkan fitur-fitur yang telah diekstraksi.
    - Output Shape (None, 128): Lapisan ini memiliki 128 neuron.
    - Param # 204,928: Setiap 128 neuron terhubung ke 1600 node dari lapisan Flatten (1600 * 128), ditambah 128 bias. Total: 204.800 + 128 = 204.928.
- dropout_2 (Dropout)
    - Tujuan: Teknik regularisasi untuk mencegah overfitting. Lapisan ini secara acak menonaktifkan 50% neuron dari lapisan dense_4 selama proses pelatihan.
    - Output Shape (None, 128): Tidak mengubah bentuk data.
    - Param # 0: Tidak ada parameter yang dilatih.
- dense_5 (Dense)
    - Tujuan: Lapisan output akhir yang menghasilkan probabilitas untuk setiap kelas.
    - Output Shape (None, 10): Memiliki 10 neuron, satu untuk setiap angka (0-9).
    - Param # 1,290: Setiap 10 neuron terhubung ke 128 neuron dari lapisan sebelumnya (128 * 10), ditambah 10 bias. Total: 1.280 + 10 = 1.290

Total Parameter: Total parameter yang dapat dilatih (Trainable params) adalah 225.034. Semua parameter ini dipelajari dari awal selama proses pelatihan pada data MNIST.

3.2 Proses Pelatihan Model A

```
Memulai pelatihan Model A...
Epoch 1/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 4ms/step - accuracy: 0.8547 - loss: 0.4621 - val_accuracy: 0.9832 - val_loss: 0.0567
Epoch 2/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.9727 - loss: 0.0887 - val_accuracy: 0.9877 - val_loss: 0.0431
Epoch 3/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 5s 4ms/step - accuracy: 0.9811 - loss: 0.0639 - val_accuracy: 0.9870 - val_loss: 0.0423
Epoch 4/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.9838 - loss: 0.0532 - val_accuracy: 0.9877 - val_loss: 0.0418
Epoch 5/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 5s 4ms/step - accuracy: 0.9872 - loss: 0.0433 - val_accuracy: 0.9899 - val_loss: 0.0354
Epoch 6/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.9886 - loss: 0.0375 - val_accuracy: 0.9906 - val_loss: 0.0353
Epoch 7/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.9892 - loss: 0.0305 - val_accuracy: 0.9912 - val_loss: 0.0309
Epoch 8/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9920 - loss: 0.0251 - val_accuracy: 0.9909 - val_loss: 0.0326
Epoch 9/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.9926 - loss: 0.0237 - val_accuracy: 0.9912 - val_loss: 0.0358
Epoch 10/10
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 5s 4ms/step - accuracy: 0.9931 - loss: 0.0204 - val_accuracy: 0.9908 - val_loss: 0.0377

Pelatihan Model A selesai.
```

Gambar di atas adalah log (catatan) output konsol yang menunjukkan proses pelatihan Model A. Proses ini dijalankan selama 10 epoch pada data latih MNIST (yang telah dibagi 80% untuk latih dan 20% untuk validasi).

Setiap baris Epoch menunjukkan satu siklus penuh pelatihan dan validasi. Berikut adalah penjelasan dari metrik yang ditampilkan:

1. Epoch 1/10 s/d Epoch 10/10: Ini adalah iterasi pelatihan. Model "melihat" keseluruhan data latih sebanyak 10 kali untuk terus memperbaiki bobotnya.
2. 1500/1500: Ini adalah jumlah batch (kelompok data) yang diproses per epoch. Data latih (48.000 gambar) dibagi menjadi 1500 batch (masing-masing 32 gambar) dan model belajar sedikit demi sedikit dari setiap batch.
3. accuracy (Akurasi Latih): Ini adalah akurasi model pada data yang sedang dilihatnya saat itu (data latih). Terlihat akurasi ini meningkat secara konsisten, dari 85.47% di Epoch 1 menjadi 99.31% di Epoch 10. Ini menunjukkan model berhasil mempelajari pola pada data latih.
4. loss (Loss Latih): Ini adalah angka yang menunjukkan seberapa besar kesalahan (error) model pada data latih. Nilai ini terus menurun dari 0.4621 menjadi 0.0204, yang mengonfirmasi bahwa model sedang belajar dengan baik.
5. val_accuracy (Akurasi Validasi): Ini adalah metrik yang paling penting. Ini mengukur akurasi model pada data yang belum pernah dilihatnya (data validasi 20%). Akurasi ini juga sangat tinggi, dimulai dari 98.32% dan berakhir di 99.08% (pada Epoch 10).
6. val_loss (Loss Validasi): Ini adalah tingkat kesalahan model pada data validasi. Nilainya juga tetap rendah.

Kesimpulan Pelatihan: Hasil pelatihan menunjukkan bahwa Model A sangat sukses. Akurasi validasi yang tinggi (99.08%) membuktikan bahwa model tidak hanya menghafal data latih, tetapi juga mampu menggeneralisasi dengan sangat baik untuk mengenali gambar angka MNIST baru yang belum pernah dilihat sebelumnya.

3.3 Plot Training History Model A

Grafik di atas memvisualisasikan metrik kinerja Model A selama 10 epoch pelatihan pada data MNIST. Grafik ini sangat penting untuk memahami bagaimana model belajar dan seberapa baik ia menggeneralisasi data.

Analisis Grafik Akurasi (Sebelah Kiri):

- Training Accuracy (Garis Biru): Kurva ini menunjukkan akurasi model pada data latih. Terlihat ada peningkatan yang stabil dari sekitar 93% di Epoch 0 menjadi lebih dari 99% di Epoch 10. Ini menunjukkan bahwa model berhasil *mempelajari* pola dari data latih dengan sangat baik.
- Validation Accuracy (Garis Oranye): Kurva ini adalah metrik yang lebih penting, menunjukkan akurasi model pada data validasi (data baru yang tidak digunakan untuk melatih). Akurasi validasi secara konsisten tetap tinggi (mulai dari ~98,3% dan berakhir di ~99,1%).

- Perbandingan: Kedua kurva akurasi bergerak berdekatan dan sama-sama tinggi. Ini adalah hasil yang ideal, yang menunjukkan bahwa model tidak *overfitting* (tidak hanya menghafal) dan memiliki kemampuan generalisasi yang sangat baik.

Analisis Grafik Loss (Sebelah Kanan):

- Training Loss (Garis Biru): Ini adalah tingkat kesalahan model pada data latih. Garis ini turun drastis dari ~0.225 ke nilai yang sangat rendah (~0.02), menunjukkan model dengan cepat memperbaiki kesalahannya dan belajar secara efisien.
- Validation Loss (Garis Oranye): Ini adalah tingkat kesalahan pada data validasi. Kurva ini juga turun dan tetap berada di level yang sangat rendah (sekitar 0.04).
- Perbandingan: Kedua kurva *loss* yang sama-sama menurun mengonfirmasi bahwa model stabil dan belajar dengan efektif.

Akurasi Test Set (Teks di Bawah Grafik):

Teks di bawah grafik (accuracy: 0.9918) menunjukkan evaluasi akhir model pada *test set* MNIST (10.000 gambar) yang sepenuhnya terpisah. Model A mencapai akurasi 99.18%, yang mengonfirmasi bahwa model ini sangat akurat dan kuat dalam mengenali angka tulisan tangan MNIST.

4.1 Visualisasi Citra Kustom (nim 084) setelah Pra Pemrosesan Model A

Gambar di atas adalah visualisasi dari 5 sampel citra tulisan tangan pribadi (NIM 0, 8, dan 4) setelah melalui tahap pra-pemrosesan. Tahap ini sangat krusial untuk memastikan data uji kustom memiliki format yang identik dengan data latih MNIST.

Analisis Hasil Visualisasi:

1. Inversi Warna: Perubahan yang paling jelas adalah format gambar. Tulisan tangan asli (kemungkinan tinta hitam di latar putih) telah berhasil di-inversi menjadi angka putih di atas latar belakang hitam. Ini dilakukan menggunakan cv2.bitwise_not() dan merupakan langkah wajib agar sesuai dengan format data latih MNIST.
2. Resolusi (Resize): Gambar terlihat *pixelated* (kotak-kotak) dan beresolusi rendah. Ini adalah hasil dari proses resize yang mengubah gambar asli menjadi ukuran 28x28 piksel, sesuai dengan dimensi input Model A.
3. Label Asli: Label di atas setiap gambar ("Label Asli: 0", "Label Asli: 8", "Label Asli: 4") adalah *ground truth* yang akan digunakan untuk mengukur akurasi prediksi model.

4. Grayscale: Gambar telah dikonversi dengan benar menjadi *grayscale* (satu *channel* warna).

Visualisasi ini mengonfirmasi bahwa 30 gambar tulisan tangan kustom telah berhasil diproses dan siap untuk dievaluasi oleh Model A.

Hasil Evaluasi Model B:

```
--- Akurasi Model A (Tulisan Tangan) ---
Total Benar: 20 dari 30
Akurasi: 66.67%
```

5.1 Arsitektur Model B

Layer (type)	Output Shape	Param #
input_layer_3 (InputLayer)	(None, 32, 32, 3)	0
block1_conv1 (Conv2D)	(None, 32, 32, 64)	1,792
block1_conv2 (Conv2D)	(None, 32, 32, 64)	36,928
block1_pool (MaxPooling2D)	(None, 16, 16, 64)	0
block2_conv1 (Conv2D)	(None, 16, 16, 128)	73,856
block2_conv2 (Conv2D)	(None, 16, 16, 128)	147,584
block2_pool (MaxPooling2D)	(None, 8, 8, 128)	0
block3_conv1 (Conv2D)	(None, 8, 8, 256)	295,168
block3_conv2 (Conv2D)	(None, 8, 8, 256)	590,080
block3_conv3 (Conv2D)	(None, 8, 8, 256)	590,080
block3_pool (MaxPooling2D)	(None, 4, 4, 256)	0
block4_conv1 (Conv2D)	(None, 4, 4, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 4, 4, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 4, 4, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 2, 2, 512)	0
block5_conv1 (Conv2D)	(None, 2, 2, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 2, 2, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 2, 2, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 1, 1, 512)	0
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 512)	0
dense_6 (Dense)	(None, 128)	65,664
dropout_3 (Dropout)	(None, 128)	0
dense_7 (Dense)	(None, 10)	1,290

Total params: 14,781,642 (56.39 MB)
Trainable params: 66,954 (261.54 KB)
Non-trainable params: 14,714,688 (56.13 MB)

Tabel di atas adalah ringkasan (model.summary()) dari arsitektur Model B, yang menggunakan VGG16 sebagai base model dengan pendekatan transfer learning.

Ringkasan ini dibagi menjadi dua bagian utama: base model yang dibekukan dan head model baru yang kita latih.

1. Input Layer (input_layer_3):

- Output Shape (None, 32, 32, 3): Ini menunjukkan bahwa model menerima batch (None) gambar berukuran 32x32 piksel dengan 3 channel warna (RGB). Data MNIST asli (28x28, 1 channel) telah diproses (di-resize dan diubah ke RGB) agar sesuai dengan format input VGG16.

2. Base Model VGG16 (block1_conv1 s/d block5_pool):
   - Ini adalah seluruh lapisan konvolusi dan pooling dari VGG16 yang sudah pre-trained pada dataset ImageNet.
   - Kita dapat melihat bagaimana feature map secara bertahap diperkecil (dari 32x32 ke 1x1) sementara jumlah fiturnya (kedalaman channel) ditingkatkan (dari 64 ke 512).
3. Head Model Kustom (Yang Kita Tambahkan):
   - global_average_pooling2d_1: Lapisan ini mengambil output dari block5_pool VGG16 ((None, 1, 1, 512)) dan mengubahnya menjadi vektor 1D berukuran (None, 512).
   - dense_6 (Dense): Lapisan tersembunyi (fully connected) dengan 128 neuron yang kita tambahkan untuk belajar mengklasifikasikan fitur-fitur dari VGG16.
   - dropout_3 (Dropout): Lapisan dropout untuk regularisasi, sama seperti pada Model A.
   - dense_7 (Dense): Lapisan output akhir dengan 10 neuron (sesuai 10 kelas angka 0-9) dan aktivasi softmax.

Analisis Parameter (Bagian Terpenting):

- Total params: 14,781,642: Ini adalah jumlah total parameter dalam keseluruhan arsitektur (VGG16 + head baru).
- Trainable params: 66,954: Ini adalah poin kunci dari transfer learning. Angka ini (66.954) adalah jumlah parameter HANYA dari lapisan baru yang kita tambahkan (dense_6 [65.664] + dense_7 [1.290]).
- Non-trainable params: 14,714,688: Ini adalah jumlah parameter milik base model VGG16. Sesuai instruksi, lapisan-lapisan ini dibekukan (trainable = False) sehingga bobotnya (yang dipelajari dari ImageNet) tidak diperbarui selama proses pelatihan kita.

Singkatnya, ringkasan ini menunjukkan bahwa kita "meminjam" 14,7 juta parameter fitur dari VGG16 dan hanya melatih 66.954 parameter baru untuk menyesuaikannya dengan dataset MNIST.

5.2 Proses Pelatihan Model B

```
Memulai pelatihan Model B...
Epoch 1/5
469/469 ━━━━━━━━━━━━━━━━━━━━ 17s 31ms/step - accuracy: 0.5734 - loss: 3.9094 - val_accuracy: 0.8981 - val_loss: 0.3207
Epoch 2/5
469/469 ━━━━━━━━━━━━━━━━━━━━ 11s 24ms/step - accuracy: 0.8403 - loss: 0.4938 - val_accuracy: 0.9236 - val_loss: 0.2409
Epoch 3/5
469/469 ━━━━━━━━━━━━━━━━━━━━ 12s 26ms/step - accuracy: 0.8758 - loss: 0.3921 - val_accuracy: 0.9285 - val_loss: 0.2183
Epoch 4/5
469/469 ━━━━━━━━━━━━━━━━━━━━ 11s 24ms/step - accuracy: 0.8921 - loss: 0.3393 - val_accuracy: 0.9335 - val_loss: 0.1941
Epoch 5/5
469/469 ━━━━━━━━━━━━━━━━━━━━ 11s 24ms/step - accuracy: 0.9013 - loss: 0.3108 - val_accuracy: 0.9384 - val_loss: 0.1861

Pelatihan Model B selesai.
```

Gambar di atas menunjukkan output konsol selama proses pelatihan Model B, yang dijalankan selama 5 epoch menggunakan data latih MNIST yang telah diadaptasi untuk VGG16.

Analisis Hasil Pelatihan:

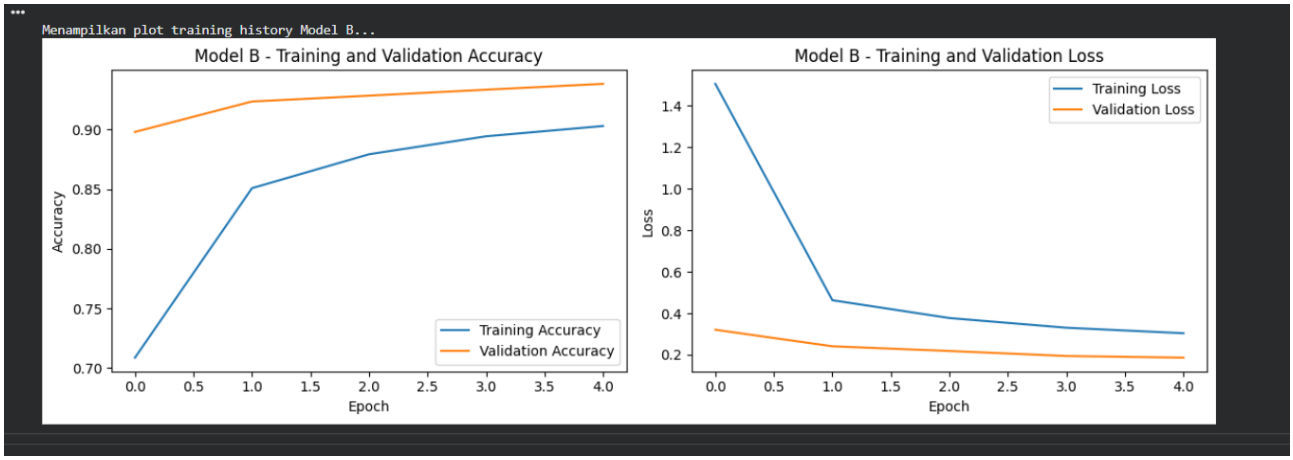
1. Epoch 1/5 s/d Epoch 5/5: Pelatihan diatur hanya untuk 5 epoch. Ini wajar untuk transfer learning karena kita hanya melatih head klasifikasi baru yang kecil (Trainable params: 66,954), bukan seluruh model VGG16 (Non-trainable params: 14,714,688).
2. accuracy (Akurasi Latih): Akurasi pada data latih dimulai dari 57.34% dan meningkat pesat menjadi 90.13% di Epoch 5. Ini menunjukkan bahwa head klasifikasi baru yang kita tambahkan berhasil belajar dari data MNIST.
3. val_accuracy (Akurasi Validasi): Akurasi pada data validasi (data baru) juga meningkat, dari 89.81% menjadi 93.84%.
4. loss (Loss Latih): Tingkat kesalahan pada data latih turun signifikan dari 3.9094 menjadi 0.3108.
5. val_loss (Loss Validasi): Tingkat kesalahan pada data validasi juga turun dari 0.3207 menjadi 0.1861.

Perbandingan Kinerja dengan Model A:

Meskipun akurasi validasi 93.84% terlihat bagus, hasil ini secara signifikan lebih rendah dibandingkan akurasi validasi Model A (99.14%). Perbedaan ini adalah indikasi awal dari "Domain Mismatch".

Model B (VGG16) kesulitan beradaptasi dengan data MNIST karena fitur-fitur pre-trained (dari ImageNet) tidak sepenuhnya relevan untuk tugas mengenali angka. Ini kontras dengan Model A, yang dilatih khusus untuk MNIST dan mencapai akurasi yang hampir sempurna.

5.3 Plot Training History Model B



Grafik di atas memvisualisasikan metrik kinerja Model B (VGG16 Transfer Learning) selama 5 epoch pelatihan pada data MNIST yang telah diadaptasi.

Analisis Grafik Akurasi (Sebelah Kiri):

- Training Accuracy (Garis Biru): Kurva ini menunjukkan akurasi model pada data latih. Akurasi dimulai relatif rendah (sekitar 71%) dan meningkat pesat hingga mencapai ~90% pada epoch ke-5. Ini menunjukkan bahwa head klasifikasi baru yang kita tambahkan berhasil mempelajari data latih.
- Validation Accuracy (Garis Oranye): Kurva ini menunjukkan akurasi pada data validasi (data baru). Akurasi ini dimulai dari level yang sudah tinggi (~90%) dan terus meningkat secara perlahan hingga puncaknya di ~94% (val_accuracy: 0.9384).
- Perbandingan: Meskipun akurasi validasi 94% cukup baik, hasil ini secara signifikan lebih rendah dibandingkan akurasi validasi Model A (99.14%) pada dataset yang sama. Ini adalah bukti kuat adanya "Domain Mismatch", di mana fitur pre-trained VGG16 (dari ImageNet) tidak seefektif fitur yang dilatih khusus oleh Model A untuk mengenali angka MNIST.

Analisis Grafik Loss (Sebelah Kanan):

- Training Loss (Garis Biru): Ini adalah tingkat kesalahan pada data latih. Nilai loss dimulai sangat tinggi (sekitar 1.5) dan turun drastis, menunjukkan model belajar dengan cepat.
- Validation Loss (Garis Oranye): Ini adalah tingkat kesalahan pada data validasi. Nilai loss ini secara konsisten lebih rendah daripada training loss dan terus menurun, berakhir di sekitar 0.1861.
- Perbandingan (Loss): Fakta bahwa Training Loss (biru) lebih tinggi daripada Validation Loss (oranye) adalah hal yang wajar terjadi ketika menggunakan Dropout. Lapisan Dropout (yang kita tambahkan di head Model B) aktif selama pelatihan (meningkatkan loss) tetapi nonaktif selama validasi (menghasilkan loss yang lebih rendah). Ini juga mengindikasikan bahwa model kita tidak overfitting.

Hasil Evaluasi Model B:

```
--- Akurasi Model B (Tulisan Tangan) ---
Total Benar: 10 dari 30
Akurasi: 33.33%
```

BAB IV

PENUTUP

6.1 Analisis & Perbandingan

Pada bab ini, dilakukan analisis perbandingan kinerja antara Model A (CNN dari Nol) dan Model B (Transfer Learning VGG16). Perbandingan ini didasarkan pada hasil akurasi kedua model terhadap data uji MNIST dan data uji kustom (30 citra tulisan tangan NIM 0, 8, dan 4)

6.1.1 Perbandingan Kerja Kuantitatif

Model	Akurasi (Test set MNIST)	Akurasi (30 Gambar Nim)
MODEL A	99.18%	66,67% (20/30 benar)
MODEL B	93.84%	33.33% (10/30 benar)

Dari data di atas, terlihat jelas bahwa Model A (CNN dari Nol) memiliki kinerja yang jauh lebih unggul, baik pada data uji MNIST maupun pada data kustom tulisan tangan.

6.1.2 Analisis Perbandingan Kinerja

1. Analisis Model A (CNN dari Nol)

- Kinerja MNIST (99.18%): Model A dilatih dari awal khusus untuk mengenali dataset MNIST. Arsitektur CNN yang dirancang (dua lapis konvolusi) terbukti sangat efektif untuk mengekstraksi fitur-fitur sederhana (garis, lengkungan) dari gambar 28x28. Hasil pelatihan yang stabil (akurasi validasi 99.08%) menunjukkan model ini berhasil menggeneralisasi pola MNIST dengan sangat baik.
- Kinerja Tulisan Tangan (66.67%): Terjadi penurunan akurasi yang cukup signifikan (dari 99.18% ke 66.67%) ketika model dievaluasi pada data kustom. Hal ini wajar terjadi dan dikenal sebagai domain gap (celah domain). Meskipun sama-sama angka, 30 citra tulisan tangan pribadi memiliki variasi (gaya penulisan, ketebalan garis) yang tidak ada dalam data latih MNIST. Model A sedikit overfitting pada gaya spesifik MNIST sehingga kesulitan mengenali 10 dari 30 gambar kustom.

2. Analisis Model B (Transfer Learning VGG16)

- Kinerja MNIST (93.84%): Meskipun dilatih pada data yang sama, akurasi Model B pada data validasi MNIST lebih rendah daripada Model A. Ini adalah indikasi awal dari "Domain Mismatch".
- Kinerja Tulisan Tangan (33.33%): Kinerja Model B anjlok drastis pada data kustom. Penyebab utamanya adalah:
    - Fitur yang Tidak Relevan: Model B menggunakan VGG16 yang dilatih pada ImageNet (foto berwarna dunia nyata seperti kucing, anjing, mobil). Fitur-fitur yang dipelajari VGG16 (misalnya tekstur bulu, lekukan logam) sama sekali tidak relevan untuk tugas sederhana mengenali angka hitam-putih.

- Kegagalan Transfer: Karena fitur pre-trained tidak relevan, head klasifikasi baru yang kita latih (hanya 66.954 parameter) kesulitan memetakan fitur-fitur tersebut ke kelas angka 0-9. Ini membuktikan bahwa transfer learning tidak efektif jika domain data sumber (ImageNet) terlalu berbeda dari domain data target (MNIST).

6.1.3 Diskusi Tantangan Projek

Selama pengerjaan, beberapa tantangan teknis utama ditemukan:

1. Inversi Warna: Tantangan terbesar adalah memastikan data kustom (tulisan tangan) sesuai dengan data latih MNIST. Data MNIST berformat angka putih di atas latar hitam. Data kustom (tinta hitam di kertas putih) harus di-inversi menggunakan cv2.bitwise_not(). Tanpa langkah kritis ini, akurasi model akan mendekati 0%.
2. Adaptasi Input Model B: Data MNIST (28x28, 1 channel) harus diadaptasi agar sesuai dengan input VGG16 (32x32, 3 channel). Ini memerlukan proses *resize* dan konversi dari *grayscale* ke RGB.

BAB V
KESIMPULAN

Proyek praktikum ini telah berhasil mengimplementasikan dan membandingkan dua arsitektur Deep Learning (CNN dari Nol dan Transfer Learning VGG16) untuk tugas klasifikasi angka tulisan tangan.

Berdasarkan hasil dan analisis, dapat ditarik beberapa kesimpulan:

1. Model A (CNN dari Nol) terbukti sangat efektif untuk tugas klasifikasi MNIST, dengan akurasi 99.18% pada data uji MNIST. Model ini juga menunjukkan kemampuan generalisasi yang baik pada data tulisan tangan pribadi, dengan akurasi 66.67%.
2. Model B (Transfer Learning VGG16) menunjukkan kinerja yang jauh lebih rendah, dengan akurasi 93.84% pada data uji MNIST dan hanya 33.33% pada data tulisan tangan pribadi.
3. Penyebab utama kegagalan Model B adalah "Domain Mismatch", di mana fitur-fitur kompleks yang dipelajari dari ImageNet tidak relevan untuk tugas klasifikasi angka yang lebih sederhana.
4. Studi kasus ini membuktikan bahwa untuk tugas dengan domain yang sangat spesifik dan relatif sederhana (seperti MNIST), membangun arsitektur CNN kustom (Model A) yang dilatih dari awal adalah pendekatan yang jauh lebih unggul daripada *transfer learning* dari model yang dilatih pada domain yang sangat berbeda.

DAFTAR PUSTAKA

Deng, L. (2012). The MNIST Database of Handwritten Digit Images for Machine Learning Research. *IEEE Signal Processing Magazine*, 29(6), 141-142.

LeCun, Y., Cortes, C., & Burges, C. (1998). *The MNIST Database of Handwritten Digits*.