PROJECT 1

Face Recongnition

PRESENTED BY

Christopher Kevin Herijanto
Thomas Dalton
Yohanes J Palis
Louis Maximilian

PENDAHULUAN

Face Recongnition atau sistem pengenalan wajah adalah teknologi yang memungkinkan sistem komputer atau perangkat untuk mengidentifikasi atau memverifikasi seseorang berdasarkan citra wajahnya. Teknologi ini bekerja dengan menganalisis karakteristik wajah seseorang, seperti jarak antara mata, bentuk hidung, atau kontur wajah, dan membandingkannya dengan data wajah yang sudah ada.

Manfaat Face Recognition

1. Keamanan yang Lebih Baik
2. Kemudahan dan Kenyamanan
3. Pengurangan Kesalahan Manual
4. Penyediaan Bukti Visual
5. Peningkatan Efisiensi

Contoh Penggunaan di Dunia Nyata

- Bandara dan Perbatasan
- Perangkat Pribadi
- Sistem Absensi

LOAD DATASET

CELEBA

CelebA_HQ_face_gender_dataset.zip

No duplicate data

Train dataset size: 23999 Test dataset size: 6001
Class names: ['female', 'male']

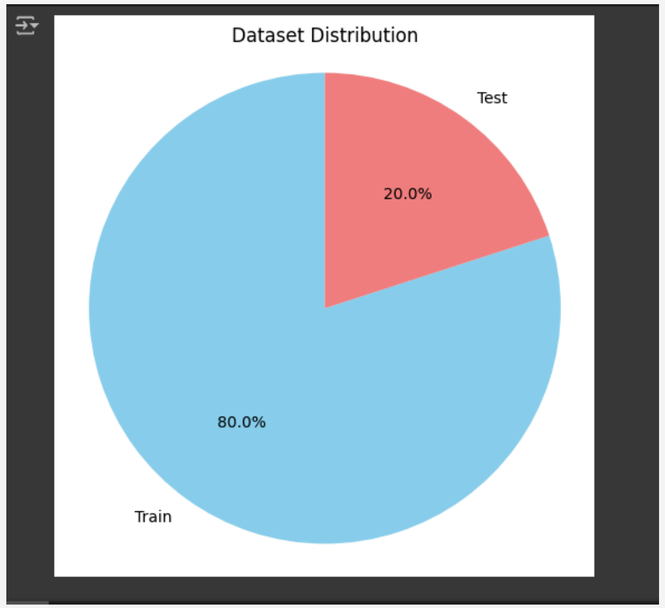
DATASET

PREPOCCESSING DATA

VISUALISASI DATASET

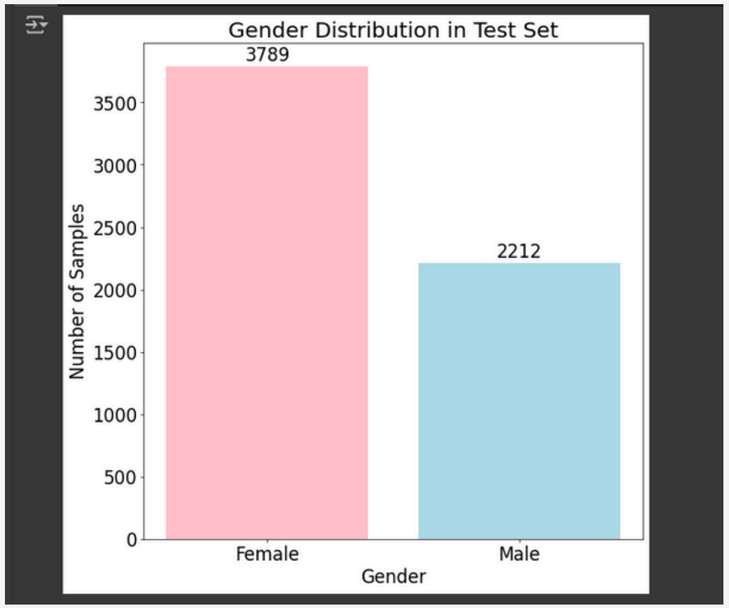
LINK TO COLAB

PEMBAGIAN DATA TRAIN DAN TEST



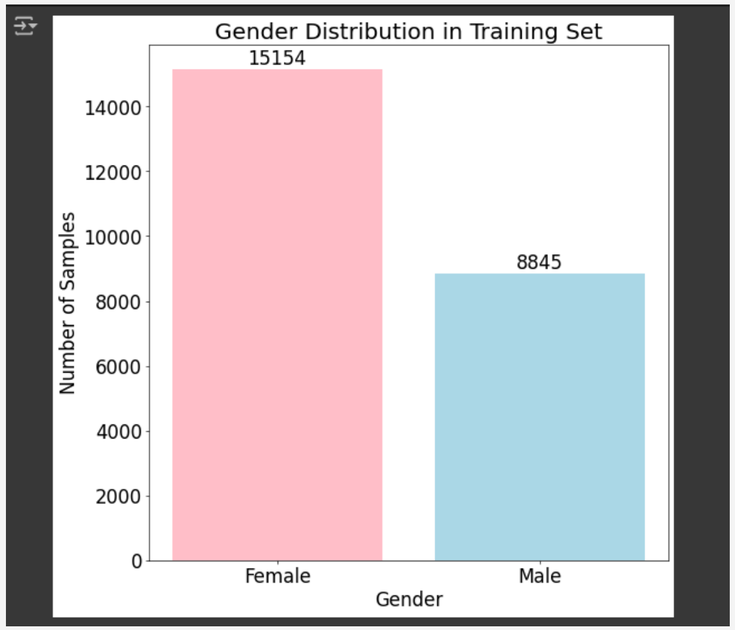
TOTAL DATASET 30000 DENGAN PEMBAGIAN
TRAIN DATASET SIZE: 23999 (80%)
TEST DATASET SIZE: 6001(20%)
CLASS NAMES: ['FEMALE', 'MALE']

CLASS DISTRIBUSI DATA TEST



TOTAL DATASET UNTUK DATA TEST ADALAH 6001 DENGAN PEMBAGIAN
FEMALE: 3789
MALE: 2212

CLASS DATA TRAIN



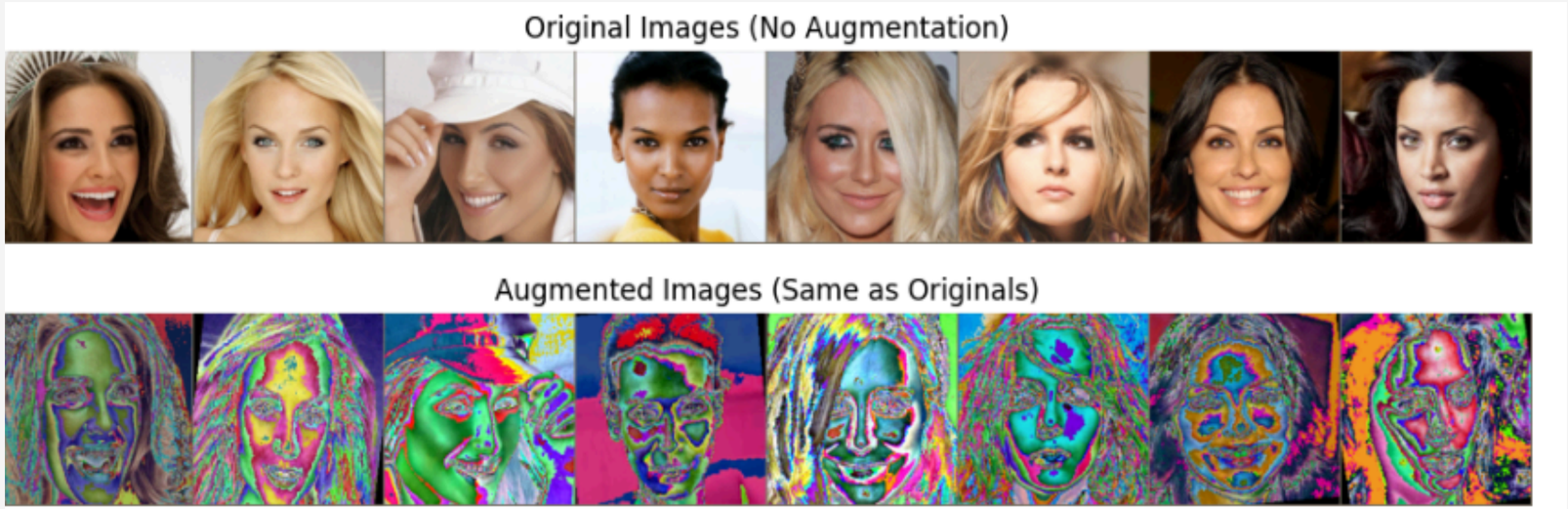
TOTAL DATASET UNTUK DATA TRAIN ADALAH 23999 DENGAN PEMBAGIAN
FEMALE: 15154
MALE: 8845

PREPROCESSING

- **BALANCING DATA**
- **AUGMENTASI**

```python
# Augmentasi tambahan untuk data training
transforms_train = transforms.Compose([
    transforms.Resize((224, 224)),                  # Ubah ukuran gambar ke 224x224
    transforms.RandomHorizontalFlip(),              # Flip horizontal acak
    transforms.RandomRotation(10),                  # Rotasi acak hingga 10 derajat
    transforms.RandomResizedCrop(224, scale=(0.8, 1.0)),  # Crop acak dengan skala
    transforms.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1),  # Jitter warna
    transforms.ToTensor(),                          # Konversi ke tensor
    transforms.Normalize([0.485, 0.456, 0.406],     # Normalisasi dengan mean & std dari ImageNet
                         [0.229, 0.224, 0.225])
])

# Transformasi tanpa augmentasi untuk data testing
transforms_test = transforms.Compose([
    transforms.Resize((224, 224)),                  # Ubah ukuran gambar ke 224x224
    transforms.ToTensor(),                          # Konversi ke tensor
    transforms.Normalize([0.485, 0.456, 0.406],     # Normalisasi dengan mean & std dari ImageNet
                         [0.229, 0.224, 0.225])
])
```

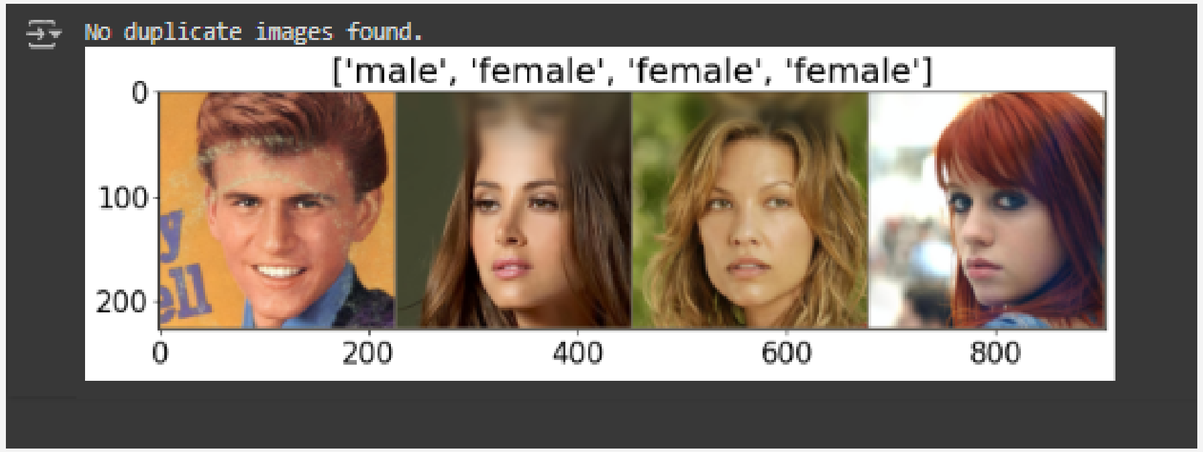



```
.ass distribution before balancing:
.ass female: 15154
.ass male: 8845

.ass distribution after balancing (approximation):
.ass female: ~23999
.ass male: ~23998
```

CEK DUPLIKASI DATASET

LINK TO COLAB




```
+ Kode + Teks

# Fungsi untuk menampilkan gambar (imshow)
def imshow(input, title):
    # torch.Tensor => numpy
    input = input.numpy().transpose((1, 2, 0))  # Mengubah tensor dari PyTorch ke numpy array dan menukar dimensi untuk ditampilkan
    # Membalik normalisasi pada gambar agar kembali ke nilai aslinya (undo normalization)
    mean = np.array([0.485, 0.456, 0.406])      # Mean dari normalisasi yang digunakan saat preprocessing
    std = np.array([0.229, 0.224, 0.225])       # Standar deviasi dari normalisasi
    input = std * input + mean                  # Kembalikan gambar ke rentang nilai aslinya
    input = np.clip(input, 0, 1)                # Potong nilai agar tetap berada di antara 0 dan 1 (untuk menyesuaikan batas piksel)
    # Menampilkan gambar menggunakan matplotlib
    plt.imshow(input)
    plt.title(title)                            # Beri judul pada gambar
    plt.show()                                  # Tampilkan gambar

# Fungsi untuk mengecek duplikasi gambar dalam satu batch
def check_for_duplicates(images):
    hashes = defaultdict(list)  # Dictionary untuk menyimpan hash gambar
    duplicate_indices = []      # List untuk menyimpan indeks gambar duplikat

    # Looping untuk setiap gambar dalam batch
    for idx, img in enumerate(images):
        img_np = img.numpy().transpose((1, 2, 0))  # Convert ke numpy array
        img_hash = generate_image_hash(img_np)     # Generate hash gambar
        if img_hash in hashes:
            duplicate_indices.append(idx)          # Simpan indeks gambar duplikat
        else:
            hashes[img_hash].append(idx)           # Simpan hash gambar baru

    return duplicate_indices

# Mengambil satu batch dari data training
iterator = iter(train_dataloader)  # Membuat iterator dari train_dataloader

# Visualisasi satu batch dari gambar training
inputs, classes = next(iterator)   # Mengambil batch pertama dari iterator (gambar dan label kelas)

# Mengecek apakah ada duplikasi dalam batch
duplicate_indices = check_for_duplicates(inputs)
if duplicate_indices:
    print(f'Duplicate images found at indices: {duplicate_indices}')
    # Jika ada duplikat, bisa melakukan tindakan tambahan di sini, misal membuang duplikat
else:
    print('No duplicate images found.')

# Jika tidak ada duplikat, lanjutkan visualisasi
out = torchvision.utils.make_grid(inputs[:4])  # Membuat grid dari 4 gambar pertama dalam batch
imshow(out, title=[class_names[x] for x in classes[:4]])  # Menampilkan gambar dengan label kelas yang sesuai
```

HYPERPARAMETER TUNING

- LEARNING RATE (LR=0.001)
- MOMENTUM (MOMENTUM=0.9)
- OPTIMIZER (OPTIM.SGD)
- LOSS FUNCTION (NN.CROSSENTROPYLOSS):

VGG

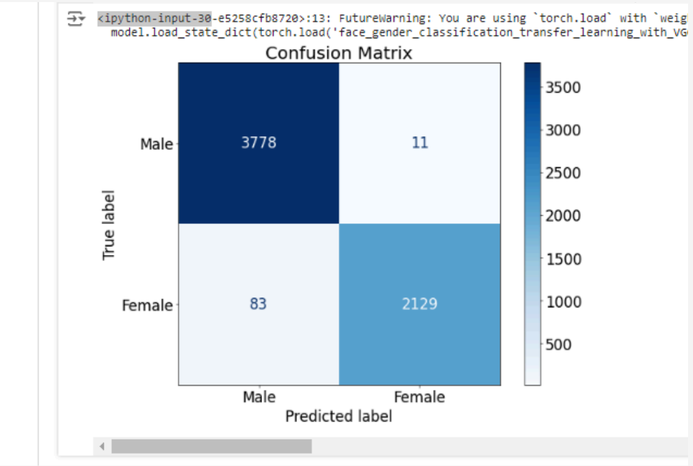
LINK TO COLAB

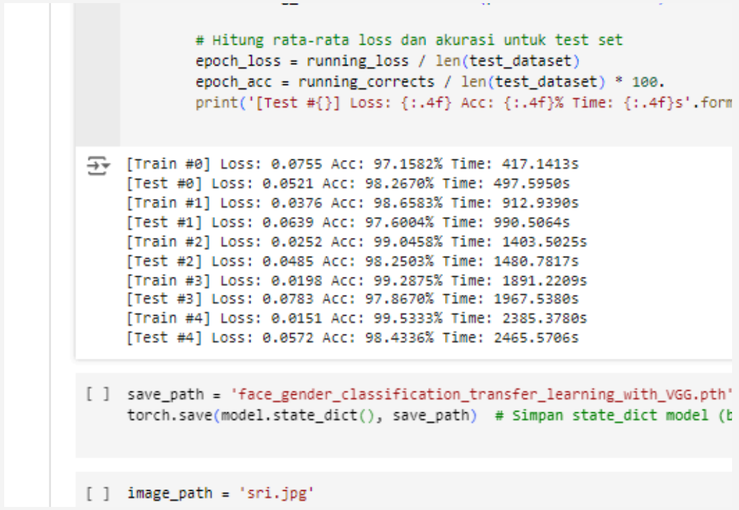
- Tahun Diperkenalkan: 2014

Fitur Utama:

- Menggunakan filter konvolusional kecil (3x3) dan arsitektur mendalam (hingga 19 lapisan).
- Arsitektur yang konsisten dengan kedalaman yang semakin meningkat.
- Mengutamakan keseragaman dalam ukuran filter dan lapisan penyatuan.

Dampak: Berpengaruh dalam menunjukkan pentingnya kedalaman dalam CNN.





GOOGLENET

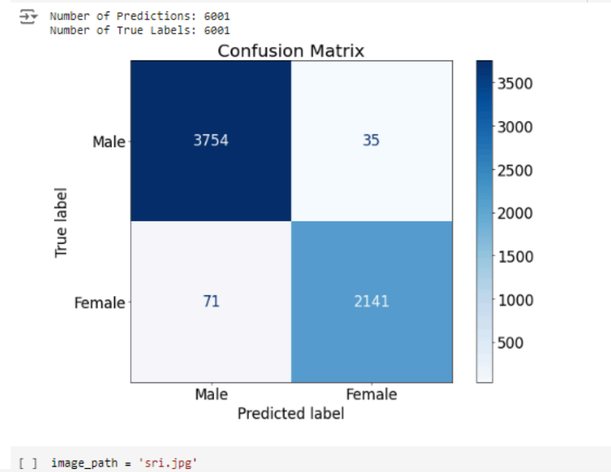
LINK TO COLAB

- Tahun Diperkenalkan: 2014

Fitur Utama:

- Memperkenalkan modul Inception, memungkinkan beberapa ukuran filter untuk menangkap fitur berbeda.
- Menggunakan konvolusi 1x1 untuk pengurangan dimensi, mengurangi biaya komputasi.
- Kedalaman 22 lapisan, dengan 9 modul awal.

Dampak: Efisiensi dalam kinerja, menekankan pemrosesan multi-skala.




```
epoch_loss = running_loss / len(test_dataset)
epoch_acc = running_corrects / len(test_dataset) * 100
print('[Test #{}] Loss: {:.4f} Acc: {:.4f}% Time: {:.4
```

```
[Train #0] Loss: 0.1255 Acc: 95.0873% Time: 304.5695s
[Test #0] Loss: 0.0594 Acc: 97.8504% Time: 376.2776s
[Train #1] Loss: 0.0546 Acc: 98.1999% Time: 685.0056s
[Test #1] Loss: 0.0530 Acc: 98.1503% Time: 757.2913s
[Train #2] Loss: 0.0389 Acc: 98.7041% Time: 1067.6936s
[Test #2] Loss: 0.0556 Acc: 98.1836% Time: 1137.0313s
[Train #3] Loss: 0.0247 Acc: 99.2583% Time: 1450.2351s
[Test #3] Loss: 0.0680 Acc: 98.1503% Time: 1523.2198s
[Train #4] Loss: 0.0200 Acc: 99.3333% Time: 1824.1704s
[Test #4] Loss: 0.0598 Acc: 98.2336% Time: 1893.8565s
```

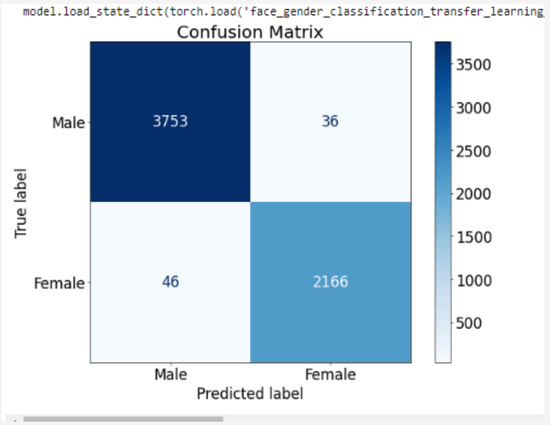
```
[ ] save_path = 'face_gender_classification_transfer_learning_with
    torch.save(model.state_dict(), save_path)  # Simpan state_dict
```

RESNET

LINK TO COLAB

- Tahun Diperkenalkan: 2015
- Fitur Utama:
- Memperkenalkan lewati koneksi atau koneksi sisa untuk memungkinkan gradien mengalir melalui jaringan tanpa menghilang.
- Dapat memiliki jaringan yang sangat dalam (misalnya 152 lapisan).
- Berfokus pada peningkatan waktu dan akurasi pelatihan untuk jaringan yang lebih dalam.

Dampak: Mencetak rekor baru dalam kompetisi ImageNet dan memengaruhi arsitektur selanjutnya.




```
            epoch_loss = running_loss / len(test_dataset)
            epoch_acc = running_corrects / len(test_dataset) *
            print('[Test #{}] Loss: {:.4f} Acc: {:.4f}% Time:
```

```
[Train #0] Loss: 0.0898 Acc: 96.7207% Time: 300.5880s
[Test #0] Loss: 0.0567 Acc: 97.6837% Time: 370.3020s
[Train #1] Loss: 0.0437 Acc: 98.4374% Time: 656.4217s
[Test #1] Loss: 0.0525 Acc: 98.4669% Time: 726.0654s
[Train #2] Loss: 0.0286 Acc: 99.0375% Time: 1017.7287s
[Test #2] Loss: 0.0442 Acc: 98.4836% Time: 1087.3307s
[Train #3] Loss: 0.0175 Acc: 99.4500% Time: 1376.7104s
[Test #3] Loss: 0.0581 Acc: 98.4669% Time: 1443.2731s
[Train #4] Loss: 0.0116 Acc: 99.6542% Time: 1736.8377s
[Test #4] Loss: 0.0535 Acc: 98.6336% Time: 1806.4469s
```

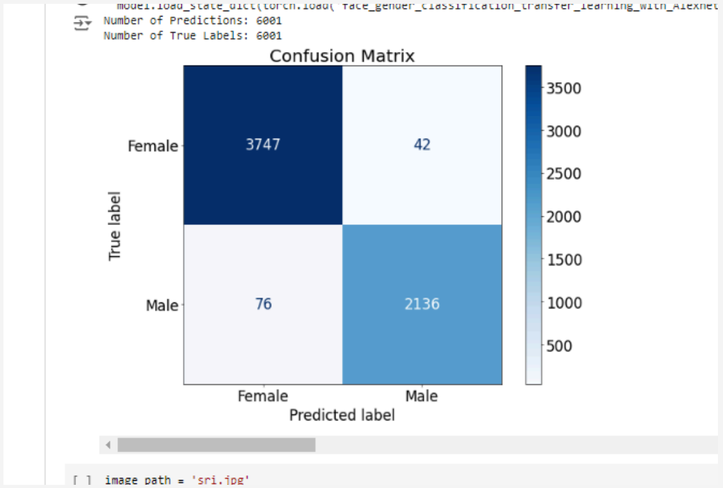
```
[37] save_path = 'face_gender_classification_transfer_learning_
     torch.save(model.state_dict(), save_path)
```

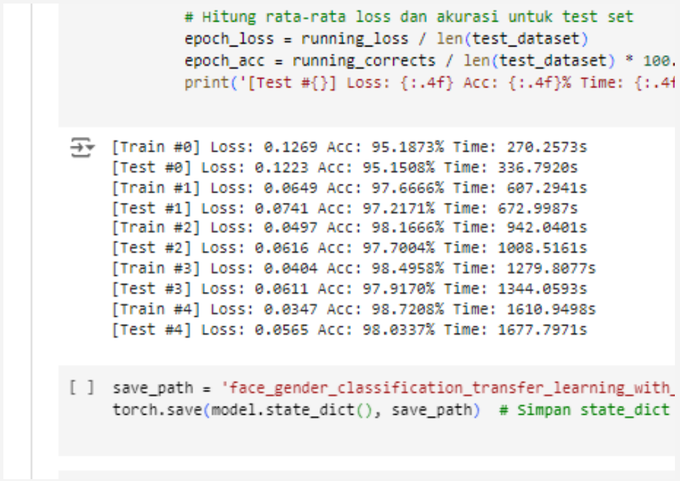
ALEXNET

LINK TO COLAB

- Tahun Diperkenalkan: 2012
- Fitur Utama:
- Model pembelajaran mendalam pertama yang memenangkan kompetisi ImageNet.
- Berisi 5 lapisan konvolusional diikuti oleh 3 lapisan yang terhubung sepenuhnya.
- Memperkenalkan fungsi aktivasi ReLU dan dropout untuk regularisasi.

Dampak: Mendemonstrasikan efektivitas pembelajaran mendalam dalam visi komputer.





CONTOH HASIL PREDIKSI SALAH

Feature	AlexNet	VGG	GoogLeNet	ResNet
Year Introduced	2012	2014	2014	2015
Depth	8 layers	16-19 layers	22 layers	34, 50, 101, 152 layers
Convolution Filters	11x11, 5x5	3x3	Mixed sizes	3x3
Unique Features	ReLU, dropout	Depth, uniformity	Inception modules, 1x1 convolutions	Residual connections
Computational Efficiency	Moderate	High	High	Very High
Performance on ImageNet	60% top-5 accuracy	71.3%	68.7%	76.5%

COMPARISON

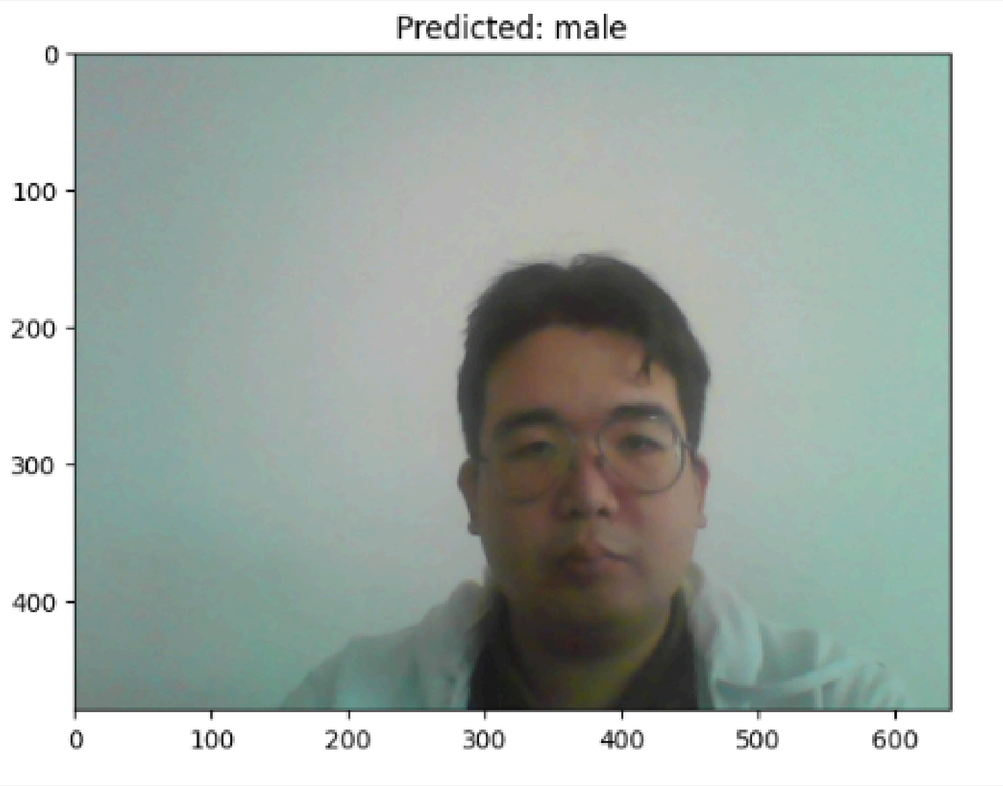
VGG

Accuracy: **98,443%**
Training Time Total: **2465,6 s**

LINK TO COLAB

GOOGLENET

Accuracy: **98,2336%**
Training Time Total: **1893,9 s**



RESNET

Accuracy: **98,6336%**
Training Time Total: **1806,4 s**

ALEXNET

Accuracy: **98,0337%**
Training Time Total: **1677, 8 s**

COMPARISON & IMPLEMENTATION

TABLE COMPARISSON

ALGORITMA	PRE-PROCESSING	ACCURACY
VGG16	Total DATASET 30000 DENGAN PEMBAGIAN Train dataset size: 23999 (80%) Test dataset size: 6001(20%) Class names: ['female', 'male']	Accuracy: **98,443%** Training Time Total: **2465,6 s**
ALEXNET	Total DATASET 30000 DENGAN PEMBAGIAN Train dataset size: 23999 (80%) Test dataset size: 6001(20%) Class names: ['female', 'male']	Accuracy: **98,0337%** Training Time Total: **1677, 8 s**
GOGGLENET	Total DATASET 30000 DENGAN PEMBAGIAN Train dataset size: 23999 (80%) Test dataset size: 6001(20%) Class names: ['female', 'male']	Accuracy: **98,2336%** Training Time Total: **1893,9 s**
RESNET	Total DATASET 30000 DENGAN PEMBAGIAN Train dataset size: 23999 (80%) Test dataset size: 6001(20%) Class names: ['female', 'male']	Accuracy: **98,6336%** Training Time Total: **1806,4 s**

KESIMPULAN

Berdasarkan hasil pengujian dan trainig dataset pada 4 algroitma , Algoritma

GITHUB

Link Address

Link To Github/kevin

Link To Github/thomas

Link To Github/Yohanes

Link To Github/Louis