



Computer Vision

Nama Kelompok :

1. Isaac Yeremia Nugroho /
223400016
2. Kevin Sidharta Handoyo /
223400006
3. Petrus Maxmiliano /
223400003
4. Michael Phrigyan Hartanto /
223400020



Face Recognition

Face recognition adalah teknologi yang mengidentifikasi atau memverifikasi identitas seseorang berdasarkan fitur wajah. Teknologi ini digunakan dalam berbagai bidang, seperti keamanan, pengawasan, otentikasi tanpa kontak, dan peningkatan pengalaman pengguna, misalnya dalam sistem pembayaran digital atau akses perangkat. Manfaatnya meliputi peningkatan keamanan, kemudahan verifikasi identitas, dan efisiensi operasional. Namun, tantangan terkait privasi dan etika juga perlu diperhatikan dalam penggunaannya.

Dataset



Dataset celeb A. dengan Label Pria dan Wanita

- Dataset yang digunakan dalam pelatihan ini 52000 gambar
- Dengan label male dan female
- Data dibagi dalam Training, Validation dan Testing

Tahap PRE-PRocessing

- Menggunakan Image Generator untuk Augmentasi Data (rescale, rotasi, flip, zoom, dsb.)
- Melakukan Balancing Data
- Membagi Data dalam 3 bagian
- Total Train Sample Images : 34638 Total Test Sample Images : 5802 Total Validation Sample Images : 2858



Balancing data

PRE PROCESSING

```
# Get the category distribution.
category_count = df["Gender"].value_counts()
print(category_count)

higher_category = list(category_count.index)[0]

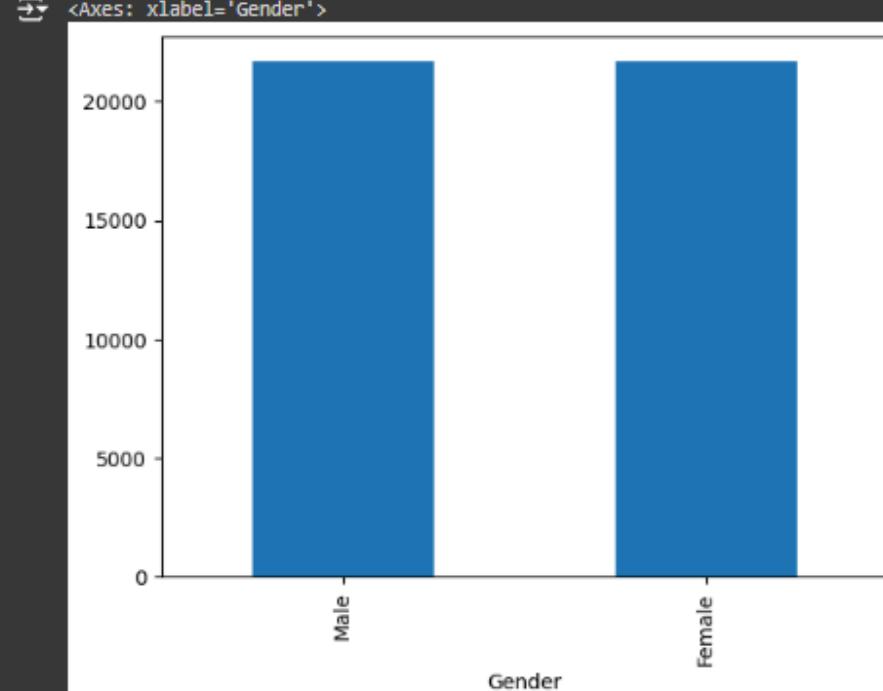
# Get the indices of the higher category images.
np.random.seed(42)
indices = df[df["Gender"] == higher_category].index
sample_size = category_count[0] - category_count[1]

# Drop the extra rows of female images to fix class imbalance problem.
drop_sample = np.random.choice(indices, sample_size, replace = False)
df = df.drop(drop_sample, axis = "index")

<ipython-input-68-6b82c4730800>:4: FutureWarning: Series._getitem_ treating keys as positions
sample_size = category_count[0] - category_count[1]

df["Gender"].value_counts().plot.bar()

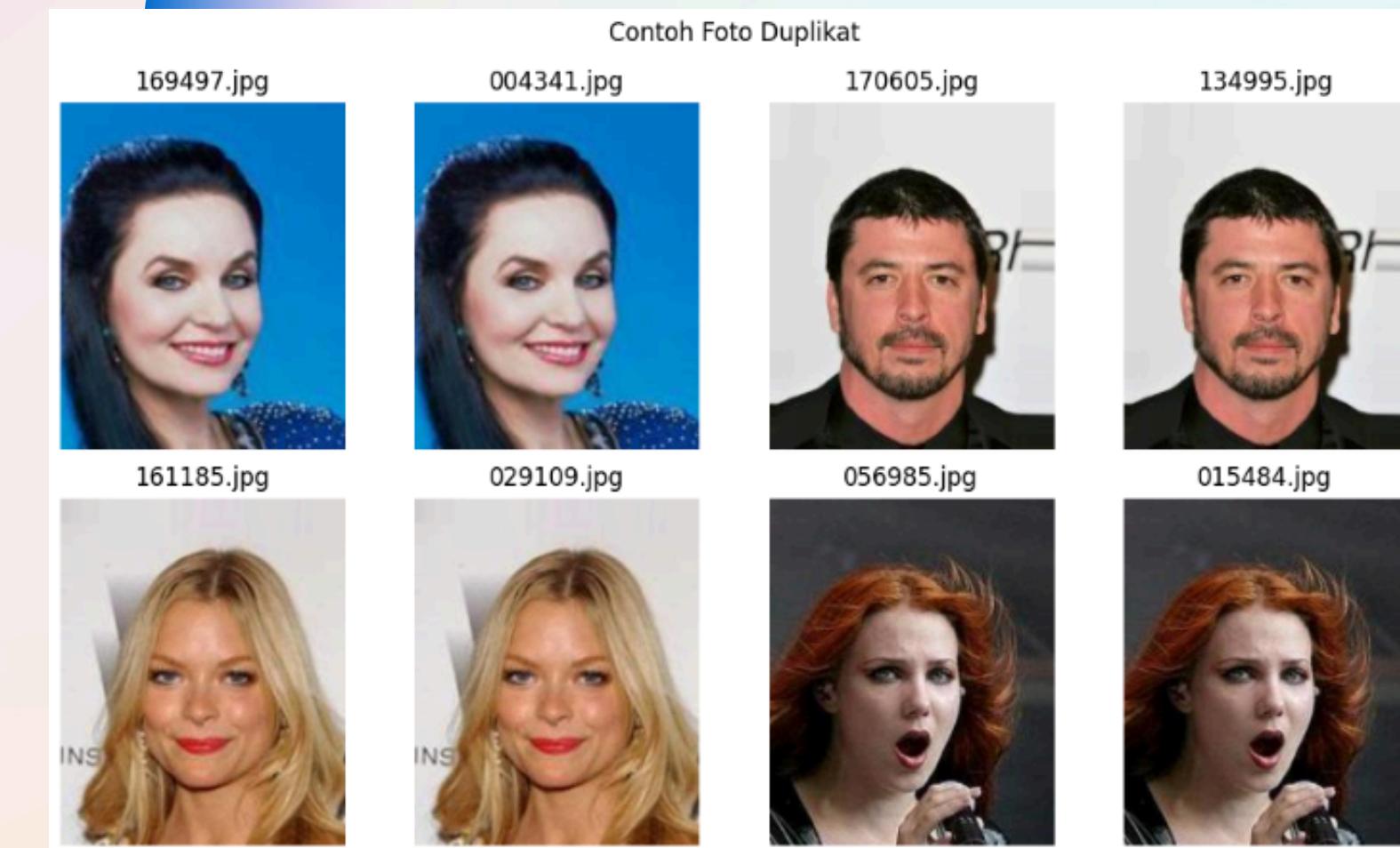
<Axes: xlabel='Gender'>
```





Remove Duplicate Picture

PRE PROCESSING





AUGMENTASI DATA



PRE PROCESSING

Lingkungan Pengujian



Dataset

Dataset yang digunakan
Terbatas pada 52000 gambar
dengan label male dan
female



Batch size

Batch size yang digunakan
adalah 128 batch size



Epoch

Epoch yang dilakukan dalam
pengujian ini 5 epoch



Learning rate

Learning rate yang
digunakan 0.00001

Arsitektur Alexnet

AlexNet adalah arsitektur jaringan saraf dalam yang dikembangkan oleh Alex Krizhevsky, Ilya Sutskever, dan Geoffrey Hinton, dan terkenal karena menang dalam kompetisi ImageNet Large Scale Visual Recognition Challenge (ILSVRC) pada tahun 2012. Dengan kedalaman 8 lapisan, termasuk 5 lapisan konvolusi diikuti oleh 3 lapisan fully connected, AlexNet merevolusi bidang computer vision dengan menunjukkan bahwa jaringan saraf dalam dapat mencapai akurasi yang jauh lebih tinggi dibandingkan metode tradisional. Arsitektur ini menggunakan teknik seperti ReLU (Rectified Linear Unit) sebagai fungsi aktivasi, pengurangan dimensi melalui max pooling, dan dropout untuk mengurangi overfitting. Selain itu, AlexNet memanfaatkan GPU untuk mempercepat pelatihan, yang menjadi salah satu faktor kunci kesuksesannya. Kontribusi AlexNet tidak hanya meningkatkan performa dalam pengenalan citra, tetapi juga menginspirasi pengembangan berbagai model jaringan saraf dalam lainnya yang lebih kompleks dan efisien.

```
[29] import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import Model
from tensorflow.keras.preprocessing.image import ImageDataGenerator

[30] Pembuatan Layer Terakhir

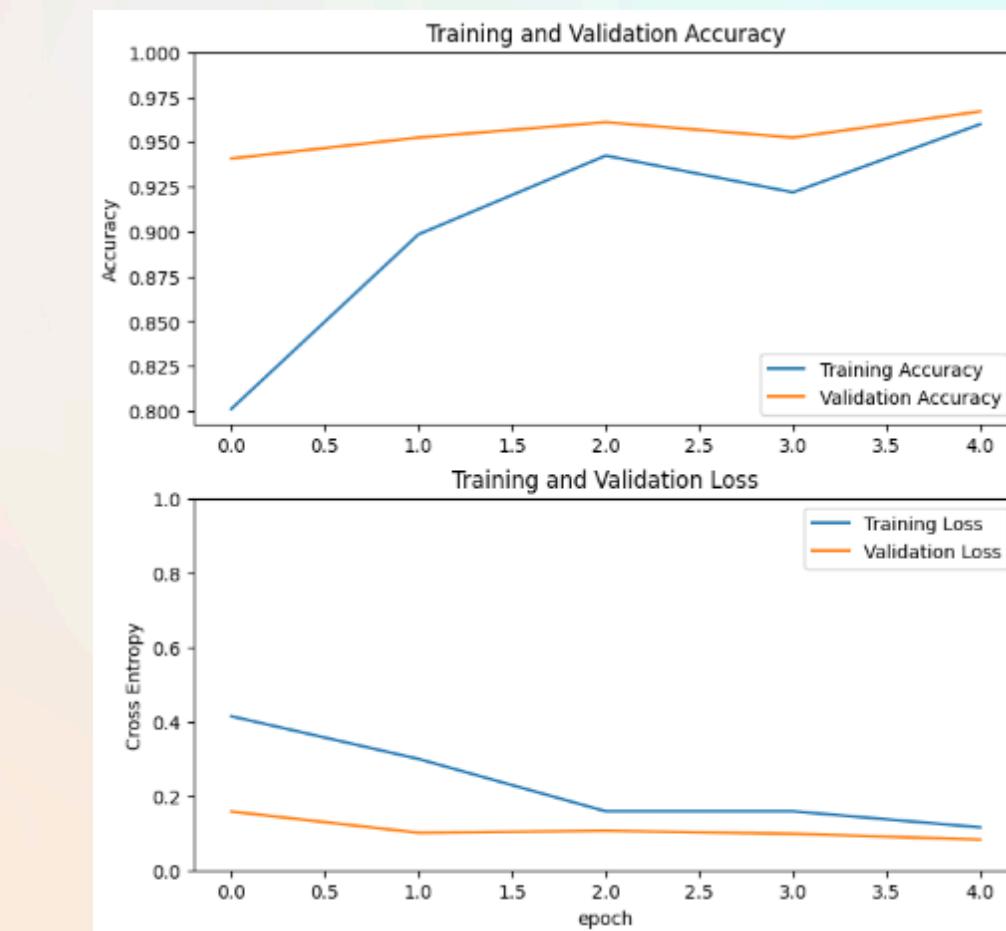
[31] from keras.optimizers import Adam
model = tf.keras.models.Sequential([
    # 1st conv
    tf.keras.layers.Conv2D(96, (11,11),strides=(4,4), activation='relu', input_shape=(218, 178, 3)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2, strides=(2,2)),
    # 2nd conv
    tf.keras.layers.Conv2D(256, (11,11),strides=(1,1), activation='relu',padding="same"),
    tf.keras.layers.BatchNormalization(),
    # 3rd conv
    tf.keras.layers.Conv2D(384, (3,3),strides=(1,1), activation='relu',padding="same"),
    tf.keras.layers.BatchNormalization(),
    # 4th conv
    tf.keras.layers.Conv2D(384, (3,3),strides=(1,1), activation='relu',padding="same"),
    tf.keras.layers.BatchNormalization(),
    # 5th Conv
    tf.keras.layers.Conv2D(256, (3, 3), strides=(1, 1), activation='relu',padding="same"),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2, strides=(2, 2)),
    # To Flatten layer
    tf.keras.layers.Flatten(),
    # To FC layer 1
    tf.keras.layers.Dense(4096, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    #To FC layer 2
    tf.keras.layers.Dense(4096, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(2, activation='sigmoid')
])
```

Model Arsitektur

Arsitektur Alexnet



Confusion Matrix



Training & Validation Accuracy and Loss

Arsitektur Alexnet

Gambar Male yang Diprediksi sebagai Female



Gambar Female yang Diprediksi sebagai Male



Arsitektur Resnet

ResNet50V2 adalah varian dari arsitektur ResNet (Residual Network) yang dirancang untuk meningkatkan performa model dalam pengenalan citra dan tugas pengolahan citra lainnya. Dengan 50 lapisan, ResNet50V2 mengimplementasikan blok residual yang memungkinkan aliran informasi lebih baik melalui shortcut connections, yang menghubungkan output dari satu blok ke blok berikutnya. Versi kedua ini memperkenalkan beberapa peningkatan, termasuk urutan konvolusi yang berbeda dan penggunaan Batch Normalization yang lebih efisien, yang membantu mempercepat proses pelatihan dan meningkatkan akurasi model. ResNet50V2 juga dikenal karena kemampuannya untuk menangkap fitur kompleks dari data gambar sambil mempertahankan jumlah parameter yang relatif rendah, menjadikannya pilihan populer untuk aplikasi seperti klasifikasi citra, deteksi objek, dan segmentasi.

```
# Transfer Learning dengan ResNet50
base_resnet_model = tf.keras.applications.ResNet50V2(weights='imagenet', include_top=False, input_shape= IMAGE_SIZE + (3,))

for layer in base_resnet_model.layers[-5:]: # Unfreeze 5 layer terakhir
    layer.trainable = True

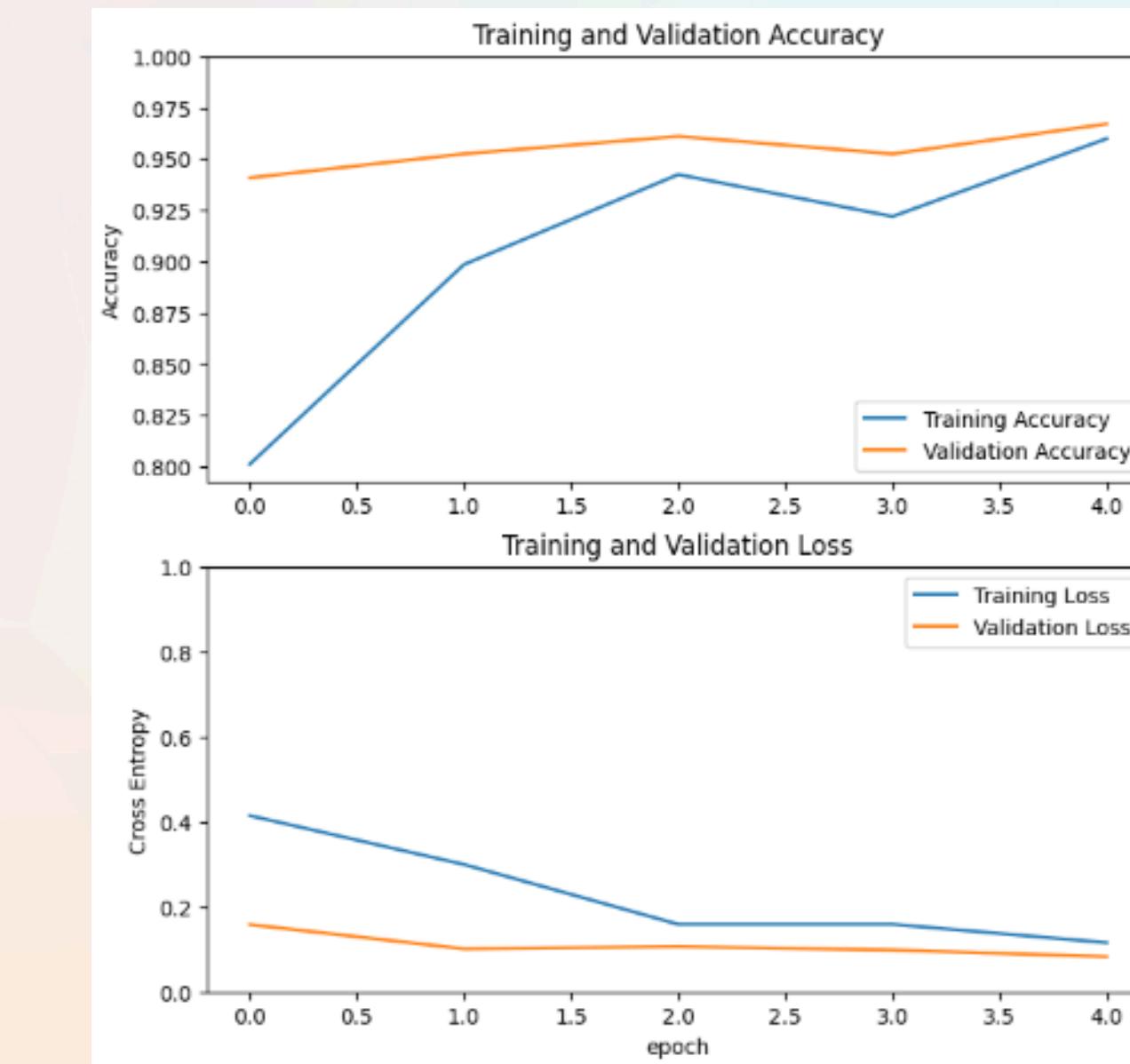
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet50v2_weights_tf_dim_ordering_t
94668760/94668760 0s 0us/step

# Membuat Arsitektur Fully Connected
resnet_model = Sequential([
    base_resnet_model,
    GlobalAveragePooling2D(),
    Dense(1024, activation='relu'),
    Dropout(0.2),
    Dense(256, activation='relu'),
    Dropout(0.2),
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dropout(0.2),
    Dense(2, activation="softmax")
])

# Kompile Model Deep Learning
base_learning_rate = 0.00001
resnet_model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics=['accuracy']
)
```

Model Arsitektur

Arsitektur Resnet



Arsitektur Resnet

Gambar Male yang Diprediksi sebagai Female



Gambar Female yang Diprediksi sebagai Male



Arsitektur GoogleNET

Arsitektur GoogLeNet adalah sebuah modifikasi arsitektur CNN yang berhasil menjadi model terbaik pada ILSVRC14. Arsitektur ini bekerja dengan mendekripsi citra dengan lapisan yang dimiliki sejumlah lima hingga 22 lapisan tetapi tetap memiliki akurasi yang tinggi. Konsep kerja arsitektur ini didasarkan pada activation values pada deep network yang tidak sepenuhnya penting karena terdapat value of zero akibat korelasi sebelumnya, sehingga dibutuhkan activation values yang tidak terkoneksi sepenuhnya. Untuk memenuhi kondisi tersebut, pada GoogLeNet terdapat lapisan inception module yang terinspirasi dari model visual cortex manusia yang berperan untuk mengoptimalkan sparse structure sehingga menunjang komputasi.

```
# Transfer Learning dengan GoogleNet (InceptionV3)
base_inception_model = tf.keras.applications.InceptionV3(weights='imagenet', include_top=False, input_shape=IMAGE_SIZE + (3,))

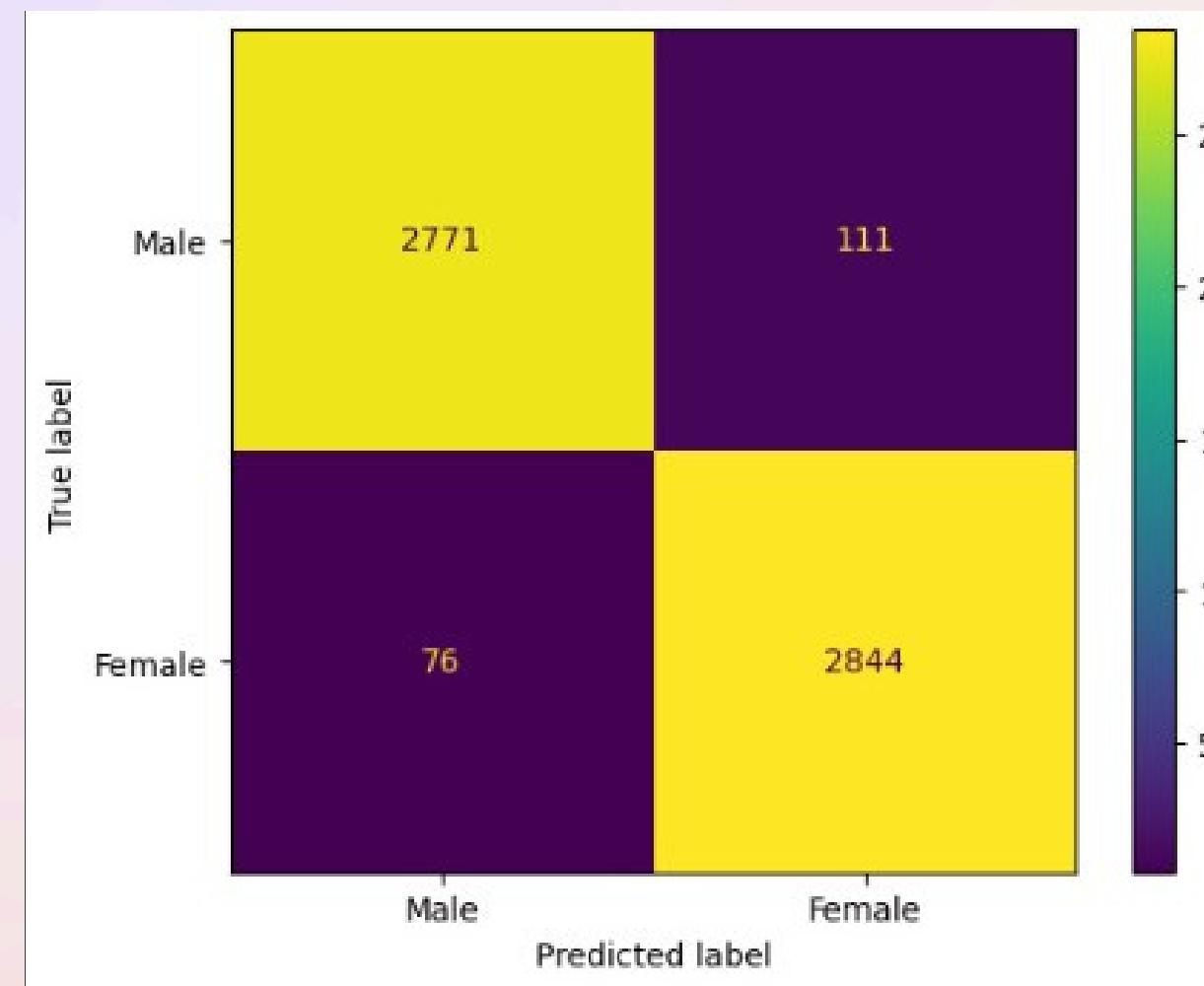
for layer in base_inception_model.layers[-5:]: # Unfreeze 5 layer terakhir
    layer.trainable = True

# Membuat arsitektur fully connected
inception_model = tf.keras.Sequential([
    base_inception_model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(2, activation='softmax') # Sesuaikan jumlah output sesuai dengan kebutuhan Anda
])

# Kompilasi model
base_learning_rate = 0.00001
inception_model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics=['accuracy']
)
```

Model Arsitektur

Arsitektur GoogleNET

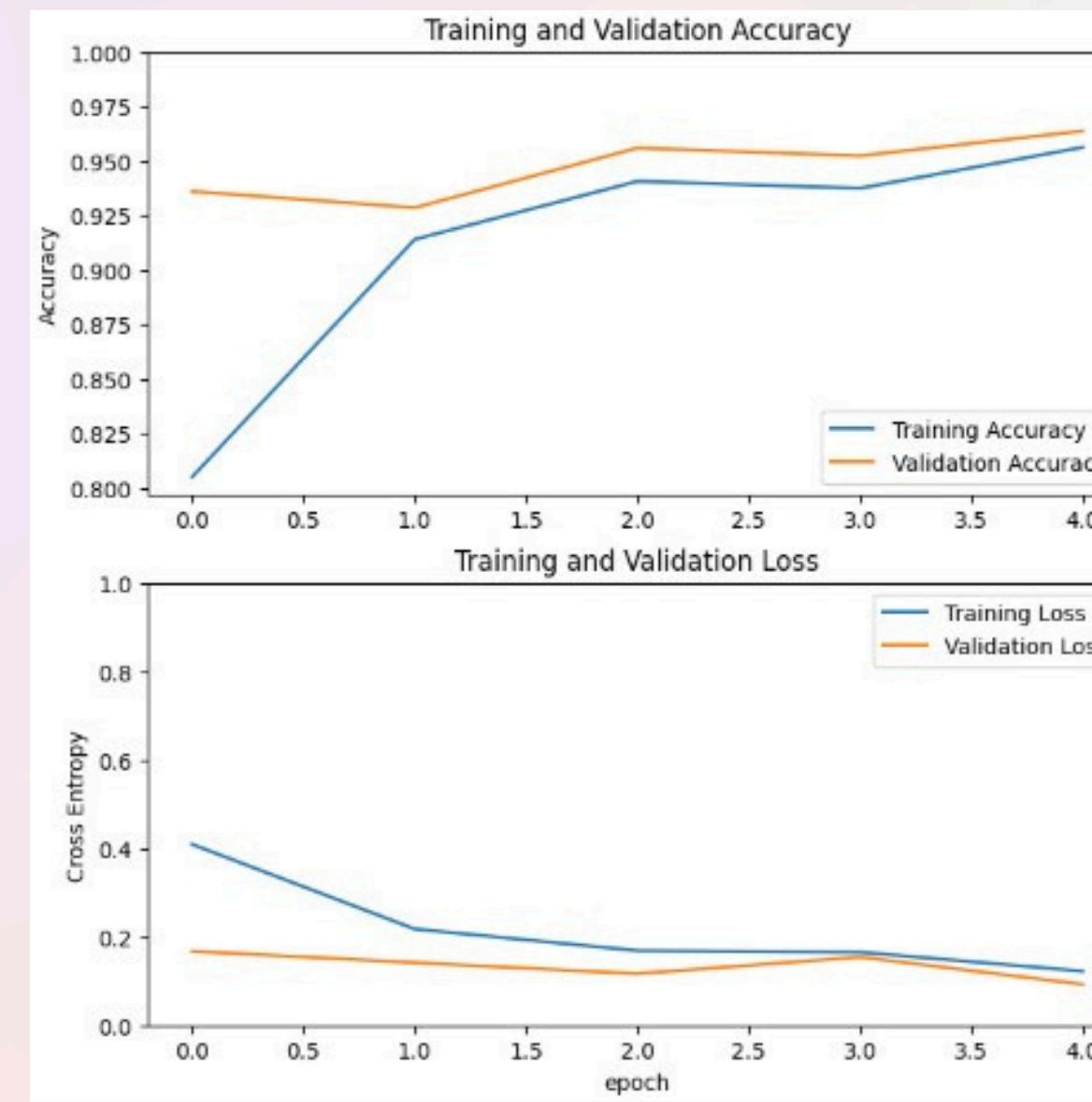


Confusion Matrix

```
Confusion Matrix:  
Male Female  
Male 2771 111  
Female 76 2844  
  
Classification Report:  
precision recall f1-score support  
Male 0.962437 0.973973 0.968170 2920.00000  
Female 0.973305 0.961485 0.967359 2882.00000  
accuracy 0.967770 0.967770 0.967770 0.96777  
macro avg 0.967871 0.967729 0.967765 5802.00000  
weighted avg 0.967835 0.967770 0.967767 5802.00000  
  
[40] # Matrix Evaluasi  
print(classification_report(test_df['Gender'], test_df['P  
  
# Code Modif from Chat GPT  
  
precision recall f1-score support  
Male 0.96 0.97 0.97 2920  
Female 0.97 0.96 0.97 2882  
  
accuracy 0.97 0.97 0.97 5802  
macro avg 0.97 0.97 0.97 5802  
weighted avg 0.97 0.97 0.97 5802
```

Confusion Matrix

Arsitektur GoogleNET



Arsitektur GoogleNET

Gambar Male yang Diprediksi sebagai Female



Gambar Female yang Diprediksi sebagai Male



Aktual Male diprediksi Female

Aktual Female diprediksi Male

Arsitektur VGG-16

VGG (Visual Geometri Group) adalah arsitektur convolutional neural network yang pertama kali diperkenalkan pada tahun 2014 oleh tim peneliti dari Universitas Oxford. Model arsitektur VGG telah menjadi salah satu model pembelajaran yang paling populer dan banyak digunakan untuk visi komputer, termasuk klasifikasi gambar, deteksi objek, dan segmentasi. VGG sendiri memiliki dua model yaitu VGG-16 dan VGG-19, yang dipakai saat ini adalah model VGG-16. Arsitektur VGG-16 menggunakan 16 layer dengan bobot dan dianggap sebagai salah satu arsitektur model visi terbaik hingga saat ini. 13 layer merupakan lapisan konvolusi, 2 lapisan digunakan sebagai fully connected, dan 1 lapisan lagi untuk klasifikasi.

```
[29] from tensorflow.keras.models import Sequential
      from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout

      # Membuat Arsitektur Fully Connected dengan VGG16
      vgg_model = Sequential([
          base_vgg_model,
          GlobalAveragePooling2D(),
          Dense(1024, activation='relu'),
          Dropout(0.2),
          Dense(256, activation='relu'),
          Dropout(0.2),
          Dense(64, activation='relu'),
          Dropout(0.2),
          Dense(32, activation='relu'),
          Dropout(0.2),
          Dense(2, activation="softmax")
      ])
```

Model Arsitektur

Arsitektur VGG-16



Confusion Matrix



Training & Validation Accuracy and Loss

Arsitektur VGG-16

Gambar Male yang Diprediksi sebagai Female



Gambar Female yang Diprediksi sebagai Male



Arsitektur DenseNet

DenseNet201 adalah varian dari arsitektur jaringan saraf dalam yang dikenal sebagai DenseNet (Densely Connected Convolutional Networks), yang memperkenalkan konsep koneksi padat antara layer. Dalam DenseNet, setiap layer menerima input dari semua layer sebelumnya, bukan hanya dari layer terdekat, yang membantu meningkatkan efisiensi model dan mengatasi masalah vanishing gradient. Dengan 201 layer dan struktur dense block yang saling terhubung, DenseNet201 mampu menangkap fitur dengan baik sambil menggunakan lebih sedikit parameter dibandingkan dengan arsitektur lain yang lebih dalam. Karena keunggulannya, DenseNet201 sering digunakan dalam berbagai tugas computer vision, seperti pengenalan citra dan deteksi objek.

```
[ ] # Transfer Learning with DenseNet201
base_densenet_model = tf.keras.applications.DenseNet201(weights='imagenet', include_top=False, input_shape=IMAGE_SIZE + (3,))

# Unfreeze the last 5 layers of the base model
for layer in base_densenet_model.layers[:-5]: # Freeze all layers except the last 5
    layer.trainable = False

# Create your full model
densenet_model = Sequential([
    base_densenet_model, # Base DenseNet201
    GlobalAveragePooling2D(), # Global Average Pooling
    Dense(1024, activation='relu'), # Fully connected layers
    Dropout(0.2),
    Dense(256, activation='relu'),
    Dropout(0.2),
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dropout(0.2),
    Dense(2, activation='softmax') # Final classification layer (adjust output size as needed)
])

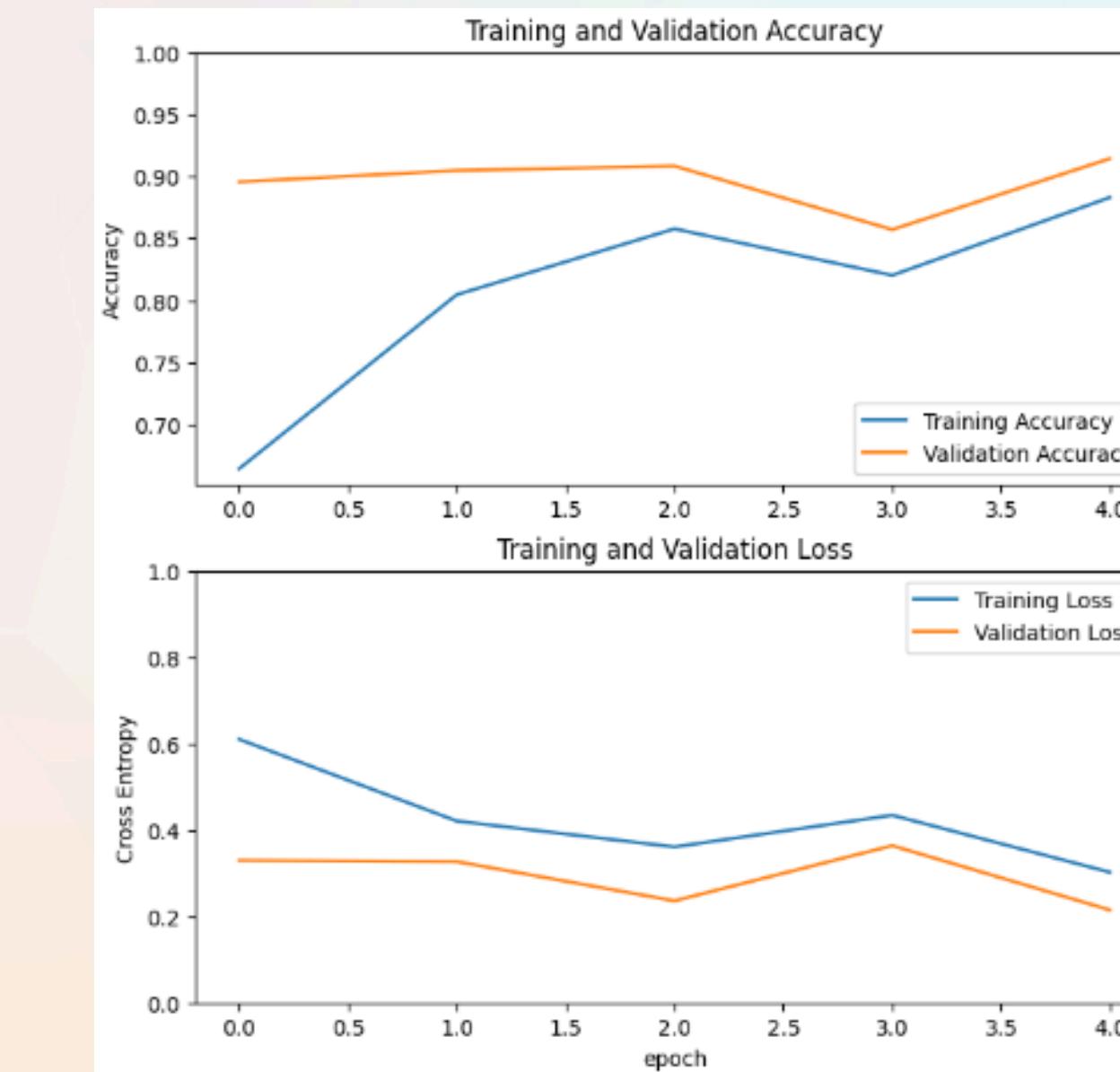
# Compile the model
base_learning_rate = 0.00001
densenet_model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False), # Remove `from_logits` since softmax is used
    metrics=['accuracy']
)
```

Model Arsitektur

Arsitektur DenseNet



Confusion Matrix



Training & Validation Accuracy and Loss

Arsitektur DenseNet

Gambar Male yang Diprediksi sebagai Female



Aktual Male diprediksi Female

Gambar Female yang Diprediksi sebagai Male



Aktual Female diprediksi Male

Akurasi Epoch

	Alexnet	Resnet	GoogleNet	VGG-16	DenseNet
EPOCH 5	0.86	0.97	0.97	0.96	0.92
EPOCH 10	0.87	0.98	0.97	0.98	0.93
EPOCH 15	0.88	0.98	0.97	0.93	0.93

Dataset: 52000
Batch Size: 128

Akurasi Dataset

	Alexnet	Resnet	GoogleNet	VGG-16	DenseNet
Dataset 52000	0.86	0,97	0.97	0.96	0.92

Epoch: 5

Batch Size: 128

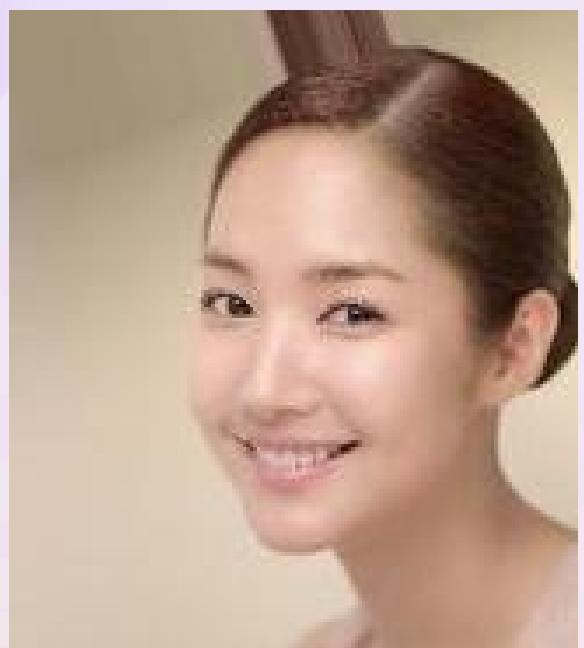
Akurasi Batch size

	Alexnet	Resnet	GoogleNet	VGG-16	DenseNet
BATCH SIZE 32	0.86	0.98	0.97	0.98	0.93
BATCH SIZE 64	0.87	0.97	0.97	0.98	0.92
BATCH SIZE 128	0.86	0.97	0.97	0.96	0.92

Dataset: 52000

Epoch: 5

Prediksi



Gambar 1



Gambar 2



Gambar 3

Pengujian Prediksi

	Alexnet	Resnet	GoogleNet	VGG-16	DenseNet
Gambar 1	Female	Female	Female	Female	Female
Gambar 2	Female	Male	Female	Female	Female
Gambar 3	Male	Male	Male	Male	Male

Kesimpulan

Berdasarkan hasil penelitian yang dilakukan ResNet50V2 dan GoogleNet adalah model yang memiliki akurasi tertinggi yaitu 97% dibanding 3 model lainnya seperti AlexNet 92%, DenseNet 92%, dan VGG 96%



ALEXNET

https://colab.research.google.com/drive/1vx7acUu4stmuCIZzhcppTjlq8VDLt_bA?usp=sharing

DENSENET

<https://colab.research.google.com/drive/1XVh1zZAym7Tre6TgXosakwnnyVp0uLby?usp=sharing>

GOOGLENET

<https://colab.research.google.com/drive/1Om6r6mMclx2KQXF2OF8i5iMzNpi6qrdk?usp=sharing>

VGG-16

https://colab.research.google.com/drive/1Hh15QILnoMJAwksWuuT9EbdB_KgSZfrf?usp=sharing

RESNET

<https://colab.research.google.com/drive/1zwyix6CTb4ugNMDKu5txwd2K2DdP9RTW?usp=sharing>

link
Github

Kevin

<https://github.com/Kevinhan1/Face-Recognition-male-female-prediction>

Petrus

<https://github.com/maxmiloiano/Face-Recognition-Gender.git>

Isaac

<https://github.com/isaacyeremia/Face-Recognition.git>

Michael

<https://github.com/lcgamingsby/Viscom>

Thank You.
Thank You.
Thank You.