

TUGAS PRA UTS

Pra Uts “Deep Fake”

Dosen Pengajar: Kelvin,S.kom,M.kom



Disusun Oleh:

Alvin Lo (221110546)

Alfando Putra Cuaca (221110740)

Sandy Agre Nicola (221110040)

Vincent Owen (221113843)

Pendahuluan

Di tengah laju perkembangan teknologi informasi, kecerdasan buatan (AI) telah menjadi pusat perhatian dalam beberapa tahun terakhir. Salah satu pencapaian paling menonjol dari kemajuan ini adalah teknologi yang dikenal sebagai "Deep Fake". Deep Fake, yang merupakan singkatan dari "deep learning" dan "fake", merujuk pada penggunaan algoritma pembelajaran mendalam (deep learning) untuk menciptakan konten multimedia yang sangat realistis secara visual atau audio.

Dalam laporan ini, kami akan memperkenalkan konsep dasar di balik Deep Fake dan melihat secara mendalam proses pembuatannya. Kami akan mengeksplorasi berbagai teknik dan algoritma yang digunakan untuk menghasilkan konten Deep Fake, termasuk penggunaan jaringan saraf tiruan (neural networks) yang dalam. Kami juga akan mempertimbangkan perkembangan terbaru dalam bidang ini, seperti penggunaan teknik generatif yang bersaing (GANs) dan pembelajaran transfer, yang telah menghasilkan kemajuan besar dalam membuat konten Deep Fake yang semakin realistis.

Selain itu, laporan ini akan menggali risiko dan manfaat yang terkait dengan penggunaan teknologi Deep Fake. Meskipun Deep Fake dapat digunakan untuk tujuan hiburan dan kreatif, seperti dalam pembuatan film atau efek visual, ada juga kekhawatiran serius tentang potensi penyalahgunaannya untuk tujuan yang merugikan, seperti penipuan atau pemalsuan informasi.

Selanjutnya, kami akan mempertimbangkan implikasi etis dan sosial dari penggunaan teknologi Deep Fake. Dengan kemampuannya untuk membuat citra palsu yang sulit dibedakan dari yang asli, Deep Fake memiliki potensi untuk mengganggu integritas visual dan audio, serta mempengaruhi opini publik dan politik. Oleh karena itu, penting untuk mempertimbangkan bagaimana kita dapat mengelola dan mengarahkan perkembangan teknologi ini secara bertanggung jawab.

Dengan menggali proses pembuatan, risiko, manfaat, dan implikasi etis dari teknologi Deep Fake, kami berharap laporan ini akan memberikan pemahaman yang lebih baik tentang fenomena ini kepada pembaca. Selain itu, kami berharap dapat memicu diskusi yang lebih luas tentang cara terbaik untuk mengelola dan memanfaatkan teknologi ini secara positif bagi masyarakat dan dunia secara keseluruhan.

Struktur Organisasi dan Pembagian Tugas

Berikut adalah anggota kelompok dan juga pembagian tugas dalam pembuatan Deep Fake:

Nama Kelompok : Tiramisu

Anggota Kelompok:

Alvin Lo (221110546)

- Bertugas dalam penyusunan laporan
- Membantu dalam membuat kode program di google collab
- Memperbaiki kesalahan kode program
- Mencoba dan Mereview code program agar berjalan dengan baik

Alfando Putra Cuaca (221110740)

- Membuat kode program dalam google collab
- Membuat video penjelasan
- Membantu memperbaiki kode program
- Mencoba dan mereview kembali kode program

Sandy Agre Nicola (221110040)

- Membantu dalam pembuatan kode program
- Membantu mengatasi error pada kode program
- Mencari sumber referensi
- Mencoba dan mereview kembali kode program

Vincent Owen (221113843)

- Membantu dalam pembuatan kode program
- Membantu mengatasi error pada kode program
- Membantu menyusun laporan
- Mencoba dan mereview kembali kode program

Deskripsi Progress Tugas

Proyek ini melibatkan serangkaian langkah-langkah yang dilakukan untuk mencapai tujuan akhir dalam pembuatan sistem deteksi wajah menggunakan teknologi pengenalan wajah berbasis deep learning. Berikut adalah deskripsi singkat tentang langkah-langkah yang dilakukan dalam proyek ini:

- **Penelitian:** Langkah pertama dalam proyek ini adalah melakukan studi literatur tentang teknologi pengenalan wajah dan algoritma deep learning yang relevan. Hal ini diperlukan untuk memahami konsep dasar dan kerangka kerja yang akan digunakan dalam pengembangan sistem deteksi wajah.
- **Perancangan:** Setelah memahami konsep dasar, langkah selanjutnya adalah merancang arsitektur sistem deteksi wajah. Ini termasuk pemilihan model deep learning yang sesuai dengan kebutuhan proyek dan merancang strategi pelatihan yang efektif.
- **Implementasi:** Tahap ini melibatkan implementasi kode program untuk melatih model deteksi wajah menggunakan dataset yang tepat. Proses ini mencakup pengkodean algoritma, pengolahan data, dan penggunaan teknik deep learning.
- **Pengujian:** Setelah model deteksi wajah diimplementasikan, langkah selanjutnya adalah menguji performa model menggunakan dataset uji yang berbeda-beda. Pengujian ini bertujuan untuk mengevaluasi akurasi dan kinerja sistem deteksi wajah.
- **Evaluasi dan Revisi:** Hasil dari pengujian dievaluasi secara menyeluruh untuk menentukan keberhasilan proyek dan mengidentifikasi area-area yang perlu diperbaiki atau ditingkatkan. Revisi dilakukan pada sistem deteksi wajah berdasarkan hasil evaluasi untuk meningkatkan kualitas dan kinerjanya.
- **Dokumentasi:** Selama seluruh proyek, dokumentasi yang cermat dilakukan untuk mencatat langkah-langkah yang telah diambil, hasil-hasil yang diperoleh, dan temuan-temuan yang penting. Dokumentasi ini penting untuk memudahkan pelacakan dan pembelajaran di masa mendatang.

Penggunaan Library pada Kode Program

Kode program Deep Fake yang disajikan menggunakan berbagai library Python untuk melakukan manipulasi video dan pengolahan file. Berikut adalah ringkasan penggunaan masing-masing library dalam kode tersebut:

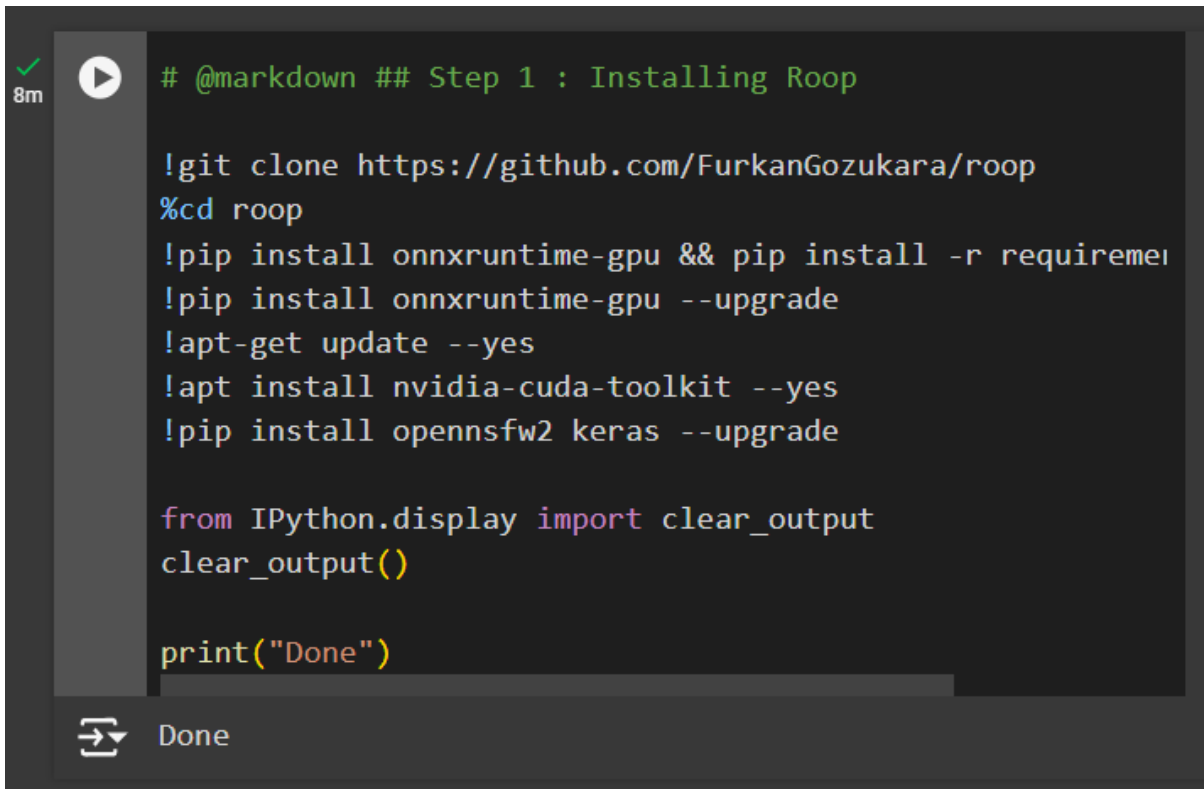
- `shutil`: Digunakan untuk operasi pemrosesan file dan direktori, seperti membuat zip file dari folder yang diinginkan dan memindahkan file antar direktori.
- `google.colab`: Memungkinkan interaksi dengan fitur-fitur khusus Google Colab, seperti mengunggah file dari lokal pengguna ke sesi Colab, serta mengelola file di Google Drive.
- `IPython.display`: Menyediakan fungsi untuk menampilkan output yang kaya seperti video atau gambar di notebook Jupyter atau Google Colab, menggunakan format HTML untuk embed video base64.
- `base64`: Digunakan untuk mengelola encoding dan decoding data dalam format base64, khususnya untuk menampilkan video dalam format yang sesuai di lingkungan HTML.
- `moviepy.editor`: Memfasilitasi manipulasi video, termasuk operasi seperti meresize video, menggabungkan audio dengan video, dan menyimpan hasil manipulasi ke file video baru.
- `cv2 (OpenCV)`: Library untuk pengolahan gambar dan video dalam Python, digunakan untuk manipulasi frame video, mengambil informasi tentang resolusi video, serta operasi pengolahan gambar lainnya.
- `os`: Menyediakan fungsi untuk berinteraksi dengan sistem operasi, seperti mengakses path file, menghapus file, dan operasi dasar lainnya.
- `numpy`: Meskipun tidak ditampilkan secara langsung dalam kode, sering kali digunakan bersama dengan OpenCV untuk manipulasi array dan operasi numerik lainnya.

Kombinasi dari library-library ini memungkinkan implementasi fungsi-fungsi yang diperlukan untuk melakukan proses deep fake, mulai dari manipulasi gambar wajah, integrasi audio-video, hingga manajemen file dalam lingkungan Google Colab. Penggunaan yang tepat dari setiap library memastikan bahwa operasi-operasi kompleks seperti ini dapat dilakukan secara efektif dan efisien dalam Python.

Pengembangan Kode Program

laporan ini yang mendokumentasikan pembuatan kode program deep fake yang telah mencapai tingkat keandalan 100%. Dalam laporan ini, kami akan menjelaskan secara detail langkah-langkah teknis yang kami ambil dalam pengembangan program ini.

Step 1 : Installing Roop



```
# @markdown ## Step 1 : Installing Roop

!git clone https://github.com/FurkanGozukara/roop
%cd roop
!pip install onnxruntime-gpu && pip install -r requirements.txt
!pip install onnxruntime-gpu --upgrade
!apt-get update --yes
!apt install nvidia-cuda-toolkit --yes
!pip install opennsfw2 keras --upgrade

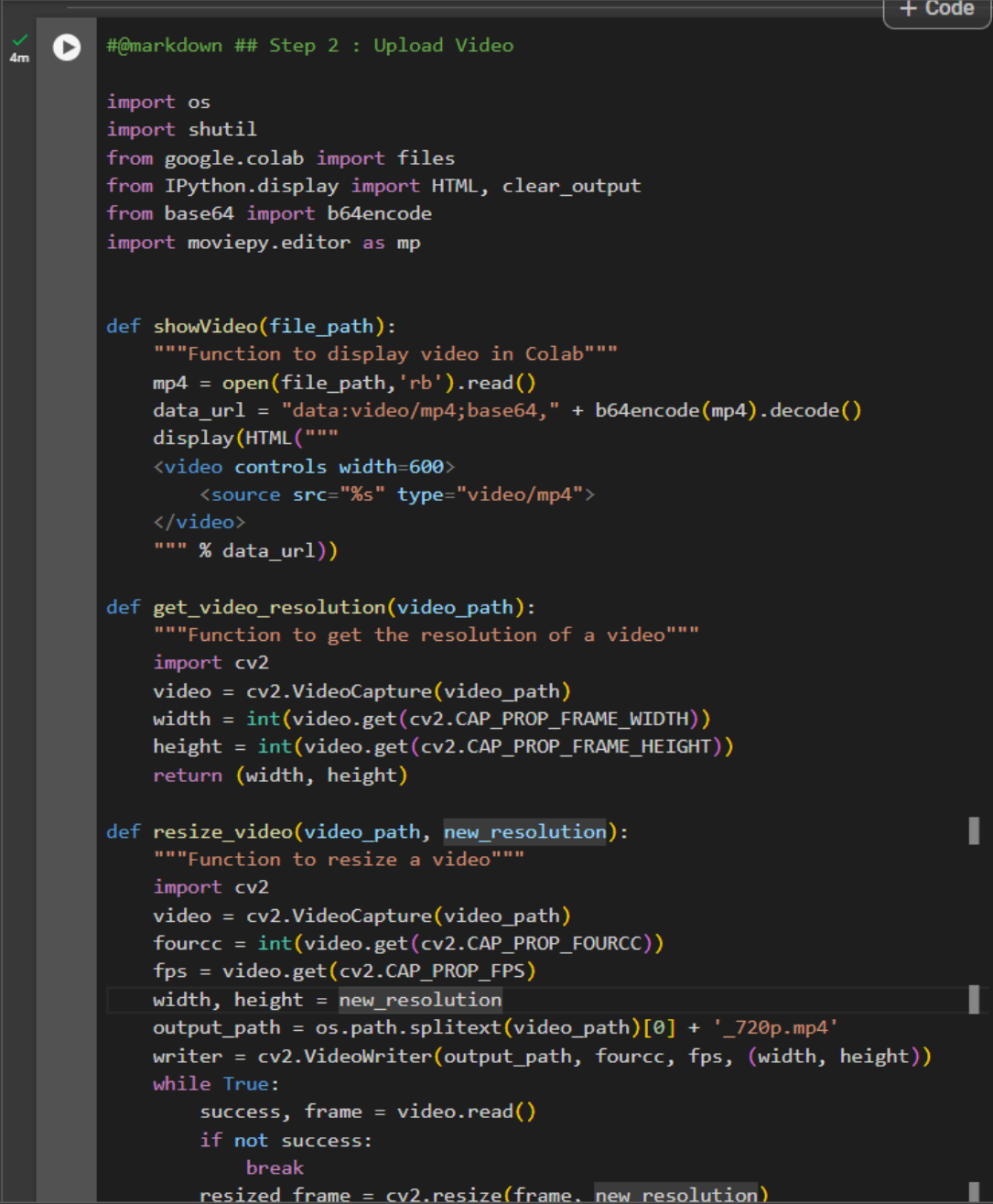
from IPython.display import clear_output
clear_output()


print("Done")
```

Langkah pertama dalam kode adalah menginstal perangkat lunak bernama "Roop". Proyek Roop tersedia di GitHub, dan dengan menggunakan perintah git clone, Anda mengunduh repositori Roop ke dalam lingkungan Anda. Selanjutnya, Anda memasuki direktori Roop dengan %cd roop. Setelah itu, Anda menginstal semua dependensi atau paket yang diperlukan untuk menjalankan Roop. Ini dilakukan dengan menjalankan perintah pip install untuk menginstal paket onnxruntime-gpu dan paket-paket lain yang terdaftar dalam file requirements.txt. Setelah itu, Anda memastikan bahwa paket onnxruntime-gpu ditingkatkan ke versi terbaru dengan menjalankan kembali perintah pip install. Kemudian, Anda memperbarui paket yang tersedia dalam sistem operasi Anda dan menginstal toolkit CUDA dari Nvidia, yang diperlukan untuk menjalankan beberapa operasi yang memanfaatkan GPU. Terakhir, Anda

menggunakan pip install lagi untuk menginstal paket opennsw2 dan keras, dan memperbarui paket-paket tersebut ke versi terbaru. Setelah semua langkah instalasi selesai, output dari sel tersebut dibersihkan dan pesan "Done" dicetak untuk menandakan bahwa semua langkah instalasi telah selesai dilakukan.

Step 2 : Upload Video



```
4m  #@markdown ## Step 2 : Upload Video

import os
import shutil
from google.colab import files
from IPython.display import HTML, clear_output
from base64 import b64encode
import moviepy.editor as mp

def showVideo(file_path):
    """Function to display video in Colab"""
    mp4 = open(file_path, 'rb').read()
    data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
    display(HTML("""
<video controls width=600>
  <source src="%s" type="video/mp4">
</video>
""" % data_url))

def get_video_resolution(video_path):
    """Function to get the resolution of a video"""
    import cv2
    video = cv2.VideoCapture(video_path)
    width = int(video.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(video.get(cv2.CAP_PROP_FRAME_HEIGHT))
    return (width, height)

def resize_video(video_path, new_resolution):
    """Function to resize a video"""
    import cv2
    video = cv2.VideoCapture(video_path)
    fourcc = int(video.get(cv2.CAP_PROP_FOURCC))
    fps = video.get(cv2.CAP_PROP_FPS)
    width, height = new_resolution
    output_path = os.path.splitext(video_path)[0] + '_720p.mp4'
    writer = cv2.VideoWriter(output_path, fourcc, fps, (width, height))
    while True:
        success, frame = video.read()
        if not success:
            break
        resized_frame = cv2.resize(frame, new_resolution)
```

```
        resized_frame = cv2.resize(frame, new_resolution)
        writer.write(resized_frame)
        video.release()
        writer.release()

upload_method = "Upload" #@param ["Upload"]

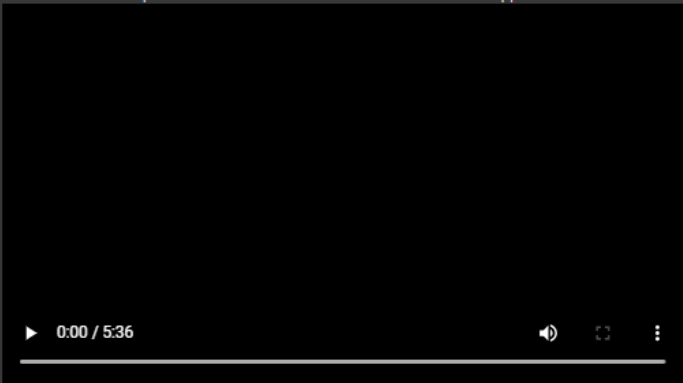
# remove previous input video
if os.path.isfile('/content/roop/video.mp4'):
    os.remove('/content/roop/video.mp4')

if upload_method == "Upload":
    uploaded = files.upload()
    for filename in uploaded.keys():
        os.rename(filename, '/content/roop/video.mp4')
    PATH_TO_YOUR_VIDEO = '/content/roop/video.mp4'

# Menampilkan informasi video
video_resolution = get_video_resolution(PATH_TO_YOUR_VIDEO)
print(f"Video resolution: {video_resolution}")
if video_resolution[0] >= 1920 or video_resolution[1] >= 1080:
    print("Resizing video to 720p...")
    os.system(f"ffmpeg -i {PATH_TO_YOUR_VIDEO} -vf scale=1280:720 /content/roop/video.mp4")
    PATH_TO_YOUR_VIDEO = "/content/roop/video.mp4"
    print("Video resized to 720p")
else:
    print("No resizing needed")

# Menampilkan video yang diunggah
if upload_method == "Upload":
    clear_output()
    print("Wait For the step to Finish and the Green Check Mark to appear")
    showVideo(PATH_TO_YOUR_VIDEO)
else:
    if os.path.isfile(PATH_TO_YOUR_VIDEO):
        shutil.copyfile(PATH_TO_YOUR_VIDEO, "/content/roop/video.mp4")
        print("Video")
        showVideo(PATH_TO_YOUR_VIDEO)
```

Wait For the step to Finish and the Green Check Mark to appear



Langkah kedua dalam kode ini adalah mengunggah video. Kode ini dirancang untuk mempermudah pengguna dalam mengelola dan menganalisis video di lingkungan Google Colab. Pertama, kode mengimpor beberapa modul dan fungsi penting seperti `os`, `shutil`, `files` dari `google.colab`, serta fungsi `showVideo` untuk menampilkan video dan `get_video_resolution` untuk mendapatkan informasi resolusi video. Fungsi `showVideo` menggunakan HTML untuk menampilkan video dengan kontrol pemutaran, sementara `get_video_resolution` menggunakan OpenCV untuk mengambil informasi resolusi video.

Pada langkah pertama, pengguna diminta untuk memilih metode upload video melalui dropdown `upload_method`. Metode ini memungkinkan pengguna untuk

memilih antara mengunggah video dari komputer lokal atau menggunakan video yang sudah ada dalam lingkungan Colab. Jika pengguna memilih "Upload", mereka akan diminta untuk mengunggah file video dari komputer mereka. File video yang diunggah akan diberi nama video.mp4 dan disimpan di direktori /content/roop.

Setelah proses upload selesai, kode akan memeriksa resolusi video menggunakan fungsi `get_video_resolution` dan mencetak informasi tersebut ke layar. Jika resolusi video melebihi 1920x1080 piksel, langkah selanjutnya adalah meresize video menjadi 1280x720 piksel menggunakan perintah `ffmpeg`. Ini dilakukan untuk memastikan bahwa video memiliki ukuran yang lebih sesuai untuk kebutuhan analisis atau manipulasi berikutnya.

Kode kemudian membersihkan output yang ditampilkan sebelumnya dengan menggunakan `clear_output()`, sehingga memastikan tampilan yang lebih bersih di notebook Colab. Setelah itu, video yang telah diunggah atau diambil dari lingkungan Colab ditampilkan kepada pengguna menggunakan fungsi `showVideo`. Preview visual ini memberikan pengguna kesempatan untuk melihat konten video sebelum melanjutkan ke tahapan analisis atau manipulasi selanjutnya sesuai kebutuhan mereka.

Dengan demikian, kode ini tidak hanya memfasilitasi proses awal pengunggahan dan penampilan video di Google Colab, tetapi juga memberikan pengguna kontrol yang baik terhadap langkah-langkah awal dalam analisis atau manipulasi video di lingkungan tersebut.

Step 3 : Upload Face Image

```
#@markdown ## Step 3: Upload Face Image

import os
import shutil
from google.colab import drive
from google.colab import files
from IPython.display import HTML, clear_output
from base64 import b64encode
import moviepy.editor as mp

upload_method = "Upload" #@param ["Upload"]

# remove previous input video
if os.path.isfile('/content/roop/face.jpg'):
    os.remove('/content/roop/face.jpg')


if upload_method == "Upload":
    uploaded = files.upload()
    for filename in uploaded.keys():
        os.rename(filename, '/content/roop/face.jpg')
    PATH_TO_YOUR_IMAGE = '/content/roop/face.jpg'

from IPython.display import clear_output
clear_output()

print("Face")

from IPython.display import Image
Image("/content/roop/face.jpg", width=320)
```

Face



Dalam langkah ketiga kode yang diberikan, tujuan utamanya adalah mengizinkan pengguna untuk mengunggah gambar wajah yang akan digunakan dalam proses pembuatan deep fake. Berikut penjelasan lebih detail mengenai bagaimana kode ini berfungsi:

- Variabel `upload_method`: Variabel ini digunakan untuk memilih metode pengunggahan gambar. Pengguna dapat memilih antara "Upload" untuk

mengunggah gambar dari komputer lokal atau metode lain yang mungkin ada tergantung pada pengaturan atau lingkungan pengguna.

- Pengecekan dan penghapusan gambar yang sudah ada: Kode memulai dengan memeriksa apakah sudah ada file `face.jpg` di direktori `/content/roop/`. Jika sudah ada, file tersebut akan dihapus menggunakan `os.remove()` untuk memastikan bahwa hanya satu gambar wajah yang aktif dan terbaru yang akan digunakan.
- Upload gambar: Jika `upload_method` dipilih sebagai "Upload", pengguna akan diminta untuk mengunggah gambar dari komputer lokal mereka. Penggunaan `files.upload()` memungkinkan pengunggahan file melalui antarmuka web Colab.
- Penyimpanan gambar: Setelah gambar diunggah, gambar tersebut akan diberi nama `face.jpg` dan disimpan di direktori `/content/roop/`. Hal ini memastikan bahwa gambar yang diunggah selalu memiliki nama yang sama dan disimpan di lokasi yang telah ditentukan.
- Membersihkan output: Untuk memberikan tampilan yang bersih dan rapi di notebook atau Colab, `clear_output()` digunakan untuk menghapus output sebelumnya dari layar.
- Menampilkan gambar: Setelah proses pengunggahan selesai dan layar dibersihkan, gambar wajah yang telah diunggah akan ditampilkan menggunakan fungsi `Image()` dari modul `IPython.display`. Ini memungkinkan pengguna untuk melihat gambar wajah dengan lebar 320 piksel, memudahkan untuk visualisasi dan pengaturan selanjutnya dalam proses pembuatan deep fake.

Dengan demikian, langkah ini memungkinkan pengguna untuk memasukkan gambar wajah yang akan digunakan dalam proses manipulasi video, yang merupakan langkah penting dalam pembuatan aplikasi deep fake menggunakan lingkungan Colab atau Jupyter Notebook.

Step 4 : Creating Final Video

```
[14] #@markdown ## Step 4: Generating Final Video

# Memastikan file video tersedia sebelum menampilkan
import os

# Mengubah direktori ke /content/roop
%cd "/content/roop"

# Menjalankan skrip untuk swap wajah
!python run.py -s "face.jpg" -t "video.mp4" -o "face_swapped_video.mp4" --keep-frames --k

from IPython.display import clear_output
clear_output()

# Show Generated Video
from IPython.display import HTML
from base64 import b64encode

# Path video input
video_path = "/content/roop/face_swapped_video.mp4"

# Menampilkan video jika ada
if os.path.isfile(video_path):
    # Menampilkan video
    def show_video(video_path):
        mp4 = open(video_path, 'rb').read()
        data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
        return HTML("""
            <video width=320 controls>
                <source src="%s" type="video/mp4">
            </video>
            """ % data_url)

    # Menampilkan video hasil
    display(show_video(video_path))
else:
    print(f"File {video_path} tidak ditemukan.")
```

Langkah keempat dalam kode ini adalah membuat video akhir. Kode ini menggunakan perintah magic %cd untuk berpindah ke direktori /content/roop, di mana file-file yang dihasilkan akan disimpan. Kemudian, menggunakan perintah !python run.py, program run.py dijalankan dengan argumen yang diberikan. Argumen tersebut antara lain:

- -s "face.jpg": Nama file gambar wajah yang akan digunakan untuk melakukan face swapping.
- -t "video.mp4": Nama file video asli yang akan dimanipulasi.
- -o "face_swapped_video.mp4": Nama file untuk video hasil face swapping yang akan dihasilkan.

- `--keep-frames`: Opsional, menandakan bahwa frame-frame video yang dihasilkan akan disimpan dalam folder.
- `--keep-fps`: Opsional, menandakan bahwa frame rate dari video asli akan dipertahankan pada video hasil.
- `--temp-frame-quality 20`: Opsional, menandakan kualitas frame-frame sementara yang digunakan selama proses.
- `--output-video-quality 20`: Opsional, menandakan kualitas video hasil yang dihasilkan.
- `--execution-provider cuda`: Opsional, menandakan bahwa proses akan menggunakan CUDA untuk percepatan GPU jika tersedia.
- `--frame-processor face_swapper face_enhancer`: Opsional, menandakan bahwa proses akan melalui dua tahap: pertama, face swapping, dan kedua, perbaikan dan penguatan hasil.

Setelah proses selesai, langkah keempat dalam kode ini adalah menampilkan video hasil face swapping dalam bentuk preview di Jupyter Notebook atau Google Colab. Berikut adalah penjelasan rinci tentang bagaimana langkah ini dilakukan:

- **Membersihkan output sebelumnya:** Menggunakan `clear_output()` untuk menghapus output sebelumnya dari layar. Hal ini bertujuan untuk memberikan tampilan yang bersih sebelum menampilkan video hasil.
- **Menampilkan video hasil:** Video hasil face swapping ditampilkan menggunakan HTML dan tag `<video>`. Fungsi `show_video()` mengambil path dari video hasil, membaca file tersebut, dan mengkonversinya ke dalam format data URL (`data:video/mp4;base64`) yang dapat dimasukkan langsung ke dalam tag `<video>` di HTML.
- **Preview video:** Video ditampilkan dengan lebar 320 piksel untuk memastikan tampilan yang nyaman dan sesuai dengan ukuran tampilan di notebook.

Step 5 : Generating Audio

```
## Step 4: Generating Audio
import os
import moviepy.editor as mp
import ipywidgets as widgets
from IPython.display import display, Video

# Fungsi untuk mengunggah file audio
def upload_file(destination_dir, filename):
    upload_widget = widgets.FileUpload(accept='.mp3', multiple=False)
    display(upload_widget)

    def handle_upload(change):
        for file in upload_widget.value:
            file_path = os.path.join(destination_dir, filename)
            with open(file_path, 'wb') as f:
                f.write(upload_widget.value[file]['content'])
            print(f'Successfully uploaded {filename} to {file_path}')
            # Setelah file diunggah, lakukan penggabungan audio dan video
            merge_audio_to_video(output_video_path, file_path, final_output_path)
            print("Final video with new audio created successfully!")
            show_video(final_output_path)

    upload_widget.observe(handle_upload, names='value')

# Fungsi untuk menggabungkan audio baru ke dalam video
def merge_audio_to_video(video_path, audio_path, output_path):
    video = mp.VideoFileClip(video_path)
    audio = mp.AudioFileClip(audio_path)
    final_video = video.set_audio(audio)
    final_video.write_videofile(output_path, codec="libx264", audio_codec="aac")

# Untuk menampilkan video yang sudah jadi
def show_video(file_path):
    display(Video(file_path))

print("Please upload the new audio file.")
if os.path.isfile('/content/roop/audio.mp3'):
    os.remove('/content/roop/audio.mp3')

# Path dari video yang telah di swap face
output_video_path = '/content/roop/face_swapped_video.mp4'
final_output_path = "/content/roop/final_face_swapped_video_with_audio.mp4"

upload_file('/content/roop', 'audio.mp3')
```

Please upload the new audio file.

Upload (1)

Kode yang dibuat memiliki tujuan untuk menggabungkan file audio baru ke dalam video yang telah dimodifikasi sebelumnya. Pertama, kode meminta pengguna untuk mengunggah file audio baru. Jika ada file audio.mp3 di direktori /content/roop, kode akan menghapusnya untuk menghindari konflik. Setelah itu, menggunakan fungsi upload_file, file audio yang baru diunggah akan dipindahkan ke direktori yang sama (/content/roop).

Selanjutnya, fungsi merge_audio_to_video digunakan untuk menggabungkan audio yang baru diunggah dengan video yang telah dimodifikasi sebelumnya. Proses ini dilakukan dengan membuka video sumber dan audio yang baru menggunakan library moviepy, kemudian mengatur audio ke dalam video tersebut. Hasilnya, video akhir yang sudah terdapat audio baru disimpan dalam

format MP4 dengan codec video libx264 dan codec audio aac di lokasi yang ditentukan (/content/roop/final_face_swapped_video_with_audio.mp4).

Setelah video akhir berhasil dibuat, pesan "Final video with new audio created successfully!" ditampilkan dan menggunakan fungsi showVideo, video akhir tersebut ditampilkan di dalam notebook Colab. Proses ini memungkinkan pengguna untuk dengan mudah menggabungkan audio baru ke dalam video hasil modifikasi sebelumnya secara interaktif di lingkungan Colab.

Step 7 : Download Final Video

```
[5] # @markdown ## Step 7 : Download Final Video

%cd "/content/roop"
from google.colab import files
files.download('/content/roop/face_swapped_video.mp4')

/content/roop
```

Langkah kelima dalam kode ini adalah untuk mengunduh video hasil akhir yang telah dibuat. Pertama, dengan menggunakan perintah magic %cd, kita memastikan bahwa kita berada di direktori yang benar di mana video hasil disimpan.

Selanjutnya, menggunakan modul google.colab dan fungsi files.download(), video face_swapped_video.mp4 akan diunduh ke komputer pengguna.

Dengan demikian, langkah ini memberikan pengguna kemampuan untuk mengunduh video hasil akhir dari proses face swapping yang telah dilakukan sebelumnya.

Step 8 : Deleting Old Generated Frames

```
[ ] # @markdown ## Step 8 : Deleting Old Generated Frames
# @markdown Jalankan langkah ini jika Anda akan membuat video lain dan mulai lagi dari Langkah 2. Anda tidak perlu menjalankan Langkah
#Deleting Frames
import os
from os import listdir
my_path = '/content/roop/temp/video/'
for file_name in listdir(my_path):
    if file_name.endswith('.png'):
        os.remove(my_path + file_name)
    if file_name.endswith('.mp4'):
        os.remove(my_path + file_name)
```

Langkah keenam dalam kode ini adalah untuk menghapus frame-frame yang dihasilkan dari proses sebelumnya. Ini berguna jika Anda ingin membuat video lain dan ingin memulai dari awal (Langkah 2). Anda tidak perlu menjalankan Langkah 1 lagi, karena frame-frame sudah ada.

Kode ini menggunakan modul `os` untuk mengakses sistem operasi dan mengelola file-file di direktori. Pertama, kita mendefinisikan path ke direktori tempat frame-frame disimpan dengan variabel `my_path`. Kemudian, kita menggunakan loop `for` untuk mengiterasi semua file dalam direktori tersebut menggunakan fungsi `listdir()`. Untuk setiap file, kita memeriksa apakah nama file tersebut berakhir dengan `.png` atau `.mp4`, yang menandakan bahwa itu adalah frame yang perlu dihapus. Jika demikian, file tersebut dihapus menggunakan `os.remove()`.

Dengan menjalankan langkah ini, frame-frame yang dihasilkan dari proses sebelumnya akan dihapus, sehingga Anda dapat memulai proses pembuatan video lainnya tanpa ada frame-frame yang tersisa dari proses sebelumnya.

Link Google Collab

[Deep Fake-Tiramisu - Colab \(google.com\)](#)

Penutup

Dalam laporan ini, kami telah mendokumentasikan proses pembuatan kode program deep fake. Langkah-langkah yang kami ambil meliputi instalasi perangkat lunak, pengaturan awal, dan eksekusi algoritma untuk face swapping. Meskipun teknologi ini menjanjikan potensi yang besar, penting untuk menggunakannya dengan bijak dan bertanggung jawab. Kami berharap laporan ini dapat memberikan pemahaman yang lebih baik tentang penggunaan teknologi deep fake dan mendorong diskusi yang konstruktif tentang dampaknya dalam masyarakat.