Practical Experiment: Extraction and Processing of Text, Image, Video, and Audio Data

Objective:

To implement Python scripts to extract and process text, image, video, and audio data from multiple platforms.

Part 1: Text Data Extraction and Processing

Steps:

1. **Extract Text Data**: Use web scraping to extract text data from a website.
2. **Process Text Data**: Perform text preprocessing steps such as tokenization, stopword removal, and stemming.

Python Script for Text Data:

python

```
import requests
from bs4 import BeautifulSoup
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer

# Download required NLTK data
nltk.download('punkt')
nltk.download('stopwords')

# Step 1: Extract Text Data
url = 'https://www.example.com'
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
text = soup.get_text()

# Step 2: Process Text Data
# Tokenization
tokens = word_tokenize(text)

# Stopword Removal
stop_words = set(stopwords.words('english'))
filtered_tokens = [word for word in tokens if word.lower() not in
stop_words]

# Stemming
ps = PorterStemmer()
stemmed_tokens = [ps.stem(word) for word in filtered_tokens]

print("Original Text:\n", text[:500])
print("\nProcessed Text Tokens:\n", stemmed_tokens[:50])
```

Part 2: Image Data Extraction and Processing

Steps:

1. **Extract Image Data**: Use web scraping to download images from a website.
2. **Process Image Data**: Perform image preprocessing steps such as resizing and grayscale conversion.

Python Script for Image Data:

python

```
import requests
from PIL import Image
from io import BytesIO
import matplotlib.pyplot as plt

# Step 1: Extract Image Data
image_url = 'https://www.example.com/image.jpg'
response = requests.get(image_url)
img = Image.open(BytesIO(response.content))

# Step 2: Process Image Data
# Resize the image
img_resized = img.resize((128, 128))

# Convert the image to grayscale
img_gray = img.convert('L')

# Display the processed images
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.title('Resized Image')
plt.imshow(img_resized)
plt.subplot(1, 2, 2)
plt.title('Grayscale Image')
plt.imshow(img_gray, cmap='gray')
plt.show()
```

Part 3: Video Data Extraction and Processing

Steps:

1. **Extract Video Data**: Use a video file for processing.
2. **Process Video Data**: Perform video preprocessing steps such as frame extraction and resizing.

Python Script for Video Data:

python

```
import cv2
```

```
# Step 1: Extract Video Data
video_path = 'path_to_video_file.mp4'
cap = cv2.VideoCapture(video_path)

# Step 2: Process Video Data
while cap.isOpened():
    ret, frame = cap.read()
    if ret:
        # Resize the frame
        frame_resized = cv2.resize(frame, (640, 360))

        # Display the frame
        cv2.imshow('Resized Frame', frame_resized)

        # Press 'q' to exit the loop
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break
    else:
        break

cap.release()
cv2.destroyAllWindows()
```

Part 4: Audio Data Extraction and Processing

Steps:

1. **Extract Audio Data**: Use an audio file for processing.
2. **Process Audio Data**: Perform audio preprocessing steps such as conversion to mono and trimming.

Python Script for Audio Data:

python

```
from pydub import AudioSegment

# Step 1: Extract Audio Data
audio_path = 'path_to_audio_file.mp3'
audio = AudioSegment.from_file(audio_path)

# Step 2: Process Audio Data
# Convert audio to mono
audio_mono = audio.set_channels(1)

# Trim the first 30 seconds of the audio
audio_trimmed = audio_mono[:30000]

# Export the processed audio
audio_trimmed.export('processed_audio.mp3', format='mp3')

print("Audio processing completed and saved as
'processed_audio.mp3'")
```

Explanation:

1. **Text Data**:
    - **Extraction**: Web scraping to extract text data from a website using BeautifulSoup.
    - **Processing**: Tokenization, stopword removal, and stemming using NLTK.
2. **Image Data**:
    - **Extraction**: Download images from a website using requests.
    - **Processing**: Resize and convert images to grayscale using PIL.
3. **Video Data**:
    - **Extraction**: Read video file using OpenCV.
    - **Processing**: Resize video frames using OpenCV.
4. **Audio Data**:
    - **Extraction**: Read audio file using PyDub.
    - **Processing**: Convert audio to mono and trim using PyDub.