Human Object Interaction Using Deep Learning

Group no.: 11
Aryan Menon (CEC21CS079)
Ken Jojo (CEC21CS057)
Sinan Noushad (CEC21CS088)
Midhun S Kumar (CEC21CS064)

Guided By : Judy Ann Joy
DEPARTMENT OF COMPUTER ENGINEERING
COLLEGE OF ENGINEERING CHERTHALA

January 8, 2025

INDEX

INTRODUCTION

- Human-Object Interaction (HOI) using Deep Learning uses Convolutional Neural Networks (CNNs) to detect and understand interactions between humans and objects in images or videos.
- CNNs are particularly well-suited for this task due to their ability to effectively extract features from visual inputs.
- This project represents a fusion of cutting-edge deep learning techniques with the fundamental challenge of understanding how humans interact with objects in visual scenes.

RELEVANCE

- Human-Object Interaction (HOI) using Convolutional Neural Networks (CNNs) is crucial in computer vision for enhancing scene understanding and improving human-centric applications.
- By recognizing human-object interactions, CNNs enhance object recognition, prediction, and smart responses in surveillance, human-robot interaction, and assistive technologies.
- In autonomous systems like self-driving cars and smart homes, HOI improves decision-making and user experience. Despite challenges, HOI with CNNs is vital for advancing AI technologies.

PROBLEM STATEMENT

- To build an application that processes image inputs and predicts the type of Human-Object Interaction (HOI) occurring in the image using deep learning models. The system will analyze visual cues to identify and label specific interactions between people and objects.

OBJECTIVES

- Develop human-object interaction detection model
- Ensure robustness across lighting, poses, objects
- Enhance interpretive accuracy
- Scalable for diverse real-world applications

LITERATURE REVIEW

Recognizing Human-Object Interactions in Still Images by Modeling the Mutual Context of Objects and Human Poses, IEEE, 2012 [1]
Authors : Bangpeng Yao, Li Fei-Fei

- The paper employs a conditional random field (CRF) model to jointly capture the mutual context between human poses and objects in human-object interactions.
- The CRF model in the paper utilizes co-occurrence context via a co-occurrence frequency algorithm.
- Using the Gray-Level Co-Occurrence Matrix (GLCM), the algorithm analyzes the spatial relationships between pixel intensities in an image to quantify texture by calculating how frequently pairs of pixel values occur in a specified direction and distance.

LITERATURE REVIEW

Advantages

- The mutual context model enhances the accuracy of both object detection and human pose estimation.
- The model excels in detecting small or partially visible objects and estimating poses.

Disadvantages

- Jointly modeling objects and human poses adds complexity.
- The effectiveness of the model may depend on the availability of high-quality, labeled datasets.

Future Scope

- Extend the method to incorporate additional contextual information and various environmental conditions for more robust and generalizable HOI detection.

LITERATURE REVIEW

Detecting and Recognizing Human-Object Interactions, IEEE, 2018 [2]

Authors: Georgia Gkioxari, Ross Girshick, Piotr Dollár, Kaiming He

- The paper utilizes InteractNet, a custom architecture that integrates multiple branches to simultaneously detect objects and recognize human actions.
- The interaction branch of InteractNet specifically targets the detection of interactions by evaluating the spatial relationships between detected humans and objects, utilizing an action-specific density estimator.
- The estimator generates a probabilistic density function (PDF) that typically predicts a single probable location for simpler actions. For more complex interactions, it can extend to multiple probable locations.

LITERATURE REVIEW

Advantages

- Improved accuracy via human-centric cues.
- Unified detection of people, objects, and interactions.
- Proven effectiveness on benchmark datasets.

Disadvantages

- High computational cost.
- Limited in rare interactions.

Future Scope

- Future work could focus on training and evaluating the model across a wider range of environments and conditions.

LITERATURE REVIEW

Human-object Interaction Recognition Using Multitask Neural Network, IEEE, 2019 [3]
Authors: Weihao Yan, Yue Gao, Qiming Liu

- The paper employs the YOLO (You Only Look Once) algorithm for real-time object detection, utilizing a single convolutional network to predict multiple bounding boxes and class probabilities simultaneously, thereby streamlining the process of identifying objects within an image.

- An attention mechanism is integrated to on relevant regions in the input image, dynamically weighing the importance of features based on the context of detected human actions.

- The attention mechanism in images computes importance scores for different regions of the feature map, enabling the model to focus on relevant areas while processing the image.

LITERATURE REVIEW

Advantages

- Improved accuracy of HOI detection by effectively utilizing the relationships between body parts and objects.
- Integrated motion and object recognition.

Disadvantages

- Requires specialized equipment (RGBD camera and digital gloves).
- Complexity in setup and training.

Future Scope

- Extend the dataset and model to recognize a broader range of human-object interactions, enhancing its applicability to more diverse and complex scenarios.

LITERATURE REVIEW

A Human-Object Interaction Detection Method Inspired by Human Body Part Information, IEEE, 2020 [4]
Authors : Hailan Kuang, Zhikun Zheng, Xinhua Liu, Xiaolin Ma

- The paper employs a hierarchical model that integrates human body part information and object features to detect human-object interactions.
- The detection algorithm leverages a human body part-based feature extraction approach, where both local and global features are analyzed.
- The algorithm functions by first segmenting the human body into various parts, then extracting key features from these parts. It then analyzes the spatial configuration of the body parts in relation to objects to infer interactions, effectively mimicking human cognitive processes in recognizing actions.

LITERATURE REVIEW

Advantages

- Improved the accuracy of HOI detection by effectively utilizing the relationships between body parts and objects.
- The method makes better use of human pose information.

Disadvantages

- Incorporating detailed human body part information may increase the computational complexity.
- The effectiveness of the method relies heavily on accurate human pose estimation.

Future Scope

- Future work could include integrating advanced techniques to improve generalization across diverse scenarios.

LITERATURE REVIEW

HOTR: End-to-End Human-Object Interaction Detection with Transformers, IEEE, 2021 [5]
Authors: Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, Hyunwoo J. Kim

- The paper presents HOTR, a transformer-based model for end-to-end human-object interaction (HOI) detection that predicts HOI triplets directly, bypassing post-processing.

- The HOTR algorithm uses a transformer encoder-decoder architecture with parallel decoders to simultaneously predict human, object, and interaction, utilizing self-attention to capture relationships between them.

- The model works by first extracting global context via a shared encoder, then using HO Pointers to associate predicted interactions with object instances, enabling efficient localization and classification of interactions.

LITERATURE REVIEW

Avantages

- Achieves high performance with minimal processing time.
- Directly predicts ⟨human, object, interaction⟩triplets, enhancing detection precision.

Disadvantages

- Relies on Object Detection Quality, accuracy depends on the effectiveness of object detection.
- Transformer models can be resource-intensive.

Future Scope

- Adapt the model for real-time applications and dynamic environments to enhance practical usability.

LITERATURE REVIEW

Spatial-Temporal Human-Object Interaction Detection, IEEE, 2021 [6]

Authors : Xu Sun, Yunqing He, Tongwei Ren, Gangshan Wu

- This paper uses the ST-HOID model to detect fine-grained human-object interactions and track trajectories of both humans and objects in videos.
- The algorithm employs visual tracking and multi-modal feature fusion, focusing on object trajectory detection and interaction reasoning.
- The reasoning module pairs human-object trajectories and classifies interactions by analyzing body motion, relative motion, and semantic context.

LITERATURE SURVEY

Advantages

- The ST-HOID task allows for detailed detection of human-object interactions and object trajectories.
- The proposed method, combined with the VidOR-HOID dataset, advances the field by providing new benchmarks.

Disadvantages

- The effectiveness of the method may be limited to the VidOR-HOID dataset.
- The method's reliance on both object trajectory detection and interaction reasoning can increase computational complexity.

Future Scope

- Future work could focus on expanding the method to diverse datasets and integrating multi-model data to enhance overall interaction detection and understanding.

LITERATURE REVIEW

Hierarchical Reasoning Network for Human-Object Interaction Detection, IEEE, 2021 [7]
Authors: Yiming Gao, Zhanghui Kuang, Guanbin Li, Wayne Zhang, Liang Lin

- This paper uses the Hierarchical Reasoning Network (HRNet) to detect human-object interactions by modeling relations between human parts and objects across multiple scales.

- The HRNet algorithm employs a graph neural network that builds a multi-level graph, with nodes representing human parts, objects, and human-object pairs, and edges capturing visual and spatial relationships.

- The algorithm refines human-object interaction representations by propagating information through intra-level (within-scale) and inter-level (cross-scale) reasoning to capture both coarse and fine details of human-object interactions.

LITERATURE REVIEW

Advantages

- The model improves accuracy by capturing detailed relationships among human parts and objects at multiple scales.
- It sets new records on major datasets, proving its effectiveness.

Disadvantages

- The hierarchical reasoning and multi-level graph structure add complexity to the model, which may affect training and deployment.
- The extensive graph-based reasoning process can be computationally intensive, requiring significant resources.

Future Scope

- Focus on optimizing the model for faster processing and reduced computational resource usage, making it more scalable for real-time applications.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
Recognizing Human-Object Interactions in Still Images by Modeling the Mutual Context of Objects and Human Poses	Bangpeng Yao, Li Fei-Fei, IEEE, 2012.	• Integrates human body part information to improve Human-Object Interaction detection. • Shows significant performance improvements on the V-COCO dataset.	• Enhances the accuracy of both object detection and human pose estimation. • Excels in detecting small or partially visible objects and estimating poses.	• Jointly modeling objects and human poses adds complexity. • Depend on the availability of high-quality, labeled datasets.	• Extend the method to include diverse contextual information for improved robustness.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
Detecting and Recognizing Human-Object Interactions	Georgia Gkioxari, Ross Girshick, Piotr Dollár, Kaiming He, IEEE, 2018	• The proposed InteractNet model leverages the appearance of detected people—such as their pose, clothing, and actions. • InteractNet jointly learns to detect people and objects, fusing these predictions to efficiently infer interaction triplets in an end-to-end system.	• Improved accuracy via human-centric cues. • Unified detection of people, objects, and interactions.	• High computational cost. • Limited in rare interactions.	• Future work could focus on training and evaluating the model across a wider range of environments and conditions.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
Human-object Interaction Recognition Using Multitask Neural Network	Weihao Yan, Yue Gao, Qiming Liu, IEEE, 2019.	• The proposed method uses a multitask 2D convolutional neural network that integrates human body motion, hand motion, and object recognition. • A YOLOv3-based object recognition network is introduced to boost the accuracy of interaction.	• Improved accuracy of HOI detection by effectively utilizing the relationships between body parts and objects. • Integrated motion and object recognition.	• Requires specialized equipment. • Complexity in setup and training.	• Extend the dataset and model to recognize a broader range of human-object interactions, enhancing its applicability to more diverse and complex scenarios.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
A Human-Object Interaction Detection Method Inspired by Human Body Part Information	Hailan Kuang, Zhikun Zheng, Xinhua Liu, Xiaolin Ma, IEEE, 2020.	• The proposed HOI detection method incorporates human body part (HBP) information to leverage the relationship between body parts and objects.	• Improved accuracy via human-centric cues. • Unified detection of people, objects, and interactions.	• High computational cost. • Effectiveness of the method relies heavily on accurate human pose estimation.	• Evaluate the model in diverse environments and conditions.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
HOTR: End-to-End Human-Object Interaction Detection with Transformers	Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, Hyunwoo J. Kim, IEEE, 2021.	• Directly predicts ⟨human, object, interaction⟩ triplets with a transformer model. • Sets new benchmarks with ¡1 ms inference time post-detection.	• Delivers top results with minimal processing time. • Directly predicts ⟨human, object, interaction⟩ triplets for better accuracy.	• Accuracy relies on the quality of object detection. • Transformers can demand significant resources.	• Optimize for real-time and dynamic environments.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
Spatial-Temporal Human-Object Interaction Detection	Xu Sun, Yunqing He, Tongwei Ren, Gangshan WuXu Sun, Yunqing He, Tongwei Ren, Gangshan Wu, IEEE, 2021.	• Detects fine-grained human-object interactions and trajectories in videos. • Provides 10,831 instances and shows better performance than existing methods.	• Enables detailed detection of interactions and object trajectories. • Advances the field with new benchmarks using VidOR-HOID.	• Effectiveness may be limited to VidOR-HOID. • Relies on object trajectory and interaction reasoning, raising computational demands.	• Expand to diverse datasets and integrate multi-modal data for improved interaction detection.

LITERATURE SURVEY

Name	Author	Methodology	Merits	Limitations	Future Scope
Hierarchical Reasoning Network for Human-Object Interaction Detection	Yiming Gao, Zhanghui Kuang, Guanbin Li, Wayne Zhang, Liang Lin, IEEE, 2021.	• Models human parts and objects with a multi-level graph and hierarchical reasoning. • Delivers leading results on HICO-DET, V-COCO, and HOI-A benchmarks.	• Improves accuracy by capturing detailed relationships among human parts and objects. • It sets new records on major datasets, proving its effectiveness.	• Adds challenges to training and deployment. • Requires significant computational resources.	• Improve processing speed and reduce resource usage for real-time scalability.

CONCLUSION FROM LITERATURE SURVEY

- The field has seen significant advancements through the use of convolutional neural networks (CNNs) and transformer-based models for detecting and understanding HOI in both static images and videos.
- While models like InteractNet and HOTR have improved accuracy by leveraging human-centric cues and predicting HOI triplets, challenges remain in computational complexity, the need for large annotated datasets, and scalability across diverse environments.
- Future work will likely focus on improving generalization, efficiency, and real-time processing capabilities to make HOI detection more practical and robust across various applications.

PRODUCT FUNCTIONS

- **Data Collection and Preprocessing:** Collect images or videos of human-object interactions and preprocess them (resizing, normalization) for consistency.
- **Object Detection:** Use a CNN (e.g., YOLO) to detect and localize objects in the images.
- **Action Recognition:** Apply action recognition models to map key human joints for interaction analysis.
- **Interaction Recognition:** Combine object and keypoint data to recognize actions like picking up or holding objects using temporal models.
- **Interaction Classification:** Classify interactions with a machine learning model (e.g., SVM) to label types of human-object engagements.

REQUIREMENTS

- **Hardware Requirements :**
  - Computer with
    - RAM: 8 GB
    - SSD: 512 GB
- **Software Requirements:**
  - OS: WINDOWS 10
- **Dataset:**
  - Verbs in COCO (V-COCO) is a dataset that builds off COCO for human-object interaction detection. V-COCO provides 10,346 images (2,533 for training, 2,867 for validating and 4,946 for testing) and 16,199 person instances.

TOOLS AND LANGUAGE

- **Language :** Python
- **Various libraries :** NumPy, PIL etc.
- **Machine learning framework :** TensorFlow, PyTorch etc.
- **FrontEnd :** Python Tkinter
- **Developement Environment :** Google Colab, VS Code

SYSTEM ARCHITECTURE

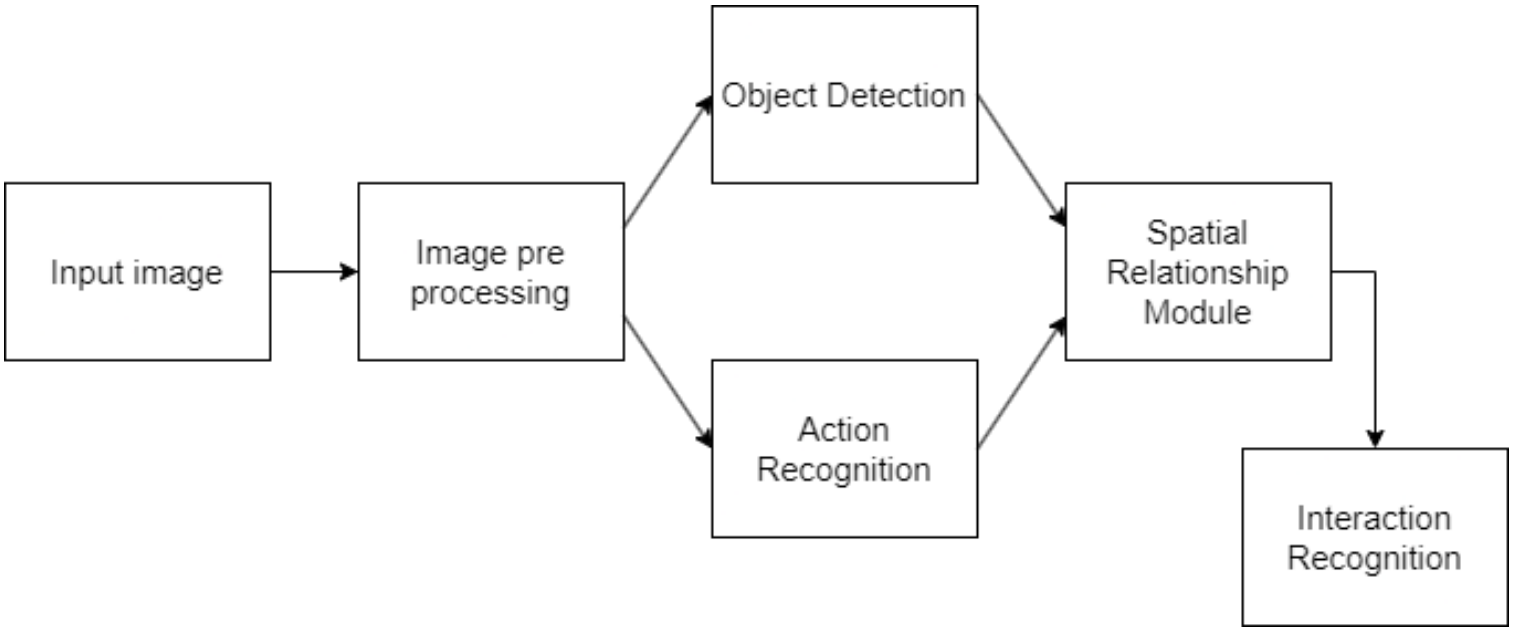

Figure 1: System Architecture

USE CASE DIAGRAM

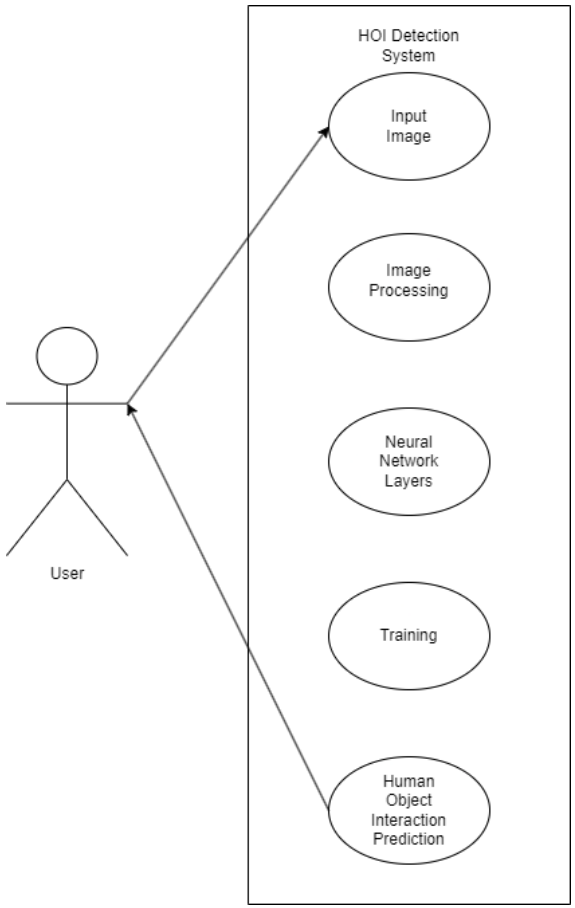

Figure 2: Use Case Diagram

DATA FLOW DIAGRAM LEVEL0

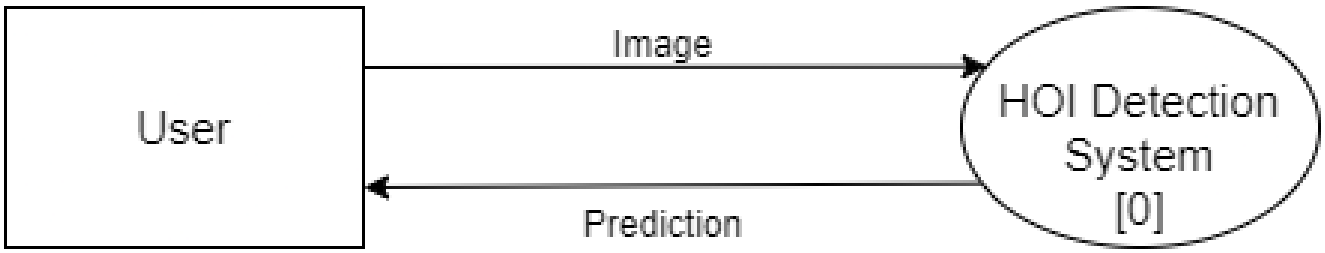

Figure 3: Level 0

DATA FLOW DIAGRAM LEVEL1

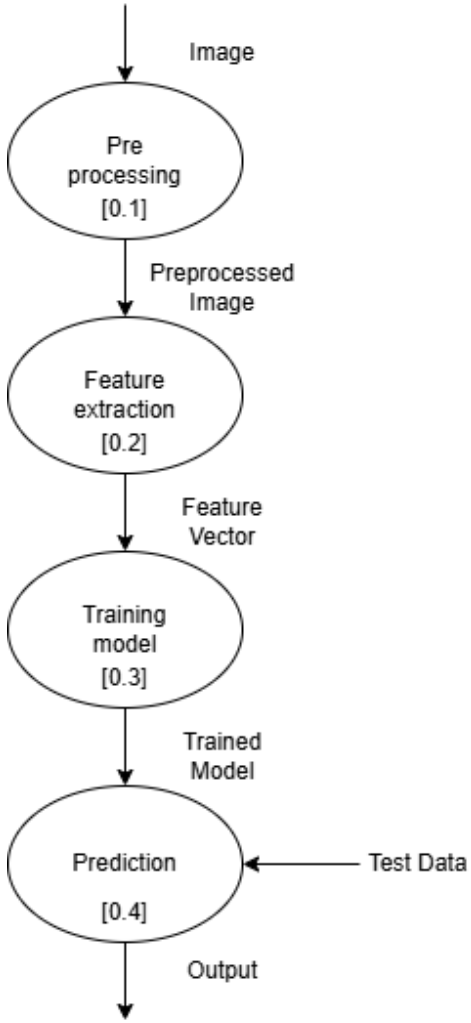

Figure 4: Level 1

GANTT CHART

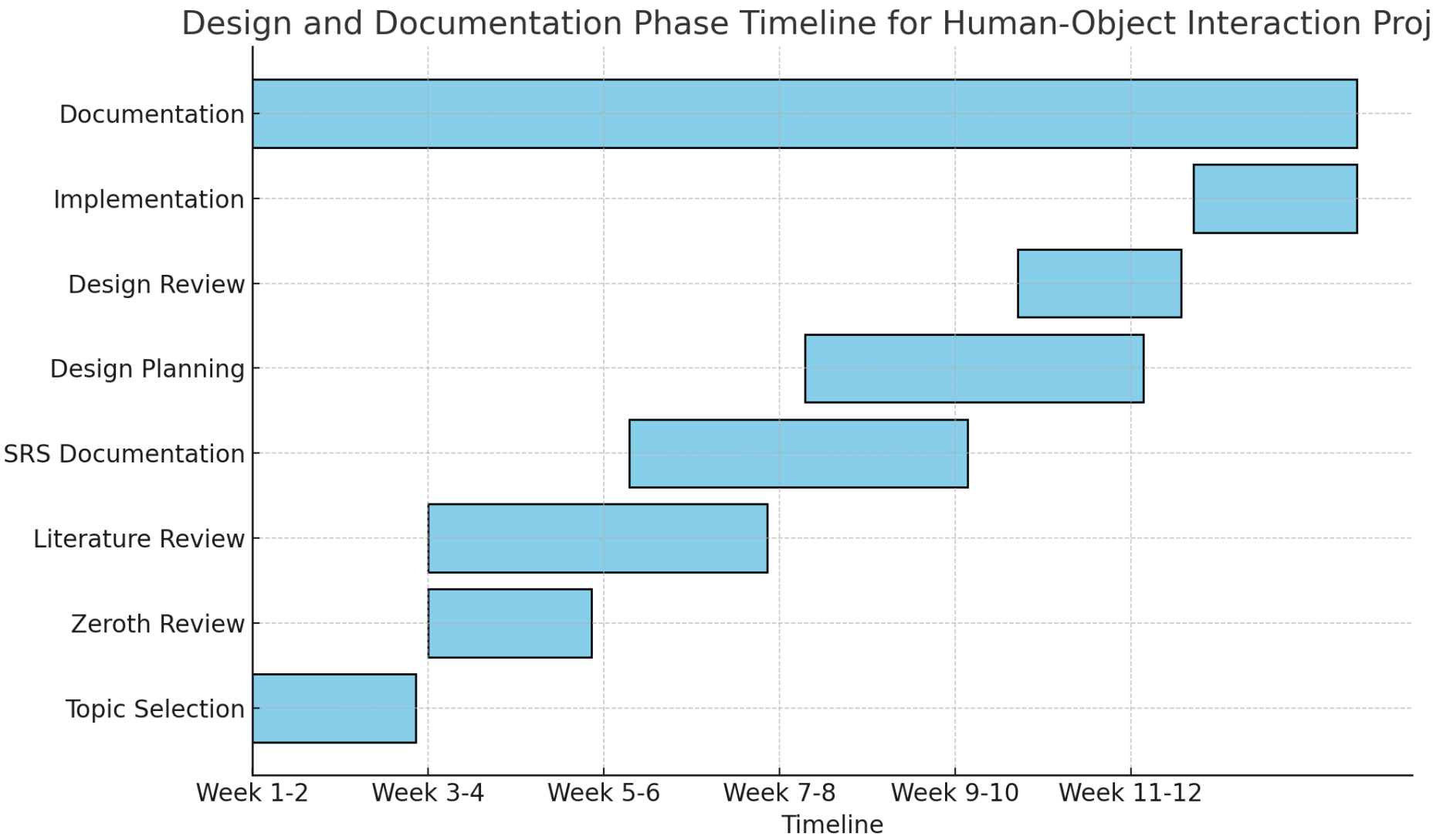

Figure 5: Gantt Chart

COST ESTIMATION

Lines of code = 211
KLOC = 211/1000 = 0.211

Effort =a(KLOC)^b
=2.4(0.211)^1.05
=0.468 PM

Duration =c(Effort)^d
=2.5(0.468)^0.38
=1.873 ≅ 2 months

Staffing =Effort/Duration
= 0.468/1.873
=0.249 ≅ 1 person

Rate per Person=15,000 Rs
Total Rate for 1 staffs for 1 months = 15,000 * 1 * 1 =15,000 Rs

IMPLEMENTATION

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
import tensorflow as tf
from tensorflow.keras.preprocessing import image_dataset_from_directory
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping
```

```
[29] # Data Preprocessing
     train_df['label'] = train_df['label'].astype('category')
     train_df['label'] = train_df['label'].cat.codes
     train_df['filepath'] = train_df['filename'].apply(lambda x: os.path.join(train_path, x))
```

```
[30] # Split training and validation set
     train_set, val_set = train_test_split(train_df, test_size=0.2, stratify=train_df['label'], random_state=42)
```

```
[31] def load_image(filepath, label):
         image = tf.io.read_file(filepath)
         image = tf.image.decode_jpeg(image, channels=3)
         image = tf.image.resize(image, [128, 128])
         image = image / 255.0
         return image, label

     # Create TensorFlow datasets
     train_dataset = tf.data.Dataset.from_tensor_slices((train_set['filepath'].values, train_set['label'].values))
     train_dataset = train_dataset.map(load_image).batch(32).shuffle(buffer_size=len(train_set))

     val_dataset = tf.data.Dataset.from_tensor_slices((val_set['filepath'].values, val_set['label'].values))
     val_dataset = val_dataset.map(load_image).batch(32)
```

```
[32] from tensorflow.keras.models import Sequential
     from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Input

     # Model Building
     model = Sequential([
         Input(shape=(128, 128, 3)),  # Use Input layer to specify input shape
         Conv2D(32, (3, 3), activation='relu'),
         MaxPooling2D((2, 2)),
         Conv2D(64, (3, 3), activation='relu'),
         MaxPooling2D((2, 2)),
         Conv2D(128, (3, 3), activation='relu'),
         MaxPooling2D((2, 2)),
         Flatten(),
         Dense(128, activation='relu'),
         Dropout(0.5),
         Dense(15, activation='softmax')
     ])

     model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

```
# Model Training
history = model.fit(train_dataset, validation_data=val_dataset, epochs=12)
```

```
Epoch 1/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 361s 1s/step - accuracy: 0.0903 - loss: 2.7202 - val_accuracy: 0.1921 - val_loss: 2.5180
Epoch 2/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 371s 1s/step - accuracy: 0.1819 - loss: 2.4830 - val_accuracy: 0.2532 - val_loss: 2.3218
Epoch 3/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 347s 1s/step - accuracy: 0.2531 - loss: 2.2906 - val_accuracy: 0.2742 - val_loss: 2.2421
Epoch 4/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 383s 1s/step - accuracy: 0.2974 - loss: 2.1367 - val_accuracy: 0.2885 - val_loss: 2.2041
Epoch 5/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 345s 1s/step - accuracy: 0.3432 - loss: 1.9993 - val_accuracy: 0.2968 - val_loss: 2.2037
Epoch 6/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 401s 1s/step - accuracy: 0.3949 - loss: 1.8284 - val_accuracy: 0.2948 - val_loss: 2.1925
Epoch 7/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 367s 1s/step - accuracy: 0.4491 - loss: 1.6551 - val_accuracy: 0.3087 - val_loss: 2.2195
Epoch 8/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 347s 1s/step - accuracy: 0.5172 - loss: 1.4565 - val_accuracy: 0.3135 - val_loss: 2.2824
Epoch 9/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 386s 1s/step - accuracy: 0.5720 - loss: 1.2741 - val_accuracy: 0.3135 - val_loss: 2.4861
Epoch 10/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 376s 1s/step - accuracy: 0.6142 - loss: 1.1259 - val_accuracy: 0.3052 - val_loss: 2.5643
Epoch 11/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 346s 1s/step - accuracy: 0.6550 - loss: 1.0144 - val_accuracy: 0.3048 - val_loss: 2.7817
Epoch 12/12
315/315 ━━━━━━━━━━━━━━━━━━━━ 385s 1s/step - accuracy: 0.6827 - loss: 0.9141 - val_accuracy: 0.3111 - val_loss: 2.8968
```

```python
# Show 10 images with actual and predicted labels from validation set
fig, axes = plt.subplots(2, 5, figsize=(20, 10))
axes = axes.flatten()
sampled_val_set = val_set.sample(10, random_state=42)
for i, (index, row) in enumerate(sampled_val_set.iterrows()):
    img = plt.imread(row['filepath'])
    actual_label = categories[row['label']]
    predicted_label = categories[val_predictions[val_set.index.get_loc(index)]]
    axes[i].imshow(img)
    axes[i].set_title(f"Actual: {actual_label}\nPredicted: {predicted_label}")
    axes[i].axis('off')
plt.tight_layout()
plt.show()
```

```python
from google.colab import files
import numpy as np
import time
import cv2
from google.colab.patches import cv2_imshow

# Upload the image file
uploaded = files.upload()

# This will prompt you to upload the image file. Use the filename returned by the upload to read the image.
image_path = None
for file_name in uploaded.keys():
    image_path = file_name
    print(f"Uploaded file: {file_name}")

# Define the paths for the YOLO files
labelsPath = 'coco.names'
weightsPath = 'yolov3.weights'
configPath = 'yolov3.cfg'

# Load the COCO class labels our YOLO model was trained on
LABELS = open(labelsPath).read().strip().split("\n")

# Initialize a list of colors to represent each possible class label
np.random.seed(42)
COLORS = np.random.randint(0, 255, size=(len(LABELS), 3), dtype="uint8")

# Load our YOLO object detector trained on COCO dataset (80 classes)
print("[INFO] loading YOLO from disk...")
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

# Verify the image path and read the image
image = cv2.imread(image_path)
if image is None:
    print(f"Failed to load image at {image_path}")
else:
    print(f"Successfully loaded image at {image_path}")

# Proceed with the rest of the script
(H, W) = image.shape[:2]

# Determine only the *output* layer names that we need from YOLO
ln = net.getLayerNames()
ln = [ln[i - 1] for i in net.getUnconnectedOutLayers()]
```

```python

# Construct a blob from the input image and then perform a forward pass
# of the YOLO object detector, giving us our bounding boxes and
# associated probabilities
blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (416, 416), swapRB=True, crop=False)
net.setInput(blob)
start = time.time()
layerOutputs = net.forward(ln)
end = time.time()

# Show timing information on YOLO
print("[INFO] YOLO took {:.6f} seconds".format(end - start))

# Initialize our lists of detected bounding boxes, confidences, and
# class IDs, respectively
boxes = []
confidences = []
classIDs = []

# Loop over each of the layer outputs
for output in layerOutputs:
    # Loop over each of the detections
    for detection in output:
        # Extract the class ID and confidence (i.e., probability) of the current object detection
        scores = detection[5:]
        classID = np.argmax(scores)
        confidence = scores[classID]

        # Filter out weak predictions by ensuring the detected probability is greater than the minimum probability
        if confidence > 0.5:
            # Scale the bounding box coordinates back relative to the size of the image
            box = detection[0:4] * np.array([W, H, W, H])
            (centerX, centerY, width, height) = box.astype("int")

            # Use the center (x, y)-coordinates to derive the top and
            # and left corner of the bounding box
            x = int(centerX - (width / 2))
            y = int(centerY - (height / 2))

            # Update our list of bounding box coordinates, confidences, and class IDs
            boxes.append([x, y, int(width), int(height)])
            confidences.append(float(confidence))
            classIDs.append(classID)

# Apply non-maxima suppression to suppress weak, overlapping bounding boxes
idxs = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.3)

```

```
# Ensure at least one detection exists
if len(idxs) > 0:
    # Loop over the indexes we are keeping
    for i in idxs.flatten():
        # Extract the bounding box coordinates
        (x, y) = (boxes[i][0], boxes[i][1])
        (w, h) = (boxes[i][2], boxes[i][3])

        # Draw a bounding box rectangle and label on the image
        color = [int(c) for c in COLORS[classIDs[i]]]
        cv2.rectangle(image, (x, y), (x + w, y + h), color, 2)
        text = "{}: {:.4f}".format(LABELS[classIDs[i]], confidences[i])
        cv2.putText(image, text, (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)

# Show the output image using cv2_imshow
cv2_imshow(image)

```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving hyundai-creta.jpeg to hyundai-creta.jpeg
Uploaded file: hyundai-creta.jpeg
[INFO] loading YOLO from disk...
Successfully loaded image at hyundai-creta.jpeg
[INFO] YOLO took 2.714727 seconds
```

CONCLUSION

- The proposed system leverages state-of-the-art technologies and innovative techniques to address the limitations of existing approaches.
- By enhancing accuracy, improving efficiency, providing better contextual understanding, and ensuring scalability and robustness, the proposed system offers a superior solution for detecting and understanding human-object interactions.
- Utilizing models like Faster R-CNN ensures quick and accurate localization of humans and objects.

REFERENCES

1 Bangpeng Yao, Li Fei-Fei "Recognizing Human-Object Interactions in Still Images by Modeling the Mutual Context of Objects and Human Poses", IEEE, 2012

2 Georgia Gkioxari, Ross Girshick, Piotr Dollár, Kaiming He "Detecting and Recognizing Human-Object Interactions", IEEE, 2018

3 Weihao Yan, Yue Gao, Qiming Liu "Human-object Interaction Recognition Using Multitask Neural Network", IEEE, 2019

4 Hailan Kuang, Zhikun Zheng, Xinhua Liu, Xiaolin Ma "A Human-Object Interaction Detection Method Inspired by Human Body Part Information", IEEE, 2020

5 Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, Hyunwoo J. Kim "HOTR: End-to-End Human-Object Interaction Detection with Transformers", IEEE, 2021

6 Xu Sun, Yunqing He, Tongwei Ren, Gangshan Wu "Spatial-Temporal Human-Object Interaction Detection", IEEE, 2021

7 Yiming Gao, Zhanghui Kuang, Guanbin Li, Wayne Zhang, Liang Lin "Hierarchical Reasoning Network for Human-Object Interaction Detection", IEEE, 2021

- Manli Zhu, Edmond S. L. Ho, and Hubert P. H. Shum "A Skeleton-Aware Graph Convolutional Network For Human-Object Interaction Detection", IEEE, 2022

THANK YOU