Task 1: Object Detection

Task 1: Approach 1 - Object Detection Using DETR

Objective:

Perform object detection on a dataset of meme images using the DETR (DEtection TRansformer) model.

Approach:

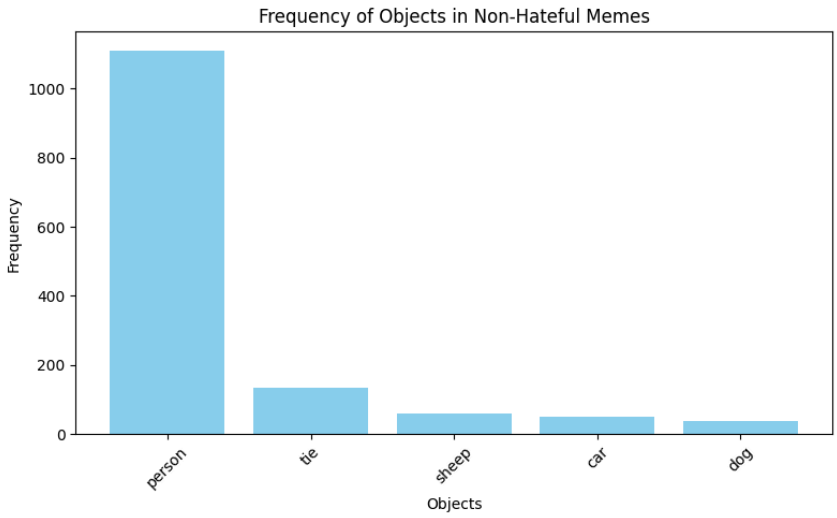
- The task involves performing object detection on a dataset of meme images.
- DETR is a powerful and state-of-the-art object detection model that has shown impressive performance in various computer vision tasks.
- DETR is based on the transformer architecture, which has proven to be effective in various natural language processing tasks and has also been successfully applied to computer vision problems.
- The pre-trained DETR model (facebook/detr-resnet-50) from the Hugging Face Transformers library is utilized, which provides a good starting point and can generalize well to diverse object categories.

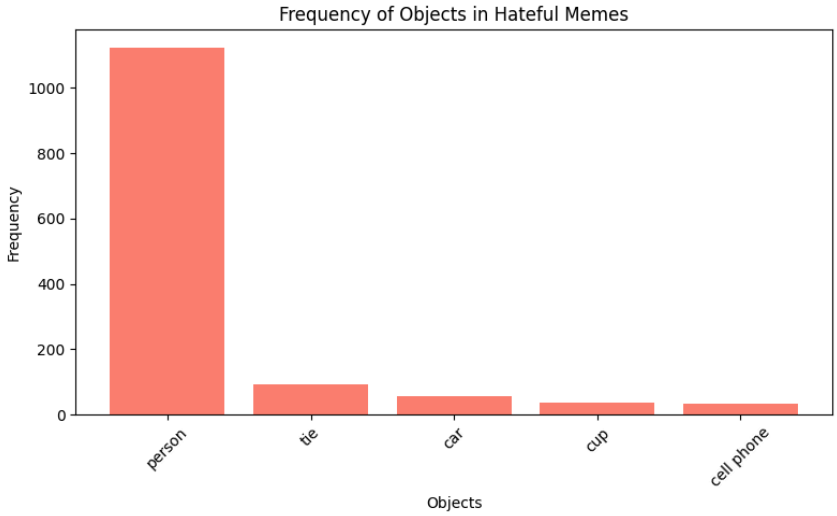
Methodology:

1. **Model Initialization**
   - Initialize the DETR object detection model (`facebook/detr-resnet-50`) from the Hugging Face Transformers library.
   - Initialize the `DetrImageProcessor` for preprocessing input images.
2. **Image Processing**
   - The `process_images` function processes images within specified categories.
   - For each category, iterate through images, open them, convert to RGB format, and preprocess using `DetrImageProcessor`.
3. **Object Detection**
   - Open and preprocess images.
   - Pass images to the DETR model for inference.
   - Post-process outputs to the COCO API format.
   - Apply a confidence threshold of 0.9 to filter low-confidence detections.
4. **Result Logging**
   - Log detection results for each image in text files named `output_results_{category}.txt`, including image path, detected object labels, confidence scores, and bounding box coordinates.
5. **Object Frequency Analysis**
   - Track object frequency across all images in each category using `object_counter`.
   - Write object frequencies to separate text files named `object_frequency_{category}.txt`.
6. **Output Files**
   - Return a list of output file paths containing detection result files and object frequency analysis files for each processed category.

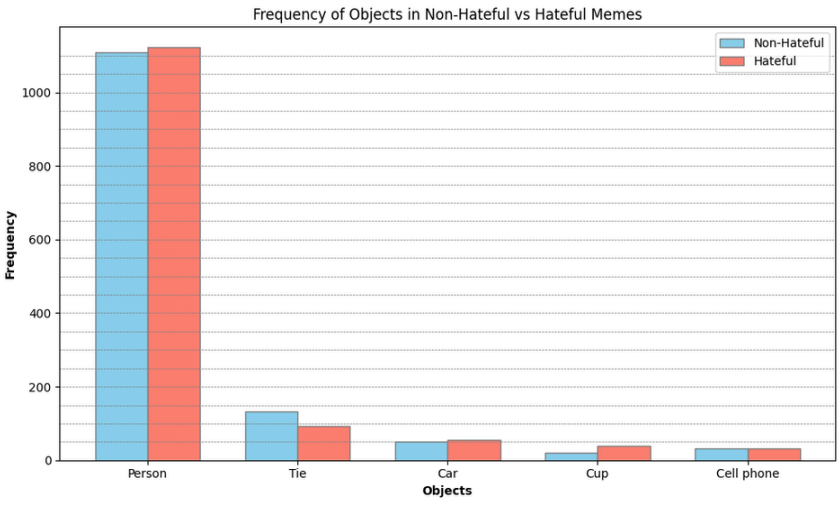
Findings:

The following graphs depict the frequency of the top 5 objects detected within two classes: hateful and non-hateful memes. These counts were derived from a catalog that lists objects along with their respective frequencies.





The following graph depicts a comparison between the frequencies of the top 5 objects.



Positives about this approach:

- DETR is a transformer-based object detection model known for its impressive performance in various computer vision tasks.
- Being pre-trained on a large dataset, DETR can generalize well to diverse object categories.
- It provides bounding box coordinates and confidence scores for detected objects, enabling further analysis and potential applications.

- The approach is well-documented and leverages popular libraries like Transformers and PyTorch, making it easy to understand and implement.

Negatives and potential failure cases:

- Object detection models, including DETR, may struggle with occlusions, unusual object orientations, and unfamiliar objects that are not well-represented in the training data. In this text occlusion is the topic we are going to focus in second task.
- Performance may degrade for low-quality, distorted, or noisy images, which are common in meme datasets.
- Object detection alone may not capture the full context present in meme images, which could be crucial for tasks like toxicity classification or understanding the meme's intended message.
- Transformer-based models like DETR can be computationally expensive, especially for high-resolution images, which may impact performance or require specialized hardware.

Task 1: Approach 2 - Object Detection Using LLaVa

Objective:
Perform object detection on meme images using the LLaVa (Large Language Vision Assistant) model.

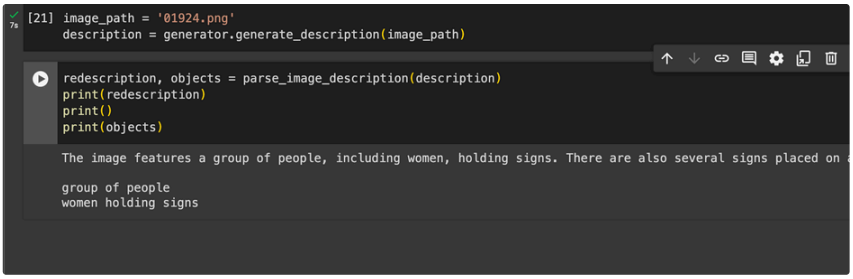
Approach:

- The task involves performing object detection on a dataset of meme images.
- LLaVa is a powerful multimodal model that can process both text and visual information, making it suitable for tasks like object detection, image captioning, and visual question answering.
- By leveraging LLaVa's multimodal capabilities, the approach aims to generate human-readable descriptions of the objects present in the meme images and explicitly list the most highlighted or prominent objects.
- The approach utilizes LLaVa's language understanding capabilities to provide flexible prompting, allowing customization of the output format or obtaining specific insights from the visual information.

Methodology:

1. **Model Initialization**
   - The `ImageDescriptionGenerator` class is responsible for loading the LLaVa model and generating textual descriptions from images.
   - The `model_id` parameter specifies the pre-trained LLaVa model to use.
   - The `BitsAndBytesConfig` is used to enable quantization for efficient model loading and inference.
2. **Model Loading**
   - The `load_model` method initializes the Hugging Face pipeline for the "image-to-text" task, using the specified LLaVa model and quantization configuration.
3. **Image Description Generation**
   - The `generate_description` method takes an image path as input and generates a textual description using the loaded LLaVa model.
   - The image is loaded using the Pillow library, and a prompt is constructed to guide the model's response.
   - The prompt instructs the model to describe objects present while ignoring text elements and provide a list of highlighted objects.
   - The model's output is obtained through the pipeline and returned as the image description.
4. **Description Parsing**
   - The `parse_image_description` function extracts the textual description and list of highlighted objects from the model's output.
   - It searches for the "ASSISTANT:" token, extracts the text after it, splits it into lines, and identifies the line where the object list begins.
   - The function separates the description and object list, removes duplicates from the description, and returns both parts.

Findings:



Positives about this approach:

- LLaVa is a powerful multimodal model that can process both text and visual data, enabling it to understand and describe complex visual scenes.
- The model can generate human-readable descriptions and explicitly list the most highlighted objects, providing interpretable results.
- LLaVa's language understanding capabilities allow for flexible prompting, enabling users to customize the desired output format or obtain various insights from visual information.

Negatives and potential failure cases:

- The LLaVa model is computationally expensive due to its large size and complexity, requiring high computational resources and GPU acceleration for efficient inference.
- The performance and quality of the generated descriptions may vary depending on the complexity of the input images with prompts and the model's understanding of the visual and textual context.

Task 1: Approach 3 - Object Detection Using YOLOv8

Objective:

Perform object detection on meme images using YOLOv8 (You Only Look Once version 8).

Approach:

- The task involves performing object detection on a dataset of meme images.
- YOLOv8 is a highly optimized and efficient object detection model known for its impressive speed and accuracy.
- By leveraging YOLOv8, the approach aims to accurately detect and localize multiple objects in the meme images with real-time performance.
- The approach generates a CSV catalog file containing the detected objects, their confidence scores, and ground truth labels, facilitating further analysis and evaluation.

Methodology:

1. **Model Installation and Import**
   - Install required dependencies, including the `ultralytics` library.
   - Import necessary modules from the `ultralytics` library, including `YOLO`.
2. **Model Loading**
   - Load the pre-trained YOLOv8 model using the `YOLO` class from the `ultralytics` library.
   - Specify the desired model configuration, such as the model version (e.g., 'yolov8n.pt').
3. **Catalog Generation**
   - The `append_to_catalog_from_json` function appends detected objects and their labels to a CSV catalog file.
   - It takes two arguments: `json_file_path` (path to a JSON file containing image paths and labels) and `catalog_path` (path to the CSV catalog file).
   - The function iterates through each entry in the JSON file, retrieving the image path and ground truth label.
   - For each image, it performs object detection using the loaded YOLOv8 model and stores the detected objects, their confidence scores, and ground truth labels in a list.
   - The detected objects, along with their labels and confidence scores, are appended to the specified CSV catalog file.
4. **Result Logging**
   - The function logs a message indicating that the detected objects have been appended to the catalog file.

Findings:

From this process, we detect objects and store their confidence levels, indicating how accurately the object is detected as belonging to a certain class. Additionally, we save the image path, the detected objects, their respective confidence levels, and the ground truth label associated with each detection. This information can be utilized for further processing and analysis.

```
    print(row)

{'Image': '/content/01235.png', 'Object': "['person']", 'Confidence': '[    0.96653]', 'Ground Truth Label': 'hateful'}
{'Image': '/content/01236.png', 'Object': "['person']", 'Confidence': '[    0.75815]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['person']", 'Confidence': '[    0.70568]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['horse']", 'Confidence': '[    0.58712]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['person']", 'Confidence': '[    0.50091]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['sheep']", 'Confidence': '[    0.48118]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['person']", 'Confidence': '[    0.46822]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['sheep']", 'Confidence': '[    0.46252]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['sheep']", 'Confidence': '[    0.36762]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['person']", 'Confidence': '[    0.33579]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['sheep']", 'Confidence': '[    0.32556]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['sheep']", 'Confidence': '[     0.2777]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['person']", 'Confidence': '[    0.26241]', 'Ground Truth Label': 'non-hateful'}
{'Image': '/content/01236.png', 'Object': "['sheep']", 'Confidence': '[     0.2592]', 'Ground Truth Label': 'non-hateful'}

Start coding or generate with AI.
```

Positives:

- YOLOv8 is a highly optimized and efficient object detection model, providing real-time performance.
- It provides bounding box coordinates, confidence scores, and class labels for each detected object, enabling further analysis and potential applications.

Negatives:

- YOLOv8 is capable of recognizing only the 80 base classes from the COCO dataset on which it was trained, limiting its ability to detect uncommon or specialized objects that may be present in meme images.