Alejandro Rebolloso

Professor Mcmanus

Computer Vision

30 October 2024

# L09 Object Detection using Transfer Learning and Pascal VOC 207 Dataset

## Conceptual Understanding

1. Image classification provides a broad categorization, while object detection gives detailed information about the location and type of various objects in the image. The goal in image classification is to categorize an entire image into a single label or class. On the other hand, the Goal in object detection is to identify and locate multiple objects within an image, classifying each one.

2. Choosing the SSD MobileNet V2 model for tasks like object detection often stems from its balance of efficiency and accuracy, especially in scenarios with limited computational resources. Most importantly, it was the most optimal choice as the assignment's focus was to understand the concepts and workflow of object detection.

## Code Interpretation

1. It is important because it is the function that helps us find images that contain the target class we are searching for. When working with large datasets, the workload can be overwhelming and require many resources. By narrowing down the relevant images, it can help save resources, power, and time during training. Also, this function allows researchers to create focused subsets for training models.

2. The threshold value in the plot_detections function determines the minimum confidence score required for a detected object to be displayed. A lower threshold includes more objects, potentially capturing false positives, while a higher threshold focuses on higher confidence detections, missing valid objects. This creates a trade-off between precision (accuracy of displayed detections) and recall (number of detected objects).

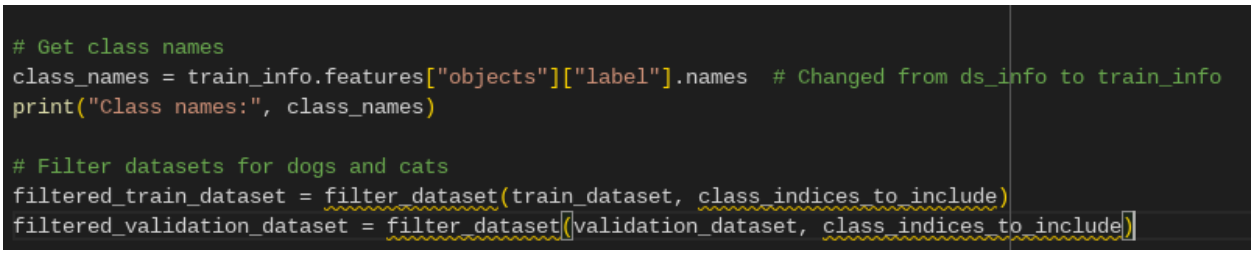
3. Heatmap visualization provides a graphical representation of the model's confidence across different areas of an image. Areas with higher confidence are indicated by warmer colors (like red), showing where the model believes objects are present, while cooler colors (like blue) indicate lower confidence. This helps identify not only where the model is making strong detections but also areas where it might be uncertain, allowing for better insights into its performance and potential weaknesses.

## Observing Results and Limitations

1. When running the exercise, I find the performance to be underachieving and plain wrong. It mistook the car hood for a chair and car door for a boat. The reason the model was performing so poorly was due to the limited computational resources available. Because of this, it resulted in lower accuracy in return for a better understanding of how the model functions.
2. Yes, there are instances where the bounding boxes are inaccurate or miss the object entirely. Factors contributing to these errors include object size and scale, occlusion, background clutter, and lighting and shadow.
3. Using the entire Pascal VOC 2007 dataset instead of a small subset would improve the model's accuracy due to the increased diversity of training examples, which helps the model learn to recognize a broader range of object variations. More examples for each class would allow the model to better capture unique features, reducing the risk of overfitting to the limited data.

Critical Thinking

1. To detect a specific set of objects, like only animals or vehicles, I would modify the code to filter for the desired classes during the detection process. Create a filter for “dog” or “cat” Update the dataset loading to focus on images containing those specific classes and adjust any functions that prepare or visualize the data accordingly. This will ensure that the model only processes and displays detections for the selected categories, enhancing the relevance of the results.



```
# Get class names
class_names = train_info.features["objects"]["label"].names  # Changed from ds_info to train_info
print("Class names:", class_names)

# Filter datasets for dogs and cats
filtered_train_dataset = filter_dataset(train_dataset, class_indices_to_include)
filtered_validation_dataset = filter_dataset(validation_dataset, class_indices_to_include)
```

2. To train my own object detection model, I would need to collect and annotate a suitable dataset, choose an appropriate model architecture, and configure the training process using a framework like TensorFlow or PyTorch. One of the main challenges I might face is dealing with computational limitations, as training these models often requires significant processing power and memory, which could lead me to consider using cloud resources or high-performance GPUs. Additionally, managing large datasets and ensuring proper data augmentation can be demanding in terms of storage and computational resources.

3. Despite its limitations, this model can be useful in real-time applications like surveillance or traffic monitoring, where quick detections are essential and moderate accuracy suffices. Its lightweight architecture makes it suitable for deployment on mobile devices

or in edge computing scenarios, such as augmented reality applications. Additionally, in low-stakes environments like basic inventory management or agricultural monitoring, the model can provide valuable insights without requiring high precision.

References

HCCS. (2024). *ITAU 1378 2024 Mod 09 Advanced Architectures* [PowerPoint slides]. Retrieved from https://eagleonline.hccs.edu/courses/266737/files/64009301?module_item_id=17334329