**📄 Experience Report: Object Detection Internship Assignment**

**Author:** Tejash Pandey  
**Project:** YOLOv3 with ResNet-50 Backbone on Pascal VOC  
**Platform:** Google Colab & GitHub  
**Dataset:** Pascal VOC 2007

**🚀 Summary of What I Did**

In this assignment, I built an object detection pipeline by integrating a YOLO-style detection head with a ResNet-50 backbone. I trained and evaluated the model on the Pascal VOC dataset and produced visual inference results with both predicted (red) and ground truth (green) boxes.

**⚙️ Technical Choices**

* **Backbone:** ResNet-50 (pretrained on ImageNet)
* **Detection Head:** YOLOv3-style multi-scale convolutional heads
* **Dataset:** Pascal VOC 2007 (auto-downloaded using torchvision)
* **Loss:** Simple composite loss (MSE + BCE) — a simplified YOLO-style loss
* **Framework:** PyTorch, with training and testing in Google Colab
* **Evaluation:** COCO API for mAP computation
* **Inference:** PNG visual outputs with bounding box overlays

**💻 AI Tools I Used**

* **ChatGPT:** I used ChatGPT during the project:
  + For code structuring and modularization (e.g., config, model, loss)
  + To debug DataLoader issues, tensor shape mismatches, and evaluation errors
  + For guidance on writing the YOLO head and decoding logic
  + To generate helper scripts for visualization, evaluation, and testing
* **GitHub Copilot:** Occasionally used inside the notebook for line completions

**😅 Challenges I Faced**

* Decoding YOLO outputs correctly was difficult. The shape transformations across feature maps needed careful handling.
* I had trouble getting meaningful evaluation scores at first due to wrong anchor scales and low confidence thresholds.
* Getting inference images saved with both GT and prediction boxes aligned took multiple fixes.
* Understanding COCO-style evaluation with Pascal VOC ground truth required extra work to format annotations.

**📚 What I Learned**

* How object detection heads differ from classification heads in CNNs
* The importance of anchor design and matching
* How mAP works in COCO evaluation and what makes predictions valid
* Real-world object detection pipelines are modular and complex — it's not just training a model
* How to use AI tools as a support system, not a replacement for understanding

**🤯 What Surprised Me**

* Even with pretrained ResNet-50, the performance was poor without correct anchor handling or extended training
* Visualization helped me debug far more than raw metrics
* Small implementation details (like proper tensor reshaping or bbox scaling) drastically affect outcome
* Training for just 12 epochs was not enough to get high precision — but the pipeline still worked

**⚖️ AI vs Manual Coding**

I personally wrote:

* Dataset loading logic
* Training loop and loss integration
* Evaluation orchestration

ChatGPT helped with:

* Model architecture definition
* Structuring functions for plotting and prediction
* Error handling and debugging

👉 I used AI more like a “pair programmer” — helpful for syntax and ideas, but I always tested and modified everything myself.

**🧠 Suggestions to Improve This Assignment**

* Recommend providing anchor size suggestions or examples to avoid the common “zero box” output problem
* Add a bonus option to export to ONNX or test on custom images
* Let interns optionally use tools like Roboflow to practice working with custom datasets
* Encourage experimenting with more than 1 detection head (SSD vs YOLO)

**✅ Conclusion**

This was a great hands-on assignment that taught me how object detection pipelines are built and evaluated.  
I now feel more confident not just in building models, but in debugging, analyzing, and documenting them effectively.

Thanks for this learning opportunity!