1. BDD Dataset - Data Analysis Report

a. *Dataset Overview*

i. Dataset Size: 100K images (~5.3GB) and corresponding annotations (~107MB) in JSON format.
ii. Classes Analyzed: Light, Signs, Person, Car, Truck, Bus, Rider, Bike, Motor, Train.
iii. Splits Used: Training and Validation sets (Test set excluded from analysis).

b. *Data Parsing*

i. A custom parser was implemented to extract bounding box information from JSON annotations.
ii. Image files were mapped to their respective annotations for efficient processing.

c. *Key Analysis and Insights*

i. *Class Distribution Analysis*

1. Observation:
    a. The dataset contains an imbalanced distribution of object classes.
    b. Cars and traffic signs are the most common, while motorcycles and buses are rare.
2. Anomalies:
    a. Some object classes appear very infrequently (e.g., motorcycles).
    b. Underrepresented classes may cause biased model predictions.
3. Recommendations:
    a. Oversample rare classes or apply class-balanced loss during training.
    b. Use data augmentation to increase representation of rare classes.

ii. *Bounding Box Size & Aspect Ratio Analysis*

1. Observation:
    a. Bounding box widths and heights show a wide variation in size.
    b. Traffic lights and signs have small bounding boxes, while cars and buses have larger bounding boxes.
    c. The aspect ratio distribution shows that some objects are unusually narrow or wide.
2. Anomalies:
    a. Some objects have extremely small bounding boxes (< 10×10 pixels), making them hard to detect.
    b. Some objects have incorrect aspect ratios, possibly due to annotation errors.
3. Recommendations:

a. Filter out bounding boxes below a minimum threshold size.
b. Normalize bounding box sizes using image resizing techniques.
c. Check for mislabeled annotations by reviewing outliers.

iii. *Occlusion & Truncation Analysis*

1. Observation:
    a. Cars and motorcycles have a higher occlusion rate due to dense traffic.
    b. Some large vehicles (buses, trucks) are often truncated at image boundaries.
2. Anomalies:
    a. Objects marked as fully visible but appear occluded in the images (possible annotation mistakes).
    b. Some occluded objects may be too difficult for the model to detect.
3. Recommendations:
    a. Train with occlusion-aware augmentation (e.g., CutMix, random occlusion).
    b. Use different confidence thresholds for occluded objects.

iv. *Metadata Analysis (Weather, Scene, Time of Day:*

1. Observation:
    a. Most images are from clear weather, with fewer from rainy or foggy conditions.
    b. More images are taken in daylight compared to nighttime.
    c. Some object classes are scene-dependent:
        i. Traffic lights & signs mostly appear in city scenes.
        ii. Pedestrians & bicycles are more common in residential areas.
2. Anomalies:
    a. Fewer images at night, meaning the model may underperform in low-light conditions.
    b. Traffic lights might not be equally distributed across different lighting conditions.
3. Recommendations:
    a. Augment night images using synthetic brightness adjustments.
    b. Apply domain adaptation techniques to improve model performance across lighting conditions.

v. *Train vs Validation Split Analysis*

1. Observation:
    a. The class distribution is mostly similar between training and validation sets.
    b. However, some rare classes appear only in training or validation.
2. Anomalies:
    a. If some classes exist only in training or validation, the model may struggle to generalize.
3. Recommendations:
    a. Ensure balanced class distribution across both sets.
    b. Use stratified sampling to maintain class balance when splitting data.

d. *Final Recommendation:*

i. Identify and filter out extreme bounding box sizes.
ii. Balance the dataset through data augmentation or weighted loss functions.
iii. Augment nighttime images to improve low-light performance.
iv. Ensure validation split contains representative samples of all classes.

2. MODEL TRAINING

a. *Model Selection*

For object detection, the YOLOv8 model was chosen due to its efficiency, real-time performance, and strong accuracy on autonomous driving datasets. YOLOv8 provides a balance between speed and detection capability, making it suitable for real-world applications like BDD100K.

b. *Dataset Preparation*

i. Data Conversion: The original BDD annotations in JSON format were converted to YOLO-compatible text format.
ii. Class Mapping: A predefined mapping was created to align BDD object classes with YOLO labels.
iii. Subset Selection: A subset of images was randomly sampled for training to accelerate the process.

c. *Training Pipeline*

i. Annotation Processing
    1. Extracted bounding boxes and converted them into normalized YOLO format.

        2. Saved annotations in separate text files corresponding to images.
    ii. Dataset YAML Configuration
        1. Created a data.yaml file specifying dataset paths, class names, and the number of classes.
    iii. Model Training
        1. The YOLOv8 model (yolov8m.pt) was fine-tuned for 10 epochs on a small subset.
        2. Training used 640x640 image size, batch size of 16, and checkpoint saving every 3 epochs.

d. *Key Observations*
    i. Efficiency: Training on a fraction of the dataset allowed quick testing of the pipeline.
    ii. Class Distribution Challenges: Class imbalance could impact detection performance, requiring additional techniques like weighted loss.
    iii. Potential Enhancements:
        1. Implementing occlusion handling and nighttime augmentations could further improve model robustness.
        2. Use of a more robust model which could handle the varying sizes of the bounding boxes , example - fasterrcnn with resnet50 as backbone.

3. Evaluation and Visualization of Model Performance

a. *Evaluation Metrics*
    i. To assess the performance of the trained YOLOv8 model, we evaluated it on the validation dataset using key object detection metrics:
    ii. Mean Average Precision (mAP @ IoU=0.50): Measures how well the model detects objects at different confidence thresholds.
    iii. Average Precision per Class: Evaluates the detection accuracy across different object categories.
    iv. Mean Recall (MR): Determines how many ground-truth objects were correctly detected.

These metrics provide insights into the model's detection capabilities, class-wise performance, and potential weaknesses.

b. *Qualitative and Quantitative Analysis*
    i. *Quantitative Analysis:*
        1. The evaluation results revealed class-wise variations in detection accuracy, with high mAP for common objects like cars and lower accuracy for rarer classes like buses and trains.

2. Class imbalance was evident in the dataset, leading to reduced performance for underrepresented categories.
3. Occlusion and nighttime images posed challenges, contributing to lower recall and misclassified detections.

ii. *Qualitative Analysis:*

1. Ground Truth vs. Predictions: Visualized bounding boxes of actual and predicted objects to assess model correctness.

c. *Model Improvement Suggestions*

i. *Handling Class Imbalance:*

1. Implement weighted loss functions or data augmentation to boost rare class representation.

ii. *Better Nighttime Detection:*

1. Introduce low-light augmentation strategies or fine-tune on nighttime-specific samples.

iii. *Occlusion-Aware Training:*

1. Incorporate synthetic occlusion augmentations to improve detection under occluded conditions.

iv. *Hyperparameter Tuning:*

1. Experiment with batch size, learning rate, and training epochs for optimized model performance.

v. *Failure Cases:*

1. Analyze the cases where the model missed detections, had overlapping boxes, or classified objects incorrectly.

vi. *Sample Images:*

1. Display examples where the model performed well versus instances where it struggled, identifying conditions affecting accuracy.