**Adaptive Traffic Algorithm of YOLO11:**

**1. box\_iou Function**

* **Purpose**: This function calculates how much two boxes (like those around detected vehicles) overlap each other.
* **How it works**:
  + It takes the coordinates of two boxes as input.
  + It creates two shapes (polygons) from these coordinates.
  + It calculates the area where they overlap and divides it by the area covered by both boxes combined.
* **Output**: It returns a number between 0 and 1, where 1 means the boxes are perfectly overlapping, and 0 means they don’t overlap at all.

**2. VehicleTracker Class**

* **Purpose**: This class keeps track of vehicles detected in the video.
* **Key Features**:
  + **Initialization (\_\_init\_\_ method)**:
    - Starts with an empty list of vehicles and sets a time limit for how long to remember a vehicle if it’s not seen.
  + **Updating Vehicle Data (update method)**:
    - Takes new detections as input and updates the list of tracked vehicles.
    - For each detection, it checks if the vehicle is already tracked. If not, it adds it and starts tracking its position.
    - If a vehicle is not seen for a while, it gets removed from tracking after the set time limit.
  + **Getting Vehicle Speed (get\_vehicle\_speed method)**:
    - Calculates the speed of a specific vehicle based on its tracked positions.
    - Uses the distance between the first and last known positions and the time between them to calculate speed.

**3. TrafficAnalyzer Class**

* **Purpose**: This class analyzes the traffic based on the vehicles tracked by the VehicleTracker.
* **Key Features**:
  + **Initialization (\_\_init\_\_ method)**:
    - Sets up the area of the road to analyze and how to identify heavy vehicles.
    - Creates an instance of VehicleTracker to manage vehicle tracking.
  + **Analyzing Traffic (analyze\_traffic method)**:
    - Takes new vehicle detections and updates the vehicle tracker.
    - Counts the total number of vehicles and the number of heavy vehicles.
    - Calculates the density of vehicles per area of the road.
    - Returns the counts and densities of vehicles.

**4. main Function**

* **Purpose**: This is the main entry point of the program, where everything starts running.
* **How it works**:
  + It loads the YOLO model for vehicle detection.
  + Initializes the TrafficAnalyzer with a specified road area.
  + Opens the video source (like a webcam or video file) for processing.
  + Enters a loop where it continually captures frames from the video.
  + For each frame, it uses the YOLO model to detect vehicles.
  + Calls the analyze\_traffic method to analyze the traffic based on detected vehicles.

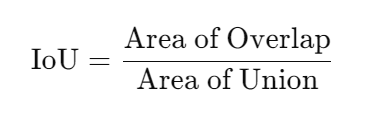
**Summary**

* **box\_iou**: Measures how much two boxes overlap.
* **VehicleTracker**: Tracks vehicles, remembers their positions, and calculates their speed.
* **TrafficAnalyzer**: Analyzes traffic by counting vehicles and calculating their density.
* **main**: Sets everything up and runs the detection and analysis on video frames.

**1. Intersection over Union (IoU)**

**Concept**: The Intersection over Union (IoU) is a common evaluation metric used to measure the accuracy of an object detection model. It calculates the overlap between two bounding boxes (in this case, the predicted bounding box and the ground truth bounding box).

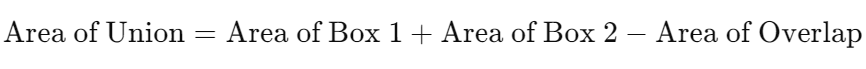
**Formula**:



Where:

* **Area of Overlap**: The area where the two boxes intersect.
* **Area of Union**: The total area covered by both boxes.

**Mathematical Steps**:

* **Area of Overlap**: This is calculated by finding the coordinates of the intersection rectangle. If the intersection rectangle has corners at (x1,y1) and (x2,y2) then: 
* **Area of Union**: This is the area of both boxes minus the area of overlap: 

**2. Calculating Distance**

**Concept**: The distance is typically calculated using the Euclidean distance formula, which measures the straight-line distance between two points in space.

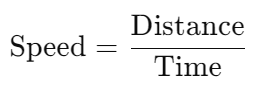
**Formula**: For two points (x1,y1) and (x2,y2):



**3. Calculating Speed**

**Concept**: Speed is calculated based on the distance traveled over time.

**Formula**:



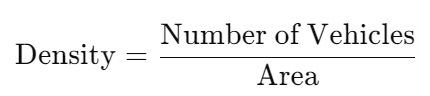
Where:

* **Distance**: The distance between the vehicle's initial and final position.
* **Time**: The difference in time between the two positions (for example, the time between two frames).

**4. Calculating Vehicle Density**

**Concept**: Vehicle density measures how many vehicles are present in a given area.

**Formula**:



Where:

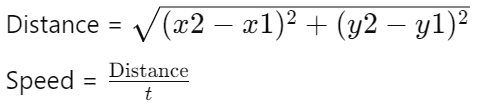
* **Number of Vehicles**: The total count of vehicles detected in the specified area.
* **Area**: The size of the area being analyzed (e.g., square meters).

**Putting It All Together**

* **Detecting Vehicles**: Using a model like YOLO, bounding boxes are created around detected vehicles.
* **Calculating IoU**: For each detected vehicle, the IoU is calculated to assess whether the detection is accurate by comparing the predicted box with a ground truth box.
* **Tracking Vehicles**: The position of each vehicle is updated over time using the Euclidean distance to calculate how far it has moved.
* **Calculating Speed**: Speed is computed using the distance moved between two consecutive frames and the time taken for those frames.
* **Analyzing Traffic Density**: Finally, the total number of detected vehicles is divided by the area to understand how congested the traffic is.

**Example Calculation**

Suppose you have:

* Two bounding boxes for vehicles: Box 1 (x1,y1,x2,y2) and Box 2 (x3,y3,x4,y4)
* You find the area of overlap using the IoU formula.
* For a vehicle moving from position (x1,y1) to (x2,y2) in t seconds, you calculate:
* 
* If there are n vehicles detected in an area of A square meters:

