Exploratory Data Analysis (EDA) Report on Cityscapes dataset

Project title: AI-based Autonomous Driving

Prepared by: S A Faraaz Ahmed

Date: 16/10/2024

Introduction

This report presents the initial Exploratory Data Analysis (EDA) conducted on the Cityscapes dataset. The dataset comprises images with pixel-level annotated counter parts of it and JSON files containing object polygon data for tasks such as lane detection, object detection/avoidance, traffic sign identification, and semantic segmentation. The goal of this analysis is to understand the characteristics of the data, assess preprocessing needs, and detect any potential challenges for training deep learning models.

Dataset and Methodology (Exploration)

Data Resolution and Channels

The resolution, defined by the dimensions of the image (2048 x 1024 pixels), informs us about the image's level of detail. Higher resolutions typically provide more detail, which can be beneficial for tasks requiring fine-grained analysis, such as object detection or semantic segmentation. However, larger images also require more computational resources, making it essential to balance resolution with processing efficiency.

The number of channels (3 in RGB) indicates that the image is in color, specifically an RGB image, where each pixel is represented by three values corresponding to the intensities of Red, Green, and Blue. Understanding the number of channels is important for selecting appropriate algorithms for tasks like image normalization or transformation.

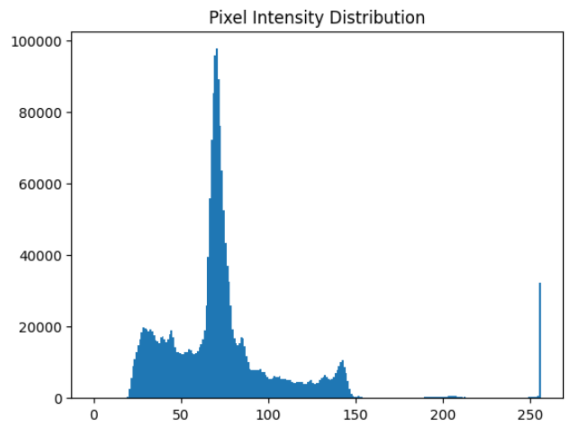
Finally, the aspect ratio, which is 2.00, shows that the width of the image is twice its height. The aspect ratio is essential for tasks like resizing or cropping, where preserving the natural proportions of the image prevents distortion and maintains the integrity of the visual content

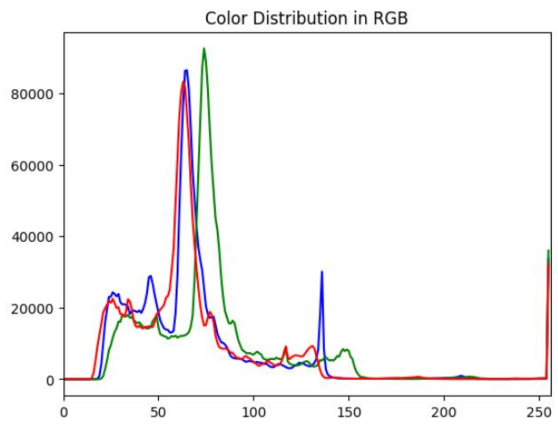
Data Composition

The dataset comprises totally of 5000 images, by itself is split into train, validation and test parts, in which 2975 are for train, 500 are for validation and 1525 for test i.e. in percentage which goes to Train images are 59.5%, Validation Images are 10.0%, Test Images are 30.5%.

Pixel and color Intensity distribution

The histograms shown represents the pixel intensity distribution & color distribution of an image, where the x-axis denotes pixel intensity values ranging from 0 (black) to 255 (white) for an 8-bit image, and the y-axis represents the number of pixels corresponding to each intensity level.





Key observations from the histogram:

> Concentration of Pixel Intensities: The distribution shows a large spike in the lower intensity range, specifically around the 50-75 range, indicating that the image contains many darker pixels or regions dominated by lower brightness. This might suggest that the image has a lot of shadows or darker areas.
>
> Mid-Range and High Intensities: There are fewer pixels in the mid-range (between 100-200) and very few in the higher intensity range, except for a sharp peak near the maximum value (255). This suggests that while the image is predominantly dark, there is a presence of some bright or white areas, which could represent lane markings, highly reflective surfaces or bright objects within the image.

Blurriness score on Laplacian scale

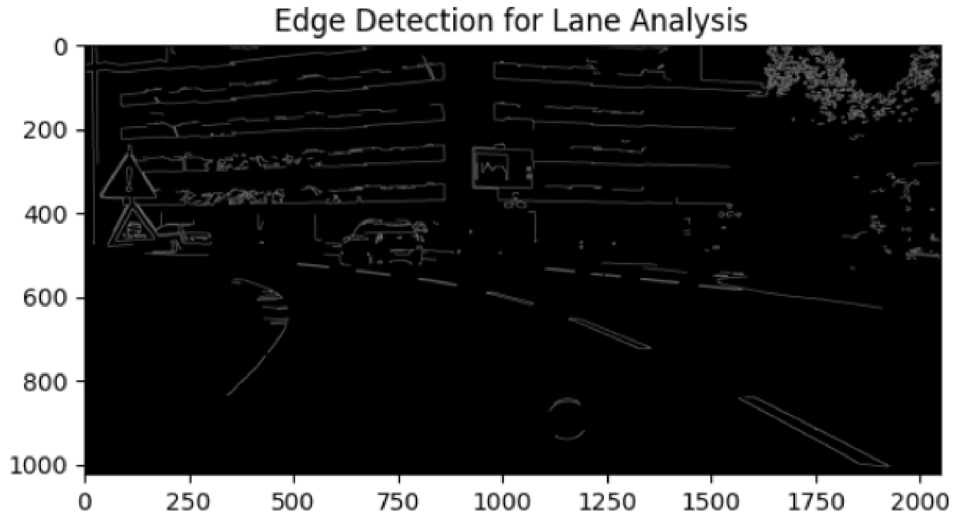
The Blurriness score 48.25 indicates a moderately blur image, on this scale higher is more sharp, considering 100 as sharp, 48.25 on our dataset wouldn't affect the model much for training, but if exposed to other images for analysis, may become a point of concern.

Edge Detection Annotation Using Canny on Grayscale

Edge detection is a fundamental image processing technique that highlights the boundaries within an image by detecting sharp changes in intensity or color. The Canny Edge Detection algorithm is a popular method for edge detection, especially for its precision and accuracy.

First, the image is converted to grayscale (if it isn't already) to simplify the analysis. In grayscale, each pixel represents brightness intensity, making it easier to identify intensity changes.

The image is smoothed using a Gaussian filter to reduce noise. The intensity gradient of the image is computed, identifying regions where there is a rapid change in pixel values. Thin the edges to retain only the most significant edges, reducing the chance of detecting false edges. Two thresholds are applied to classify strong and weak edges, and the algorithm traces the edges through the image.



Resizing

Resizing images is a common preprocessing step, especially when working with datasets for machine learning. The aim is to bring all images to a uniform size, which simplifies training models by ensuring consistent input dimensions.

Resizing is mandatorily not necessary for our dataset since all the images are in a uniform Resolution, but to enhance the training process and to reduce the computational resources, Resizing can be performed in accordance with the Aspect ratio.

Normalizing

Normalization is the process of scaling the pixel values of an image to a specific range, typically between 0 and 1 or -1 and 1. This is especially important in machine learning models, as it helps with Stability, Normalized data leads to faster convergence during training by ensuring the model doesn't get stuck due to outlier pixel values.

Uniformity, different images can have varying pixel intensity ranges, and normalization brings consistency across the dataset.

Data Integrity

Ensuring data integrity is crucial when working with large datasets, especially in projects involving image datasets and their corresponding annotations (stored in PNG masks & JSON format). It is required to clean the dataset in terms of removing the corrupted values and filling the missing values, in our dataset upon check there was no corrupted image, ensuring our data is clean.

Results

The dataset consists of images taken in dim daylight, with pixel intensities concentrated around 75 across all three channels, which doesn't hinder training but may affect generalization to other dataset.

Resizing the images isn't strictly necessary but to save up on some computational resources can be resized in accordance with aspect ratio. The images were normalized between [0 1].

Since the dataset lacks bounding box annotations, the training must rely on polygon-based annotations to form bounding boxes and masks to be used for object detection.

Manual annotation of 40-50 samples for YOLOv8 training failed to generalize to unseen images, primarily due to high blurriness (with a score of 48.25) and the limited number of annotated samples. The blurriness impacted edge detection and feature extraction, while the small dataset size limited the model's performance. To improve results, a larger, more diverse dataset with better-quality images would be necessary.

The Annotation part for lane detection is done using Canny algorithm, since there is no annotations present in gtfine set for it. This will help in detecting the edges and to proficiently map the lane markings

Conclusion

The Cityscapes dataset of Images is already preprocessed upto some extent & much preprocessing may not be needed, but since annotation and labelling are absent for some modules like object and lane detection, annotation processing is necessary.

The model trained with this dataset can well perform on the test images of same dataset, but may cause some discrepancy on the outside data.

Future Objectives (for the next two weeks)

Data Loading Functions: Data loading functions facilitate the efficient retrieval and management of image and annotation files from the dataset, ensuring smooth access for subsequent processing steps.

Image Preprocessing: Image preprocessing involves operations such as resizing, normalization, and edge detection to enhance image quality and prepare it for analysis, ensuring that the data is suitable for model training.

Annotation Preprocessing: Annotation preprocessing includes converting polygon-based annotations into bounding boxes or segmentation masks, ensuring compatibility with the chosen object detection framework and improving the accuracy of model predictions.

Dataset Splitting & Preparation: This step might not be necessary because our dataset has already been divided into Train, validation and test. But to receive more precise output we can change the proportion of the already divided dataset.

References

https://paperswithcode.com/task/lane-detection

https://arxiv.org/abs/1506.01497

https://www.analyticsvidhya.com/blog/2021/12/traffic-signs-recognition-using-cnn-and-keras-in-python/

https://www.mathworks.com/solutions/image-video-processing/semantic-segmentation.html#:~:text=Semantic%20segmentation%20is%20a%20deep,pavement%2C%20and%20other%20road%20features.