Car segmentation

Tags	Autoimport
Can be used	Yes

Introduction

At Arealize, we use semantic segmentation to identify and classify elements within architectural floor plans. In this assignment, we invite you to demonstrate your ability to build a segmentation model, process image data, and develop a clear and functional inference pipeline.

Your task is to design a solution that reflects thoughtful model selection, data preprocessing, and clean, maintainable code. While we are interested in the model's performance, it is not the primary focus. Instead, we will evaluate your reasoning behind model choices, the structure and clarity of your code, and your ability to deliver a robust and reproducible pipeline.

Please prioritize code readability, documentation, and modularity. Aim for simplicity and explainability over extensive tuning or achieving top accuracy metrics.

We will not run the code directly, so please make a rapport with the dice score and IoU on the test set.

The data

For this assignment, you will work with a publicly available semantic segmentation dataset hosted on Kaggle:

https://www.kaggle.com/datasets/intelecai/car-segmentation

The dataset consists of images of cars taken primarily from side views, along with corresponding segmentation masks. Each image file in the `images/` folder has a matching mask in the `masks/` folder, with filenames aligned (e.g., `003.png` in `images/` corresponds to `003.png` in `masks/`).

The segmentation masks are labeled pixel-wise with the following five classes:

- `0` – background
- `1` – car
- `2` – wheel
- `3` – light
- `4` – windows

The dataset is not pre-split into training, validation, and test sets. You are expected to implement a logical data split as part of your solution.

Task

Your goal is to build a semantic segmentation pipeline using the provided dataset. The pipeline should include data preprocessing, model implementation, and result visualization. Please focus on producing clean, modular, and understandable code.

Your tasks are as follows:

- Preprocess the data: Load the dataset and implement a logical split into training, validation, and test sets. Explain your reasoning behind the chosen split strategy.
- Implement a segmentation model: Choose and implement a semantic segmentation model appropriate for this task. Justify your choice based on the problem context, not performance alone.

- Apply data augmentations (optional but recommended): To improve generalization, we encourage you to include meaningful augmentations during training (e.g., flips, rotations, color jitter, etc.).
- Create an inference pipeline: Build a reusable function or module that accepts a trained model and performs inference on new images. The output should include predicted masks.
- Visualize the results: Show a selection of input images alongside their predicted and ground truth masks. This helps to qualitatively assess your model’s performance.

While we appreciate good performance, your solution will primarily be evaluated on clarity, reasoning, and code quality. Aim for simplicity, structure, and reproducibility.