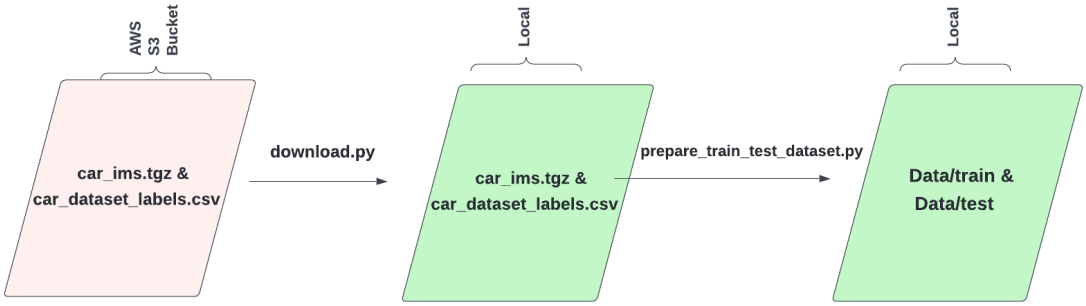
Vehicle Classification Workflow

1 - *Data Collection & EDA*

The dataset was store on a S3 Bucket on AWS Cloud. To download it locally, I coded the *download.py* file which does the job.

Once I had the dataset and its labels, I had to create the Train and Test folders. Each one of them has sub-folders, one per class.

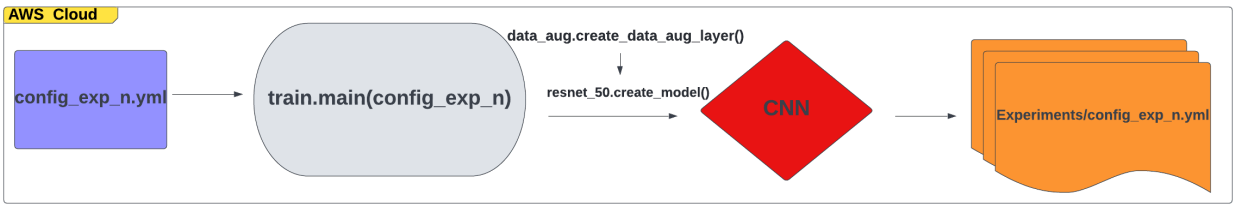


Then, the EDA was done on a Jupyter Notebook, in order to gain more knowledge of the dataset.

2 - *Model Training & Evaluation*

After we have our images in place, it´s time to create our first CNN and train it on our dataset. To do so, we will make use of *scripts/train.py*.

The only input argument it receives is the *config_exp_n.yml* file with all the experiment settings like dataset, model, output folder, epochs, learning rate, data augmentation, etc.



Each time a new model was going to be trained, a new *config_exp_n.yml*, with “n” as the counter, is created with different values for the hyperparameters. This is locate inside a new sub-folder inside *experiments/*. Also the results are store on the same sub-folder of the specific experiment with the model weights and training logs.

This process is done on a Clone repository of the project on the AWS Cloud. This will let you use GPU instead of your local CPU, and will extremely accelerate the training process. This is an iterative process, personally I trained around 30 CNN. Once you

have your final model, you will have to use the saved weights to evaluate its performance on the Evaluation Notebook.

3- *Improve Classification by Removing Noisy Background*

This was the last part of the project workflow, as it was a way of improving our "Baseline CNN". So, it was done once I had the base model I just mentioned.

As we saw on the EDA, most of the images have a noisy background which may affect our model learning during the training process. What I did was to isolate the vehicle from the rest of the content in the picture.

I used Detectron2 Framework for this, more specifically a model called "R101-FPN" trained on COCO dataset (it has a good balance between accuracy and speed).
The objects assigned to be detected were only "car" and "truck".

The code for this tasks had been coded in *scripts/remove_background.py*, were the initial dataset was used to remove the background from pictures, and then storing them on a new folder. Finally, on *utils/detection.py*, the logic to get the vehicle coordinates from the image was implemented.

Once this was done, I repeat the second step of the workflow until I got a Model which evaluation performance satisfy m.