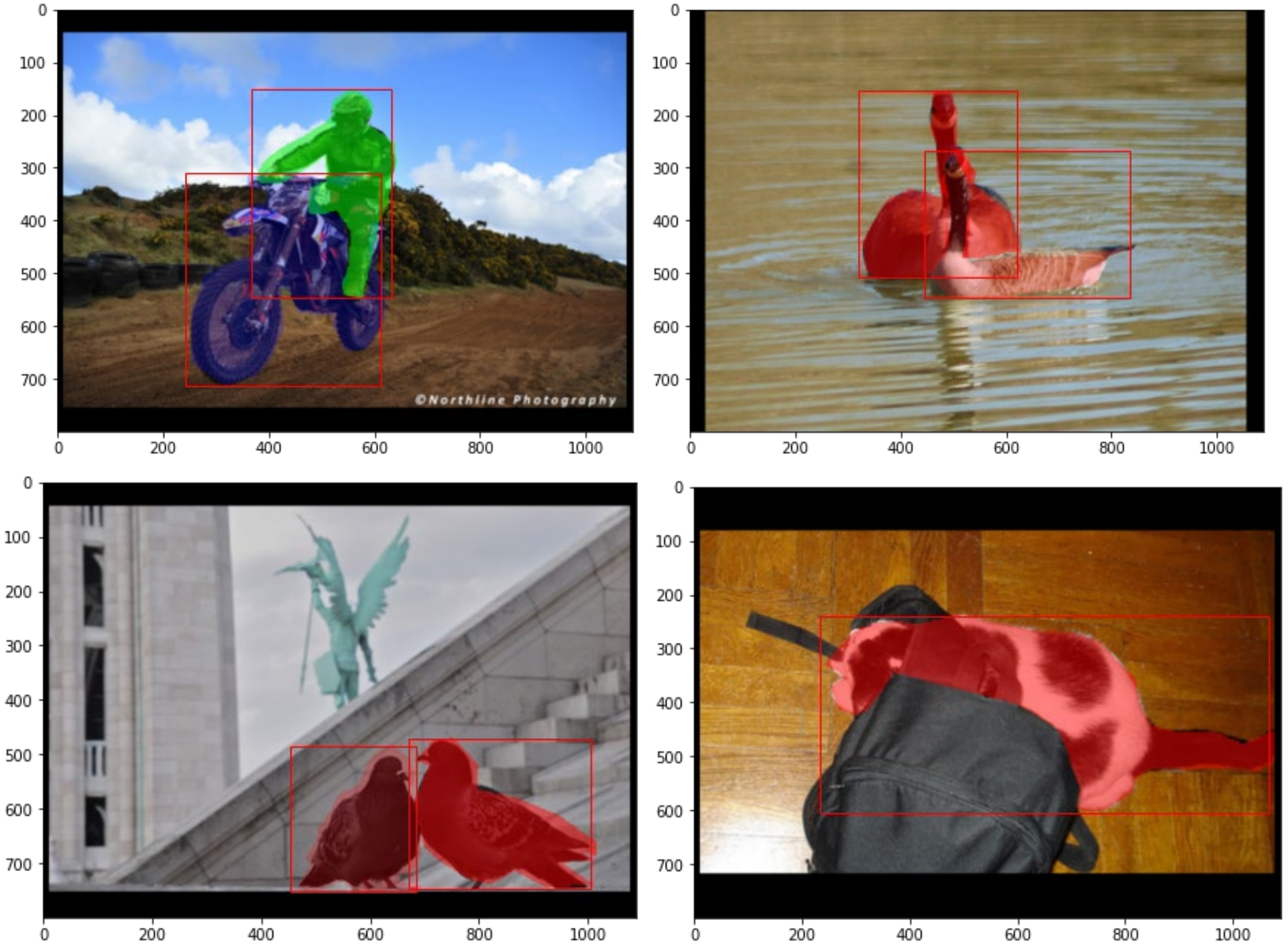
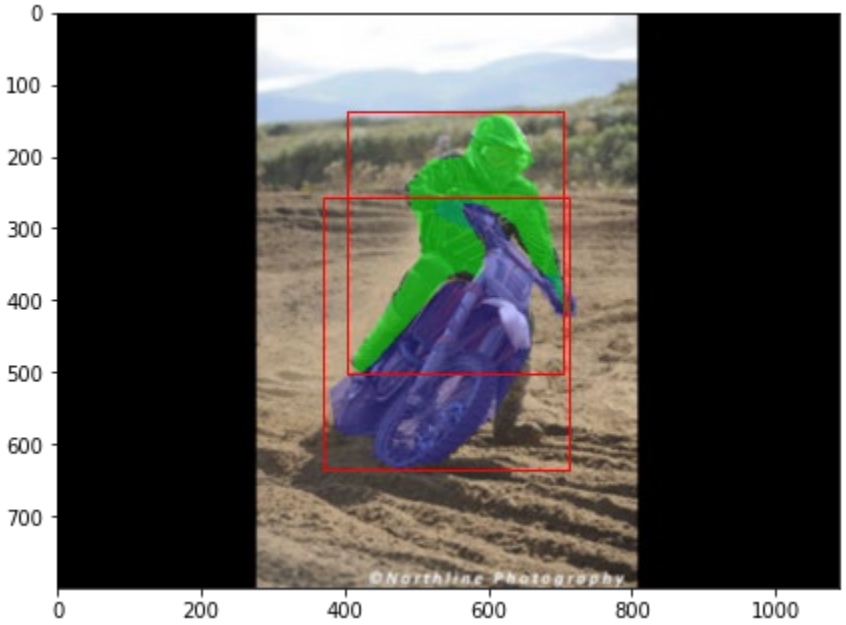
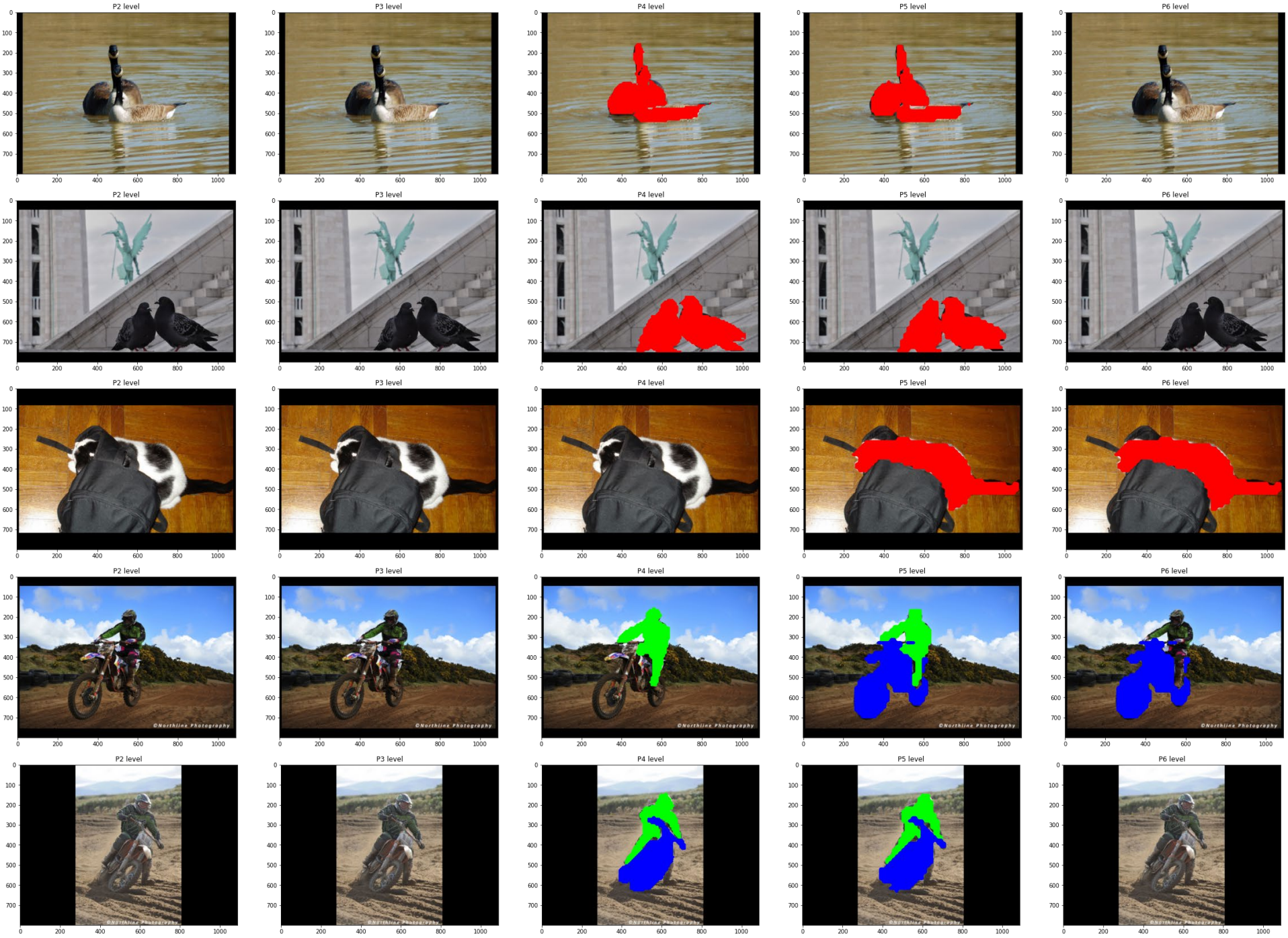
Dataset Visualization



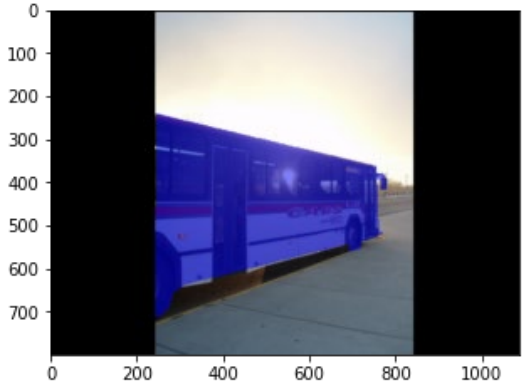

0
100
200
300
400
500
600
700
0
200
400
600
800
1000
©Northline Photography

Target Assignment Plot



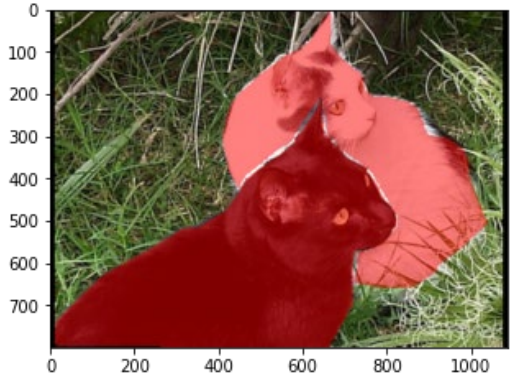
Final Inference Result

Ground Truth

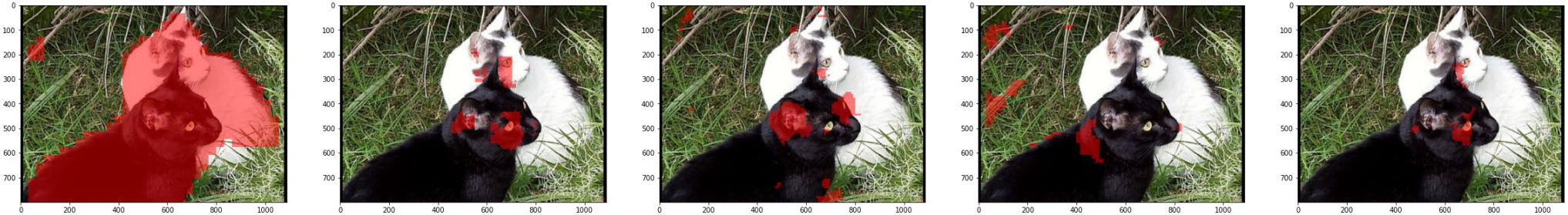


Result

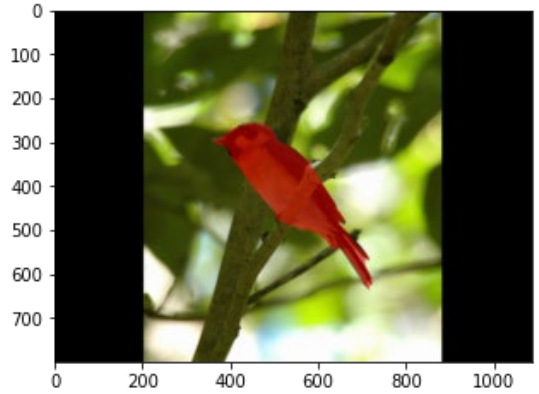
Ground Truth



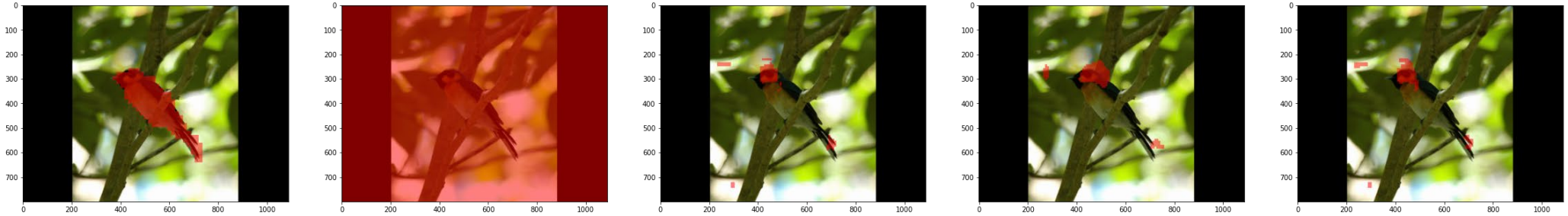
Result



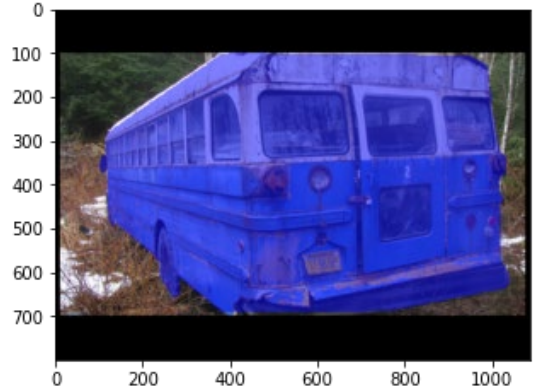
Ground Truth



Result

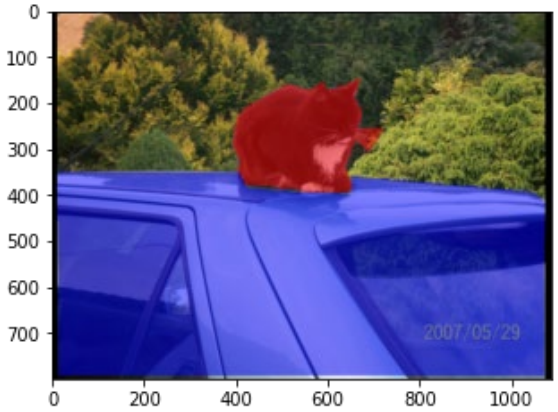


Ground Truth

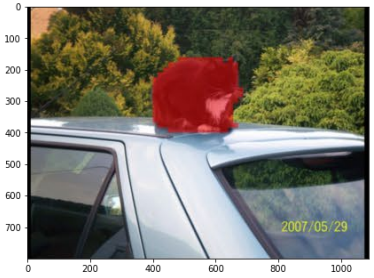


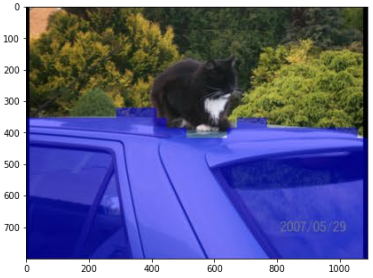
Result

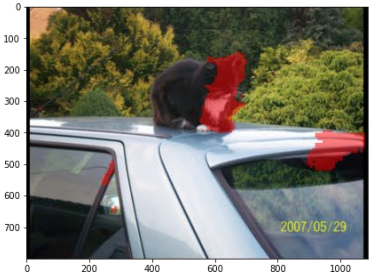
Ground Truth



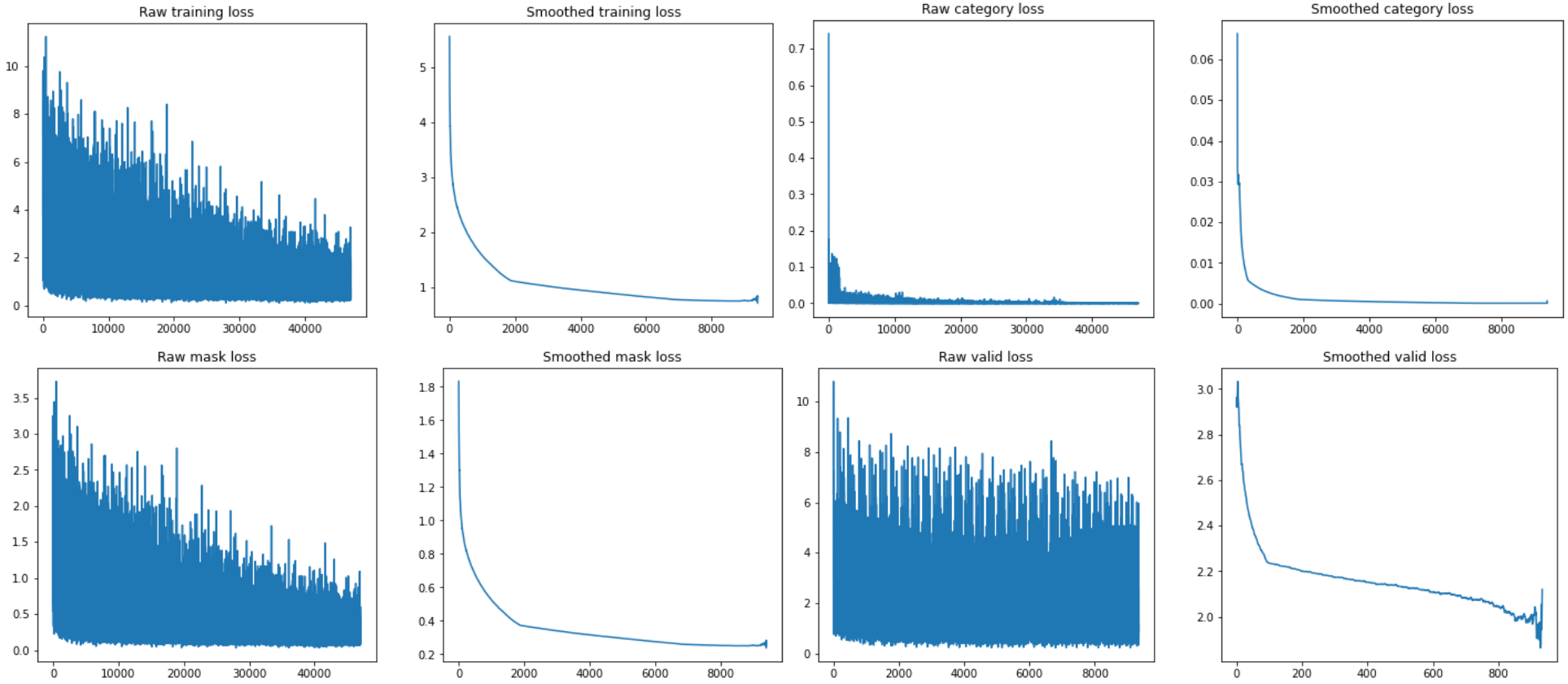
Result







Training and validation loss curve



Discussions:

1. Implementation of SOLO:

We basically followed the instructions and finished the task. There were some challenges and our solutions that we think are worth mentioning.

a. sparse representation for reducing RAM

In the implementation, one main challenge was that the storing and processing of data would takes up a lot of RAM and time. We noticed that for masks, whose matrix representation has a lot of zeros (indicating sparsity), can be stored and processed in another way. We only saved the active masks and recorded corresponding indices to process the data and neglected all other zeros. Which helped to save a lot of RAM and time.

b. flattening representation for sorting and matrix NMS

In post-processing, we first coded several loops in order to compute the score and do sorting. Then, we referred to the live-coding session last year and got to know how to compute the score and by flattening the data, which helped a lot.

c. Quick debugging using small-sized data

Our solo result was not so ideal at first. However, we couldn't judge whether the problem came from post-processing or training implementation, and training for SOLO took a lot of time. Therefore, we tried to train very small-sized data (around 100 images) and fewer epochs so that we could do trial and error quickly, and that method helped us to find the problem.

2. SOLO Performance:

Our model performs quite good in masking, however for category, we found that it usually produces relative high score for animals and relative low scores for the other two. After carefully thinking, we doubt that it is due to the unbalanced data distribution. In all the labels, there are 990 vehicle labels, 882 human labels and 1971 animal labels. We think the dominant number of animal labels cause the preference of category of our model