LINC - Tryolabs

Documentation of design process and model specifications

Introduction

The purpose of this paper is to document the project that took place in June-July 2019 between Tryolabs and LINC, with the objective of detecting lion body parts and whisker spots in pictures.

This is stage one of a three stage project by LINC with the general purpose of identifying lions based on their pictures. Stages 2 and 3 will consist of using whisker spot pattern matching and face-id algorithms to identify the lions, by taking the normalized images from stage 1 as inputs.

Objective

The objective of this project was to build two models, one for detecting certain body parts in lions, and one for detecting whisker spots. Each model had to be able to run one inference in under 5 seconds on a medium range laptop using just its CPU.

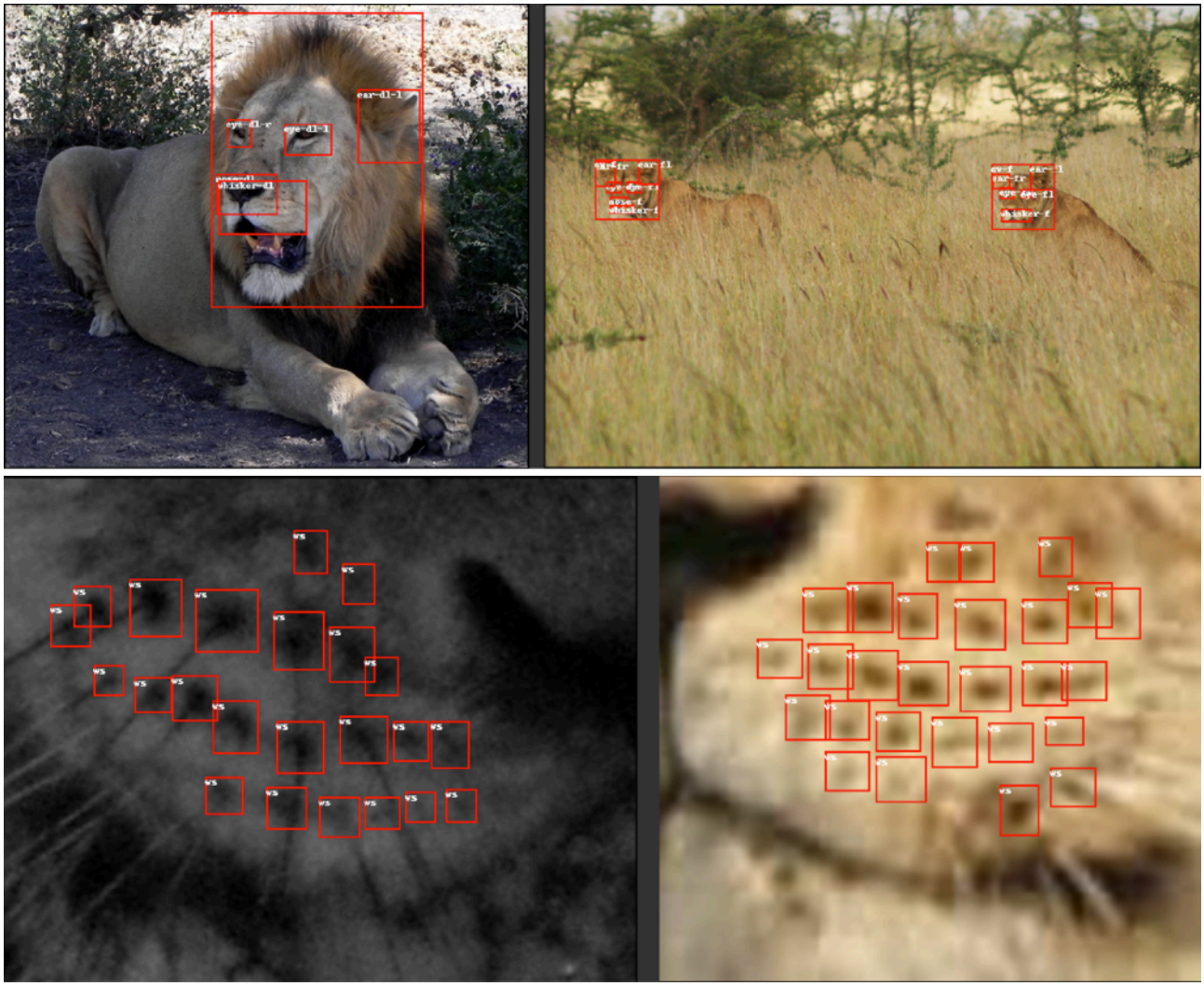
Data

The tagged data provided by LINC consists of 1355 tagged pictures of lions. The tags consisted of bounding boxes drawn around each relevant body part, and they were divided between the following tag classes:

```
cv-f            (head-Front)
ear-fl          (ear-FrontLeft)
ear-fr          (ear-FrontRight)
eye-fr          (eye-FrontRight)
eye-fl          (eye-FrontLeft)
nose-f          (nose-Front)
cv-sl           (head-Left)
ear-sl          (ear-SideLeft)
eye-sl          (eye-SideLeft)
nose-sl         (nose-Left)
whisker-sl      (whisker-Left)
cv-sr           (head-Right)
ear-sr          (ear-SideRight)
eye-sr          (eye-SideRight)
nose-sr         (nose-Right)
whisker-sr      (whisker-Right)
cv-dl           (head-DiagonalLeft)
ear-dl-l        (ear-DiagonalLeft-Left(ear))
ear-dl-r        (ear-DiagonalLeft-Right(ear))
eye-dl-l        (eye-DiagonalLeft-Left(eye))
eye-dl-r        (eye-DiagonalLeft-Right(eye))
nose-dl         (nose-DiagonalLeft)
whisker-dl      (whisker-DiagonalLeft)
cv-dr           (head-DiagonalRight)
ear-dr-r        (ear-DiagonalRight-Right(ear))
```

```
ear-dr-l          (ear-DiagonalRight-Left(ear))
eye-dr-r          (eye-DiagonalRight-Right(eye))
eye-dr-l          (eye-DiagonalRight-Left(eye))
nose-dr           (nose-DiagonalRight)
whisker-dr        (whisker-DiagonalRight)
ws                (whisker-Spot)
full-body         (full-body)
```

Notice that each body part is labeled as centered/diagonal/side, as this pose information was deemed to be important for obtaining good normalized data for stages 2 and 3.



Some sample pictures of the data provided by LINC.

Approach

Initially we thought about using the same model for both body parts and whiskers, but after examining the data we noticed that the only way to detect the general body parts and the whisker spots in one single inference step was to process the very high resolution original images without doing any downsampling. This was due to the fact that downsampling the full body lion image would make the whisker spots be too low resolution to be distinguishable.

This was a problem because processing the full body images in a high enough resolution for the whisker spots to be distinguishable was prohibitive, as processing such a large image would make inference time go way above our limit. It was just faster to separate the full body images and the whisker spot images and process each at the minimum resolution that didn't significantly reduce accuracy.

The model we chose was a FasterRCNN object detector with a Resnet-50 FPN base network for both models. The FasterRCNN architecture has proven to be very accurate and the Resnet-50 FPN backbone provides a good balance between inference speed and accuracy.

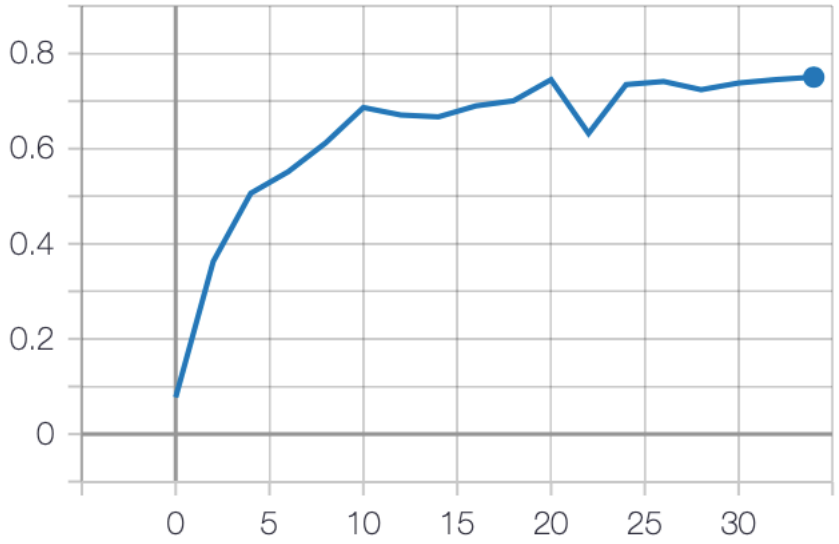
On the whisker spot model we considered using only the RPN part of the FasterRCNN model, as we only had one class, but benchmarking showed that it would reduce inference time only marginally.

The codebase is built using torchvision (pytorch's default computer vision module), and we used the library's reference FasterRCNN model as a starting point.

The nature of the labels demanded some changes in the network. First, we modified the horizontal flip data augmentation to change left/right labels coherently when flipping pictures. Second, we modified the NMS filter to group all labels of the same type together, so all eyes get filtered together, all ears get filtered together, and so on.

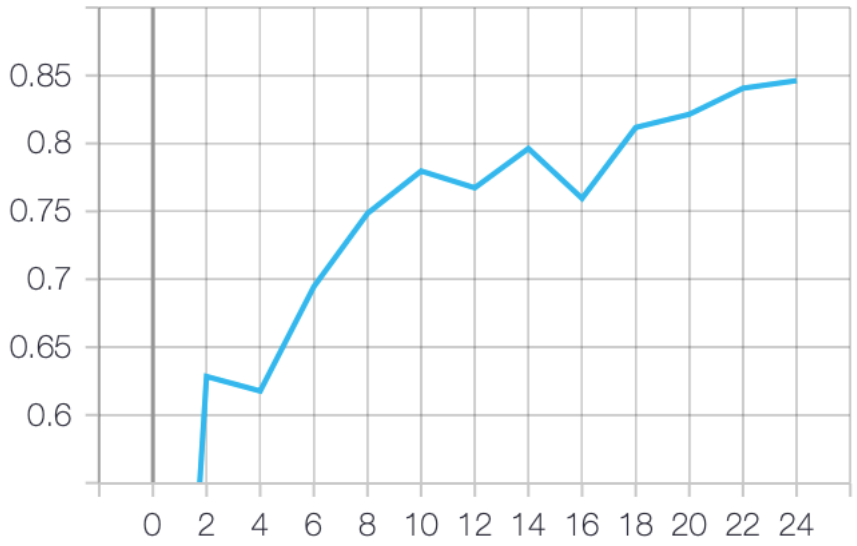
Results

The body parts model trains in about 2.5 hours on a GTX1080 and reaches a mAP@50 of 0.75. Errors consisted mostly of diagonal/front and front/side edge cases which were difficult for humans to distinguish as well. It was trained in 35 epochs and the validation curve can be seen in the following graph:



mAP@.50 vs epochs while training body parts model

The whisker spots model trains in about 13 minutes on a GTX1080 and reaches a mAP@50 of 0.84. Errors consisted mostly of whiskers which were ambiguous and hard for humans to classify as well. It was trained in 25 epochs and the validation curve can be seen in the following graph:



mAP@.50 vs epochs while training whiskers model

Inference time on both models is under 5 seconds in several laptops we tried, using only CPU, and less than 0.16 seconds on an Nvidia GTX1080 GPU enabled desktop machine.

All code and jupyter notebooks with demos are in: https://github.com/tryolabs/LINC