A Survey of Deep CNNs for Mouse Paw Localization

Molly Lucas[1,2,3] & Konstantin Kaganovsky[1]

1. Wu Tsai Neurosciences Institute, Stanford University, Stanford, CA 94305
2. Department of Psychiatry and Behavioral Sciences, Stanford University, Stanford, CA 943053
3. Veterans Affairs Palo Alto Healthcare System, and the Sierra Pacific Mental Illness, Research, Education, and Clinical Center (MIRECC), Palo Alto, CA, 94394, USA

Motivation

To better understand the neural basis of motion, neuroscience studies record cellular activity in the brain while mice perform an action. In order to link movement behavior with brain activity, mouse foot placement is manually labeled by experimenters in each image frame. This is incredibly time consuming and also involves human error (differences in how each individual labels a foot using an X,Y coordinate). A recent Nature Neuroscience paper used convolutional neural network (CNN) to automate this task. DeepLabCut has a high accuracy rate but takes enormous computational resources and time. Here, we both explore ways to improve this new field standard (DeepLabCut) and also test two variants of an object detection algorithm, Faster R-CNN, to see if we can improve performance (maintain high accuracy while improving speed) of labeling mouse feet in this type of image.

Data & Preprocessing

- Grayscale images were collected using a high resolution (800x600 pixels) camera under infrared light illumination at 150 frames per second. Each frame constitutes a single image.
- Mouse position is fairly constant on the ball (left sagittal view). Example image (left foot marked in pink) seen on right. Left foot X,Y coordinates were used as the label in each image (hand labeled by experts in Ding Lab at Stanford University).
- Data were not pre-processed or augmented (low bias results did not indicate more training data was needed).
- Training set: 973 images (60% of data), Development set: 325 images (20% of data), Test set: 325 images (20% of data). The same training, development, and test images were used in each independently tested model. Development set was used to test variations within each model. Test set was only used one as a final comparison across the three models.

Models

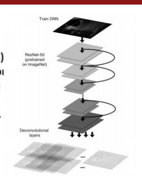
We tested four models: DeepLabCut, DeepLabCut with ResNet101, Faster RCNN, and Faster R-CNN combined with ResNet50.

DeepLabCut

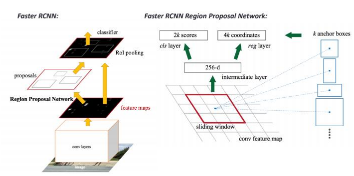
- DeepLabCut[1] is the current best field standard for markerless labeling of experimental subjects' joints. DeepLabCut (DLC) uses the feature detector portion of the best human pose estimation model[2]. It uses a ResNet architecture pre-trained on the ImageNet dataset; by using deconvolutional layers, the final layers are modified to output a feature map that matches the input size. This feature map is then passed through a regression layer to refine the x,y position and finally a sigmoid activation layer. Cross entropy loss is combined with Huber loss (for regression layer) and the network is trained with SGD.

DeepLabCut (ResNet101)

- To test whether a deeper network would perform better, we exchanged the standard ResNet architecture with ResNet101. All other layers, input, and output remained consistent with the original DeepLabCut. To test intermediate supervision, an intermediate detection/classification layer was added to the output of the conv4 block of ResNet-101 and intermediate CE loss was added to the total loss for SGD.



DeepLabCut. Adapted from Mathis et al. 2018 Nat Neuro



Faster RCNN. Adapted from Ren et al. 2016 CVPR

Faster R-CNN

- Faster R-CNN[3] is an object detection network that increases the speed of Fast R-CNN[4] by adding a novel Region Proposal Network (RPN). This RPN is a fully convolutional deep network combining bounding box prediction with object scores, which eliminates the main computational bottleneck of Fast R-CNN. The output layer of Faster R-CNN classifies proposed regions to match input labels of interest. Loss function uses binary classification (correct/incorrect) evaluated by measuring intersection over union of the proposed bounding box and the true label bounding box. It is trained using SGD.

Faster RCNN with ResNet50

- Here, we exchanged the convolutional layers of Faster R-CNN with ResNet50. The same Regional Proposal Network was used to estimate bounding box location, and the same classifier layer was used as the model output.

Error Analysis

- DeepLabCut showed both low bias and low variance. Bayes error for this task can be approximated by human ability, which gives a 0% error rate (with some individual variability in X,Y coordinate label placement). Therefore, even the low error rate seen initially has room for improvement. To improve bias, we used a deeper CNN (exchanging the ResNet50 component in DeepLabCut with a ResNet101).
- Based on the difference between train and CV RMSE, the larger network (DeepLabCut ResNet101 – especially the version with intermediate supervision) is overfitting the training data so increasing the amount of training data may help overcome this problem and further decrease the CV RMSE.

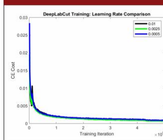
References: [1] A. Mathis *et al.*, DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nat. Neurosci.* **21**, 1281 (2018). [2] E. Insafutdinov *et al.* in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, M. Welling, Eds. (Springer International Publishing, 2016), *Lecture Notes in Computer Science*, pp. 34–50.[3] S. Ren *et al.*, Faster RCNN: Towards Real-Time Object Detection with Regional Proposal Networks, *CVPR* (2016). [4] R. Girshick, Fast R-CNN. *CVPR* (2015).

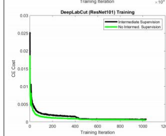
Error Analysis continued

- A main difference between DeepLabCut and Faster R-CNN is that Faster R-CNN uses bounding boxes as labels, whereas DeepLabCut uses X,Y coordinates. We initially used a small box to resemble an X,Y point. This resulted in 0% accuracy even on training (high bias). After manually analyzing 100 incorrectly labeled images, we found that the model was identifying light/dark edges, likely due to the X,Y label being placed at the edge of the foot. To fix this, we made the input label bounding box larger and centered it around the foot. We tested 5 variants of this before finding the input label with the highest accuracy on training (RMSE of 80 pixels). This was used in subsequent testing for Faster R-CNN.
- Using Faster R-CNN, variance was quite low (minimal difference between training and development error), but bias was higher than when using DeepLabCut (higher error both in training and development). Therefore, we tried changing the Faster RCNN architecture to mirror DeepLabCut, hoping this hybrid could improve the bias while still maintaining the speed / low computational requirements of Faster RCNN.

Results

We decided to use root mean square error (RMSE) as a single-value metric to benchmark performance across models because our end goal is to find consistent X,Y coordinates of a feature on the mouse paw to correlate with recordings from the mouse neurons. Further, the gold standard in the field is using RMSE to benchmark the model[1].





DeepLabCut Training

Learning rate	Training RMSE	Dev. RMSE
0.0005,0.002,0.0002	1.285	6.1486
0.0025,0.01,0.001	**1.0097**	4.4437
0.005,0.02,0.002	0.9204	5.41
0.01,0.04, 0.004	0.9081	7.3286

DeepLabCut (ResNet101) Training

DeepLabCut ResNet101	Training RMSE	Dev. RMSE
No Intermed. Supervision	0.6251	4.9553
Intermediate Supervision	0.567	9.51730

Experiment 1: Improving DeepLabCut (ResNet50)

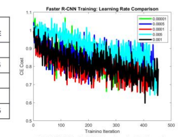
- AIM: While DeepLabCut is a field standard, it is very new (2018). Here, we tested learning rate and model architecture to try to improve accuracy.
- The scaling factor of 0.8 was the best, but there is surprisingly no major difference from 0.5 to 1.2 scaling - indicating that at our resolution, detecting a simple white paw on a black background does not require high resolution.
- The deeper ResNet architectures overfit the training data, adding more training data could overcome this problem.

Experiment 2: DeepLabCut with ResNet101

- AIM: To see whether a deeper network would improve accuracy.
- The deeper ResNet architectures overfit the training data, adding more training data could overcome this problem.

Faster R-CNN Training

Learning rate	Training RMSE	Dev. RMSE
0.001	80.1	86.7
0.005	103.5	107.5
0.0001	80.1	83.6
0.0005	77.3	82.9
0.00001	174.3	166.6



Experiment 3: Comparing Faster RCNN to DeepLabCut

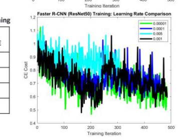
- AIM: Since DeepLabCut is the only algorithm used for this task, we tested whether another object detection model (Faster R-CNN) performed similarly to DeepLabCut (RMSE as output metric).
- Learning rate of 0.0005 showed the lowest development error. The model had greater bias (assuming Bayes error is close to zero) but fairly low variance. This suggests either the model architecture is not optimal or else more training data could be useful.

Experiment 4: Creating a hybrid between DeepLabCut and Faster RCNN

- AIM: Faster RCNN trained and ran in a fraction of the time as DeepLabCut but had a lower accuracy on both training and development sets. To try to improve accuracy while maintaining superior speed, we integrated ResNet50 (as used in DeepLabCut) with Faster RCNN to create a hybrid between the two. The goal was to use accuracy as a satisficing measure (RMSE < 10 pixels) and use testing speed as our optimizing measure.

Faster R-CNN (ResNet50) Training

Learning rate	Training RMSE	Dev. RMSE
0.001	31.8	32.4
0.005	20.3	22.9
0.0001	29.0	32.5
0.00001	39.7	42.7



Final Comparison Across Four Models

Model	Training RMSE	Dev. RMSE	Test RMSE	Average time to run	Hardware (GPU)
DeepLabCut	**1.0097**	4.4437	3.1686	24hr train 3 min test	GeForce GTX 2080
DeepLabCut (ResNet101)	0.6251	4.9553	2.9650	36hr train 3 min test	GeForce GTX 2080
Faster R-CNN	77.3	82.9	81.3	0.8 hrs training 6 min. testing	GeForce GTX 1050
Faster R-CNN (ResNet50)	20.3	22.9	19.9	1.9 hrs training 20 min. testing	GeForce GTX 1050

Final Comparison Across Four Models

- The best model performs at: 2.9650 pixels RMSE. We argue that this performance is more than satisfactory, as human variability in this task is reported to have an RMSE of 2.7 pixels[1]; further, a mouse's paw takes up an area of 1000-2000 pixels^2 at this resolution.
- DLC and DLC 101 were the only model to achieve our satisficing metric (RMSE < 10 pixels).
- Faster R-CNN had a much faster train and run time, even using a smaller GPU, than DeepLabCut, however the RMSE was much higher. Faster R-CNN with ResNet50 cut the RMSE in half while maintaining relatively low computing time.

Conclusions & Future Directions

While DeepLabCut still had the greatest accuracy (lowest RMSE) of all the models tested, we were able to achieve fairly high accuracy using a modified version of Faster R-CNN (using ResNet50 instead of the usual convolutional layers). This new model was significantly faster both at training and testing new data compared to DeepLabCut, even while using a much smaller GPU on workstation computer. We believe further modifying this new algorithm (adding more layers, training data, and training time) could ultimately lead to similar performance as DeepLabCut while requiring far less computational resources. This would be a significant improvement for the field.