DeepLabCut workshop
Session 2: training & network selection

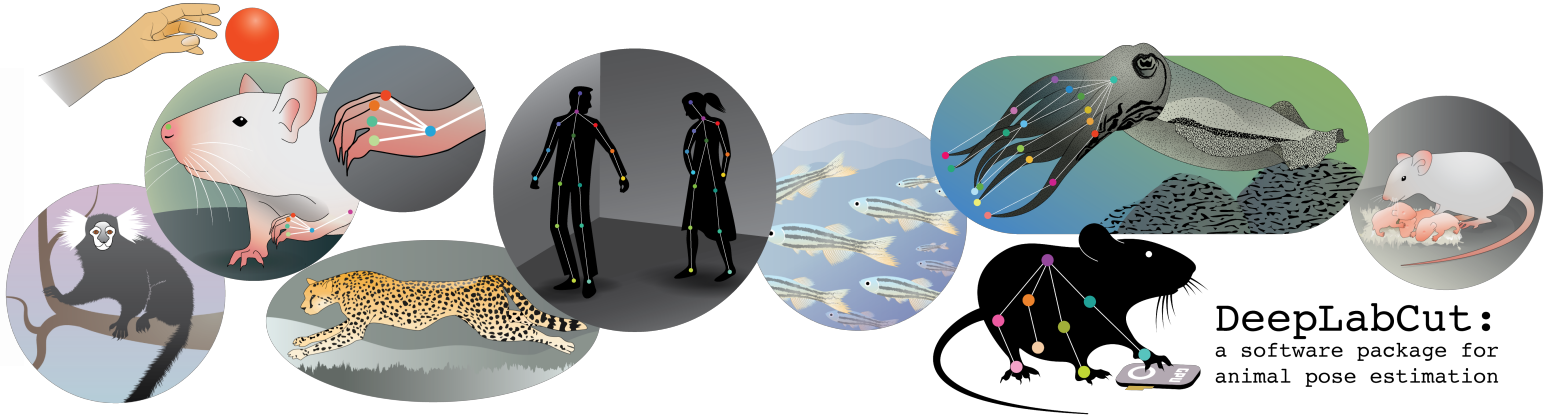
16th and 17th of January 2020

Rowland Institute, Harvard University

Mackenzie Mathis
Alexander Mathis
Jessy Lauer

Chan Zuckerberg Initiative

HARVARD UNIVERSITY



“Software 2.0” – integration of annotation, training and inference

Train DNN

Create a project, extract frames, + GUIs to label your data

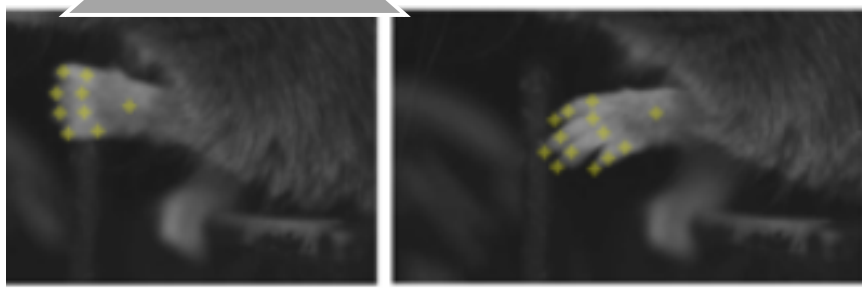
Select + Train your deep neural network

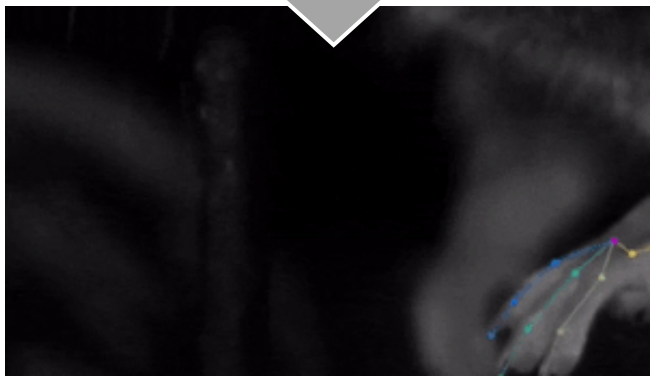
Evaluate network performance

(active learning + GUIs if improvement needed)

Run inference on new videos, create labeled videos, + plot your results!

refine?





Mathis et al. Nature Neurosci, 2018
Nath* & Mathis* et al. Nature Protocols 2019

What network & augmentation method?

Network architecture:

- Network backbones: MobileNets V2, *ResNets*
- Scale (level of deconvolution)
- Differences: **Inference & training speed, resolution**

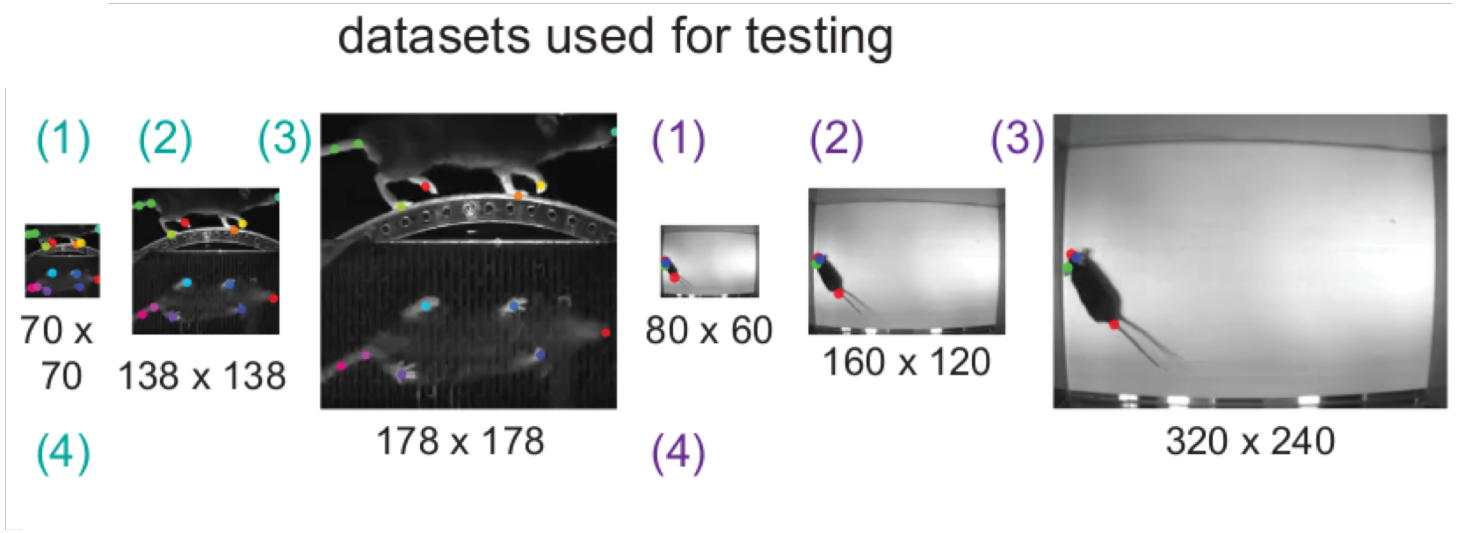
Pre-training (Weights):

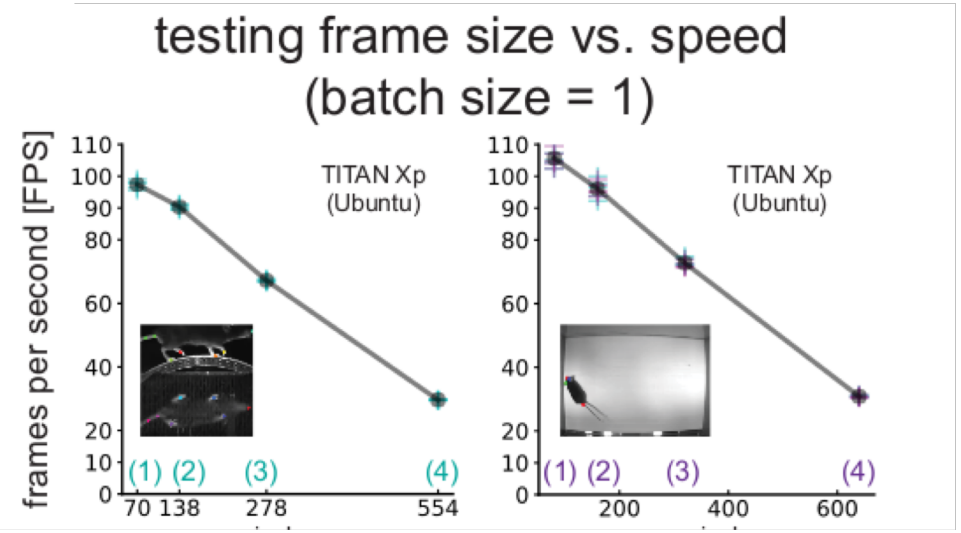
- *ImageNet pretrained (the generalist)*
- MPII pose pretrained
- DeepLabCut model zoo….

Training method:

- Learning rate & optimizer (Adam/SGD, batch_size, …)
- Augmentation transformations (rotation, scaling, …)

Inference speed strongly depends on image size

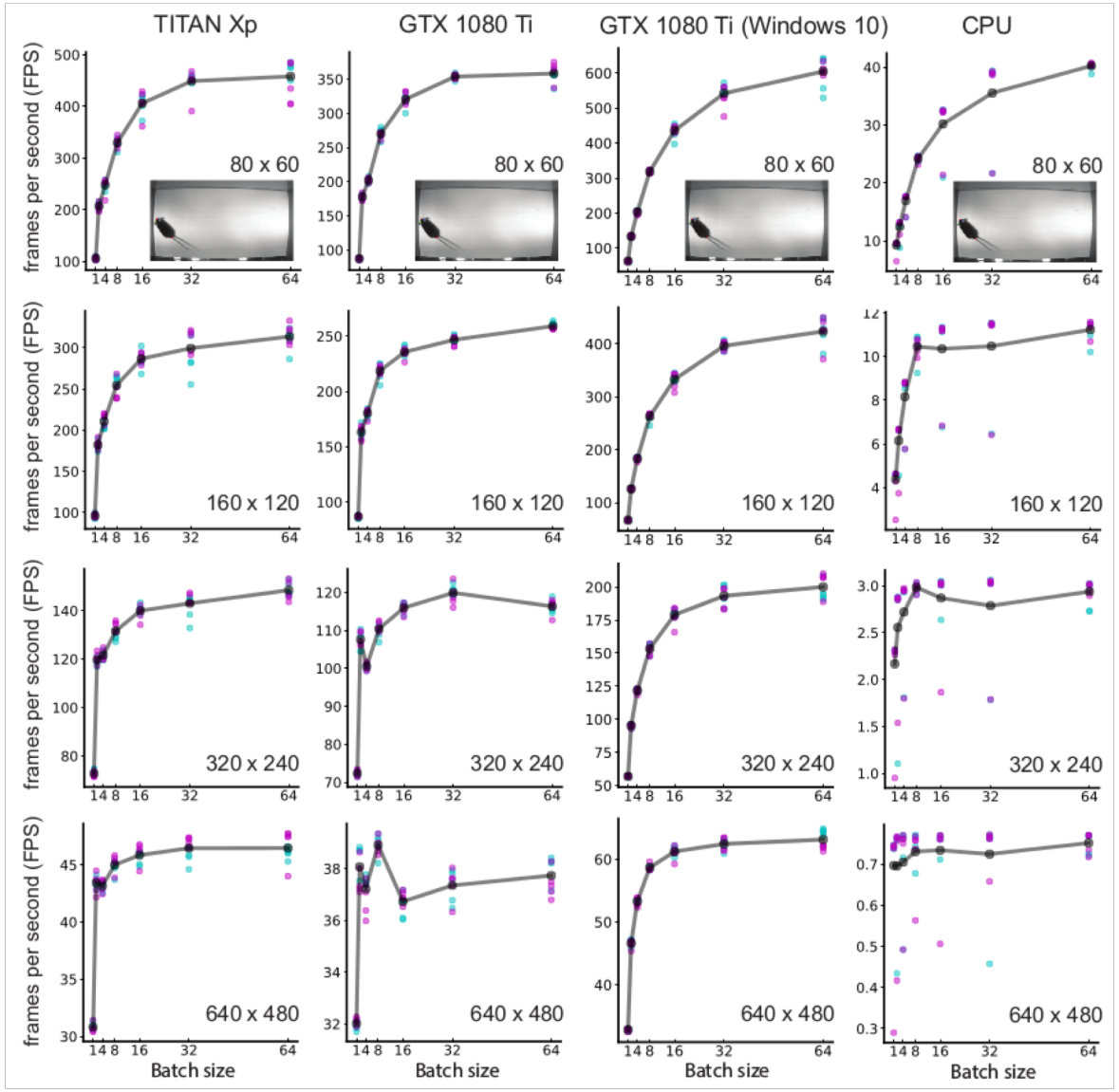




Mathis, Warren BioRxiv 2018

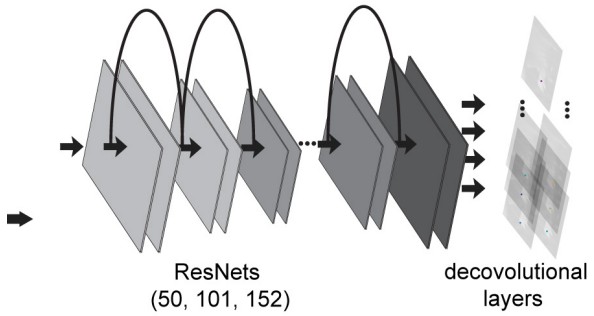
Inference speed depends on hardware!

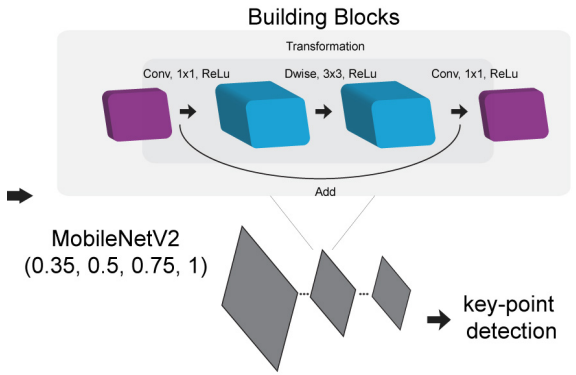
- Get a GPU!

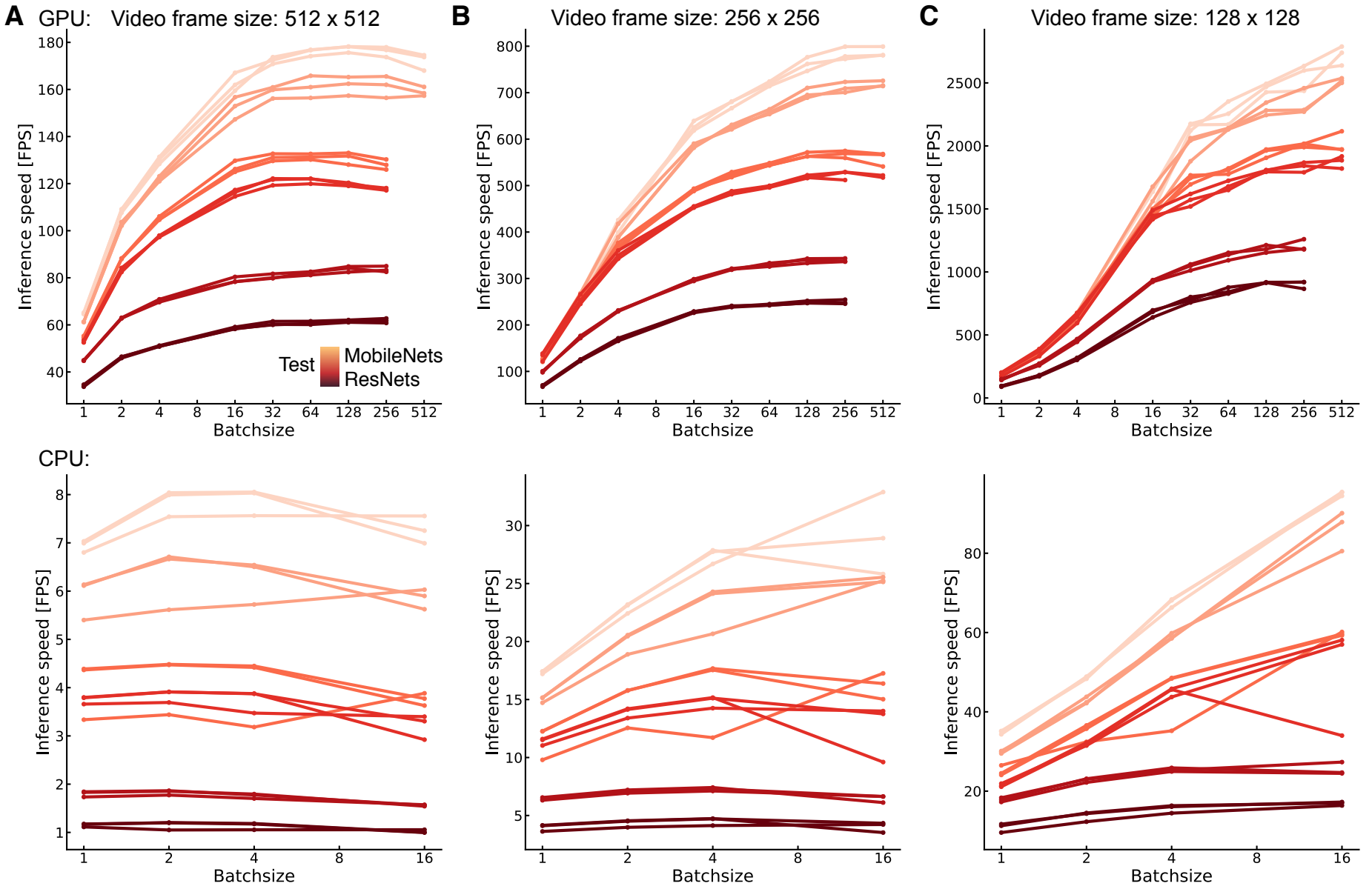


Mathis, Warren BioRxiv 2018

MobileNets are faster than ResNets

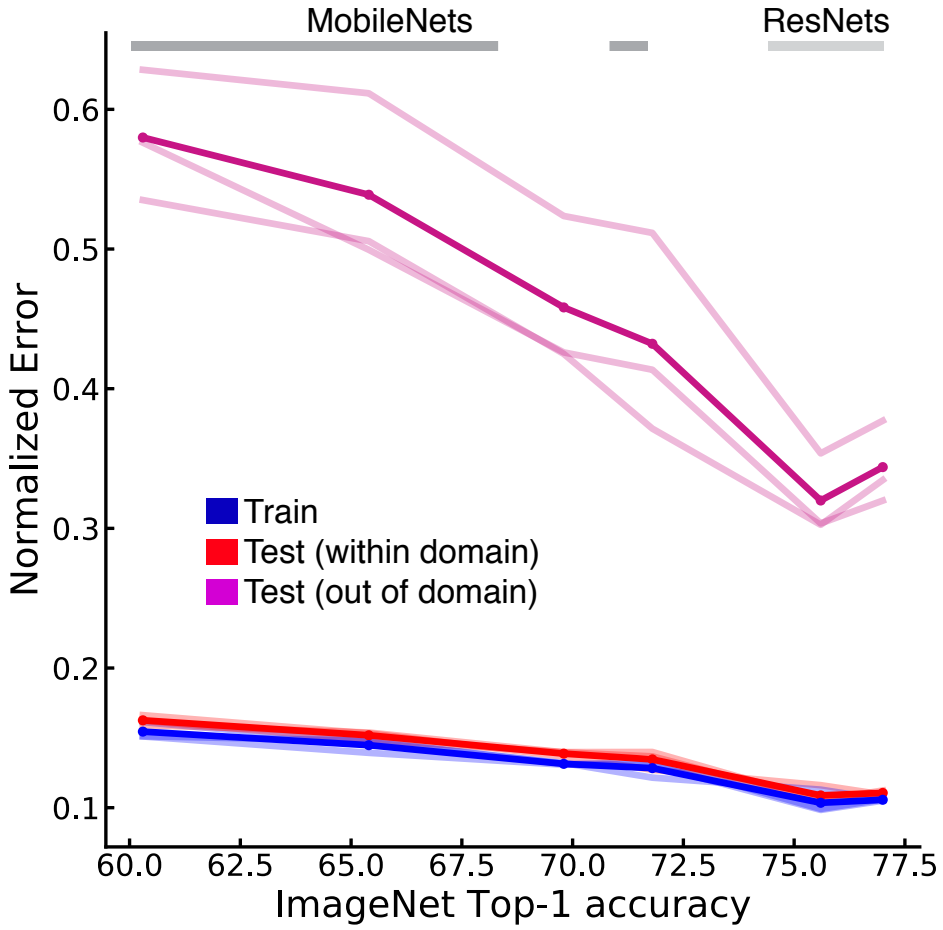






Mathis et al. arXiv:1909.11229 2019

ImageNet accuracy **correlates** with performance



Within domain: horses as contained in data set
Out of domain: different horses & background

Mathis et al. arXiv:1909.11229 2019

How to pick the network type?

- **Performance vs. speed trade-off.**
  Higher robustness requirements: ResNet 50 (or 101),
  High speed requirements: MobileNetsV2 0.35

- Try to down sample/crop videos (the smaller videos, the faster irrespective of model)

- How to choose in DLC?
  - By default uses “default_net_type” from config file of project!
  - Otherwise during **training set creation** you can pass your choice, e.g.:

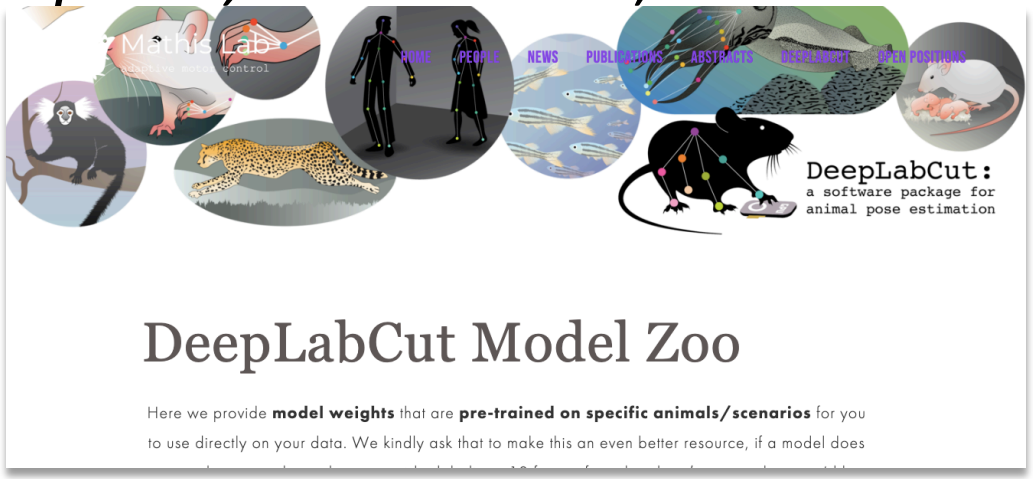
    deeplabcut.create_training_dataset(config, net_type=‘mobilenet_v2_1.0’)

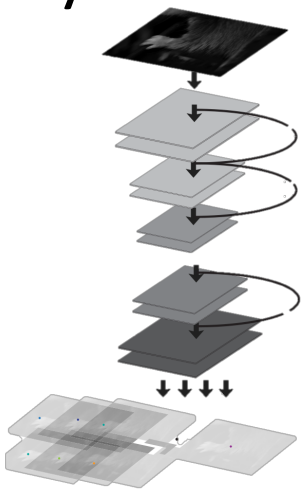
    Pick from *resnet_50, resnet_101, resnet_152, mobilenet_v2_1.0, mobilenet_v2_0.75, mobilenet_v2_0.5, mobilenet_v2_0.35*
    Soon: EfficientNet

  - Alternative: change **net_type** in **config_pose.yaml** (for both train & test folders!)

What type of pre-training (weights)?

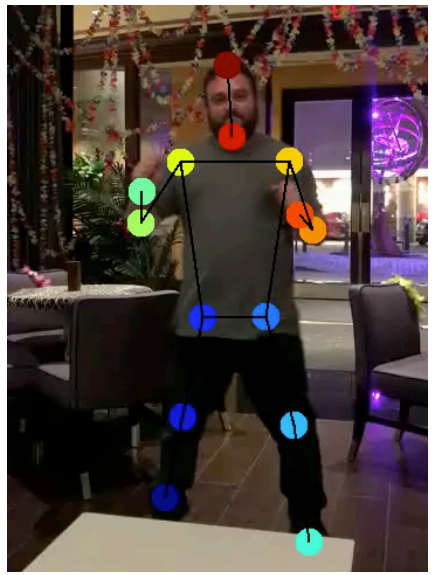
- Why **pre-training**? (*faster training, less data required, more robust*; see Mathis et al. 2019 arxiv)
- Currently: “generic” ImageNet pre-training
- **Use weights from** previous training/others!
- Soon: *pre-trained models for typical lab task*
- Already available: MPII pose pretrained





Never re-trained >

DeepLabCut includes the ability to use ResNet-101 pre-trained on MPII
26K images, 40K humans

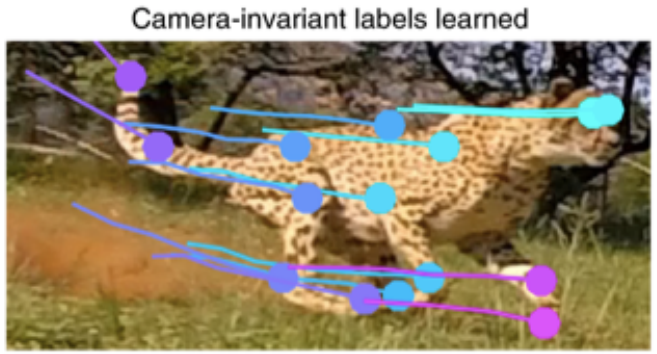


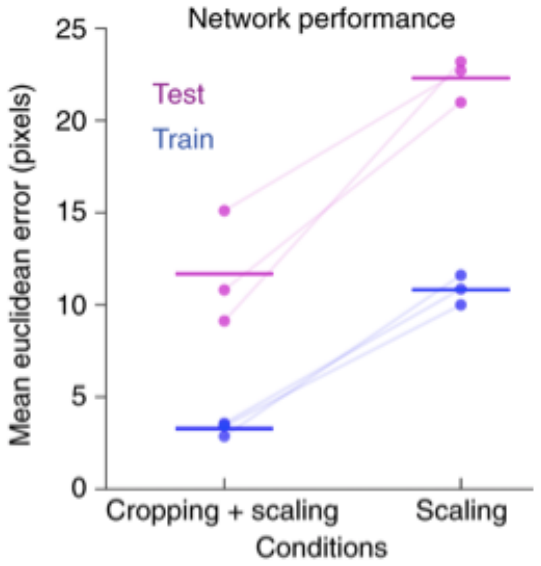
For loading different weighs:
Set path in
config_pose.yaml file!

Code: deeplabcut.create_pretrained_human_project(…)

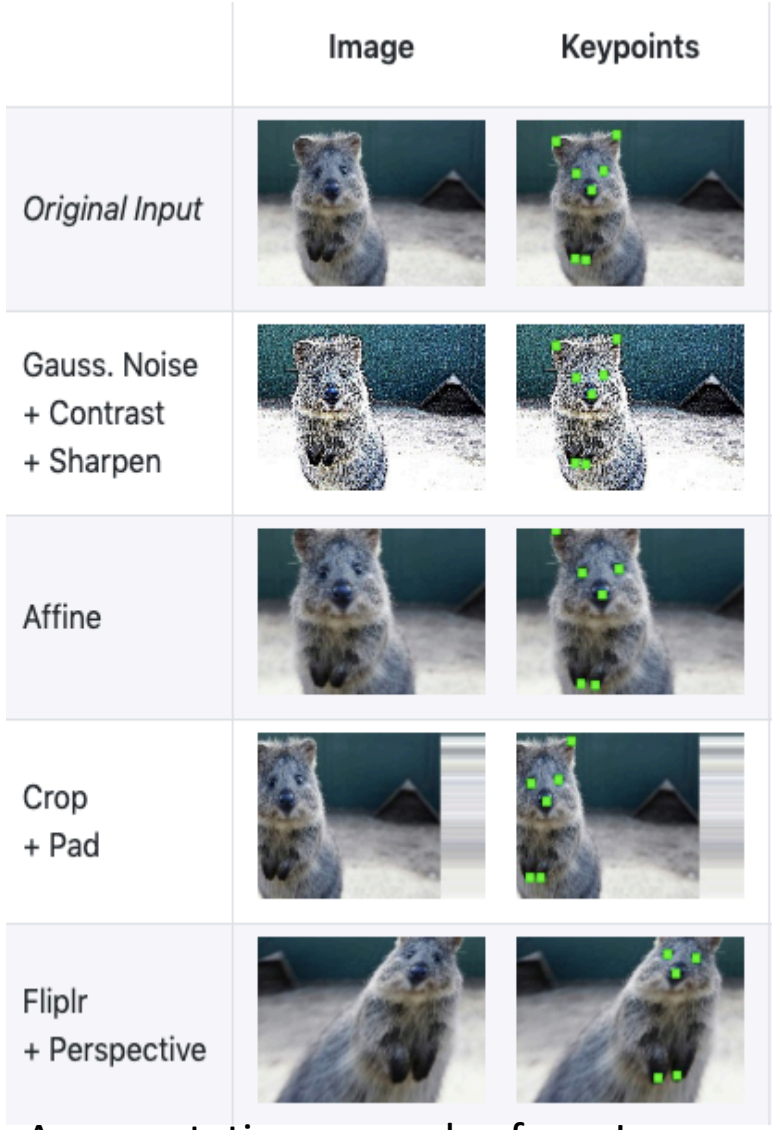
What type of training/augmentation?

- Batch size (the larger the better; but large memory demand)
- Learning schedule Adam, SGD; rates are good
- **Augmentation**





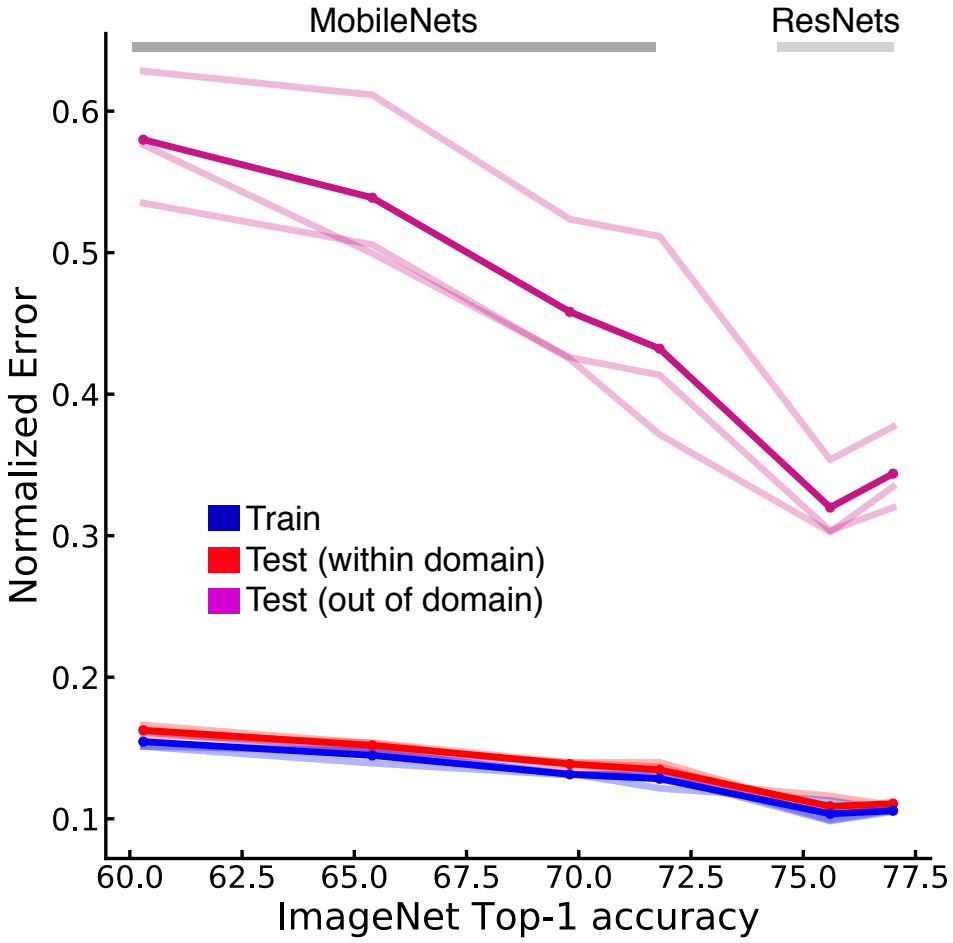
Nath* & Mathis* et al. Nature Protocols 2019



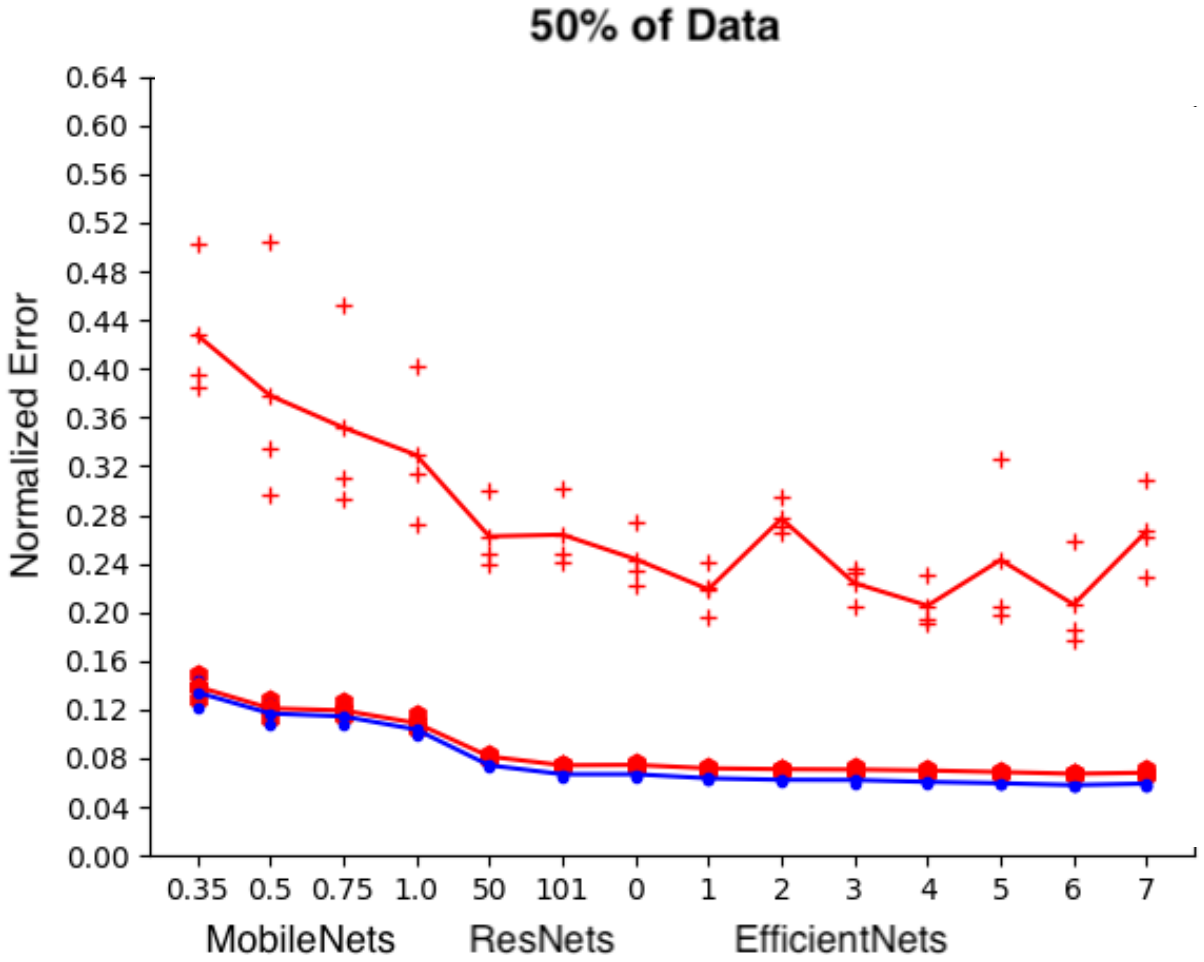
Augmentation examples from Imgaug

The impact of augmentation methods

default loader
(SGD, scaling + cropping)



imgaug loader
(Adam, scaling + cropping, covering, brightness,…)



Mathis et al. arXiv:1909.11229 2019

How to set augmentation methods in DLC?

By default uses “default_augmenter” in config file of project!

Alternatively, during **training set creation** you can pass your choice, e.g.:
deeplabcut.create_training_dataset(config, augmenter_type=‘imgaug’)

Pick from *default, imgaug, tensorpack & deterministic*

You can also change **dataset_type** in **config_pose.yaml** as well as other variables of interest (i.e. global_scale, multi_step, …).

See BOX 2 in Nath*, Mathis* et al for details

Box 2 | Parameters of interest in the network configuration file, *pose_cfg.yaml*

Please note, there are more parameters that typically never need to be adjusted; they can be found in the default *pose_cfg.yaml* file at https://github.com/AlexEMG/DeepLabCut/blob/master/deeplabcut/pose_cfg.yaml.

- `display_iters`: An integer value representing the period with which the loss is displayed (and stored in *log.csv*).
- `save_iters`: An integer value representing the period with which the checkpoints (weights of the network) are saved. Each snapshot has >90 MB, so not too many should be stored.
- `init_weights`: The weights used for training. Default: `<DeepLabCut_path>/Pose_Estimation_Tensorflow/pretrained/resnet_v1_50.ckpt`. For ResNet-50 or 101, -- this will be automatically created. The weights can also be changed to restart from a particular snapshot if training is interrupted, e.g., `<full path>-snapshot-5000` (with no file-type ending added). This would re-start training from the loaded weights (i.e., after 5,000 training iterations, the counter starts from 0).
- `multi_step`: These are the learning rates and number of training iterations to perform at the specified rate. If the users want to stop before 1 million, they can delete a row and/or change the last value to be the desired stop point.
- `max_input_size`: All images larger with size width × height > `max_input_size*max_input_size` are not used in training. The default is 1500 to prevent crashing with an out-of-memory exception for very large images. This will depend on your GPU memory capacity. However, we suggest reducing the pixel size as much as possible; see Mathis and Warren[27].

The following parameters allow one to change the resolution:

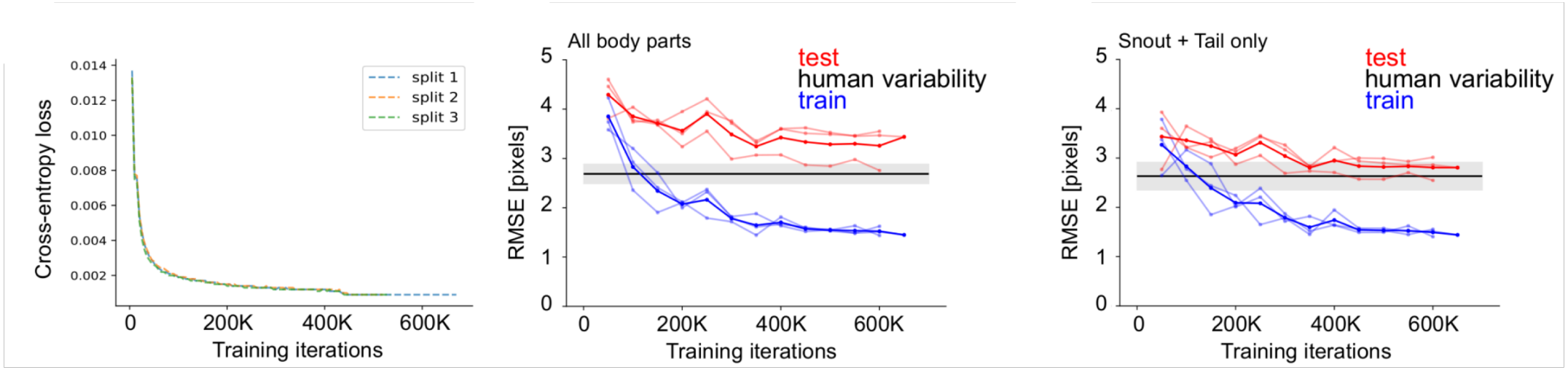
- `global_scale`: All images in the dataset will be rescaled by the following scaling factor to be processed by the convolutional neural network. You can select the optimal scale by cross-validation (see discussion in Mathis et al.[12]). Default is 0.8.
- `pos_dist_thresh`: All locations within this distance threshold (measured in pixels) are considered positive training samples for detection (see discussion in Mathis et al.[12]). Default is 17.

The following parameters modulate the data augmentation. During training, each image will be randomly rescaled within the range `[scale_jitter_lo, scale_jitter_up]` to augment training:

- `scale_jitter_lo`: 0.5 (default).
- `scale_jitter_up`: 1.5 (default).
- `mirror`: If the training dataset is symmetric around the vertical axis, this Boolean variable allows random augmentation. Default is `False`.
- `cropping`: Allows automatic cropping of images during training. Default is `True`.
- `cropratio`: Fraction of training samples that are cropped. Default is 40%.
- `minsize`, `leftwidth`, `rightwidth`, `bottomheight`, `topheight`: These define dimensions and limits for auto-cropping.

Nath* & Mathis* et al. Nature Protocols 2019

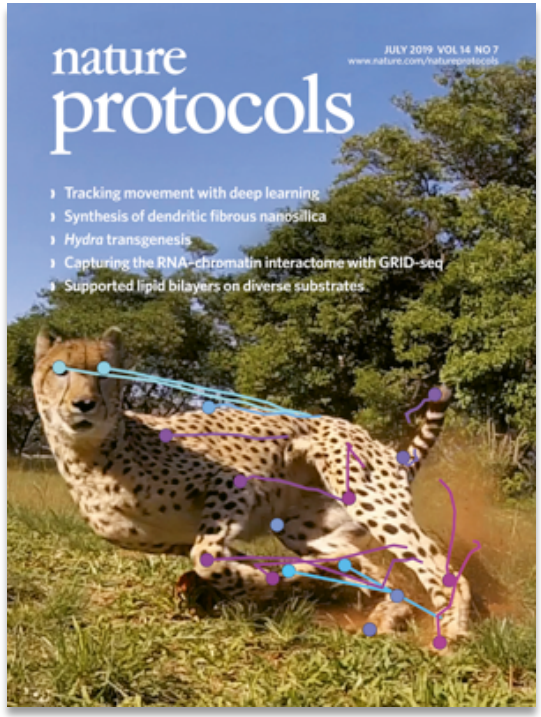
Training considerations



Mathis et al. Nature Neurosci, 2018

- Cross entropy loss minimized (measures difference in probabilities between epsilon disk around ground-truth and predicted scoremap)
- Training speed (mostly) depends on frame size! (and GPU, CPU type)
- Train until the “loss flattens”
- Store as many **snapshots** as necessary (intermediate weight steps; consider early stopping)

Tutorial: create your own training dataset + start training



nature protocols

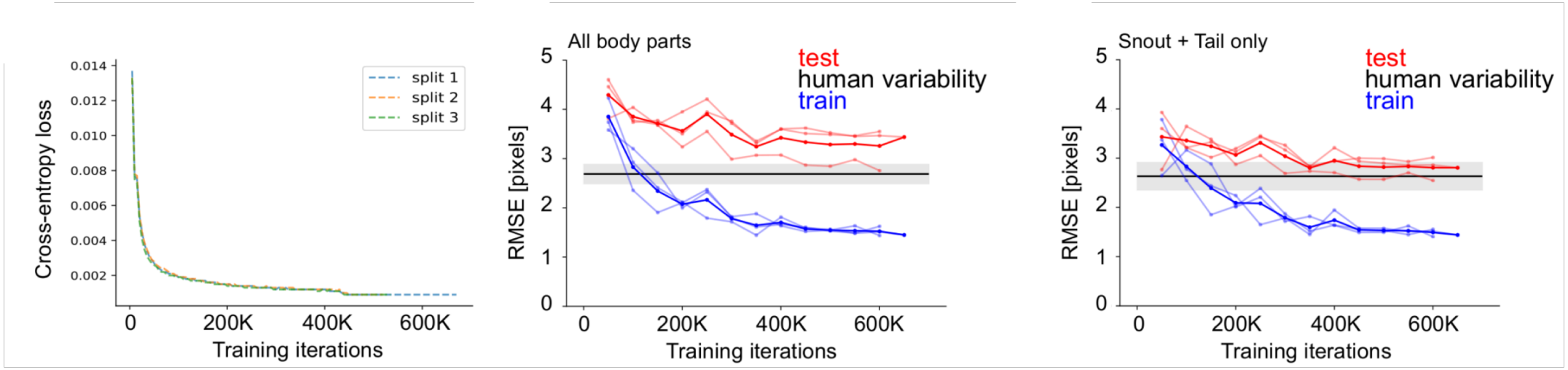
PROTOCOL

https://doi.org/10.1038/s41596-019-0176-0

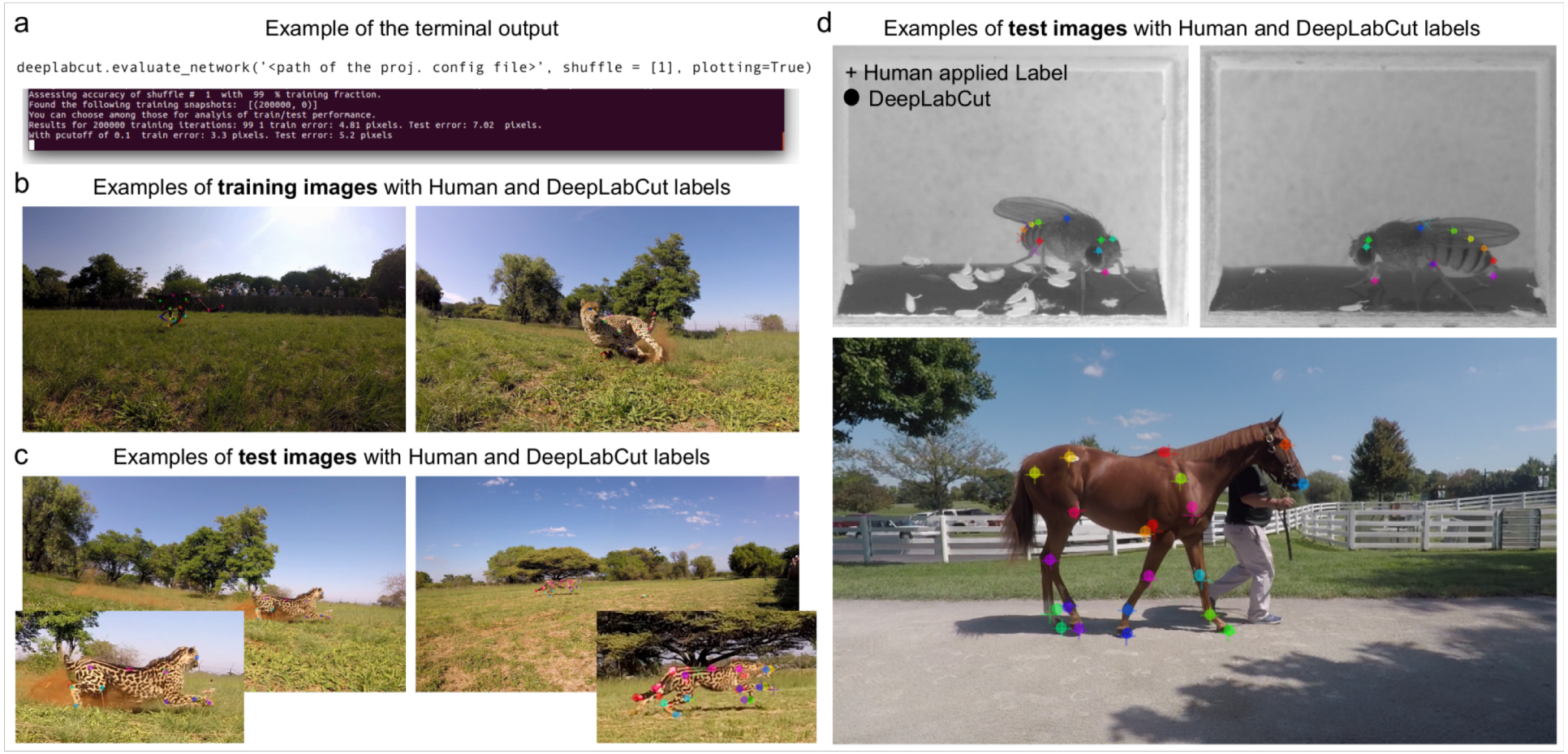
Using DeepLabCut for 3D markerless pose estimation across species and behaviors

Tanmay Nath[1,5], Alexander Mathis[1,2,5], An Chi Chen[3], Amir Patel[3], Matthias Bethge[4] and Mackenzie Weygandt Mathis[1*]

Evaluation: *Compare test RMSE and ideally your labeling error!*



Evaluation >> look at test images!



Nath* & Mathis* et al. Nature Protocols 2019