SuperAnimal pretrained pose estimation models for behavioral analysis

Shaokai Ye[1], Anastasiia Filippova[1], Jessy Lauer[1], Maxime Vidal[1], Steffen Schneider[1], Tian Qiu[1], Alexander Mathis*[1], & Mackenzie Weygandt Mathis* ✉[1]

[1]École Polytechnique Fédérale de Lausanne (EPFL), Brain Mind Institute & Neuro-X Institute. Geneva, Switzerland

Quantification of behavior is critical in applications ranging from neuroscience, veterinary medicine and animal conservation efforts. A common key step for behavioral analysis is first extracting relevant keypoints on animals, known as pose estimation. However, reliable inference of poses currently requires domain knowledge and manual labeling effort to build supervised models. We present a series of technical innovations that enable a new method, collectively called SuperAnimal, to develop and deploy deep learning models that require zero additional human labels and model training. SuperAnimal allows video inference on over 45 species with only two global classes of animal pose models. If the models need fine-tuning, we show SuperAnimal models are 10× more data efficient and outperform prior transfer-learning-based approaches. Moreover, we provide an unsupervised video-adaptation method to refine keypoints in videos. We illustrate the utility of our model in behavioral classification in mice and gait analysis in horses. Collectively, this presents a data-efficient solution for animal pose estimation for downstream behavioral analysis.

Introduction

Measuring and modeling behavior is an important step in many clinical, biotechnological, and scientific quests (1–6). A key part of many behavioral analysis pipelines is animal pose estimation, yet this requires domain knowledge and labeling efforts to obtain reliable pose models (2, 3, 7, 8). Open-source pose estimation software, such as DeepLabCut (9, 10) and other tools (11–14) also reviewed in (7), have gained popularity in the research community interested in understanding animal behavior. Compared to commercial solutions constrained to fixed cage and camera settings (15), DeepLabCut offers flexibility to train customized pose models of various animals in diverse settings. Notably, it requires few human-labeled images (around 100–800) to train a typical lab animal pose estimator that matches human level accuracy (9, 10) due to its transfer learning abilities (9, 16).

However, regardless of the data efficiency of current solutions, their flexibility still comes with the cost of requiring users to label if they want to define keypoints and then train deep neural networks, an effort that is often duplicated across labs. A solution is to build generalized, foundation-like models (17), for common model organisms across labs and in-the-wild settings (discussed in (7)). Such models, once trained, can be used across labs and settings without further training and/or requiring little fine-tuning.

To provide the research community with easy access to such high performance models we present a new panoptic paradigm – which we call the SuperAnimal method – for building pre-trained pose models, and the ability to perform video adaptation across many species, environments and video sizes (e.g., fine-tune them if needed; Figure 1a). SuperAnimal combines diverse datasets into two broad unified pose models that cover over 45 species of mammals with 27–39 keypoints. In brief, our new approach allows for merging and training diverse, differently labeled datasets. We developed an optimal keypoint matching algorithm to automatically align out-of-distribution datasets with our models. Then, at inference time, to minimize domain shifts, we developed a spatial-pyramid search method to account for changes in animal size (or use a top-down detector). We also developed a rapid, unsupervised video-adaptation method that uses pseudo-labeling to minimize temporal jitter in videos and allows users to fine-tune videos without any data labeling.

We developed models based on state-of-the-art convolutional neural networks (CNNs), such as HRNet (18) and DLCRNet (10), and introduce AnimalTokenPose that uses transformers (19–21). We show that the resulting models have excellent zero-shot performance (i.e., with no additional training, tested on new data), and our approach outperforms ImageNet-pretraining on five benchmarks. If users then want to use these new weights for fine-tuning, we show they are 10X more data efficient, and our video adaptation method allows for smooth, refined videos that can be used in behavioral analysis pipelines.

Results

The SuperAnimal method comprises generalized data conversion, training with keypoint gradient masking and memory-replay, a keypoint matching algorithm, and the ability to fine-tune with video adaptation (Figure 1a), which will be explained below. Collectively, this is a formulation that treats diverse pose datasets as if they collectively formed one single super-pose template. This effectively allowed us to overcome a major challenge with combining datasets that are

✉mackenzie@post.harvard.edu | *Jointly supervised this work.

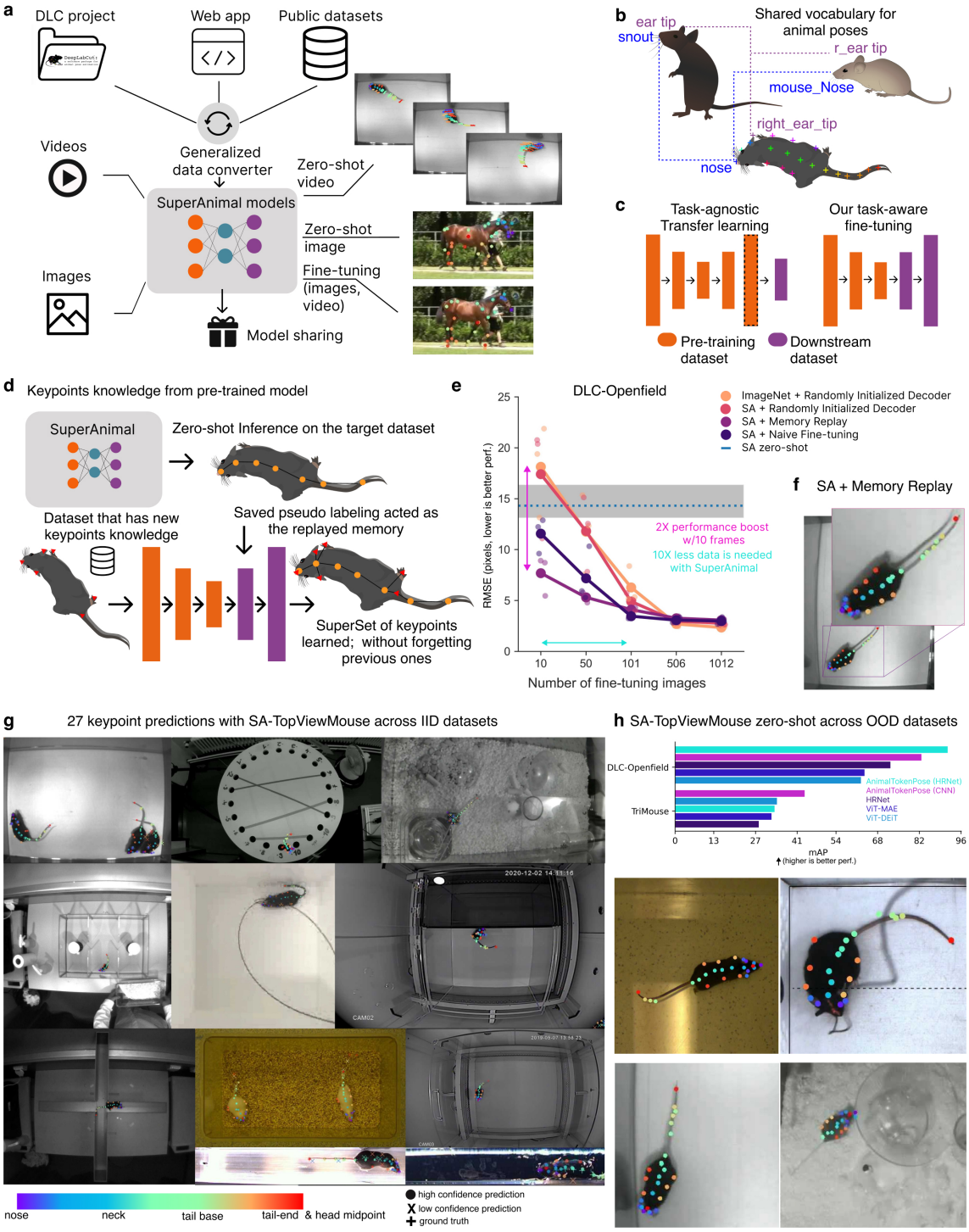

Figure 1. The DeepLabCut Model Zoo, the SuperAnimal method, and SuperAnimal-TopViewMouse model performance. **a**: The website can collect data shared by the research community; SuperAnimal models are trained, and can be used for inference on novel images and videos with or without further training. **b**: The panoptic animal pose estimation approach unifies the vocabulary of pose data across labs, such that each individual dataset is a subset of a super-set keypoint space, independently of its naming. **c**: For canonical, task-agnostic transfer learning, the encoder learns universal visual features from ImageNet, and a randomly initialized decoder is used to learn the pose from the downstream dataset. For task-aware fine-tuning, both encoder and decoder learn task-related visual-pose features in the pre-training datasets and the decoder is fine-tuned to update pose priors in downstream datasets. Crucially, the network has pose-estimation specific weights. **d**: Memory-replay combines the strengths of SuperAnimal models' zero-shot inference, data combination strategy, and leveraging labeled data for fine-tuning (if needed). It achieves better data efficiency. **e**: Data efficiency of baseline (ImageNet) and various SuperAnimal fine-tuning methods on the DLC-Openfield OOD dataset. Grey shadow represents minimum, maximum and blue dash is the mean for zero-shot performance across three shuffles. Large, connected dots represent mean results across three shuffles and smaller dots represent results for individual shuffles. **f**: Using memory replay avoids catastrophic forgetting; here all keypoints are predicted. **g**: Top: SuperAnimal-TopViewMouse qualitative results on the within distribution test images (IID). They were randomly selected based on visibility of the keypoints within the figure (but not on performance). Full keypoint color and mapping is available in Extended Data Figure S1). **h**: Top: Transformers also perform well on OOD tasks. Bottom: visualization of model performance on OOD images using DLCRNet.

not identically labeled across labs or datasets, as it is often the case even for the same species (Figure 1b, c and Extended Data Figure S1a). SuperAnimal enables multi-dataset training, allowing the model to receive richer learning signals (Figure 1c, d), resulting in the model having "pose priors" (whereas ImageNet pre-training, the current state-of-the-art in animal pose (10, 16, 22) has no pose-specific features).

In order to demonstrate the strength of our SuperAnimal method, we present two datasets that cover over 45 species: TopViewMouse-5K and Quadruped-40K, which are built from over 45,000 images sourced from diverse laboratory settings and in-the-wild data (Extended Data Figure S1a, b). First, we used a new generalized data converter (see Methods) to unify the annotation space of those datasets and named the first dataset TopViewMouse-5K (as it contains approximately 5k images). Specifically, we merged 13 overhead-camera view-point lab mice datasets from across the research community (9, 10, 15, 23, 24) (see Methods) and from our own experiments (Figure 1e, h). Similarly, we collected side-view quadruped datasets (16, 25–30), including a new annotated rodent dataset with images sourced from iNaturalist (see Methods), to form Quadruped-40K (Extended Data Figure S1b). However, below we leave out all benchmark datasets in order to show performance of the model on unseen data. Our released weights are trained on all available data described above (Extended Data Figure S1b).

The SuperAnimal method is a series of solutions that show excellent generalization on unseen images and videos (Figure 1a-e, Tables S1, S2). For training, we developed keypoint gradient masking (Extended Data Figure S1c, d) to train neural networks across disjoint datasets without penalizing "missing" ground truth data from the superset of keypoints (Figure 1b). To note, typically transfer learning involves fine-tuning a pre-trained encoder, but using a randomly initialized decoder in the downstream dataset (9). In contrast, we fine-tuned the pre-trained encoder and decoder (Figure 1c). Additionally, inspired by the excellent zero-shot inference of pre-trained models (31) and continual learning (32), we developed a tailored fine-tuning approach that combines zero-shot inference and few-shot learning, which we call "memory-replay" (Figure 1d). This allowed us to combine datasets and extract all keypoints independently of how many were labeled in any one dataset (Figure 1d). We also developed a keypoint matching algorithm (Methods and Extended Data Figure S2a, b) to help minimize the mismatch caused by annotator bias in the ground truth datasets (see Suppl. Note).

SuperAnimal models.

SuperAnimal-TopViewMouse

To evaluate our models we tested performance "within distribution" also known as "independent and identically distributed" (IID), and on images considered "out-of-distribution" (OOD). IID images are similar in appearance, but not identical to those used in training. OOD data stems from images that were never included in training and differ from the training data (33).

One use case of the new models we provide is to run video inference without any additional training, called "zero-shot". Therefore, to test performance, we built a SuperAnimal-TopViewMouse model that did not contain data from the DLC-Openfield (9) or related TriMouse dataset (9, 10). Collectively, we find that the SuperAnimal methods are critical to avoid catastrophic forgetting (see Suppl. Video 1), and show excellent zero-shot performance (Figure 1e) on the DLC-Openfield benchmark. SuperAnimal-TopViewMouse performed well within distribution (IID) and OOD across diverse camera and cage settings (Figure 1g, h).

Zero-shot SuperAnimal-TopViewMouse shows an RMSE error of 14.31 pixels on the DLC-Openfield dataset, where the average mouse's nose width is approximately 10 pixels (9) (Figure 1e, f). Namely, we found that without any labeling we could outperform ImageNet-based transfer learning (Figure 1e; mixed-effect model; in the low-data regime d=3.06 [1.99, 4.13]; p<.0001; see Tables S4- S12). A user would need to label 30–700 frames to achieve the same performance as our SuperAnimal pre-trained models without any need to data labeling (see Figure 1e where zero-shot performance intersects with either SuperAnimal fine-tuning or ImageNet fine-tuning; Extended Data S3b). This also effectively removes the time needed for labeling and training models (which typically comprises several hours (34)). Note that the performance of zero-shot inference is likely underestimated by annotator bias (see Suppl. Note). We also show our method is not limited to convolutional neural networks (CNNs), but can be used with transformers (Figure 1h, and see Methods), which, we find, have especially good zero-shot performance and can outperform CNNs.

Depending on the zero-shot performance, one might want to still label their own frames to increase performance. We demonstrate that our SuperAnimal weights are better than using ImageNet weights. When we fine-tuned with labeled images and evaluated performance (called few-shot), SuperAnimal-TopViewMouse pre-trained models significantly outperformed ImageNet pre-trained models by a 10× data efficiency factor and large margin of performance gain (Figure 1e, Suppl. Tables). We found that both keypoint gradient masking and memory-replay boosted performance (Figure 1e-h, Extended Data Figure S1, Suppl. Videos 1 & 2), as keypoints that were not annotated would otherwise cause a false penalty and therefore degrade performance (see Methods). This combined encoder-decoder fine-tuning often improved efficiency over naive fine-tuning (Figure 1e, f).

Using this optimal SuperAnimal setting, if the model is fine-tuned with only 10 images (randomly selected) on DLC-Openfield, the SuperAnimal pre-trained model obtained an RMSE of 7.68 pixels, whereas ImageNet pre-training was 18.14 pixels. We show that the baseline ImageNet pre-trained model required 101 (randomly selected) images to reach a similar performance (6.28) to SuperAnimal pre-trained models (Figure 1e). Therefore, we outperformed DeepLabCut-ResNet-50 (i.e., ImageNet baseline) by over 2X in the low data regime (i.e., with 10 frames of labeling), and we can

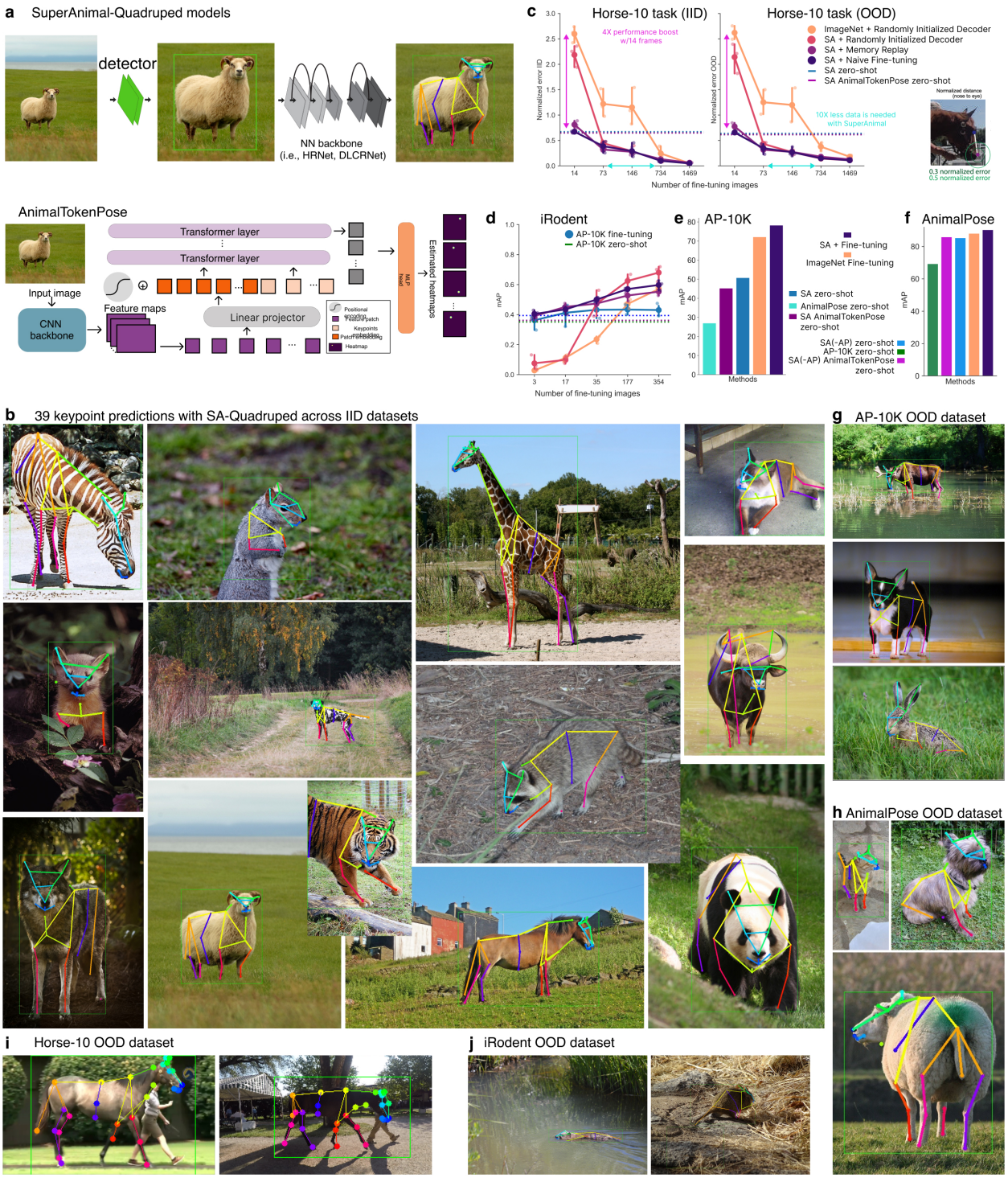

Figure 2. SuperAnimal-Quadruped a: Conceptual diagram to demonstrate top-down detection and CNN-based (here we used HRNet) pose estimation, and the AnimalTokenPose architecture (see Methods and diagram at Extended Data Figure S4a-d). **b**: Qualitative performance with SuperAnimal-Quadruped (HRNet) performance, randomly selected based on visibility of the keypoints within the figure (but not on performance). A likelihood cutoff of 0.65 was applied for keypoint visualization. Full keypoint color and mapping is available in Extended Data Figure S1). **c**: Performance on the OOD Horse-10 dataset, using the official IID and OOD splits, reported as a normalized error from eye to nose, see inset adopted from (16). Results are with HRNet, unless noted. **d**: Performance on the OOD iRodent dataset, reported as a normalized bounding-box error. Colors as in **c** with additions noted in **c**. **e**: Performance on the OOD AP-10K dataset, reported as mAP, legend to the right of **f**. **f**: Performance on AnimalPose (AP), where AP was removed from the SuperAnimal-Quadruped base model, reported as mAP. **f, g, h**: Qualitative performance on various OOD datasets, as noted and selected as in **b**. A likelihood cutoff of 0.6 was applied for keypoint visualization in all datasets except for iRodent, where 0.5 was used.

achieve the same performance as DeepLabCut-ImageNet weights with 10X less data.

One important point is that the SuperAnimal pre-trained model is now imbued with a "pose prior". Historically,

the fine-tuning strategies assumed no "task priors" in the pretrained model, a paradigm adopted from previous task-agnostic transfer learning (35). Yet, here we show that naively fine-tuning on datasets that do not have the full superset of points might cause catastrophic forgetting (Figure 1g, see Methods and Suppl. Video 2). Specifically, if we fine-tuned with the four keypoint dataset from DLC-Openfield, the model would forget the full 27 keypoints. We show this is prevented by our memory-replay method. This method can also improve the fine-tuning performance (e.g., by 3.9 pixels in the DLC-Openfield dataset at N=10 training images; Cohen's d=-1.7494, Figure 1f).

SuperAnimal-Quadruped

We also developed a SuperAnimal-Quadruped model using both a top-down CNN (HRNet-W32) (10, 13, 37), which recently haven been shown to be excellent in crowded animal scenes (38) and our modified transformer (called AnimalTokenPose, see Figure Figure 2a). IID performance was quantitatively and qualitatively excellent (mean Average Precision (mAP) = 86.1 for SuperAnimal-HRNet-W32, Figure 2b).

We then tested its performance on four OOD benchmarks that had various metrics: Horse-10 (16) which reports the normalized error (normalized by the animals size, see inset in Figure 2b), a new dataset we present called iRodent, AP-10k (30) and lastly we consider AnimalPose (27) all which we report mAP. To probe the model performance we did not include Horse-10, iRodent, or AP-10K in the base model we call SuperAnimal-Quadruped here.

Horse-10 is a benchmark challenge that tests OOD robustness. We evaluated on the official splits and show zero-shot performance is similar across IID and OOD splits, which are both OOD to our model (Figure 2c). Next we show that with minimal fine-tuning using our SuperAnimal method we can match ImageNet-based transfer learning with 10X less data (Figure 2c, i.e, fine-tune with 73 frames vs 734 for the same level of performance).

iRodent is a challenging new dataset comprising a diverse set of images of rodents, yet with our top-down HRNet CNN we can achieve excellent zero-shot performance (Figure 2d, h), although there is still a gap to close in future work, as even fine-tuning on over 350 frames only slightly improves the mAP.

Next, we tested our model on the AP-10K benchmark. Here we show that when fine-tuned, our SuperAnimal-Quadruped HRNet outperforms ImageNet-based fine-tuning (Figure 2e), and our zero-shot performance was better than a model that performs well on the AnimalPose benchmark (27). AnimalPose is a benchmark dataset of dog, cat, cow, horse, sheep with 20 keypoints (27).

Lastly, we benchmarked our SuperAnimal model directly on AnimalPose. For testing AnimalPose we made a SuperAnimal model variant that dropped AnimalPose during training for both HRNet and AnimalTokenPose backbones (Figure 2f). Notably, we find that our zero-shot performance is almost on par with fully supervised models (Figure 2e), and beats zero-shot of a only AP-10K dataset trained model (39) (Figure 2f).

Collectively, the SuperAnimal method presents an efficient way to achieve strong zero-shot performance and also provides better starting weights for fine-tuning (vs. ImageNet-fine tuning). Also note that both sets of SuperAnimal models—TopViewMouse and Quadruped—learned to predict the union of all keypoints defined in multiple datasets even if no single dataset had defined all of these keypoints. Of course, despite strong generalization, there can still be failures (Extended Data Figure S2c).

Unsupervised Video Adaptation.

Independent of the use case (i.e., zero-shot or few-shot fine-tuning), to optimize performance on unseen user data we also developed two unsupervised methods for video inference that help overcome differences in the data SuperAnimal models were trained on compared to what data users might have. These so-called distribution shifts can come in various forms (e.g., spatial or temporal; see Methods). For example, the model can not perform well if the video is dramatically different in size than what we trained on (Figure 3a, b, c). Therefore, inspired by (40), we developed an unsupervised test-time augmentation called spatial-pyramid search that significantly boosted performance in three OOD videos (Figure 3b, c, Suppl. Video 3, Table S13; see Methods). This is unsupervised, as the user does not need to label any data, they simply give a range of video sizes. Note that in practice this does slow down inference time depending on the search parameter space.

Secondly, to improve temporal video performance we propose a new unsupervised domain adaptation method (41, 42) tailored for pose estimation called video adaptation that mitigates the jittering in predictions (Figure 3d-f, Suppl. Video 4). The method runs pose inference on the videos and treats the output predictions as the pseudo ground-truth labels and then fine-tunes the model. This does not take extensive training time, and can be run during video analysis. For example, if a video (of a given size) can be run at 40 FPS, this video adaptation would slow down processing to approx. 12 FPS. In Figure 3e-i we demonstrate the qualitative performance gains in video smoothness across frames with or without the video adaptation, and show quantification in Extended Data Figure S4e.

Unsupervised behavioral analysis.

To illustrate the value of zero-shot predictions for behavioral quantification (Figure 4a), we turned to an open-source dataset that was used to benchmark the performance of open-source machine learning tools vs. some commercially available solutions (15). Specifically, we used the open-field test (OFT) dataset presented in Sturman et al. (15). We evaluated the performance of SuperAnimal weights in an action segmentation task. To make OFT out-of-distribution, we made a variant of the SuperAnimal-TopViewMouse model

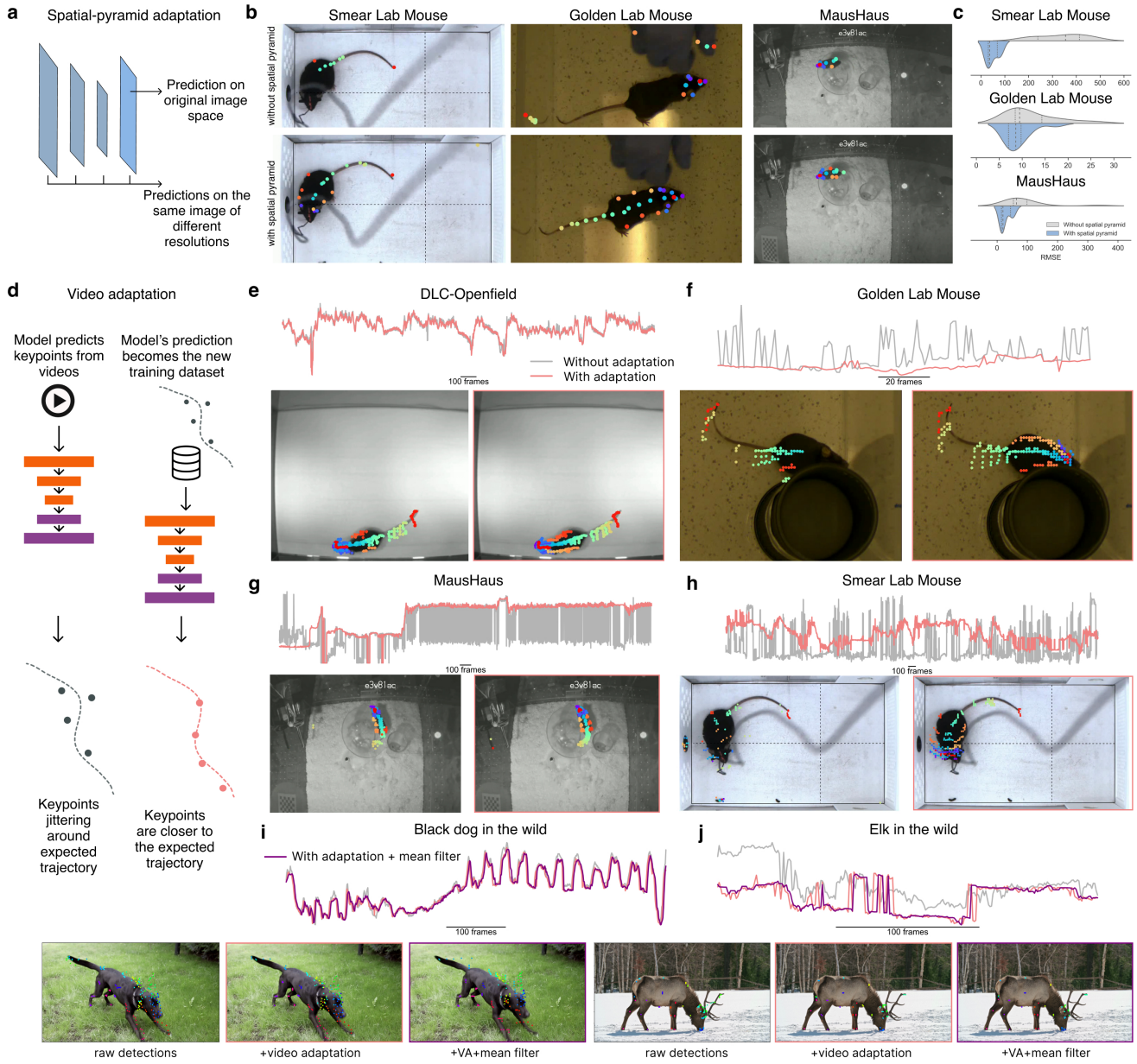

Figure 3. Unsupervised video adaptation methods. a: Conceptual diagram to demonstrate that the spatial-pyramid search leverages prediction from multiple resolutions. **b**: SuperAnimal-TopViewMouse model was used to infer poses on three OOD videos. Visual inspection shows zero-shot inference with vs. without the spatial-pyramid search. **c**: Quantification between with and without using spatial-pyramid search. **d**: Illustration of the unsupervised video adaptation algorithm. **e-h**: Animal size described by convex hull of keypoints. Frequent changes of the convex hull indicates non-smooth keypoint predictions, and below are example images with and without video adaptation showing the trailing keypoints for 10 past frames of data (to demonstrate the motion smoothness). **i-j**: Same as in **e-h** with an additional median filtering post-video adaptation examples (dark purple line).

by excluding the OFT dataset during training from full SuperAnimal-TopViewMouse.

As a strong baseline, we used the DeepLabCut keypoints trained by Sturman et al., who trained in a supervised way on each video specifically, thus making it in-domain (Figure 4a, b). We asked if the SuperAnimal model variant, which has never been trained on the 20 videos they present, is sufficient to classify two critical kinematic-based postures: unsupported rearing in the open field, and supported rearing against the box wall (Figure 4a, b, see also Suppl. Video 5). If the keypoints were too noisy, this task would be very challenging.

In order to transform keypoints into behavioral actions via segmentation, we used skeleton-based features to convert keypoints to feature vectors (see Methods). We then either used only a MLP-based classifier as in Sturman et al., or we used a newly described non-linear clustering algorithm called CEBRA (36) to further improve the feature space, followed by the same classifier (see Methods and Figure 4c-e).

We found that SuperAnimal zero-shot could be as good as the supervised keypoint model in predicting both behaviors (Figure 4d-f, g; linear mixed effect model, fixed effect of 'method': F=0.999, p=0.393; see Table S14). Moreover, using CEBRA slightly improved upon the behavior classifica-

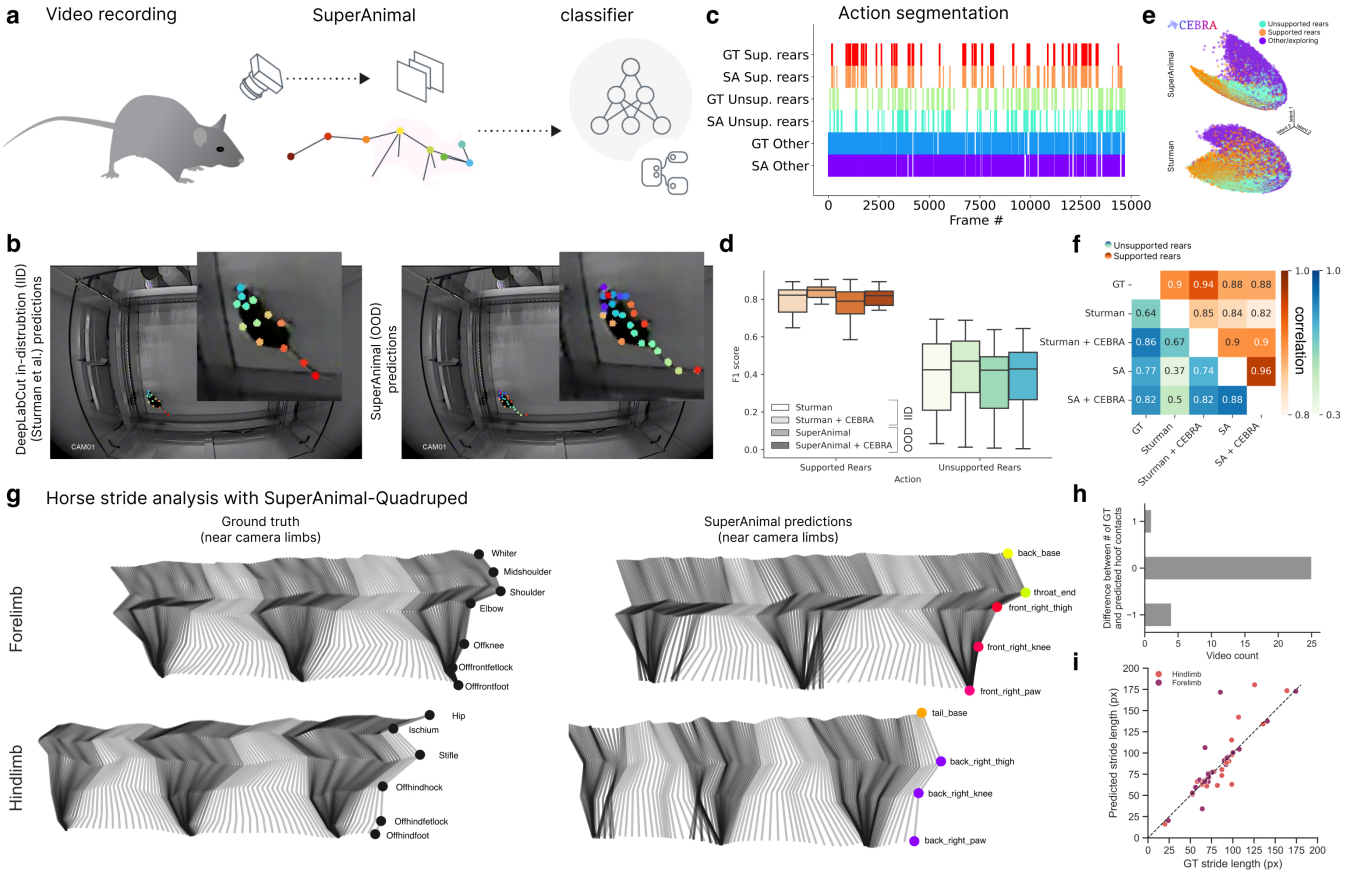

Figure 4. Zero-shot behavioral quantification with SuperAnimal. **a**: Workflow overview for behavioral analysis with SuperAnimal. **b**: Images of the open-source dataset from Sturman et al. (15) with their DeepLabCut "in distribution" model and our SuperAnimal zero-shot, out-of-distribution, results. **c**: Ethogram comparing ground truth annotations vs. zero-shot predictions from SuperAnimal-TopViewMouse. **d**: F1 score computed across IID (Sturman) and SuperAnimal with, or without CEBRA on the two behavioral classes. **e**: CEBRA (36) embedding on Sturman keypoints and SuperAnimal-based keypoints in 3D, transformed with FastICA for visualization. **f**: Correlation matrix that demonstrates the correlation between SuperAnimal-TopViewMouse and ground-truth annotations averaged across 3 annotators and across the model and keypoint configurations. **g**: We analyzed 30 horse videos where every frame had a ground truth (GT) annotation of keypoints (16) (left) vs. our SuperAnimal-Quadruped model (representative of performance on most videos, right). Light grey zones denoted the swing phase, black denotes the stance, for both ground truth (left) and classified (right). Only the right legs (closest to the camera) are shown. **h**: Histogram delineating the number of videos where the ground contact by the hoof were identical to the GT vs. over or under counted by 1 stride (no error larger than 1 was found). **i**: We computed the error between the GT stride length vs. model prediction for the hoofs (i.e, right_back_paw vs. Offhindfoot, etc). Each dot represents a stride, color denotes hindlimb vs. forelimb, near legs only.

tion, independent of which keypoints were used (Figure 4e, f). We also compared the correlation of our result based on SuperAnimal or Sturman keypoint data against the three annotators per video and find that our model is well correlated to the ground truth annotations, particularly when using CEBRA (Figure 4f). SuperAnimal-TopViewMouse also performed well with post-hoc unsupervised analysis of behavior with the newly introduced Keypoint-MoSeq (43) and AmadeusGPT (44). Therefore, collectively this demonstrates that without any pose training data, SuperAnimal models can be used for downstream behavioral analysis on out-of-distribution data.

Lastly, to show the utility of the SuperAnimal-Quadruped model in video analysis we performed gait analysis in horses. Here, we turn to a ground truth video dataset where every frame of the video was annotated by an equine expert (16). We computed the stride and swing phase of the gait and show that the SuperAnimal-Quadruped model with video adaptation can match ground truth (Figure 4g) in 24 out of 30 videos, where we only miss one stride detection (either over or under, (Figure 4h). A failure case is shown in Extended Data Figure S4). We also computed the hoof-ground contacts and find generally reasonable agreement between ground truth and predictions (Figure 4i). The fraction of contacts within 1-5 frames of ground truth was 60%-80.5%, respectively, averaged across front and hind limbs across all videos. Collectively, this suggests our SuperAnimal models can be used in real-world tasks both in and outside the laboratory.

Discussion

Biology often uses a few, common model organisms such as rodents, zebrafish, Drosophila and non-human primates (45, 46). Here we propose an approach to create robust, cross-lab neural network models that are applicable for rodents and many other quadrupeds (>40 species). What keypoints are of relevance, also depends on the experiment. For instance, in reaching experiments (9, 47), different keypoints are of interest than in open field studies. Our approach is general, and it will be an important future goal to expand the DeepLabCut

Model Zoo to additional animals (e.g., insects, birds, or fish) and behavioral contexts.

Building a pretrained pose model via supervised learning benefits from the availability of the annotated pose datasets, and we show that our formulation removes the obstacles of leveraging inhomogenous pose datasets, which enables SuperAnimal models to benefit from learning pose prior and larger datasets. Alternatively, unsupervised keypoint discovery can be used (48, 49). While the unsupervised approach requires no pose annotations, the learned keypoints might lack interpretability and it is not clear whether it allows zero-shot inference on OOD data. Therefore, both approaches that create predictions based on the superset of annotated keypoints from different studies and unsupervised keypoint discovery are promising, complementary directions.

Taken together, we aimed to reduce the (human and computing) resources needed to create or adapt animal pose models in both lab and in-the-wild animal studies, thereby increasing access to critical tools in animal behavior quantification. We developed a new framework called panoptic pose estimation, where models can be used across various environments in a zero-shot manner and if fine-tuned, they require $10\times$ less labeled data than previous models. Our method might lead to the homogenization of animal pose estimation models, making them useful to a broader range of users, thus lowering the entry barrier and reducing resource consumption. Moreover, labs can now share their data and we can leverage a global community effort to build more powerful models. The DeepLabCut Model Zoo web platform allows access to SuperAnimal pre-trained models, aids in collecting and labeling (Extended Data Fig S2d) more data, and hosts other user-shared models at http://modelzoo.deeplabcut.org.

References

1. Sandeep Robert Datta, David J Anderson, Kristin Branson, Pietro Perona, and Andrew Leifer. Computational neuroethology: a call to action. *Neuron*, 104(1):11–24, 2019.
2. Mackenzie Weygandt Mathis and Alexander Mathis. Deep learning tools for the measurement of animal behavior in neuroscience. *Current Opinion in Neurobiology*, 60:1–11, 2020.
3. Talmo D Pereira, Joshua W Shaevitz, and Mala Murthy. Quantifying behavior to understand the brain. *Nature neuroscience*, pages 1–13, 2020.
4. Lukas von Ziegler, Oliver Sturman, and Johannes Bohacek. Big behavior: challenges and opportunities in a new era of deep behavior profiling. *Neuropsychopharmacology*, 46(1):33–44, 2021.
5. Sébastien B Hausmann, Alessandro Marin Vargas, Alexander Mathis, and Mackenzie W Mathis. Measuring and modeling the motor system with machine learning. *Current opinion in neurobiology*, 70:11–23, 2021.
6. Devis Tuia, Benjamin Kellenberger, Sara Beery, Blair R. Costelloe, Silvia Zuffi, Benjamin Risse, Alexander Mathis, Mackenzie W. Mathis, Frank van Langevelde, Tilo Burghardt, Roland W. Kays, Holger Klinck, Martin Wikelski, Iain D. Couzin, Grant Van Horn, Margaret C. Crofoot, Chuck Stewart, and T. Berger-Wolf. Perspectives in machine learning for wildlife conservation. *Nature Communications*, 13, 2021.
7. Alexander Mathis, Steffen Schneider, Jessy Lauer, and Mackenzie W. Mathis. A primer on motion capture with deep learning: Principles, pitfalls, and perspectives. *Neuron*, 108:44–65, 2020.
8. David J. Anderson and Pietro Perona. Toward a science of computational ethology. *Neuron*, 84:18–31, 2014.
9. Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience*, 21:1281–1289, 2018.
10. Jessy Lauer, Mu Zhou, Shaokai Ye, William Menegas, Steffen Schneider, Tanmay Nath, Mohammed Mostafizur Rahman, Valentina Di Santo, Daniel Soberanes, Guoping Feng, Venkatesh N. Murthy, George Lauder, Catherine Dulac, Mackenzie W. Mathis, and Alexander Mathis. Multi-animal pose estimation, identification and tracking with deeplabcut. *Nature Methods*, 19:496 – 504, 2022.
11. Jacob M Graving, Daniel Chae, Hemal Naik, Liang Li, Benjamin Koger, Blair R Costelloe, and Iain D Couzin. Deepposekit, a software toolkit for fast and robust animal pose estimation using deep learning. *Elife*, 8:e47994, 2019.
12. Semih Günel, Helge Rhodin, Daniel Morales, João Campagnolo, Pavan Ramdya, and Pascal Fua. Deepfly3d, a deep learning-based approach for 3d limb and appendage tracking in tethered, adult drosophila. *Elife*, 8:e48571, 2019.
13. Talmo D Pereira, Nathaniel Tabris, Junyu Li, Shruthi Ravindranath, Eleni S Papadoyannis, Z Yan Wang, David M Turner, Grace McKenzie-Smith, Sarah D Kocher, Annegret Lea Falkner, et al. Sleap: multi-animal pose tracking. *bioRxiv*, 2020. doi: 10.1101/2020.08.31.276246.
14. Praneet C Bala, Benjamin R. Eisenreich, Seng Bum Michael Yoo, Benjamin Yost Hayden, Hyun Soo Park, and Jan Zimmermann. Automated markerless pose estimation in freely moving macaques with openmonkeystudio. *Nature Communications*, 11, 2020.
15. Oliver Sturman, Lukas von Ziegler, Christa Schläppi, Furkan Akyol, Mattia Privitera, Daria Slominski, Christina Grimm, Laetitia Thieren, Valerio Zerbi, Benjamin Grewe, et al. Deep learning-based behavioral analysis reaches human accuracy and is capable of outperforming commercial solutions. *Neuropsychopharmacology*, 45(11):1942–1952, 2020.
16. Alexander Mathis, Thomas Biasi, Steffen Schneider, Mert Yuksekgonul, Byron Rogers, Matthias Bethge, and Mackenzie W Mathis. Pretraining boosts out-of-domain robustness for pose estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1859–1868, 2021.
17. Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir P. Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel J. Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher R'e, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models. *ArXiv*, abs/2108.07258, 2021.
18. Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2020.
19. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Min-

derer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.
20. Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Keypoint localization via transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11802–11812, October 2021.
21. Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *arXiv preprint arXiv:2204.12484*, 2022.
22. Jinkun Cao, Hongyang Tang, Hao-Shu Fang, Xiaoyong Shen, Cewu Lu, and Yu-Wing Tai. Cross-domain adaptation for animal pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9498–9507, 2019.
23. Isaac Chang. Trained DeepLabCut model for tracking mouse in open field arena with topdown view. https://doi.org/10.5281/zenodo.3955216. *Zenodo*, July 2020. doi: 10.5281/zenodo.3955216.
24. Simon RO Nilsson, Nastacia L. Goodwin, Jia Jie Choong, Sophia Hwang, Hayden R Wright, Zane C Norville, Xiaoyu Tong, Dayu Lin, Brandon S. Bentzley, Neir Eshel, Ryan J McLaughlin, and Sam A. Golden. Simple behavioral analysis (simba) – an open source toolkit for computer classification of complex social behaviors in experimental animals. *bioRxiv*, 2020. doi: 10.1101/2020.04.19.049452.
25. Daniel Joska, Liam Clark, Naoya Muramatsu, Ricardo Jericevich, Fred Nicolls, Alexander Mathis, Mackenzie W. Mathis, and Amir Patel. Acinoset: A 3d pose estimation dataset and baseline models for cheetahs in the wild. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13901–13908, 2021.
26. Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization : Stanford dogs. In *Semantic Scholar*, 2012.
27. Jinkun Cao, Hongyang Tang, Haoshu Fang, Xiaoyong Shen, Cewu Lu, and Yu-Wing Tai. Cross-domain adaptation for animal pose estimation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9497–9506, 2019.
28. Prianka Banik, Lin Li, and Xishuang Dong. A novel dataset for keypoint detection of quadruped animals from images. *arXiv preprint arXiv:2108.13958*, 2021.
29. Benjamin Biggs, Oliver Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out? 3d animal reconstruction with expectation maximization in the loop. In *European Conference on Computer Vision*, pages 195–211. Springer, 2020.
30. Hang Yu, Yufei Xu, Jing Zhang, Wei Zhao, Ziyu Guan, and Dacheng Tao. Ap-10k: A benchmark for animal pose estimation in the wild. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
31. Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.
32. Gido M van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):1–14, 2020.
33. Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.
34. Tanmay Nath, Alexander Mathis, An Chi Chen, Amir Patel, Matthias Bethge, and Mackenzie W Mathis. Using deeplabcut for 3d markerless pose estimation across species and behaviors. *Nature protocols*, 14:2152–2176, 2019.
35. Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *ArXiv*, abs/1310.1531, 2013.
36. Steffen Schneider, Jin H Lee, and Mackenzie W Mathis. Learnable latent embeddings for joint behavioral and neural analysis. *CoRR*, abs/2204.00673, 2022. doi: 10.48550/ARXIV.2204.00673.
37. Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. A simple yet effective baseline for 3d human pose estimation, 2017.
38. Mu Zhou, Lucas Stoffl, Mackenzie W. Mathis, and Alexander Mathis. Rethinking pose estimation in crowds: overcoming the detection information-bottleneck and ambiguity. *IEEE/CVF International Conference on Computer Vision*, 2023.
39. Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose+: Vision transformer foundation model for generic body pose estimation. *ArXiv*, abs/2212.04246, 2022.
40. Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.
41. Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pages 5468–5479. PMLR, 2020.
42. Evgenia Rusak, Steffen Schneider, George Pachitariu, Luisa Eck, Peter Vincent Gehler, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. If your data distribution shifts, use self-learning. *Transactions on Machine Learning Research*, 2021.
43. Caleb Weinreb, Mohammed Osman Abdal Monium, Libby Zhang, Sherry Lin, Jonah Pearl, Sidharth Annapragada, Eli Conlin, Winthrop F. Gillis, Maya Jay, Shaokai Ye, Alexander Mathis, Mackenzie Weygandt Mathis, Talmo Pereira, Scott W. Linderman, , and Sandeep Robert Datta. Keypoint-moseq: parsing behavior by linking point tracking to pose dynamics. *bioRxiv*, 2023. doi: https://doi.org/10.1101/2023.03.16.532307.
44. Shaokai Ye, Jessy Lauer, Mu Zhou, Alexander Mathis, and Mackenzie W. Mathis. Amadeusgpt: a natural language interface for interactive animal behavioral analysis, 2023.
45. Michael M Yartsev. The emperor's new wardrobe: rebalancing diversity of animal models in neuroscience research. *Science*, 358(6362):466–469, 2017.
46. Georg Striedter. *Model Systems in Biology: History, Philosophy, and Practical Concerns*. MIT Press, 2022.
47. Roger N Lemon. Descending pathways in motor control. *Annu. Rev. Neurosci.*, 31:195–218, 2008.
48. Jennifer J Sun, Serim Ryou, Roni H Goldshmid, Brandon Weissbourd, John O Dabiri, David J Anderson, Ann Kennedy, Yisong Yue, and Pietro Perona. Self-supervised keypoint discovery in behavioral videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2171–2180, 2022.
49. Praneet Bala, Jan Zimmermann, Hyun Soo Park, and Benjamin Y Hayden. Self-supervised secondary landmark detection via 3d representation learning. *International Journal of Computer Vision*, pages 1–15, 2023.
50. Lukas von Ziegler, Oliver Sturman, and Johannes Bohacek. Videos for deeplabcut, noldus ethovision X14 and TSE multi conditioning systems comparisons. https://doi.org/10.5281/zenodo.3608658. *Zenodo*, January 2020. doi: 10.5281/zenodo.3608658.
51. Jared M. Cregg, Roberto Leiras, Alexia Montalant, Paulina Wanken, Ian R. Wickersham, and Ole Kiehn. Brainstem neurons that command mammalian locomotor asymmetries. *Nature neuroscience*, 23:730 – 740, 2020.
52. Prianka Banik, Lin Li, and Xishuang Dong. A novel dataset for keypoint detection of quadruped animals from images. *ArXiv*, abs/2108.13958, 2021.
53. Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.
54. Benjamin Biggs, Thomas Roddick, Andrew Fitzgibbon, and Roberto Cipolla. Creatures great and smal: Recovering the shape and motion of animals from video. In *Asian Conference on Computer Vision*, pages 3–19. Springer, 2018.
55. iNaturalist. OGBIF Occurrence Download. https://doi.org/10.15468/dl.p7nbxt. *iNaturalist*, July 2020. doi: https://doi.org/10.15468/dl.p7nbxt.

56. Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
57. Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2016.
58. Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.
59. Sam Golden. Open-field Social Investigation Videos, https://edspace.american.edu/openbehavior/project/open-field-social-investigation-videos-donated-sam-golden/. *Open Behavior*, July 2022.
60. Matt Smear. Olfactory Search Video, https://edspace.american.edu/openbehavior/project/olfactory-search-video-donated-matt-smear/. *Open Behavior*, July 2022.
61. Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9404–9413, 2019.
62. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
63. Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.
64. Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.
65. Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020.
66. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
67. Gary A Kane, Gonçalo Lopes, Jonny L Saunders, Alexander Mathis, and Mackenzie W Mathis. Real-time, low-latency closed-loop feedback using markerless posture tracking. *Elife*, 9:e61909, 2020.
68. Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.
69. Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. Pmlr, 2018.
70. Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *International conference on machine learning*, pages 1802–1811. PMLR, 2019.
71. Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. *Advances in neural information processing systems*, 32, 2019.
72. Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.
73. James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.
74. Matteo Ruggero Ronchi and Pietro Perona. Benchmarking and error diagnosis in multi-instance pose estimation, 2017.
75. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2022.
76. Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. Fitting linear mixed-effects models using lme4. *arXiv preprint arXiv:1406.5823*, 2014.
77. Naturalist. inaturalist. available from https://www.inaturalist.org. *Naturalist*, 2019.
78. Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981, 2020.
79. MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020.
80. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Acknowledgments. The authors thank Mu Zhou, Lucas Stoffl and Shilong Zhang for discussions and feedback. We also thank Ole Kiehn, Jared Cregg, Carmelo Bellardita, Johannes Bohacek, Sam Golden and Nastacia Goodwin for generously sharing data on OpenBehavior, zenodo.org, or with us directly. Funding was provided by the EPFL, CZI grant DAF2020-207363 from the Chan Zuckerberg Initiative DAF, an advised fund of Silicon Valley Community Foundation, The Vallee Foundation (MWM), SNSF grant no. 310030_201057 (MWM), a Novartis Foundation for Medical-Biological Research Young Investigator Grant (MWM). MWM is the Bertarelli Foundation Chair of Integrative Neuroscience.

Author Contributions. Conceptualization: MWM, AM, SY; Methodology: SY, AF, JL, StS, AM; Software: SY, AF, MV, StS, JL, AM, MWM; Investigation: SY, AF, JL, AM; Dataset Curation: MWM, AM, SY, TQ; MausHaus data collection and labeling: MWM; iNaturalist data curation and labeling: TQ; WebApp: StS, MV; Writing-Original Draft: MWM, SY; Writing-Editing: MWM, AM, SY, MV, StS, JL, AF.

Conflicts: M.W. Mathis and S. Schneider are co-founders at Kinematik AI. The other authors declare no conflicts of interest. The funders had no role in the conceptualization, design, data collection, analysis, decision to publish, or preparation of the manuscript.

Data Availability. The publicly available datasets are detailed in the Methods. SuperAnimal model weights are banked at HuggingFace: https://huggingface.co/mwmathis/DeepLabCutModelZoo-SuperAnimal-Quadruped and https://huggingface.co/mwmathis/DeepLabCutModelZoo-SuperAnimal-TopViewMouse.

Code Availability. Code to use the DeepLabCut Model Zoo: https://github.com/DeepLabCut/DeepLabCut; it is available since version 2.3.1. Code and data to reproduce the figures: https://github.com/AdaptiveMotorControlLab/modelzoo-figures. All other requests should be made to the corresponding authors.

Methods

Datasets

We collected publicly available datasets from the community, as well as in-house datasets for building the SuperAnimal models. Thereby, we sought to cover diverse individuals, backgrounds, scenarios, and postures. In the following we detail the references for those datasets.

TopViewMouse-5k. 3CSI, BM, EPM, LDB, OFT See full details at (15) and in (50). **BlackMice** See full details at (23). **WhiteMice** Courtesy of Prof. Sam Golden and Nastacia Goodwin. See details in SIMBA (24). **TriMouse** See full details at (10). **DLC-Openfield** See full details at (9). **Kiehn-Lab-Openfield, Swimming, and treadmill** Courtesy of Prof. Ole Kiehn, Dr. Jared Cregg, and Prof. Carmelo Bellardita; see details at (51). **MausHaus** We collected video data from five single-housed C57BL/6J male and female mice in an extended home cage, carried out in the laboratory of Mackenzie Mathis at Harvard University and also EPFL (temperature of housing was 20-25C, humidity 20-50%). Data were recorded at 30Hz with 640×480 pixels resolution acquired with White Matter, LLC eV cameras. Annotators localized 26 keypoints across 322 frames sampled from within DeepLabCut using the k-means clustering approach (34). All experimental procedures for mice were in accordance with the National Institutes of Health Guide for the Care and Use of Laboratory Animals and approved by the Harvard Institutional Animal Care and Use Committee (IACUC) (n=1 mouse), and by the Veterinary Office of the Canton of Geneva (Switzerland; license GE01) (n=4 mice).

Quadruped-40K. AwA-Pose Quadruped dataset, see full details at (52). **AnimalPose** See full details at (27). **AcinoSet** See full details at (25). **Horse-30** Horse-30 dataset, benchmark task is called Horse-10; See full details at (16). **StanfordDogs** See full details at (53, 54). **AP-10K** See full details at (30). **iRodent** We utilized the iNaturalist API functions for scraping observations with the taxon ID of Suborder Myomorpha (55). The functions allowed us to filter the large amount of observations down to the ones with photos under the CC BY-NC creative license. The most common types of rodents from the collected observations are Muskrat (Ondatra zibethicus), Brown Rat (Rattus norvegicus), House Mouse (Mus musculus), Black Rat (Rattus rattus), Hispid Cotton Rat (Sigmodon hispidus), Meadow Vole (Microtus pennsylvanicus), Bank Vole (Clethrionomys glareolus), Deer Mouse (Peromyscus maniculatus), White-footed Mouse (Peromyscus leucopus), Striped Field Mouse (Apodemus agrarius). We then generated segmentation masks over target animals in the data by processing the media through an algorithm we designed that uses a Mask Region Based Convolutional Neural Networks(Mask R-CNN) (56) model with a ResNet-50-FPN backbone (57), pretrained on the COCO datasets (58). The processed 443 images were then manually labeled with both pose annotations and segmentation masks. Data is banked at `https://zenodo.org/record/8250392`.

Additional OOD Videos. In Figure 3, for video testing we additionally used the following data: **Golden Lab mouse**: see details at (59). **Smear Lab Mouse**: see details at (60). **Mathis Lab MausHaus**: New video conditions, but the same as MausHaus ethics approval as above.

Panoptic pose estimation

We cast animal pose estimation as panoptic segmentation (61) on the animal body; i.e., every pixel on the body is potentially a semantically meaningful keypoint that has an individual identity. Ideally, an infinite collection of diverse pose datasets covers this and the union of keypoints that are defined across datasets make the label space of panoptic pose estimation.

Data conversion and panoptic vocabulary mapping (generalized data converter). Data came from multiple sources and in multiple formats. To homogenize different annotation formats (COCO-style, DeepLabCut format, etc.), we implemented a generalized data converter. We parsed more than 20 public datasets and re-formatted them into DeepLabCut projects. Besides data conversion, the generalized data converter also implements key steps for the panoptic animal pose estimation task formulation. These steps include:

1. **Hand-crafted conversion mapping.** The same anatomical keypoint might be named differently in different datasets, or different anatomical locations might correspond to different labels in different datasets. Thus, the generalized data converter used a hand-crafted conversion mapping (see Extended Data Figs. S1a, S3) to enforce a shared vocabulary among datasets. We checked the visual appearance of keypoints to determine whether two keypoints (in different datasets) should be regarded as identical. In such cases, the model had to learn (possible) dataset-bias in a data-driven way. We can also think of it as a form of data augmentation that randomly shifts the coordinate of keypoints by a small magnitude, which is the case for keypoints which most dataset creators agree on (e.g., keypoints on the face). For keypoints on the body, the quality of the conversion table can be critical for the model to learn a stable representation of animal bodyparts.

2. **Vocabulary projection.** After the conversion mapping is made, keypoints from various datasets were projected to a super-set keypoint space. Every keypoint became a one-hot vector in the union of keypoint spaces of all datasets. Thereby the animal pose vocabularies were unified.
3. **Dataset merging.** After annotations were unified into the super-set annotation space, we merged annotations from datasets by concatenating them into a collection of annotation vectors. Note that if the images only displayed a single species, we essentially built a specialized dataset for that species in different cage and camera settings. If there were multiple species present, we essentially grouped them in a species-invariant way to encourage the model to learn species-agnostic keypoint representations, as is the case for our SuperAnimal-Quadruped model.

Data split and training fraction. We used a 80:20 and 95:5 training:test ratio to split Quadruped-40K and TopViewMouse-5K, respectively. To measure in model performance, we used three data splits for all our experiments.

For all benchmark testing we used the data splits of the original papers. One exception is AnimalPose (22) in which there is no available data split that includes all 5 species. Therefore, we randomly split the dataset into 80:20 to create our own split.

Training protocol. The SuperAniml-Quadruped model used the same training protocol described in AP-10K paper (30). Specifically, we used the Adam optimizer (62) with an initial learning rate of $5e-4$. The total training epochs was 210, and we used a step decay for the learning rate at 170 and 200 epochs. We used batchsize 64.

For fine-tuning models with a very small number of unique images (e.g. less than 64 images in the training set), we did not train with batch-norm and used an initial learning rate $5e-5$ and observed stable training with this setting.

Model architectures

To show our methods are consistent regardless of the choice of architectures and data splits, we provided results using both variants of convolutional neural networks (CNNs) and several Transformer architectures.

DLCRNet. We use DLCRNet_ms5 (10) as the baseline network architecture for its excellent performance on animal pose estimation. A batch size of 8 was used and the SuperAnimal-TopViewMouse and were trained for a total of 750k iterations, respectively. In the fine-tuning stage, a batch size of 8 was used for 70k iterations. The Adam optimizer (62) was used for all training instances, and we otherwise used default parameters. We follow DeepLabCut's multi-step learning rate scheduler to drop learning rates three times from 1e-4 to 1e-5. Cross-entropy is used for learning heatmaps. For fine-tuning experiments, we keep the same optimizer, batch size and learning rate scheduler. The total number of training steps is adjusted to 70K iterations. During video adaptation, we keep the same optimizer and learning rate scheduler, but with batch size 1 and total training steps as 1000. We observe low computational budget as described is sufficient for the model to adapt.

HRNet-w32. We also used HRNet (18) for SuperAnimal-Quadruped and further experiments shown in the Extended Data Figures S3b, 1h, and 2. Note that in the Extended Data the data splits are different from the ones in the main text and this split was also used for the transformers, therefore can be considered as another fully independent replication.

Transformers. Inspired by recent results of Vision Transformers (19) on human pose estimation tasks (21) we assessed ViT's zero-shot performance. We conducted experiments with the original ViT architecture in three setups: with masked auto-encoder (MAE) (63) initialization, DeiT (64) initialization and truncated normal initialization with standard deviation 0.02 and 0 mean. Following the original setup (19), we did not use a convolutional backbone. The input image of size 224×224 was split into patches of 16×16 pixels, the depth of the transformer encoder was equal to 12 and each attention layer had 12 heads with a feature dimension of 768. It was crucial to use a pre-trained vision transformer; without pre-training, the model did not converge for either dataset (data not shown).

We also adapted the TokenPose model by Yang et al. (20), which adds information about each keypoint in learnable queries called keypoint embeddings. The model was originally used for human pose estimation with a fixed number of keypoints. Combining TokenPose and panoptic animal pose estimation, we obtain AnimalTokenPose models that are able to achieve high zero-shot performance in OOD datasets we prepared (Fig. 1 and Figure 2).

For keypoint estimation, 12 transformer encoder blocks with feature vector of size 192 were stacked. While the ViT encoder received raw pixels as an input, in TokenPose (20) the images of size 256×256 are first processed by a convolutional backbone and captured abstract features are then split into patches of size 4×4. As in TokenPose (20), we used the first 3 stages of HRNet (65) and 2 stacked residual blocks from a ResNet (66).

Algorithmic enhancements for training and inference

Keypoint gradient masking. First we manually verified a semantic mapping of the datasets with diverse naming (i.e., nose in dataset 1 and snout in dataset 2). Then, we defined a master keypoint space naming, where no one dataset needed to have all the named identified. This yielded sparse keypoint annotations into the super-set keypoint space (Extended Data Figs. S1b, c). Training naively on these projected annotations would harm the training stability, as the loss function penalizes undefined keypoints, as if they were not visible (i.e., occluded).

For stable training of our panoptic pose estimation model, we mask components of the loss function across keypoints. The keypoint mask n_k is set to 1 if the keypoint k is present in the annotation of the image and set to 0 if the keypoint is absent. We denote the predicted probability for keypoint k at pixel (i,j) as $p_k(i,j) \in [0,1)$ and the respective label as $t_k(i,j) \in \{0,1\}$, and formulate the masked L_k error loss function as

$$\mathcal{L}_{\mathrm{L_k}} = \sum_{k=1}^{m} \sum_{i,j} n_k \cdot \|p_k(i,j) - t_k(i,j)\|^k, \tag{1}$$

with $k = 2$ for mean square error and $k = 1$ for L1 loss (e.g. used for locref maps in DLCRNet (10)) and the masked cross-entropy loss function as

$$\mathcal{L}_{\mathrm{CE}} = \sum_{k=1}^{m} \sum_{i,j} n_k t_k(i,j) \log p_k(i,j). \tag{2}$$

Note that we make distinct the difference between not annotated and not defined in the original dataset and we only mask undefined keypoints. This is important as, in the case of sideview animals, "not annotated" could also mean occluded/invisible. Adding masking to not annotated keypoints will encourage the model to assign high likelihood to occluded keypoints.

Also note that the network predictions $p_k(i,j)$ are generated by applying a softmax to the logits $l_k(i,j)$ across all possible keypoints, including masked ones:

$$p_k(i,j) = \frac{\exp l_k(i,j)}{\sum_{j=1}^{M} \exp l_j(i,j)}. \tag{3}$$

The masking in the loss function then ensures that probability assigned to non-defined keypoints is neither penalized nor encouraged during training.

Handling the train and test time resolution discrepancy. One notable challenge our models face at inference time is the discrepancy in the resolution of images between train and test stages. Even though scale jitter augmentation is part of most pose estimation frameworks' data augmentation pipeline, including DeepLabCut's (10, 34, 67), one does not expect to handle dramatic change in the resolution. Indeed, it is well known that scale-augmentation greatly helps robustness to image and animal sizes (67). However, for shared models, we cannot anticipate the animal scale of user datasets.

In the case of fine-tuning, the downstream dataset (and the animals present in it) could have a very different resolution from the pre-training datasets. To handle those outliers, we apply resizing (height 400 pixels and same aspect ratio) to downstream datasets if their sizes are drastically different from our training images. To further deal with scale changes, we employ spatial-pyramid search at test-time (see below).

Domain shifts and unsupervised adaptation. These domain shifts (68) describe a classical vulnerability of neural networks, where a model takes inputs from a data domain that is dissimilar from the training data domain, which usually leads to large performance degradation. We empirically observe three types of domain shifts when applying our models in a zero-shot manner. These domain shifts range from pixel statistics shift (69), to spatial shift (70), to semantic shift (68, 69). To mitigate those, we applied two methods, test time spatial-pyramid search and video adaptation.

Test time spatial-pyramid search. Even though during training time, our model learns images of various resolutions, it might only perform the best if the appearance size of the animal is close to those in the training images. For inferencing OOD data, We propose to search for proper resolutions at test time and aggregate good candidates for the final predictions.

Therefore, during inference, we build a spatial-pyramid composed of model's predictions for multiple copies of the original image at different resolutions. We used model's confidence as the criterion to filter out the resolutions that give sub-optimal performance and aggregate (taking median) predictions from resolutions that have above-threshold confidence as our final prediction.

The train-test resolution discrepancy (71) has been studied actively and most approach it through multi-resolution fusion (10, 40, 65). Previous work mostly focuses on IID setting where the resolution of testing images did not vary considerably from

the training images. Moreover, prior work approaches multi-resolution fusion via deep features, requiring modifications of the architecture and adding more parameters. In contrast, the proposed spatial-pyramid search is designed to aid SuperAnimal models in zero-shot scenario where the resolutions of testing images are most likely out of distribution to our training images. We did not apply multi-resolution fusion via deep features for that requires fixing choice of architectures. On the other hand, commonly used multi-scale testing in IID setting does not need to carefully filter out very noisy predictions. This method can also be used for calibration to find the optimal scale.

Spatial-pyramid pseudo-code:

```
def  spatial_pyramid_search(images, model, scale_list, confidence_threshold, cosine_threshold):
  # generate rescaled version of original images with multiple scaling factor
        rescaled_images = rescale_images(images, scale_list)
        preds_per_scale = []
        # gather predictions of the model, assuming the final pred_keypoints are projected to the
    original image space by the forward function
  for rescaled_image in rescaled_images:
    pred_keypoints = model(rescaled_image)
    preds_per_scale.append(pred_keypoints)

  # using median to get a good estimate of expected keypoint positions
        median_keypoint = get_median_keypoint(preds_per_scale)
        # If the rescaled image is not suitable for the model, we expect the model have a confidence less
     than a given threshold
        pred_keypoints = filter_by_confidence(pred_keypoints, confidence_threshold )
        # A median filter alone does not remove outliers. After confidence filtering, we compare the
    remained predictions to the median keypoint and drop the low quality predictions
        pred_keypoints = filter_by_cosine_similarity(pred_keypoints, median_keypoint, cosine_threshold)

        return get_median_keypoints(pred_keypoints)
```

Video adaptation.

To aid SuperAnimal models to adapt to novel videos, we inference the model on the videos, and treat these predictions as the pseudo ground-truth (72) labels to train on. We remove the predictions that have low confidence to filter out unreliable predictions. Empirically, 1000 iterations with batch size 1 is sufficient to greatly reduce the jitter. The optimal number of iterations and the confidence threshold are hyperparameters for different videos.

Video adaptation pseudo-code:

```
def get_pseudo_predictions(frame_id):
  # return pseudo prediction by frame id

def video_adaptation(model, video_data_loader, optimizer, threshold):
  for data in video_data_loader:
    frame_id = data['frame_id']
    Image = data['image']
    pseudo_keypoints = get_pseudo_predictions(frame_id)
    preds = model(image)
    loss = criterion(preds, pseudo_keypoints, mask_by_threshold = threshold)
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

Combined together. The two methods above can be combined for better results: the pseudo labels can come from test time spatial-pyramid search which gives more accurate pseudo labels for the model to learn.

Memory replay. Catastrophic forgetting (73) describes a classic problem in continual learning (32). Indeed, a model gradually loses its ability to solve previous tasks after it learns to solve new ones.

Fine-tuning a SuperAnimal models falls into the category of continual learning: the downstream dataset defines potentially different keypoints than those learned by the models. Thus, the models might forget the keypoints they learned and only pick up those defined in the target dataset. Here, retraining with the original dataset and the new one, is not a feasible option as datasets cannot be easily shared and more computational resources would be required.

To counter that, we treat zero-shot inference of the model as a memory buffer that stores knowledge from the original model. When we fine-tune a SuperAnimal model, we replace the model predicted keypoints with the ground-truth annotations, resulting in hybrid learning of old knowledge and new knowledge. The quality of the zero-shot predictions can vary and we use the

confidence of prediction as a threshold to filter out low confidence predictions. With threshold set to one, memory replay fine-tuning becomes naive-fine-tuning.

Memory replay pseudo-code:

```
def is_defined(keypoints):
  # check whether the original dataset defines each keypoint. We use a flag `-1` to denote that a given keypoint is not defined in the original dataset. Note this is different from not annotated, which use flag `0`
  return True if keypoints[2] >= 0 else False

def load_pseudo_keypoints(image_ids):
  # get the pseudo keypoints by image IDs.
  # note, pseudo keypoints are loaded from disk and fixed throughout the process, so not drifting as is expected in typical online pseudo labeling
  return pseudo_keypoints

def get_confidence(keypoints):
  # get the model confidence of the predicted keypoints. Unlike ground truth data that have 3 discrete flags, predicted keypoints have confidence that can be used as likelihood readout for post-inference analysis
  return keypoints[2]

def memory_replay(model, superset_gt_data_loader, optimizer, threshold):

    # gt data is preprocessed such that annotations are now in superset keypoint space.
    # every gt keypoint has 3 flags (-1: not defined, 0: not labeled, 1: annotated)

    For batch_data in superset_gt_data_loader:

    gt_keypoints = batch_data['keypoints']
    image_ids = batch_data['image_ids']
    images = batch_data['images']
    # model() is a pytorch style forward function
    preds = model(images)
    pseudo_keypoints = load_pseudo_keypoints(image_ids)
    # 3 here is (x, y, flag)
    batch_size, num_kpts, 3 = gt_keypoints
    # iterate through batch
    For b_id in batch_size:
  # iterate through keypoints
        For kpt_id in range(num_kpts):
  # since this specific body part is not defined in the new dataset, we use saved pseudo labels (zero-shot prediction) as gt. This prevents catastrophic forgetting and drifting. We can also use confidence to filter the pseudo keypoints
  If not is_defined(gt_keypoints[b_id, kpt_id]) and get_confidence(pseudo_keypoints[b_id][kpt_id]) > threshold:
      # we assume a single animal scenario for simplicity. For multiple animals, matching between gt and pseudo keypoints need to be completed.
      gt_keypoints[b_id][kpt_id] = pseudo_keypoints[b_id][kpt_id]

    loss = criterion(preds, gt_keypoints)
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

Automatic keypoint matching. In cases where users want to apply our models to an existing, annotated pose dataset, we recommend to use our keypoint matching algorithm. This step is important because our models define their own vocabulary of keypoints that might differ from the novel pose dataset. To minimize the gap between the model and the dataset, we propose a matching algorithm to minimize the gap between the models' vocabulary and the dataset vocabulary. Thus, we use our model to perform zero-shot inference on the whole dataset. This gives pairs of prediction and ground-truth for every image. Then, we cast the matching between models' predictions (2D coordinates) and ground-truth as bipartite matching using the Euclidean distance as the cost between paired of keypoints. We then solve the matching using the Hungarian algorithm. Thus for every image, we end up getting a matching matrix where 1 counts for match and 0 counts for non-matching. Because the models' predictions can be noisy from image to image, we average the aforementioned matching matrix across all the images and perform another bipartite matching, resulting in the final keypoint conversion table between the model and the dataset (example affinity matrices are shown in Figure S2a,b).

Note that the quality of the matching will impact the performance of the model, especially in fine-tuning. In the case where,

e.g., the annotation *nose* is mistakenly converted to keypoint *tail* and vice versa, the model will have to de-learn the channel that corresponds to nose and tail.

Evaluation metrics

Supervised metrics for pose estimation.

RMSE. Root Mean Squared Error is a metric to measure the distance between prediction and ground truth annotations in pixel space (7, 9). However for pose estimation, it does not take the scale of the image and individuals into consideration and the distance is thus non-normalized. As our data is highly variable, we also sometimes use normalized errors. We use RMSE for the DLC-Openfield benchmarking, as this was the original authors main reported metric. Note that during evaluating RMSE, we do not remove predictions that have low confidence due to occlusion. Therefore, all predictions including outliers are penalized by RMSE.

Normalized Error. For Horse-10 experiments we use the eye-to-nose distance for normalization (computed ground truth is avilable in Horse-30 (16)).

mAP. Mean average precision (mAP) is the averaged precision of object keypoint similarity (OKS) (74):

$$OKS = \frac{\sum_{i=1}^{n} \left[\exp\left(-d_i^2/2s^2{k_i}^2\right)\delta\left(v_i > 0\right)\right]}{\sum_{i=1}^{n}\left[\delta\left(v_i > 0\right)\right]},$$

where d_i is the Euclidean distances between each corresponding ground truth and detected keypoint and v_i is the visibility flags of the ground truth, s is the object scale and k_i is a per keypoint constant that controls falloff (see full implementation details at (58)), and see Methods Table 1. s is the square root of bounding box area (product of width X height of the bounding box).

Unsupervised metrics for video prediction smoothness. Convex hull body area measurement

To evaluate the smoothness of SuperAnimal model predictions in video, we utilize a simple unsupervised heuristic. It computes the area of a polygon encompassing all keypoints, the idea being that the smoother the detections, the lower the variance of this polygon's area. This is formally noted by A_{body}, to estimate the animal body area. A_{body} is calculated using the convex hull

Body Part	k, in pixels	Body Part	k, in pixels
nose	0.026	upper_jaw	0.067
lower_jaw	0.067	mouth_end_right	0.067
mouth_end_left	0.067	right_eye	0.025
right_earbase	0.067	right_earend	0.067
right_antler_base	0.067	right_antler_end	0.067
left_eye	0.025	left_earbase	0.067
left_earend	0.067	left_antler_base	0.067
left_antler_end	0.067	neck_base	0.035
neck_end	0.067	throat_base	0.067
throat_end	0.067	back_base	0.067
back_end	0.067	back_middle	0.035
tail_base	0.067	tail_end	0.079
front_left_thai	0.072	front_left_knee	0.062
front_left_paw	0.079	front_right_thigh	0.072
front_right_knee	0.062	front_right_paw	0.089
back_left_paw	0.107	back_left_thigh	0.107
back_right_thai	0.087	back_left_knee	0.087
back_right_knee	0.089	back_right_paw	0.067
belly_bottom	0.067	body_middle_right	0.067
body_middle_left	0.067		

Table 1. We used the following k values per bodypart for the SuperAnimal-Quadruped evaluation.

containing all keypoints over time. Let $\mathcal{K}$ represent the set of all keypoints for the animal at each time step, and conv($\mathcal{K}$) denote the convex hull containing all keypoints. The animal body area, A_{body}, is then given by the area of the convex hull:

$$A_{\text{body}} = \text{Area}(\text{conv}(\mathcal{K}))$$

where Area(conv($\mathcal{K}$)) is the function that calculates the area of the convex hull conv($\mathcal{K}$) containing all keypoints over time.

Jittering metric

We define jittering, denoted by J, as the average of the absolute values of centered, non-signed speeds across all examples and all keypoints. For a given keypoint k and example e, the jittering value is computed as follows:

$$\text{J}_{k,e} = \frac{1}{N_{k,e}} \sum_{i=1}^{N_{k,e}} \left| v_{k,e,i} \right|$$

where: - $\text{J}_{k,e}$ is the jittering value for keypoint k in example e, - $N_{k,e}$ is the total number of centered, non-signed speed measurements for keypoint k in example e, - $v_{k,e,i}$ is the i-th centered, non-signed speed measurement for keypoint k in example e.

Keypoint dropping metric

Let K_{total} be the total number of keypoints in the video sequence, and K_{dropped} be the count of keypoints that are below a defined threshold $T_{\text{threshold}}$ and considered for dropping in environments with little occlusion and a top view.

We define "keypoint dropping" as the process of tracking the number of keypoints below the threshold for each video frame:

$$K_{\text{dropped}}(t) = \sum_{i=1}^{K_{\text{total}}} \delta_i(t)$$

where $K_{\text{dropped}}(t)$ is the count of keypoints dropped at time t, and $\delta_i(t)$ is an indicator function that returns 1 if the i-th keypoint is below the threshold at time t, and 0 otherwise:

$$\delta_i(t) = \begin{cases} 1, & \text{if score}_i(t) < T_{\text{threshold}} \\ 0, & \text{otherwise} \end{cases}$$

where $\text{score}_i(t)$ is the confidence score or measurement of the i-th keypoint at time t.

Statistical analysis. Linear mixed-effects models were fitted in R (75) using the `lme4` package (v1.1.31; (76)). Training data fraction (or, equivalently, the number of images) and fine-tuning methods were defined as fixed effects, whereas the various datasets and shuffles were treated as random effects; random intercepts and slopes were also added at the dataset level. The best models were selected based on the Akaike Information Criterion (AIC); adding complexity did not result in lower AIC, and even led to singular fits, indicative of overfitting. The weight of evidence for an effect was computed using likelihood ratio tests, as well as with *p*-values provided by `lmerTest` (v.3.1.3). Pairwise contrasts and Cohen's *d* standardized effect sizes were computed with the `emmeans` package (v.1.8.3), and degrees of freedom estimated with the Kenward-Roger method. Distributions of prediction errors with and without spatial-pyramid search were compared with the two-sample, one-sided (alternative hypothesis: "less") Kolmogorov-Smirnov test. The significance threshold was set at 0.05.

Behavioral Action Segmentation

As our benchmark dataset, we used the openfield test (OFT) task from Sturman et al. (15). We calculated the same skeleton-based features by concatenating 10 distances between keypoints, 6 angles, 4 body areas and two additional boolean variables coding whether the nose and head center were inside the arena, resulting in a 22D vector at each time step. For the action classifier, we used an MLP neural network as the action decoder that acted as a sliding window across 31 time steps to perform action segmentation and used F1 score on supported and unsupported rears as evaluation metrics. As in the original paper, we performed leave-one-out cross-validation on 20 videos and across 3 annotators.

For CEBRA (36), we used the model architecture 'offset10-model'. The output dimension was set to 32, as found via a simple grid search over the following values: [4, 8, 16, 32]. We trained it for 5000 iterations with batch size 4096, the Adam optimizer, and learning rate 1e-4.

Note that the original model for OFT task from Sturman et al. includes the center and four corners of the mouse cage, which is critical for their handcrafted features to determine the relative distance between the mouse and the walls. As our SuperAnimal models focus on animal bodyparts only, we take the corner coordinates from their released data for the sake of comparison. In practice, those static environmental keypoints can be provided by taking users' inputs via interactive GUI for videos.

Our SuperAnimal-Quadruped model was run on the videos from Horse-30 (16). The start (2 s) and end (2 s) of each of the 30 videos were removed from the analysis, to ignore instants when the horse is only partially seen. Front and back hoof contacts and lifts were identified using respectively peak and valley detection from the 2D kinematic traces of the front and back hooves. Beforehand, these trajectories were smoothed using a 2nd-order, low-pass, zero-lag Butterworth filter (cutoff=3 Hz) and centered on a keypoint located on the animal's back; this effectively expresses keypoint coordinates in a reference frame stationary relative to the moving horse, facilitating event detection. We extracted fore and hind limb strides between consecutive ground contacts, and stance phases between a contact of one hoof until it is lifted off the ground. Stride lengths (in pixels), stances, and the number of identified hoof contacts were then computed, and qualitatively compared to those obtained using the densely annotated (ground truth) keypoints (Figure 4g,h,i).

Code API

High-level inference API (with spatial-pyramid search) for using SuperAnimal models in DeepLabCut:

```
video_path = 'demo-video.mp4'
superanimal_name = 'superanimal_topviewmouse'
scale_list = range(200, 600, 50)  # image height pixel size range and increment

deeplabcut.video_inference_superanimal(
    [video_path],
    superanimal_name,
    scale_list=scale_list,
    video_adapt=True,  # slow but likely most accurate
)
```

Low-level API that combines spatial-pyramid search and video adaptation:

```
from  deeplabcut.modelzoo.api import SpatiotemporalAdaptation
video_path = 'demo_video.mp4'
videotype = 'mp4'
superanimal_name = 'superanimal_topviewmouse'
scale_list = range(200, 600, 50)

adapter = SpatiotemporalAdaptation(
    video_path,
    superanimal_name,
    modelfolder="weight_directory",
    scale_list=scale_list,
)
adapter.before_adapt_inference()
adapter.adaptation_training()
adapter.after_adapt_inference()
```

Web App

Many labs use DeepLabCut to define, annotate, and refine animal bodyparts, resulting in high quality, diverse keypoint annotations for animals in different contexts (10, 34). In order to enable a positive feedback loop to turn the collection of animal pose data and models into a community effort we developed a Web App.

The app is available at https://contrib.deeplabcut.org/. This app allows anyone, within their browser, to a) upload their own image and label, b) annotate community images, c) run inference of available community models on their own data, d) share models to be hosted. The website is written using JavaScript with the Svelte framework, and the models are run on cloud servers.

Data collection. The website has an upload portal for groups to upload their models and labeled data in DeepLabCut format to help grow the pre-training datasets and allow researchers to build on top of varied models and data.

Annotation. Additionally, the website hosts a labeling web app that allows users to annotate curated images. The datasets currently available for annotation are from iNaturalist (77) and the OpenImage Datase (78). After selecting which dataset to label, images are displayed successively with the target animal prominently shown in front of an opaque masked background (which can be toggled off). The keypoint set is selected taking into account the species morphology and keypoint value in subsequent analysis. Once the annotation is complete, the data is saved to the database and made available for use in further research.

Online inference. To allow testing DeepLabCut models in the browser, the user selects a few images, which model to run, and receives predictions along with confidence scores for each keypoint. Users are then able to adjust or delete keypoints, as well as download the model weights. This allows for a quick and hassle-free evaluation of DeepLabCut's capabilities and suitability for specific tasks, making it available to a wider range of users.

Extended Data Figures

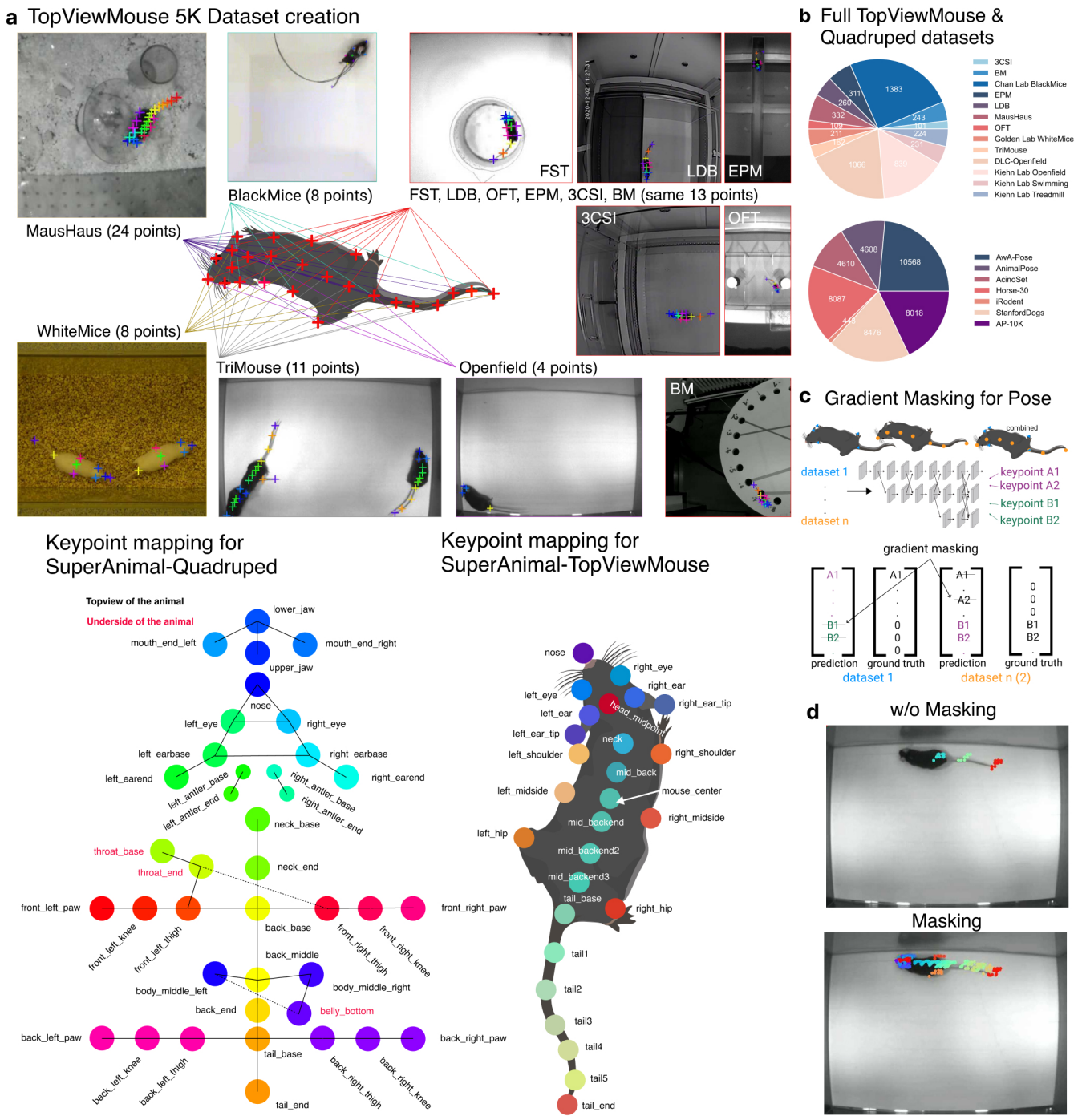

Figure S1. Constructing SuperAnimal models and keypoint gradient masking. **a**: Demonstration of how multiple pose datasets are merged into a single dataset. We created a main keypoint names to cover all keypoints we observe from datasets. Then we built a conversion table to map keypoints from each dataset to the main keypoint names. We design a corresponding conversion table such that anatomically similar keypoints are mapped to the same keypoint. Below we add the keypoint naming map for both SuperAnimal-TopViewMouse and SuperAnimal-Quadruped models. **b**: Composition of the SuperAnimal-Quadruped (left) and SuperAnimal-TopViewMouse (right) datasets. **c**: Demonstration of keypoint gradient masking algorithm. Keypoints that were not defined in the original datasets introduce false penalties for the model training. Therefore, during back-propagation, the gradients of those undefined keypoints are artificially masked. **d**: With masking, the model is able to learn a pose representation that is the union of training datasets. Without masking, the model has severe degraded pose representation.

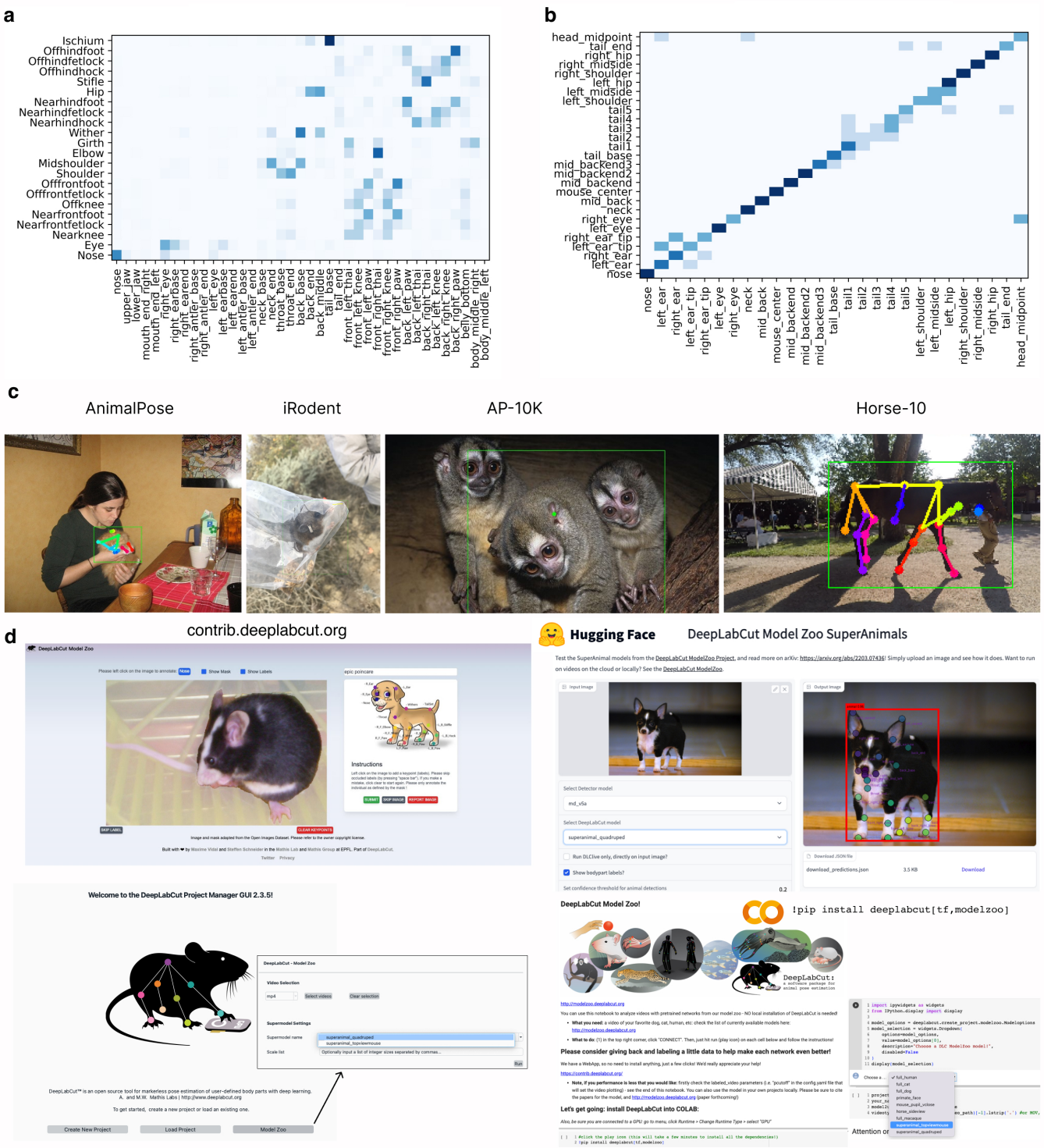

Figure S2. Dataset and WebApp considerations. a: The affinity matrix represents the semantic similarity between keypoint defined by the model and keypoint defined by dataset annotations across images. The affinity matrix is obtained by hard voting. The voting per image is obtained via pairwise euclidean distance between SuperAnimal-Quadruped model's zero-shot predictions and Horse-30 dataset ground truth. **b**: Affinity matrix for Golden Lab Mouse (see Methods) video (bottom at Figure 3), where we deliberately tried to match the keypoint space to model's zero-shot prediction. The noise in the affinity matrix suggests annotator bias for hard keypoints (e.g., tail points along the tail where the exact position is not visually concretely defined, as say opposed to the nose). For this analysis we annotated 20 frames of the Golden Lab Mouse data to illustrate our matching process. **c**: Examples of OOD failures from several datasets, as noted, from our SuperAnimal-Quadruped model. **d**: Top Left: An example of the current WebApp interface at `contrib.deeplabcut.org`. Users can add and edit the annotations from images we collect, following an anatomical figure that aids the expected location of bodyparts. Top Right: Example of current Gradio App on HuggingFace. Bottom Left: our current stand-alone GUI for local computer use showing a simple ModelZoo with SuperAnimal weights. Bottom Right: example of the Google Colaboratory interface with ModelZoo inference with SuperAnimal weights.

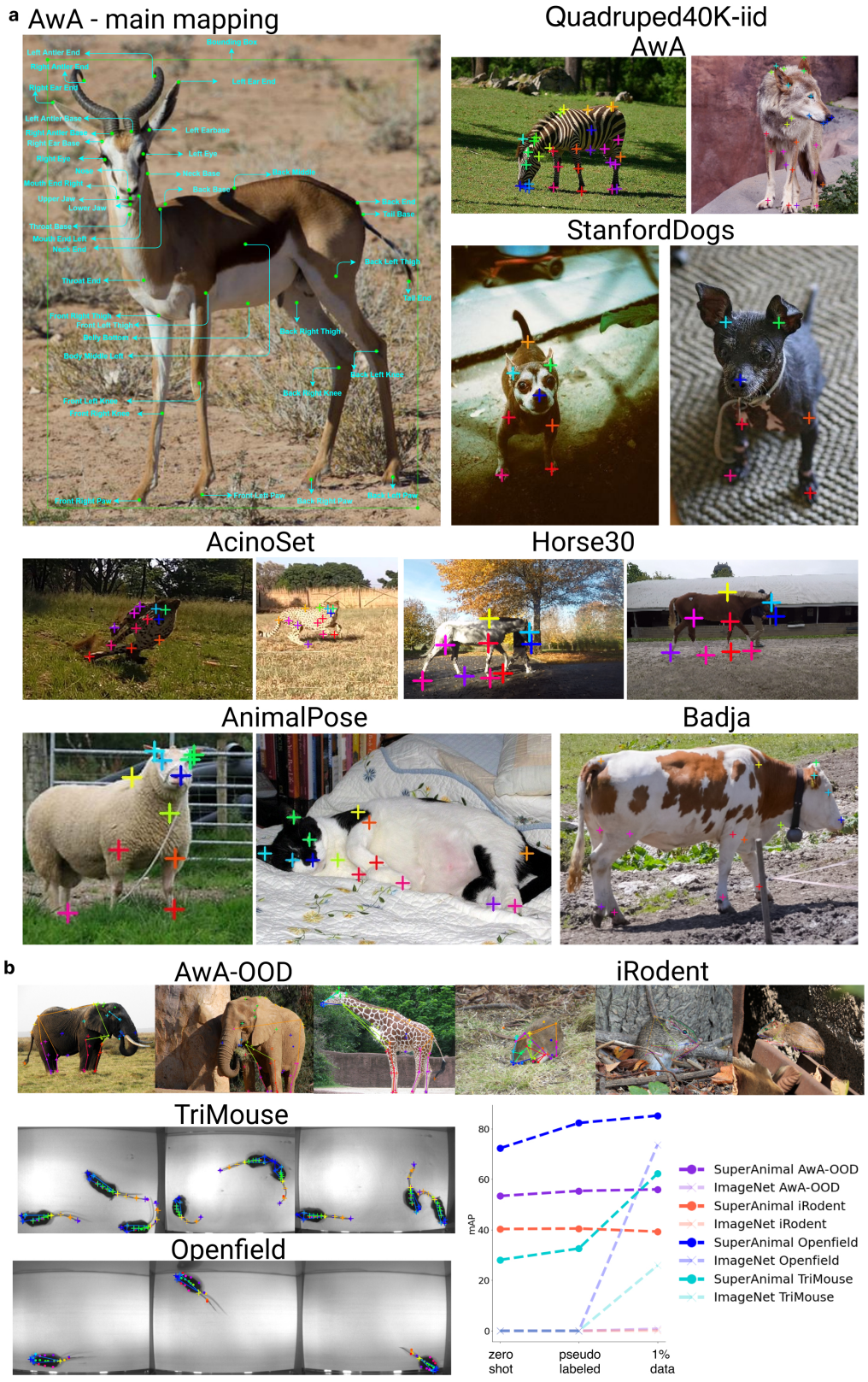

Figure S3. Quadruped dataset creation and additional performance results on held-out (OOD) data. **a**: Visual illustration of the datasets that compose Quadruped-40K (see Extended Data Figure S1 for TopViewMouse). **b**: Zero-shot examples and performance quantification. SuperAnimal models outperform ImageNet pretraining, independently of the data (image) split or framework used (here we used HRNet implemented within mmpose). Here, elephants, giraffes, iRodents, and two mice datasets were held out for model training, respectively, and still show excellent zero-shot performance (higher mAP), and show good fine-tuning with as little as 1% of added data.

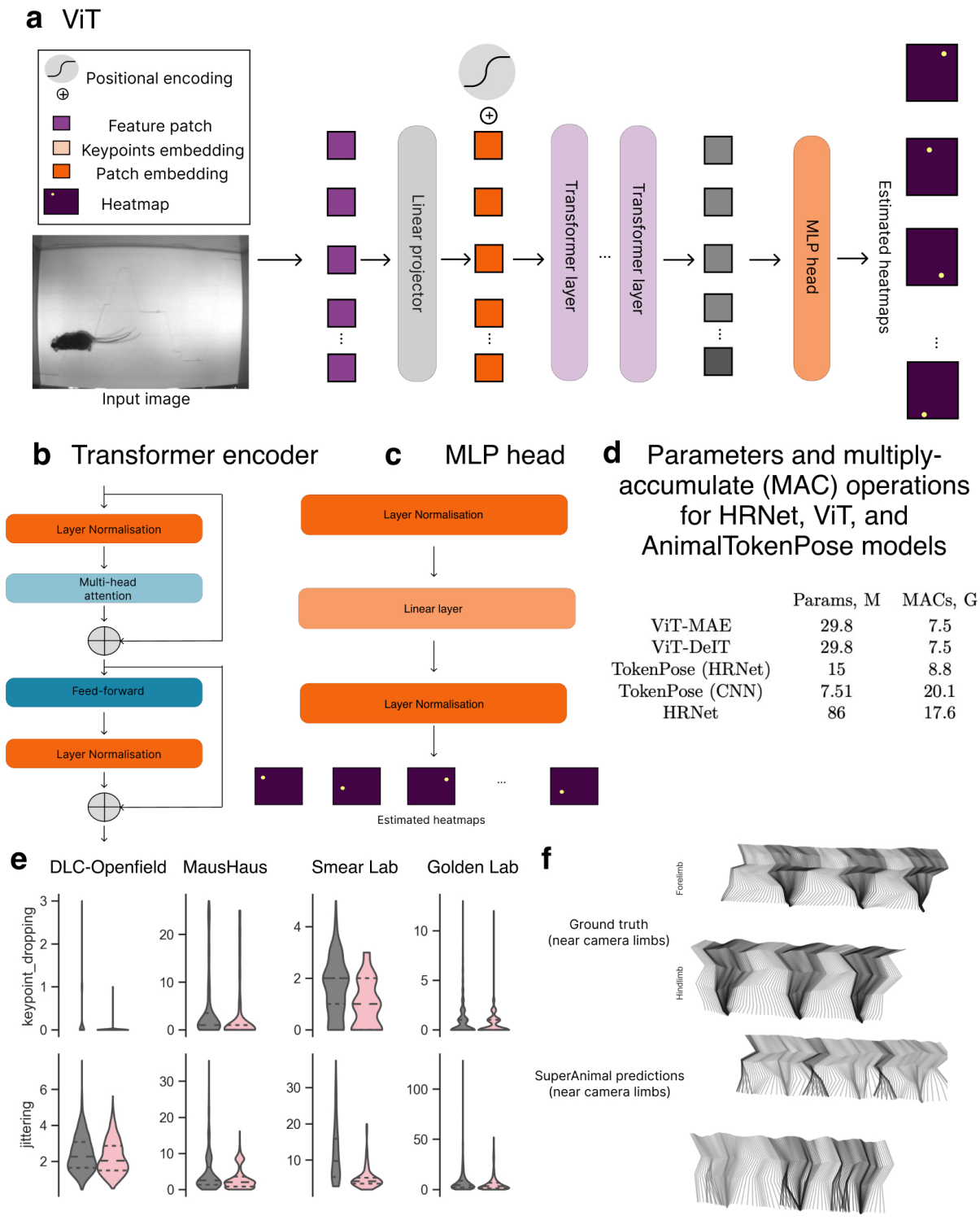

Figure S4. SuperAnimal pre-trained transformers for animal pose estimation. **a**: Visual Transformer architecture with additional MLP head for heatmap estimation. In our experiments we use ViT-Base with 12 heads, 12 layers and 786 hidden layer size. **b**: Transformer encoder architecture. **c**: MLP head architecture. **d**: The model complexity for the transformer models and HRNet. **e**: Quantification of video adaptation in terms of jitter and keypoint dropping (see Methods) across the four test videos. Grey is without, pink is with video adaptation. Quartiles are shown as dashed lines. **f**: Kinematic plots as in Figure 4g where the SuperAnimal-Quadruped tracking performance was poor.

Supplementary Information

Results Table Summary

To summarize the results in Figure 1 and Figure 2 on OOD Benchmarks, we provide two Tables:

Table S1. Main Results on Mouse Benchmarks. The mAP comparison between CNN baseline (HRNet) and Transformer based models (TokenPose models and ViTs) on SuperAnimal-TopViewMouse. The performance of HRNet is comparable to performance shown in Extended Data Fig. 3c), and transformers most-always outperformed HRNet. Here we used mmpose (79)) as well. Suggesting that users also may consider other architectures when building SuperAnimal models, especially for zero-shot inference.

Model	TriMouse	DLC-Openfield
ViT-MAE	32.4	63.6
ViT-DeIT	34.2	62.4
AnimalTokenPose (HRNet)	33.4	**91.5**
AnimalTokenPose (CNN)	**43.5**	82.7
HRNet	28.1	72.3

Table S2. Main Results on Quadruped Benchmarks. Here, the base SuperAnimal-Quadruped model had none of the heldout datasets, with the exception of AnimalPose (AP). For the AnimalPose experiments we make a new variant that dropped AnimalPose data (SA-AP), thereby making it OOD for this setting as well. Full results can be found in Figure 2 for fine-tuning with different amounts of data, but the best-case fine-tuning performance is shown, which matches the top-performance of the SuperAnimal (SA) variant as shown in Figure 2. *NOTE: Cao et al.(22) do not report a unified single mAP, rather per animal, therefore we trained a model using their dataset to estimate top-line performance if only trained on AP. **Number as reported in (39) using the data from (30).

Benchmark:	AnimalTokenPose (HRNet)	HRNet (w32)
mAP (↑, higher the better)		
iRodent: SA zero-shot	36.2	39.3
iRodent: fine-tuning ImageNet w/1%	-	02.8
iRodent: fine-tuning SA w/1%	-	40.4
iRodent: fine-tuning ImageNet w/100%	-	57.0
iRodent: fine-tuning SA w/100%	-	68.0
AP-10K: SA zero-shot	45.3	50.8
AP-10K: AnimalPose weights, zero-shot	-	27.0
AP-10K: ** (39)		
fine-tuning w/ImageNet	-	72.2
AP-10K: fine-tuning w/SA	-	78.3
AnimalPose: SA-AP zero-shot	86.0	85.4
AnimalPose: SA+AP-10K zero-shot	-	69.4
AnimalPose: Fine-tuning ImageNet*	-	88.2
AnimalPose: SA-AP Fine-tuning	-	90.4
Normalized Error (↓, lower the better)		
Horse10: IID SA zero-shot	0.647	0.673
Horse10: OOD SA zero-shot	0.613	0.640
Horse10: IID SA fine-tuning ImageNet	-	0.049
Horse10: OOD SA fine-tuning ImageNet	-	0.179
Horse10: IID SA fine-tuning	-	0.047
Horse10: OOD SA fine-tuning	-	0.109

How to use the DeepLabCut Model Zoo

The DeepLabCut Model Zoo consists of two parts. The first is a web-based platform that accepts pose data contributions, ranging from a DeepLabCut project, labeled images from our WebApp, and public animal pose datasets (See Figure S2d). As these data come in different formats, we implement a software-based data layer dubbed “generalized data converter" (see

Methods) that convert data of various forms to DeepLabCut pose format. We call models we provide Super-Animal models for their generalization powers. After users download these super models from our website or via DeepLabCut APIs, they can either use the models as a plug-and-play solution or alternatively choose to adapt or fine-tune these models from videos or pose datasets.

Considerations on building general datasets for pretraining

To build generalizable pose models, a large-scale pre-training dataset is the key. It has been shown in both computer vision and natural language processing that pre-trained models significantly improve the generalization of models and data efficiency in the downstream datasets (63, 80). However, data of lab animals are not ubiquitous on the internet. To get large scale animal pose data, it is critical to gather the data directly from the research community in a responsible and transparent way. A platform that actively interacts with the community is thus required to build such a pre-training dataset. As such a vocabulary is built on top of a wide range of pose datasets, it can be used across different research needs and it is also key to for useful zero-shot inference (see Methods).

We acknowledge that these SuperAnimal models would not have been be possible without the accumulated data from the community. In the future, feedback from the community for models' efficacy and failure modes (Extended Data Fig. S2) in different downstream data will be critical for updated model releases and algorithmic updates. As publicly available data increase, we expect the performance will improve.

Annotator bias in labeled data. Unlike previous works that require labeling data to create a working model, our models can be used as they are. For the purpose of evaluation, we could use the ground-truth of the target dataset or label frames of a novel video. We note when it comes to evaluating the performance of zero-shot inference, there will always be a systematic errors between the model and the annotator of the target dataset. We refer this type of error to be caused annotator bias, meaning annotators of different datasets try to place keypoints in slightly different places due to the bias of annotators. Therefore, the supervised metrics will tend to be an over-estimation of the error.

Reversely, SuperAnimal models can be used to monitor annotator bias as the model's predictions are consistent across frames while in many cases human annotators annotate keypoints in a inconsistent way.

Supervised metrics do not capture the richness of SuperAnimal-models

In pose estimation literature, work mostly report supervised metrics (RMSE, Normalized Error, and mAP). What is shared in the supervised metrics is that the metrics do not penalize keypoints that are not annotated in the dataset. In contrast to other pose models, our SuperAnimal models can predict keypoints that are not annotated in the labeled dataset. For instance, if we apply only supervised metrics to evaluate SuperAnimal models, catastrophic forgetting is not detected as metrics do not penalize keypoint predictions that are not annotated.

Video captions

Suppl_video1.mp4 Video prediction results by comparison model trained with and without gradient masking.

Suppl_video2.mp4 Video prediction results by SuperAnimal model fine-tuned with naive-fine-tuning and SuperAnimal model fine-tuned with memory replay.

Suppl_video3.mp4 Video prediction results by SuperAnimal-TopViewMouse model with and without spatial pyramid inference. Note that because we use a detector for the SuperAnimal-Quadruped, this is not needed.

Suppl_video4.mp4 Video prediction results by SuperAnimal models with and without video adaptation.

Suppl_video5.mov Example video from Sturman et al (15) vs. SuperAnimal-TopViewMouse without any training.

Tables

The inclusion of a fixed effect for the fine-tuning methods in the mixed model formula was statistically significant ($\chi^2(2, N = 5) = 10.4, p = .006$). The model was further improved by adding an interaction term between method and training data fraction ($\chi^2(12, N = 17) = 51.4, p < .0001$) and random slopes at the dataset level ($\chi^2(14, N = 31) = 46.5, p < .0001$).

Table S3. Linear mixed models goodness-of-fit.
model_1: $rmse \sim (1|dataset)$
model_2: $rmse \sim method + (1|dataset)$
model_3: $rmse \sim method * frac + (1|dataset)$
model_4: $rmse \sim method * frac + (1 + frac|dataset)$
model_5: $rmse \sim method * frac + (1 + frac|dataset) + (1|shuffle)$

	npar	AIC	BIC	logLik	deviance	Chisq	Df	Pr(>Chisq)
model_1	3	1509.83	1518.55	-751.92	1503.83			
model_2	5	1503.43	1517.96	-746.72	1493.43	10.40	2	0.0055
model_3	17	1476.09	1525.48	-721.04	1442.09	51.35	12	0.0000
model_4	31	1457.60	1547.66	-697.80	1395.60	46.49	14	0.0000
model_5	32	1459.60	1552.57	-697.80	1395.60	0.00	1	0.9999

Table S4. Pairwise contrasts adjusted with Tukey's method. MR: memory replay; TL; transfer learning.

	contrast	frac	estimate	SE	df	lower.CL	upper.CL	t.ratio	p.value
1	TL - MR	0.01	132.76	20.10	137.65	85.13	180.39	6.60	<.0001
2	TL - zero-shot	0.01	123.35	20.10	137.65	75.72	170.98	6.14	<.0001
3	MR - zero-shot	0.01	-9.41	20.10	137.65	-57.04	38.22	-0.47	0.89
4	TL - MR	0.05	39.58	20.10	137.65	-8.05	87.21	1.97	0.12
5	TL - zero-shot	0.05	22.07	20.10	137.65	-25.56	69.70	1.10	0.52
6	MR - zero-shot	0.05	-17.51	20.10	137.65	-65.14	30.12	-0.87	0.66
7	TL - MR	0.1	31.58	20.10	137.65	-16.05	79.21	1.57	0.26
8	TL - zero-shot	0.1	5.97	20.10	137.65	-41.66	53.60	0.30	0.95
9	MR - zero-shot	0.1	-25.61	20.10	137.65	-73.24	22.03	-1.27	0.41
10	TL - MR	0.5	1.40	20.10	137.65	-46.23	49.03	0.07	1.00
11	TL - zero-shot	0.5	-26.84	20.10	137.65	-74.47	20.80	-1.33	0.38
12	MR - zero-shot	0.5	-28.23	20.10	137.65	-75.86	19.40	-1.40	0.34
13	TL - MR	1.0	-2.70	20.10	137.65	-50.33	44.93	-0.13	0.99
14	TL - zero-shot	1.0	-33.15	20.10	137.65	-80.78	14.49	-1.65	0.23
15	MR - zero-shot	1.0	-30.44	20.10	137.65	-78.07	17.19	-1.51	0.29

Table S5. Type-III Analysis of Variance Table for Horse-10 IID mixed model.

	Sum Sq	Mean Sq	NumDF	DenDF	F value	Pr(>F)
method	0.26	0.09	3.00	38.00	12.28	<0.0001
n	5.73	1.43	4.00	38.00	201.94	<0.0001
method:n	0.39	0.03	12.00	38.00	4.61	0.0001

Table S6. Pairwise contrasts adjusted with Tukey's method for the Horse-10 IID mixed model. MR: memory replay; TL: transfer learning; NF: naive fine-tuning; RID: randomly initialized decoder.

contrast	estimate	SE	df	lower.CL	upper.CL	t.ratio	p.value
n = 14							
TL - MR	0.3866	0.0688	38	0.2018	0.5714	5.620	<.0001
TL - NF	0.3843	0.0688	38	0.1995	0.5692	5.587	<.0001
TL - RID	0.0156	0.0688	38	-0.1692	0.2004	0.227	0.9958
MR - NF	-0.0023	0.0688	38	-0.1871	0.1826	-0.033	1.0000
MR - RID	-0.3710	0.0688	38	-0.5558	-0.1862	-5.393	<.0001
NF - RID	-0.3687	0.0688	38	-0.5536	-0.1839	-5.360	<.0001
n = 73							
TL - MR	0.2996	0.0688	38	0.1148	0.4844	4.355	0.0005
TL - NF	0.2959	0.0688	38	0.1111	0.4807	4.301	0.0006
TL - RID	0.1145	0.0688	38	-0.0703	0.2993	1.664	0.3562
MR - NF	-0.0037	0.0688	38	-0.1885	0.1811	-0.054	0.9999
MR - RID	-0.1851	0.0688	38	-0.3699	-0.0003	-2.691	0.0495
NF - RID	-0.1814	0.0688	38	-0.3662	0.0034	-2.637	0.0560
n = 146							
TL - MR	0.1214	0.0688	38	-0.0634	0.3063	1.765	0.3055
TL - NF	0.1114	0.0688	38	-0.0734	0.2962	1.619	0.3803
TL - RID	0.0602	0.0688	38	-0.1246	0.2450	0.875	0.8177
MR - NF	-0.0101	0.0688	38	-0.1949	0.1748	-0.146	0.9989
MR - RID	-0.0613	0.0688	38	-0.2461	0.1236	-0.890	0.8099
NF - RID	-0.0512	0.0688	38	-0.2360	0.1336	-0.744	0.8785
n = 734							
TL - MR	-0.0360	0.0688	38	-0.2208	0.1488	-0.523	0.9531
TL - NF	-0.0328	0.0688	38	-0.2176	0.1520	-0.476	0.9638
TL - RID	-0.0051	0.0688	38	-0.1899	0.1797	-0.074	0.9998
MR - NF	0.0032	0.0688	38	-0.1816	0.1880	0.046	1.0000
MR - RID	0.0309	0.0688	38	-0.1540	0.2157	0.448	0.9695
NF - RID	0.0277	0.0688	38	-0.1572	0.2125	0.402	0.9777
n = 1469							
TL - MR	-0.0290	0.0688	38	-0.2138	0.1558	-0.421	0.9745
TL - NF	-0.0222	0.0688	38	-0.2070	0.1626	-0.322	0.9882
TL - RID	-0.0002	0.0688	38	-0.1850	0.1846	-0.003	1.0000
MR - NF	0.0068	0.0688	38	-0.1780	0.1916	0.099	0.9996
MR - RID	0.0288	0.0688	38	-0.1561	0.2136	0.418	0.9751
NF - RID	0.0220	0.0688	38	-0.1629	0.2068	0.319	0.9886

Table S7. Type-III Analysis of Variance Table for Horse-10 OOD mixed model.

	Sum Sq	Mean Sq	NumDF	DenDF	F value	Pr(>F)
method	0.79	0.26	3.00	38.00	54.05	<0.0001
n	4.93	1.23	4.00	38.00	254.52	<0.0001
method:n	0.82	0.07	12.00	38.00	14.10	<0.0001

Table S8. Pairwise contrasts adjusted with Tukey's method for the Horse-10 OOD mixed model. MR: memory replay; TL: transfer learning; NF: naive fine-tuning; RID: randomly initialized decoder.

contrast	estimate	SE	df	lower.CL	upper.CL	t.ratio	p.value
n = 14							
TL - MR	0.5559	0.0568	38	0.4033	0.7086	9.784	<.0001
TL - NF	0.5734	0.0568	38	0.4208	0.7261	10.092	<.0001
TL - RID	-0.0215	0.0568	38	-0.1741	0.1312	-0.378	0.9814
MR - NF	0.0175	0.0568	38	-0.1352	0.1702	0.308	0.9897
MR - RID	-0.5774	0.0568	38	-0.7301	-0.4247	-10.161	<.0001
NF - RID	-0.5949	0.0568	38	-0.7476	-0.4422	-10.469	<.0001
n = 73							
TL - MR	0.4726	0.0568	38	0.3200	0.6253	8.318	<.0001
TL - NF	0.4503	0.0568	38	0.2976	0.6029	7.924	<.0001
TL - RID	0.1137	0.0568	38	-0.0390	0.2663	2.001	0.2057
MR - NF	-0.0224	0.0568	38	-0.1750	0.1303	-0.393	0.9790
MR - RID	-0.3590	0.0568	38	-0.5116	-0.2063	-6.317	<.0001
NF - RID	-0.3366	0.0568	38	-0.4893	-0.1840	-5.924	<.0001
n = 146							
TL - MR	0.2025	0.0568	38	0.0499	0.3552	3.565	0.0053
TL - NF	0.1982	0.0568	38	0.0455	0.3509	3.488	0.0066
TL - RID	0.0468	0.0568	38	-0.1058	0.1995	0.824	0.8428
MR - NF	-0.0043	0.0568	38	-0.1570	0.1483	-0.077	0.9998
MR - RID	-0.1557	0.0568	38	-0.3084	-0.0031	-2.741	0.0441
NF - RID	-0.1514	0.0568	38	-0.3040	0.0013	-2.664	0.0527
n = 734							
TL - MR	0.0103	0.0568	38	-0.1424	0.1629	0.180	0.9979
TL - NF	0.0046	0.0568	38	-0.1481	0.1572	0.081	0.9998
TL - RID	0.0066	0.0568	38	-0.1460	0.1593	0.116	0.9994
MR - NF	-0.0057	0.0568	38	-0.1583	0.1470	-0.100	0.9996
MR - RID	-0.0036	0.0568	38	-0.1563	0.1490	-0.064	0.9999
NF - RID	0.0020	0.0568	38	-0.1506	0.1547	0.036	1.0000
n = 1469							
TL - MR	-0.0336	0.0568	38	-0.1863	0.1190	-0.592	0.9339
TL - NF	-0.0232	0.0568	38	-0.1759	0.1294	-0.409	0.9766
TL - RID	-0.0152	0.0568	38	-0.1679	0.1375	-0.267	0.9932
MR - NF	0.0104	0.0568	38	-0.1422	0.1631	0.183	0.9978
MR - RID	0.0184	0.0568	38	-0.1342	0.1711	0.324	0.9880
NF - RID	0.0080	0.0568	38	-0.1446	0.1607	0.141	0.9990

Table S9. Type-III Analysis of Variance Table for DLC-Openfield mixed model.

	Sum Sq	Mean Sq	NumDF	DenDF	F value	Pr(>F)
method	137.15	45.72	3.00	38.00	9.29	0.0001
n	1049.81	262.45	4.00	38.00	53.31	<0.0001
method:n	197.61	16.47	12.00	38.00	3.34	0.0022

Table S10. Pairwise contrasts adjusted with Tukey's method for the Openfield mixed model. MR: memory replay; TL: transfer learning; NF: naive fine-tuning; RID: randomly initialized decoder.

contrast	estimate	SE	df	lower.CL	upper.CL	t.ratio	p.value
n = 10							
TL - MR	10.4477	1.8117	38	5.5806	15.3147	5.767	<.0001
TL - NF	6.5661	1.8117	38	1.6990	11.4331	3.624	0.0045
TL - RID	0.7263	1.8117	38	-4.1408	5.5933	0.401	0.9779
MR - NF	-3.8816	1.8117	38	-8.7486	0.9855	-2.143	0.1582
MR - RID	-9.7214	1.8117	38	-14.5885	-4.8543	-5.366	<.0001
NF - RID	-5.8398	1.8117	38	-10.7069	-0.9728	-3.223	0.0133
n = 50							
TL - MR	6.4436	1.8117	38	1.5766	11.3107	3.557	0.0054
TL - NF	4.5692	1.8117	38	-0.2978	9.4363	2.522	0.0724
TL - RID	-0.0791	1.8117	38	-4.9462	4.7879	-0.044	1.0000
MR - NF	-1.8744	1.8117	38	-6.7415	2.9927	-1.035	0.7304
MR - RID	-6.5228	1.8117	38	-11.3898	-1.6557	-3.600	0.0048
NF - RID	-4.6484	1.8117	38	-9.5154	0.2187	-2.566	0.0658
n = 101							
TL - MR	2.1848	1.8117	38	-2.6823	7.0518	1.206	0.6269
TL - NF	2.8322	1.8117	38	-2.0348	7.6993	1.563	0.4112
TL - RID	1.3784	1.8117	38	-3.4886	6.2455	0.761	0.8714
MR - NF	0.6474	1.8117	38	-4.2196	5.5145	0.357	0.9841
MR - RID	-0.8064	1.8117	38	-5.6734	4.0607	-0.445	0.9702
NF - RID	-1.4538	1.8117	38	-6.3209	3.4133	-0.802	0.8528
n = 506							
TL - MR	-0.5671	1.8117	38	-5.4342	4.2999	-0.313	0.9892
TL - NF	-0.3719	1.8117	38	-5.2390	4.4952	-0.205	0.9969
TL - RID	-0.2698	1.8117	38	-5.1369	4.5972	-0.149	0.9988
MR - NF	0.1952	1.8117	38	-4.6718	5.0623	0.108	0.9995
MR - RID	0.2973	1.8117	38	-4.5698	5.1643	0.164	0.9984
NF - RID	0.1021	1.8117	38	-4.7650	4.9691	0.056	0.9999
n = 1012							
TL - MR	-0.7258	1.8117	38	-5.5928	4.1413	-0.401	0.9779
TL - NF	-0.6376	1.8117	38	-5.5046	4.2295	-0.352	0.9848
TL - RID	-0.4115	1.8117	38	-5.2785	4.4556	-0.227	0.9958
MR - NF	0.0882	1.8117	38	-4.7788	4.9553	0.049	1.0000
MR - RID	0.3143	1.8117	38	-4.5527	5.1814	0.174	0.9981
NF - RID	0.2261	1.8117	38	-4.6410	5.0932	0.125	0.9993

Table S11. Type-III Analysis of Variance Table for iRodent mixed model.

	Sum Sq	Mean Sq	NumDF	DenDF	F value	Pr(>F)
method	25.70	8.57	3.00	38.00	15.68	<0.0001
n	11.17	2.79	4.00	38.00	5.11	0.0021
method:n	11.08	0.92	12.00	38.00	1.69	0.1082

Table S12. Pairwise contrasts adjusted with Tukey's method for the iRodent mixed model. MR: memory replay; TL: transfer learning; NF: naive fine-tuning; RID: randomly initialized decoder.

contrast	estimate	SE	df	lower.CL	upper.CL	t.ratio	p.value
n = 3							
TL - MR	1.9452	0.6035	38	0.3240	3.5664	3.223	0.0133
TL - NF	2.0036	0.6035	38	0.3824	3.6248	3.320	0.0103
TL - RID	0.0362	0.6035	38	-1.5851	1.6574	0.060	0.9999
MR - NF	0.0584	0.6035	38	-1.5628	1.6796	0.097	0.9997
MR - RID	-1.9090	0.6035	38	-3.5302	-0.2878	-3.163	0.0155
NF - RID	-1.9675	0.6035	38	-3.5887	-0.3462	-3.260	0.0121
n = 17							
TL - MR	1.4040	0.6035	38	-0.2172	3.0253	2.327	0.1098
TL - NF	1.2256	0.6035	38	-0.3957	2.8468	2.031	0.1948
TL - RID	-1.5161	0.6035	38	-3.1373	0.1051	-2.512	0.0740
MR - NF	-0.1785	0.6035	38	-1.7997	1.4427	-0.296	0.9909
MR - RID	-2.9202	0.6035	38	-4.5414	-1.2990	-4.839	0.0001
NF - RID	-2.7417	0.6035	38	-4.3629	-1.1205	-4.543	0.0003
n = 35							
TL - MR	0.6765	0.6035	38	-0.9448	2.2977	1.121	0.6791
TL - NF	0.6196	0.6035	38	-1.0016	2.2408	1.027	0.7349
TL - RID	-1.2197	0.6035	38	-2.8409	0.4016	-2.021	0.1983
MR - NF	-0.0568	0.6035	38	-1.6780	1.5644	-0.094	0.9997
MR - RID	-1.8961	0.6035	38	-3.5173	-0.2749	-3.142	0.0164
NF - RID	-1.8393	0.6035	38	-3.4605	-0.2181	-3.048	0.0209
n = 177							
TL - MR	-0.0076	0.6035	38	-1.6288	1.6136	-0.013	1.0000
TL - NF	0.0444	0.6035	38	-1.5768	1.6656	0.074	0.9999
TL - RID	-0.6193	0.6035	38	-2.2405	1.0019	-1.026	0.7352
MR - NF	0.0520	0.6035	38	-1.5692	1.6733	0.086	0.9998
MR - RID	-0.6117	0.6035	38	-2.2329	1.0095	-1.014	0.7425
NF - RID	-0.6637	0.6035	38	-2.2849	0.9575	-1.100	0.6918
n = 354							
TL - MR	-0.1036	0.6035	38	-1.7248	1.5176	-0.172	0.9982
TL - NF	-0.0871	0.6035	38	-1.7083	1.5341	-0.144	0.9989
TL - RID	-0.7305	0.6035	38	-2.3517	0.8907	-1.211	0.6241
MR - NF	0.0165	0.6035	38	-1.6047	1.6377	0.027	1.0000
MR - RID	-0.6269	0.6035	38	-2.2482	0.9943	-1.039	0.7279
NF - RID	-0.6434	0.6035	38	-2.2647	0.9778	-1.066	0.7118

Table S13. Exact two-sample one-sided Kolmogorov-Smirnov test.

	D	*p*
Smear Lab mouse	1	<.0001
ood_ Mathis MausHaus	0.33	.27
Golden Lab Mouse	0.75	<.0001

Table S14. Type-III Analysis of Variance Table for OFT linear mixed effect model.

	Sum Sq	Mean Sq	NumDF	DenDF	F value	Pr(>F)
method	0.1371	0.0457	3	453	0.9988	0.3932
action	12.1056	12.1056	1	453	264.6151	<0.0001
method:action	0.0202	0.0067	3	453	0.1474	0.9313