A Primer on Motion Capture with Deep Learning: Principles, Pitfalls and Perspectives

Alexander Mathis[1,2,3,5,*], **Steffen Schneider**[3,4], **Jessy Lauer**[1,2,3], **and Mackenzie W. Mathis**[1,2,3,*]

[1]Center for Neuroprosthetics, Center for Intelligent Systems, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland
[2]Brain Mind Institute, School of Life Sciences, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland
[3]The Rowland Institute at Harvard, Harvard University, Cambridge, MA USA
[4]University of Tübingen & International Max Planck Research School for Intelligent Systems, Germany
[5]lead Contact
[*]Corresponding authors: alexander.mathis@epfl.ch, mackenzie.mathis@epfl.ch

Extracting behavioral measurements non-invasively from video is stymied by the fact that it is a hard computational problem. Recent advances in deep learning have tremendously advanced predicting posture from videos directly, which quickly impacted neuroscience and biology more broadly. In this primer we review the budding field of motion capture with deep learning. In particular, we will discuss the principles of those novel algorithms, highlight their potential as well as pitfalls for experimentalists, and provide a glimpse into the future.

Introduction

The pursuit of methods to robustly and accurately measure animal behavior is at least as old as the scientific study of behavior itself (1). Trails of hominid footprints, "motion" captured by pliocene deposits at Laetoli that date to 3.66 million years ago, firmly established that early hominoids achieved an upright, bipedal and free-striding gait (2). Beyond fossilized locomotion, behavior can now be measured in a myriad of ways: from GPS trackers, videography, to microphones, to tailored electronic sensors (3–5). Videography is perhaps the most general and widely-used method as it allows noninvasive, high-resolution observations of behavior (6–8). Extracting behavioral measures from video poses a challenging computational problem. Recent advances in deep learning have tremendously simplified this process (9, 10), which quickly impacted neuroscience (10, 11).

In this primer we review markerless (animal) motion capture with deep learning. In particular, we review principles of algorithms, highlight their potential, as well as discuss pitfalls for experimentalists, and compare them to alternative methods (inertial sensors, markers, etc.). Throughout, we also provide glossaries of relevant terms from deep learning and hardware. Furthermore, we will discuss how to use them, what pitfalls to avoid, and provide perspectives on what we believe will and should happen next.

What do we mean by "markerless motion capture?" While biological movement can also be captured by dense, or surface models (10, 12, 13), here we will almost exclusively focus on "keypoint-based pose estimation." Human and many other animal's motion is determined by the geometric structures formed by several pendulum-like motions of the extremities relative to a joint (6). Seminal psychophysics studies by Johansson showed that just a few coherently moving keypoints are sufficient to be perceived as human motion (6). This empirically highlights why pose estimation is a great summary of such video data. Which keypoints should be extracted, of course, dramatically depends on the model organism and the goal of the study; e.g., many are required for dense, 3D models (12–14), while a single point can suffice for analyzing some behaviors (10). One of the great advantages of deep learning based methods is that they are very flexible, and the user can define what should be tracked.

Principles of deep learning methods for markerless motion capture

In raw video we acquire a collection of pixels that are static in their location and have varying value over time. For analyzing behavior, this representation is sub-optimal: Instead, we are interested in properties of objects in the images, such as location, scale and orientation. Objects are collections of pixels in the video moving or being changed in conjunction. By decomposing objects into *keypoints* with semantic meaning —such as body parts in videos of human or animal subjects—a high dimensional video signal can be converted into a collection of time series describing the movement of each keypoint (Figure 1). Compared to raw video, this representation is easy to analyze, and semantically meaningful for investigating behavior and addressing the original research question for which the data has been recorded.

Motion capture systems aim to infer keypoints from videos: In marker-based systems, this can be achieved by manually enhancing parts of interest (by colors, LEDs, reflective markers), which greatly simplifies the computer vision challenge, and then using classical computer vision tools to extract these keypoints. Markerless pose estimation algorithms directly map raw video input to these coordinates. The conceptual difference between marker-based and marker-less approaches is that the former requires special preparation or equipment, while the latter can even be applied *post-hoc*, but

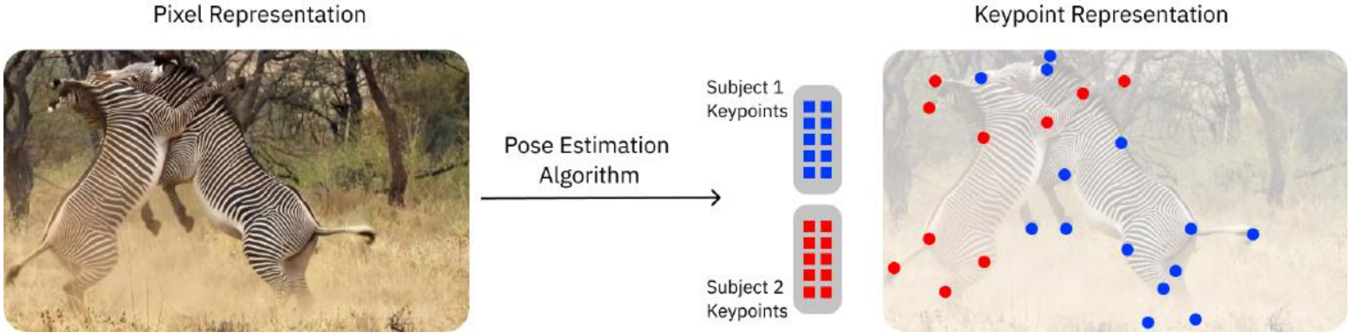


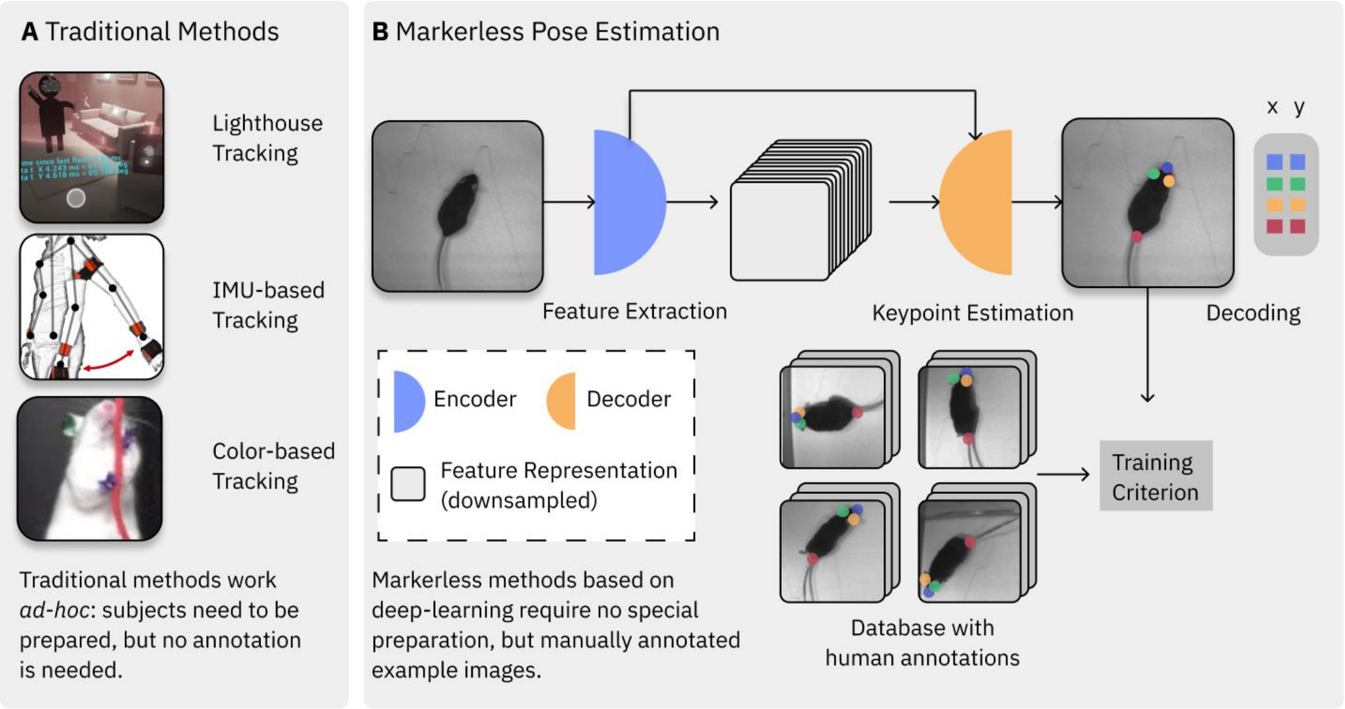
Figure 1. Schematic overview of markerless motion capture or pose estimation. The pixel representation of an image (left) or sequence of images (video) is processed and converted into a list of keypoints (right). Semantic information about object identity and keypoint type is associated to the predictions. For instance, the keypoints are structures with a name e.g. ear, the x and y coordinate as well as a confidence readout of the network (often this is included, but not for all pose estimation packages), and are then grouped according to individuals (subjects).

typically requires ground truth annotations of example images (i.e., a training set). Notably, markerless methods allow for extracting additional keypoints at a later stage, something that is not possible with markers (Figure 2).

Fundamentally, a pose estimation algorithm can be viewed as a function that maps frames from a video into the coordinates of body parts. The algorithms are highly flexible with regard to what body parts are tracked. Typically the identity of the body parts (or objects) have semantically defined meaning (e.g., different finger knuckles, the head), and the algorithms can group them accordingly (namely, to assemble an individual) so that the posture of multiple individuals can be extracted simultaneously (Figure 1). For instance, for an image of one human the algorithm would return a list of pixel coordinates (these can have subpixel resolution) per body part and frame (and sometimes an uncertainty prediction; 18–20). Which body parts the algorithm returns depends on both the application and the training data provided—this is an important aspect with respect to how the algorithms can be customized for applications.

Overview of algorithms.

While many pose estimation algorithms (21, 22) have been proposed, algorithms based on deep learning (23) are the most powerful as measured by performance on human pose estimation benchmarks (18, 24–29). More generally, pose es-



Traditional methods work *ad-hoc*: subjects need to be prepared, but no annotation is needed.

Markerless methods based on deep-learning require no special preparation, but manually annotated example images.

Figure 2. Comparison of marker-based (traditional) and markerless tracking approaches. (A) In marker-based tracking, *prior* to performing an experiment, special measures have to be taken regarding hardware and preparation of the subject (images adapted from 15, 16; IMU stands for inertial measurement unit). **(B)** For markerless pose estimation, raw video is acquired and processed post-hoc: Using labels from human annotators, machine learning models are trained to infer keypoint representations directly from video (on-line inference without markers is also possible (17)). Typically, the architectures underlying pose estimation can be divided into a feature extractor and a decoder: The former maps the image representation into a feature space, the latter infers keypoint locations given this feature representation. In modern deep learning systems, both parts of the systems are trained end-to-end.

timation algorithms fall under "object detection", a field that has seen tremendous advances with deep learning (aptly reviewed in Wu et al., 9). In brief, pose estimation can often intuitively be understood as a system of an encoder that extracts important (visual) features from the frame, which are then used by the decoder to predict the body parts of interests along with their location in the image frame.

In classical algorithms (see 9, 21, 22), handcrafted feature representations are used that extract invariant statistical descriptions from images. These features were then used together with a classifier (decoder) for detecting complex objects like humans (21, 30). Handcrafted feature representations are (loosely) inspired by neurons in the visual pathway and are designed to be robust to changes in illumination, and translations; typical feature representations are Scale Invariant Feature Transform (SIFT; 31), Histogram of Gradients (HOG; 30) or Speeded Up Robust Features (SURF; 32).

In more recent approaches, both the encoder and decoders (alternatively called the backbone and output heads, respectively) are deep neural networks (DNN) that are directly optimized on the pose estimation task. An optimal strategy for pose estimation is jointly learning representations of the raw image or video data (encoder) and a predictive model for posture (decoder). In practice, this is achieved by concatenating multiple layers of differentiable, non-linear transformations and by training such a model as a whole using the back propagation algorithm (9, 23, 33). In contrast to classical approaches, DNN based approaches directly optimize the feature representation in a way most suitable for the task at hand (For a glossary of deep learning terms see Box 1).

Machine learning systems are composed of a dataset, model, loss function (criterion) and optimization algorithm (33). The dataset defines the input-output relationships that the model should learn: In pose estimation, a particular pose (output) should be predicted for a particular image (input), see Figures 1 & 2B. The model's parameters (weights) are iteratively updated by the optimizer to minimize the loss function. Thereby the loss function measures the quality of a predicted pose (in comparison to the ground truth data). Choices about these four parts influence the final performance and behavior of the pose-estimation system and we discuss possible design choices in the next sections.

Datasets & Data Augmentation.

Two kinds of datasets are relevant for training pose estimation systems: First, one or multiple datasets used for related tasks—such as image recognition—can be used for *pre-training* computer vision models on this task (also known as transfer learning; see Box 1). This dataset is typically considerably larger than the one used for pose estimation. For example, ImageNet (34), sometimes denoted as ImageNet-21K, is a highly influential dataset and a subset was used for the ImageNet Large Scale Visual Recognition Challenge in 2012 (ILSRC-2012; 35) for object recognition. Full ImageNet contains 14.2 million images from 21K classes, the ILSRC-2012 subset contains 1.2 million images of 1,000 different classes (such as car, chair, etc; 35). Groups working

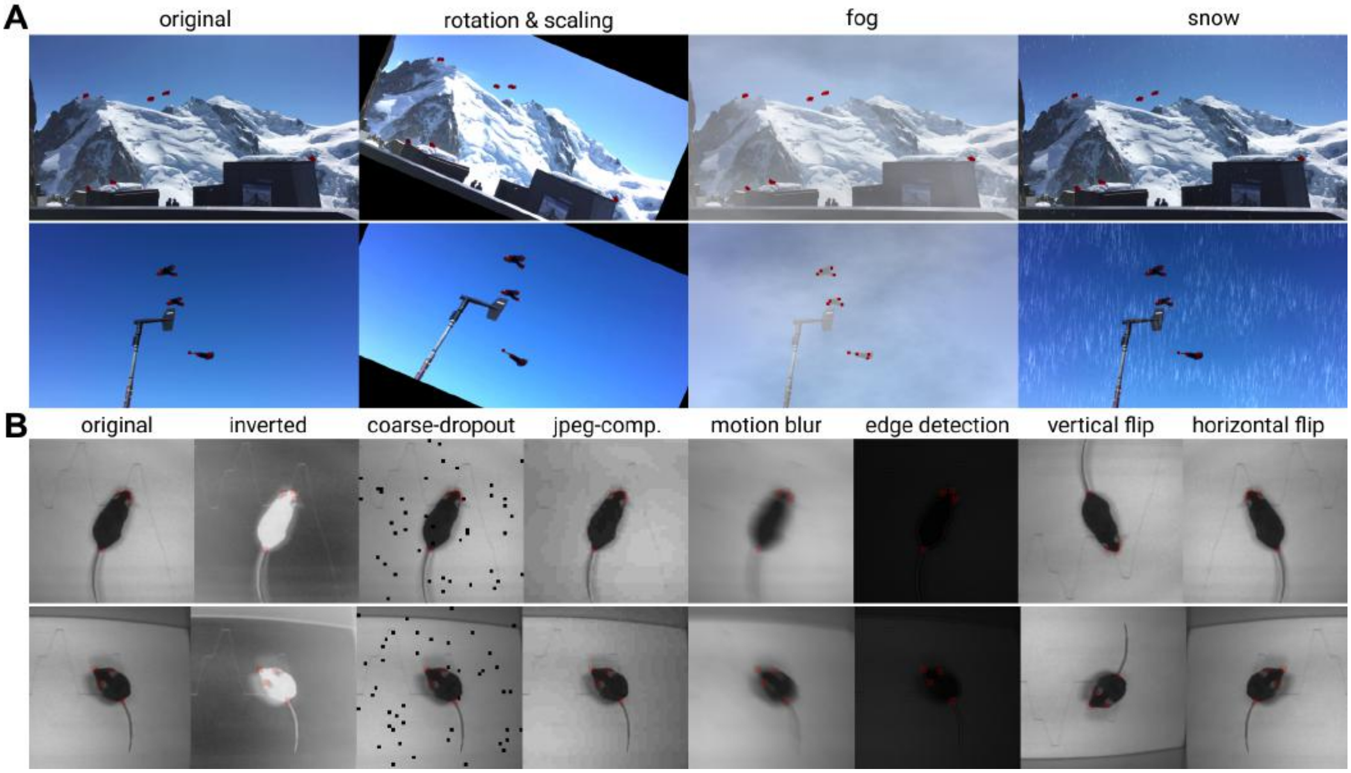


Figure 3. Example augmentation images with labeled body parts in red. **(A)** Two example frames of Alpine choughs (Pyrrhocorax graculus) near Mont Blanc with human-applied labels in red (original). The images to the right illustrate three augmentations (as labeled). **(B)** Two example frames of a trail-tracking mouse (mus musculus) from (20) with four labeled bodyparts as well as augmented variants. Open in Google Colaboratory

towards state-of-the-art performance on this benchmark also helped push the field to build better DNNs and openly share code. This dataset has been extensively used for pre-training networks, which we will discuss in the model and optimization section below.

The second highly relevant dataset is the one curated for the task of interest– Mathis et al. (20) empirically demonstrated that the size of this dataset can be comparably small for typical pose estimation cases in the laboratory. Typically, this dataset contains 10–500 images, vs. the standard human pose estimation benchmark datasets, such as MS COCO (36) or MPII pose (37), which has annotated 40K images (of 26K individuals). This implies that the dataset that is curated is highly influential on the final performance, and great care should be taken to select diverse postures, individuals, and background statistics and labeling the data accurately (discussed below in "pitfalls").

In practice, several factors matter: the performance of a fine-tuned model on the task of interest, the amount of images that need to be annotated for fine-tuning the network, and the convergence rate of optimization algorithm—i.e., how many steps of gradient descent are needed to obtain a certain performance. Using a pre-trained network can help with this in several regards: He et al. (38) show that in the case of large training datasets, pre-training typically aids with convergence rates, but not necessarily the final performance. Despite this evidence that under the right circumstances (i.e., given enough task-relevant data) and with longer training, randomly initialized models can match the performance of fine-tuned ones for key point detection on COCO (38) and horses (39), however, the networks are less robust (39). Beyond robustness, using a pre-trained model is generally advisable when the amount of labeled data for the target task is small, which is true for many applications in neuroscience, as it leads to shorter training times and better performance with less data (20, 38–40). Thus, pre-trained pose estimation algorithms save training time, increase robustness, and require substantially less training data. Indeed, most packages in Neuroscience now use pre-trained models (20, 40–44), although some do not (45–47), which can give acceptable performance for simplified situations with aligned individuals.

More recently, larger datasets like the 3.5 billion Instagram dataset (48), JFT which has 300M images (49, 50) and OpenImages (51) became popular, further improving performance and robustness of the considered models (50). What task is used for pre-training also matters. Corroborating this insight, Li et al. showed that pre-training on large-scale object detection task can improve performance for tasks that require fine, spatial information like segmentation (52).

Besides large datasets for pre-training, a curated dataset with pose annotations is needed for optimizing the algorithm on the pose estimation task. The process is discussed in more detail below and it typically suffices to label a few (diverse) frames. Data augmentation is the process of expanding the training set by applying specified manipulations (like rotate, scaling image size). Based on the chosen corruptions, models become more invariant to rotations, scale changes or translations and thus more accurate (with less training data). Augmentation can also help with improving robustness to noise, like jpeg compression artefacts and motion blur (Figure 3). To note, data augmentation schemes should not affect the semantic information in the image: for instance, if color conveys important information about the identity of an animal, augmentations involving changes in color are not advisable. Likewise, augmentations which change the spatial position of objects or subjects should always be applied to both the input image and the labels (Box 2).

Model architectures.

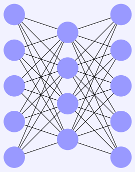
Systems for markerless pose estimation are typically composed of a *backbone* network (encoder), which takes the role of the feature extractor, and one or multiple *heads* (decoders). Understanding the model architectures and design choices common in deep learning based pose estimation systems requires basic understanding of convolutional neural networks. We summarize the key terms in Box 1, and expand on what encoders and decoders are below.

Instead of using handcrafted features as in classical systems, deep learning based systems employ "generic" encoder architectures which are often based on models for object recognition. In a typical system, the encoder design affects the most important properties of the algorithms such as its inference speed, training-data requirements and memory demands. For the pose estimation algorithms so far used in neuroscience the encoders are either stacked hourglass networks (26), MobileNetV2s (53), ResNets (54), DenseNets (55) or EfficientNets (56). These encoder networks are typically pre-trained on one or multiple of the larger-scale datasets introduced previously (such as ImageNet), as this has been shown to be an advantage for pose estimation on small lab-scale sized datasets (20, 39, 40). For common architectures this pre-training step does not need to be carried out explicitly-trained weights for popular architectures are already available in common deep learning frameworks.

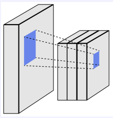
The impact of the encoder on DNN performance is a highly active research area. The encoders are continuously improved in regards to speed and object recognition performance (9, 53, 55–57). Naturally, due to the importance of the ImageNet benchmark the accuracy of network architectures continuously increases (on that dataset). For example, we were able to show that this performance increase is not merely reserved for ImageNet, or (importantly) other object recognition tasks (57), but in fact that better architectures on ImageNet are also better for pose estimation (44). However, being better on ImageNet, also comes at the cost of decreasing inference speed and increased memory demands. DeepLabCut (an open source toolbox for markerless pose estimation popular in neuroscience) thus incorporates backbones from MobileNetV2s (faster) to EfficientNets (best performance on ImageNet; 39, 44).

Box 1: Glossary of Deep Learning terms

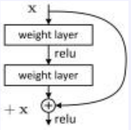
An excellent textbook for Deep Learning is provided by Goodfellow et al. (33). See Dumoulin & Visin (58) for an in-depth technical overview of convolution arithmetic.



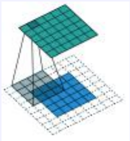
Artificial neural network (ANN): An ANN can be represented by a collection of computational units ("neurons") arranged in a directed graph. The output of each unit is computed as a weighted sum of its inputs, followed by a nonlinear function.



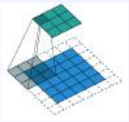
Convolutional neural network (CNN): A CNN is an ANN composed of one or multiple convolutional layers. Influential early CNNs are the LeNet, AlexNet and VGG16 (9, 23, 33).



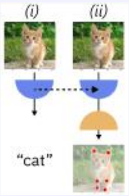
Residual Networks (ResNets): Increasing network depth makes deep neural networks (DNNs) more expressive compared to adding units to a shallow architecture. However, optimization becomes hard for standard CNNs beyond 20 layers, at which point depth in fact decreases the performance (54). In residual networks, instead of learning a mapping $y = f(x)$, the layer is re-parametrized to learn the mapping $y = x + f(x)$, which improves optimization and regularizes the loss landscape (59). These networks can have much larger depth without diminishing returns (54) and are the basis for other popular architectures such as MobileNets (53) and EfficientNets (56).



Convolution: A convolution is a special type of linear filter. Compared with a full linear transformation, convolutional layers increase computational efficiency by weight sharing (23, 33). By applying the convolution, the same set of weights is used across all locations in the image. **Deconvolution:** Deconvolutional layers allow to upsample a feature representation. Typically, the kernel used for upsampling is optimized during training, similar to a standard convolutional layer. Sometimes, fixed operations such as bilinear upsampling filters are used.



Stride, Downsampling and Dilated (atrous) Convolutions: In DNNs for computer vision, images are presented as real-valued pixel data to the network and are then transformed to symbolic representations and image annotations, such as bounding boxes, segmentation masks, class labels or keypoints. During processing, inputs are consecutively abstracted by aggregating information from growing "receptive fields". Increasing the receptive field of the unit is possible by different means: Increasing the stride of a layer computes outputs only for every n-th input and effectively downsamples the input with a learnable filter. Downsampling layers perform the same operation, but with a fixed kernel (e.g. taking the maximum or mean activation across the receptive field). In contrast, *atrous* or dilated convolutions increase the filter size by adding intermittent zero entries between the learnable filter weights—e.g., for a dilation of 2, a filter with entries $(1, 2, 3)$ would be converted into $(1, 0, 0, 2, 0, 0, 3)$. This allows increases in the receptive field without loosing resolution in the next layers, and is often applied in semantic segmentation algorithms (60), and pose estimation (18, 20).



Transfer learning: The ability to use parameters from a network that has been trained on one task—e.g. classification, see *(i)*—as part of a network to perform another task—e.g. pose estimation, see *(ii)*. The approach was popularized with DeCAF (61), which used AlexNet (62) to extract features to achieve excellent results for several computer vision tasks. Transfer learning generally improves the convergence speed of model training (38–40, 63) and model robustness compared to training from scratch (39).

Box 2: Key parameters and choices.

The key design choices of pose estimation systems are dataset curation, data augmentation, model architecture selection, optimization process, and the optimization criterions.

- **Data Augmentation**: is the technique of increasing the training set by converting images and annotations into new, altered images via geometric transformations (e.g. rotation, scaling..), image manipulations (e.g. contrast, brightness,...), etc. (Figure 3) . Depending on the annotation data, various augmentations, i.e. rotation symmetry/etc. are ideal. Packages such as Tensorpack (64) and imgaug (65) as well as tools native to PyTorch (66) and TensorFlow (67) provide common augmentation methods and are used in many packages.
- **Model Architecture**: Users should select an architecture that is accurate and fast (enough) for their goal. Top performing networks (in terms of accuracy) include Stacked Hourglass (26), ResNets (54) and EfficientNets (56) with appropriate decoders (18, 19, 28, 44) as well as recent high-resolution nets (29, 68). Performance gains in speed at the expense of slightly worse accuracy are possible with (optimized) lightweight models such as MobileNetV2 (53) in DeepLabCut (39) and stacked hourglass networks with DenseNets (55) as proposed in DeepPoseKit (41); and often this performance gap can be rescued with good data augmentation.

In (standard) convolutional encoders, the high-resolution input images get gradually downsampled while the number of learned features increases. Regression based approaches which directly predict keypoint locations from the feature representation can potentially deal with this downsampled representation. When the learning problem is instead cast as identifying the keypoint locations on a grid of pixels, the output resolution needs to be increased first, often by deconvolutional layers (18, 28). We denote this part of the network as the decoder, which takes downsampled features, possibly from multiple layers in the encoder hierarchy, and gradually upsamples them again to arrive at the desired resolution. The first models of this class were Fully Convolutional Networks (69), and later DeepLab (60). Many popular architectures today follow similar principles. Design choices include the use of skip connections between decoder layers, but also regarding skip connections between the encoder and decoder layers. Example encoder–decoder setups are illustrated in Figure 4. The aforementioned building blocks—encoders and decoders—can be used to form a variety of different approaches, which can be trained end-to-end directly on the target task (i.e., pose estimation).

Pre-trained models can also be adapted to a particular application. For instance, DeeperCut (18), which was adapted by the animal pose estimation toolbox DeepLabCut (20), was built with a ResNet (54) backbone network, but adapted the stride by atrous convolutions (60) to retain a higher spatial resolution (Box 1). This allowed larger receptive fields for predictions, but retains a relatively high speed (i.e., for video analysis) but most importantly because ResNets can be pretrained on ImageNet, those initialized weights could be used. Other architectures, like stacked hourglass networks (26) used in DeepFly3D (70) and DeepPoseKit (41), retain feature representations at multiple scales and pass those to the decoder (Figure 4A, B).

Loss functions: training architectures on datasets.

Keypoints (i.e., bodyparts) are simply coordinates in image space. There are two fundamentally different ways for estimating keypoints (i.e., how to define the loss function). The problem can be treated as a regression problem with the coordinates as targets (24, 71). Alternatively, and more popular, the problem can be cast as a classification problem, where the coordinates are mapped onto a grid (e.g. of the same size as the image) and the model predicts a heatmap (scoremap) of location probabilities for each bodypart (Figure 4C). In contrast to the regression approach (24), this is fully convolutional, and allows modeling of multi-modal distributions and aids the training process (18, 26, 27, 72). Moreover, the heatmaps have the advantage that one can naturally predict

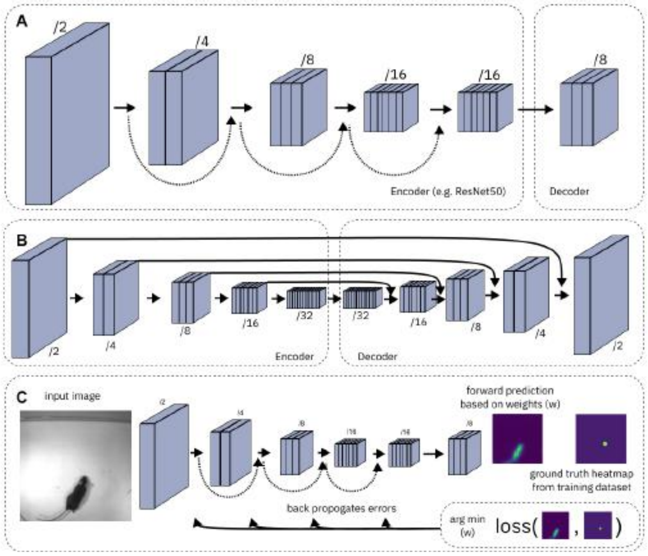


Figure 4. Schematic overview of possible design choices for model architectures and training process (A) A simple, but powerful variant (18) is a ResNet-50 (54) architecture adapted to replace the final down-sampling operations by atrous convolutions (60) to keep a stride of 16, and then a single deconvolution layer to upsample to output maps with stride 8. It also forms the basis of other architectures (e.g. 28). The encoder can also be exchanged for different backbones to improve speed or accuracy (see Box 2). **(B)** Other approaches like stacked hourglass networks (26), are not pre-trained and employ skip connections between encoder and decoder layers to aid the up-sampling process. **(C)** For training the network, the training data comprising input images and target heatmaps is used. The target heatmap is compared with the forward prediction. Thereby, the parameters of the network are optimized to minimize the loss that measures the difference between the predicted heatmap and the target heatmap (ground truth).

multiple locations of the “same” bodypart in the same image (i.e., 2 elbows) without mode collapse (Figure 5A).

Loss functions can also reflect additional priors or inductive biases about the data. For instance, DeepLabCut uses location refinement layers (locref), that counteract the downsampling inherent in encoders, by training outputs to predict corrective shifts in image coordinates relative to the downsampled output maps (Figure 5A). In pose estimation, it is possible to define a *skeleton* or graph connecting keypoints belonging to subjects with the same identity (see below) (18, 27). When estimating keypoints over time, it is also possible to employ temporal information and encourage the model to only smoothly vary its estimate among consecutive frames (73–76). Based on the problem, these priors can be directly encoded and be used to regularize the model.

How can pose estimation algorithms accommodate multiple individuals? Fundamentally, there are two different approaches: bottom-up and top-down methods (Figure 5). In top-down methods, individuals are first localized (often with another neural network trained on object localization) then pose estimation is performed per localized individual (26, 28, 68). In bottom-up methods all bodyparts are localized, and networks are also trained to predict connections of bodyparts within individuals (i.e., limbs). These connections are then used to link candidate bodyparts to form individuals (19, 27, 29, 73). To note, these techniques can be used on single individuals for increased performance, but often are not needed and usually imply reduced inference speed.

Optimization.

For pre-training, stochastic gradient descent (SGD; 77) with momentum (78) is an established method. Different variants of SGD are now common (such as Adam; 79) and used for fine-tuning the resulting representations. As mentioned above, pose estimation algorithms are typically trained in a multi-stage setup where the backbone is trained first on a large (labeled) dataset of a potentially unrelated task (like image classification). Users can also download these pretrained weights. Afterwards, the model is fine-tuned on the pose-estimation task. Once trained, the quality of the prediction can be judged in terms of the root mean square error (RMSE), which measures the distance between the ground truth keypoints and predictions (20, 45), or by measuring the percentage of correct keypoints (PCK, 37, 39); i.e., the fraction of detected keypoints that fall within a defined distance of the ground truth.

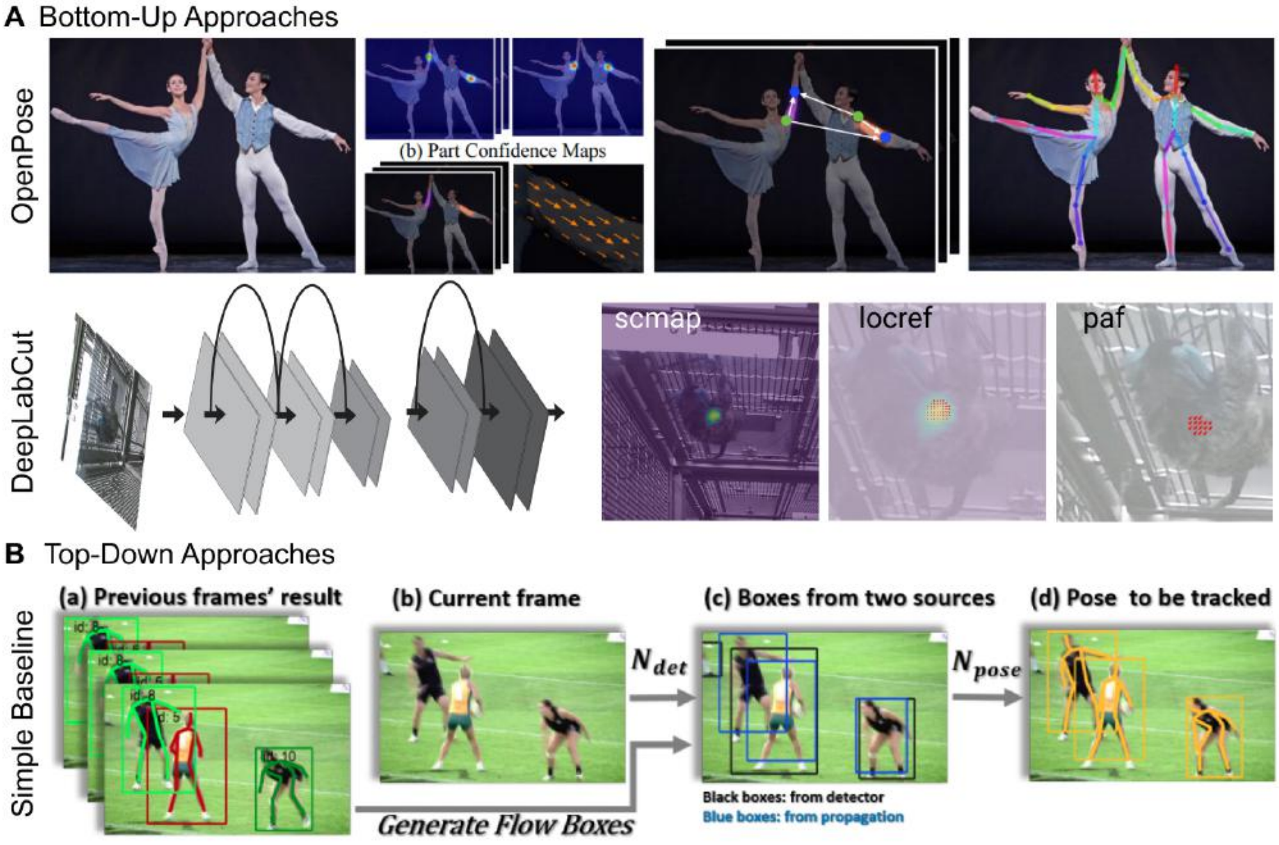


Figure 5. Multi-animal pose estimation approaches. A: Bottom-up approaches detect all the body parts (e.g. elbow and shoulder in example) as well as “limbs” (part confidence maps). These limbs are then used to associate the bodyparts within individuals correctly (Figure from OpenPose, 27). For both OpenPose and DeepLabCut, the bodyparts and part confidence maps, and part affinity fields (paf’s) are predicted as different decoders (aka output heads) from the encoder. **B**: Top-down approaches localize individuals with bounding-box detectors and then directly predict the posture within each bounding box. This does not require part confidence maps, but is subject to errors when bounding boxes are wrongly predicted (see black bounding box encompassing two players in (c)). The displayed figures, adapted from Xiao et al. (28), improved this disadvantage by predicting bounding boxes per frame and forward predicting them across time via visual flow.

To properly estimate model performance in an application setting, it is advisable to split the labeled dataset at least into train and test subsets. If systematic deviations can be expected in the application setting (e.g., because the subjects used for training the model differ in appearance from subjects encountered at model deployment (39), this should be reflected when choosing a way to split the data. For instance, if data from multiple individuals is possible, distinct individuals should form distinct subsets of the data. On the contrary, strategies like splitting data by selecting every n-th frame in a video likely overestimates the true model performance.

The model is then optimized on the training dataset, while performance is monitored on the validation (test) split. If needed, hyperparameters—like parameter settings of the optimizer, or also choices about the model architecture—of the model can be adapted based on an additional validation set.

All of the aforementioned choices influence the final outcome and performance of the algorithm. While some parts of the training pipeline are well-established and robust—like pretraining a model on ImageNet—choices about the dataset, architecture, augmentation, fine-tuning procedure, etc. will inevitably influence the quality of the pose estimation algorithm (Box 2). See Figure 3 for a qualitative impression of augmentation effects of some of these decisions (see also Figure 8). We will discuss this in more detail in the Pitfalls section.

So far, we considered algorithms able to infer 2D keypoints from videos, by training deep neural networks on previously labeled data. Naturally, there is also much work in computer vision and machine learning towards the estimation of 3D keypoints from 2D labels, or to directly infer 3D keypoints. In the interest of space, we had to omit those but refer the interested reader to (74, 81–84) as well specifically for neuroscience (42, 47, 70, 74, 80, 85).

Lastly, it is not understood how CNNs make decisions and they often find "shortcuts" (86). While this active research area is certainly beyond the scope of this primer, from practical experience we know that at least within-domain—i.e., data that is similar to the training set—DNNs work very well for pose estimation, which is the typical setting relevant for downstream applications in neuroscience. It is worth noting that in order to optimize performance, there is no one-size-fits-all solution. Thus, we hope by building intuition in users of such systems, we provide the necessary tools to make these decisions with more confidence (Figure 6).

Scope and applications

Markerless motion capture can excel in complicated scenes, with diverse animals, and with any camera available (monochrome, RGB, depth cameras, etc). The only real requirement is the ability of the human to be able to reliably label keypoints (manually or via alternative sources). Simply, you need to be able to see what you want to track. Historically, due to limitations in computer vision algorithms experimentalists would go to great lengths to simplify the environment, even in the laboratory (i.e., no bedding, white or black walls, high contrast), and this is no longer required with deep learning-based pose estimation. Now, the aesthetics one might want for photographs or videos taken in daily life are the best option.

Indeed, the field has been able to rapidly adopt these tools for neuroscience. Deep learning-based markerless pose estimation applications in the laboratory have already been published for flies (20, 41, 43, 45, 70, 85), rodents (20, 40, 41, 43, 45, 47, 70, 87), horses (39), dogs (74), rhesus macaque (42, 74, 88, 89) and marmosets (90); the original architectures were developed for humans (18, 26, 27). Outside of the laboratory, DeepPoseKit was used for zebras (41) and DeepLabCut for 3D tracking of cheetahs (80), for squirrels (91) and macaques (89), highlighting the great "in-the-wild" utility of this new technology (10). As outlined in the principles section, and illustrated by these applications, these deep learning architectures are general-purpose and can be broadly applied to any animal as well as condition.

Recent research highlights the prevalent representations of action across the brain (92), which emphasizes the importance of quantifying behavior even in non-motor tasks. For instance, pose estimation tools have recently been used to elucidate the neural variability across cortex in humans during thousands of spontaneous reach movements (93). Pupil tracking is of great importance for visual neuroscience. One recent study by Meyer et al. used head-fixed cameras and DeepLabCut to reveal two distinct types of coupling between eye and head movements (94). In order to accurately correlate neural activity to visual input tracking the gaze is crucial. The recent large, open dataset from the Allen Institute includes imaging data of six cortical and two thalamic regions in response to various stimuli classes as well as pupil tracking with DeepLabCut (95). The International Brain Lab has integrated DeepLabCut into their workflow to track multiple bodyparts of decision-making mice including their pupils (96).

Measuring relational interactions is another major direction, that has been explored less in the literature so far, but is feasible. Since the feature detectors for pose estimation are of general nature one can easily not only track the posture of individuals but also the tools and objects one interacts with (e.g. for analyzing golf or tennis). Furthermore, social behaviors, and parenting interactions (for example in mice) can now be studied noninvasively.

Due to the general capabilities, these tools have several applications for creating biomarkers by extracting high fidelity animal traits, for instance in the pain field (97) and for monitoring motor function in healthy and diseased conditions (98). DeepLabCut was also integrated with tools for x-ray analysis (99). For measuring joint center locations in mammals, arguably, x-ray is the gold standard. Of course, x-ray data also poses challenges for extracting body part locations from x-ray data. A recent paper shared methodology to integrate

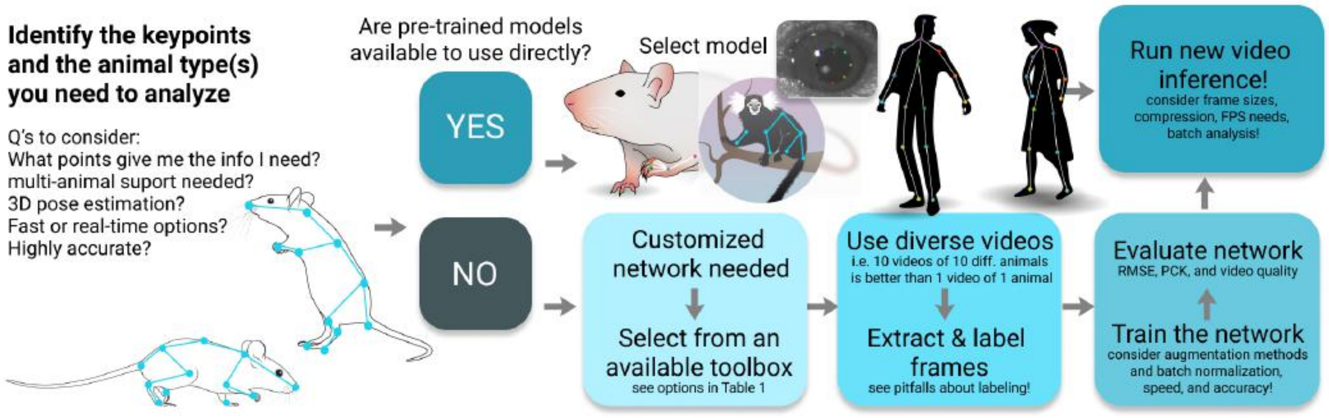

Identify the keypoints and the animal type(s) you need to analyze
Q's to consider:
What points give me the info I need?
multi-animal suport needed?
3D pose estimation?
Fast or real-time options?
Highly accurate?
Are pre-trained models available to use directly?
YES
NO
Select model
Run new video inference!
consider frame sizes, compression, FPS needs, batch analysis!
Customized network needed
Select from an available toolbox
see options in Table 1
Use diverse videos
i.e. 10 videos of 10 diff. animals is better than 1 video of 1 animal
Extract & label frames
see pitfalls about labeling!
Evaluate network
RMSE, PCK, and video quality
Train the network
consider augmentation methods and batch normalization, speed, and accuracy!

Figure 6. An overview of the workflow for deep learning based pose estimation, which highlights several critical decision points.

DeepLabCut with XROMM, a popular analysis suite, to advance the speed and accuracy for x-ray based analysis (99).

How do the (current) packages work?

Here we will focus on packages that have been used in behavioral neuroscience, but the general workflow for pose estimation in computer vision research is highly similar. What has made experimentalist-focused toolboxes different is that they provide essential code to generate and train on one's own datasets. Typically, what is available in computer vision focused pose estimation repositories is code to run inference (video analysis) and/or run training of an architecture for specific datasets around which competitions happen (e.g., MS COCO; 36 and MPII pose; 37). While these are two crucial steps, they are not sufficient to develop tailored neural networks for an individual lab or experimentalist. Thus, the "barrier to entry" is often quite high to use these tools. It requires knowledge of deep learning languages to build appropriate data loaders, data augmentation pipelines, and training regimes. Therefore, in recent years several packages have not only focused on animal pose estimation networks, but in providing users a full pipeline that allows for (1) labeling a customized dataset (frame selection and labeling tools), (2) generating test/train datasets, (3) data augmentation and loaders, (4) neural architectures, (5) code to evaluate performance, (6) run video inference, and (7) post-processing tools for simple readouts of the acquired machine-labeled data.
Thus far, around 10 packages have become available in the past 2 years (20, 40–43, 45, 47, 70). Each has focused on providing slightly different user experiences, modularity, available networks, and balances to the speed/accuracy trade-off for video inference. Several include their (adapted) implementations of the original DeepLabCut or LEAP networks as well (41, 43). But the ones we highlight have the full pipeline delineated above as a principle and are open source, i.e., at minimum inference code is available (see Table 1). The progress gained and challenges they set out to address (and some that remain) are reviewed elsewhere (10, 100). Here, we discuss collective aims of these packages (see also Figure 6).

Current packages for animal pose estimation have focused on primarily providing tools to train tailored neural networks to user-defined features. Because experimentalists need flexibility and are tracking very different animals and features, the most successful packages (in terms of user base as measured by citations and GitHub engagement) are species agnostic. However, given they are all based on advances from prior art in human pose estimation, the accuracy of any one package given the breadth of options that could be deployed (i.e, data augmentation, training schedules, and architectures) will remain largely comparable, if such tools are provided to the user. What will determine performance the most is the input training data provided, and how much capacity the architectures have.

It is notable that using transfer learning has proven to be ad-

Table 1. Overview of popular deep learning tools for animal motion capture (or newly presented packages that, minimally, include code). Here, we denote if it can be used to create tailored networks, or only specific animal tools are provided, i.e., only work "as-is" on a fly or rat. We also only highlight if beyond human pre-trained neural networks (PT-NNs) are available. We also provide the release date and current citations for noted references, including those to related preprints (indexed from google scholar). *note, this code is deprecated and supplanted by SLEAP.

	Any species	3D	>1 animal	Training Code	Full GUI	Ex. Data	PT-NNs	Released	Citations
DeepLabCut (20, 80)	yes	yes	yes	yes	yes	yes	many	4/2018	491
LEAP (45)	yes	no	yes	yes	yes	yes	no	6/2018*	98
DeepBehavior (40)	no	yes	yes	no	no	no	no	5/2019	15
DeepPoseKit (41)	yes	no	no	yes	partial	yes	no	8/2019	48
DeepFly3D (70)	no	yes	no	2D only	partial	yes	fly	5/2019	21
FreiPose (47)	no	yes	no	partial	no	yes	no	2/2020	1
Optiflex (43)	yes	no	no	yes	partial	yes	no	5/2020	0

vantageous for better robustness (i.e., its ability to generalize, see 20, 39, 40), which was first deployed by DeepLabCut (see Table 1). Now, training on large animal-specific datasets has recently been made available in DeepLabCut as well (such as a horse pose dataset with >8,000 annotated images of 30 horses; 39). This allows the user to bypass the only manual part of curating and labeling ground truth data, and these models can directly be used for inference on novel videos. For DeepLabCut, this is an emerging community-driven effort, with external labs already contributing models and data[1].

Box 3: Computing hardware

- **CPU**: The central processing unit (CPU) is the core of a computer and executes computer programs. CPUs work well on sequential or lightly parallelized routines due to the limited number of cores.
- **GPU**: A graphical processing unit (GPU) is a specialized computing device designed to rapidly process and alter memory. GPUs are ideal for computer graphics and often located in graphics cards. Their highly parallel architecture enables them to be more efficient[a] than CPUs for algorithms with many small subroutines which can be launched in parallel. They can be applied to run DNNs at higher speed (62) and pose estimation in particular (17, 39, 87).
- **Affordability of GPUs**: Modern GPUs are affordable (around 300 - 800 USD for cards than can be used for the pose estimation tools mentioned here; and up to 10,000 USD for high end cards) and ideally suited to run video processing within a single lab in a decentralized way. They can be placed into standard desktop computers, or even "gaming" laptops are options. However, to get started it might be easier to test software in cloud computing services first for ease-of-use (i.e. no driver installation).
- **Cloud computing**: Ability to use resources online rapidly (minimal installation) often in a pay-per-use scheme. Two relevant examples are *Google Colaboratory* and *My Binder*. *Google Colaboratory* is an online platform for hosted free GPU use with run times of up to 6 hours. *My Binder* allows turning a Git repository into a collection of interactive notebooks by running them in an executable environment, making your code immediately reproducible by anyone, anywhere (mybinder.org).

[a] Link to NVIDIA Data Center Deep Learning Product Performance: developer.nvidia.com/deep-learning-performance-training-inference

In the future, having the ability to skip labeling and training and run video inference with robust models will lead to more reproducible and scalable research. For example, as we show in other sections of the primer, if the labeling accuracy is not of a high quality, and the data is not diverse enough, then the networks are not able to generalize to so-called "out-of-domain" data. If as a community we collectively build stable and robust models that leverage the breadth of behaviors being carried out in laboratories worldwide, we can work towards models that would work in a plug-in-play fashion. We anticipate new datasets and models to become available in the next months to years.

All packages, just like all applications of deep learning to video, prefer access to GPU computing resources (See Box 3). On GPUs one experiences faster training and inference times but the code can also be deployed on standard CPUs or laptops. With cloud computing services, such as Google Colaboratory and JupyterLab, many pose estimation packages can simply be deployed on remote GPU resources. This still requires (1) knowledge about these resources, and (2) toolboxes providing so-called "notebooks" that can be easily deployed. But, given these platforms have utility beyond just pose estimation, they are worthwhile to learn about.

For the non-GPU aspects, only a few packages have provided easy-to-use graphical user interfaces that allow users with no programming experience to use the tool (see Table 1). Lastly, the available packages vary in their access to 3D tools, multi-animal support, and types of architectures available to the user, which is often a concern for speed and accuracy. Additionally, some packages have limitations on only allowing the same sized videos for training and inference, while others are more flexible. These are all key considerations when deciding which eco-system to invest in learning (as every package has taken a different approach to the API).

Perhaps the largest barrier to entry for using deep learning-based pose estimation methods is managing the computing resources (See Box 3, Box 4). From our experience, installing GPU drivers and the deep learning packages (TensorFlow, PyTorch), that all the packages rely on, is the biggest challenge. To this end, in addition to documentation that is "user-focused" (i.e., not just an API for programmers), resources like webinars, video tutorials, workshops, Gitter and community-forums (like StackOverflow and Image Forum SC) have become invaluable resources for the modern neuroscientist. Here, users can ask questions and get assistance from developers and users alike. We believe this has also been a crucial step for the success of DeepLabCut.

While some packages provide full GUI-based control over the packages, to utilize more advanced features at least minimal programming knowledge is ideal. Thus, better training for the increasingly computational nature of neuroscience will be crucial. Making programming skills a requirement of graduate training, building better community resources, and leveraging the fast-moving world of technology to harness those computing and user resources will be crucial. In animal pose estimation, while there is certainly an attempt to make many of the packages user-friendly, i.e., to onboard users

[1] modelzoo.deeplabcut.org

and have a scalable discussion around common problems, we found user forums to be very valuable (101). Specifically, DeepLabCut is a member of the Scientific Community Image Forum[2] alongside other packages that are widely used for image analysis in the life sciences such as Fiji (102), napari, CellProfiler (103) Ilastik (104) and scikit-image (105).

Box 4: Reproducible Software

Often installation of deep learning languages like TensorFlow/Keras (67) and PyTorch (66) is the biggest hurdle for getting started.

- **Python virtual environments**: Software often has many dependencies, and they can conflict if multiple versions are required for different needs. Thus, placing dependencies within a contained environment can minimize issues. Common environments include Anaconda (conda) and virtualenv, both for Python code bases.
- **Docker** delivers software in packages called containers, which can be run locally or on servers. Containers are isolated from one another and bundle their own software, libraries and configuration files (docker.com; 106).
- **GitHub**: github.com is a platform for developing and hosting software, which uses Git version control. Version control is excellent to have history-dependent versions and discrete workspaces for code development and deployment. GitLab gitlab.com/explore also hosts code repositories.

Practical considerations for pose estimation (with deep learning)

As a recent field gaining traction, it is instructive to regard the operability of deep learning-powered pose estimation in light of well-established, often gold standard, techniques.

General considerations and pitfalls.

As discussed in *Scope and applications* and as evidenced by the strong adaptation of the tools, deep learning-based pose estimation work well in standard setups with visible animals. The most striking advantage over traditional motion capture systems is the absence of any need for body instrumentation. Although seemingly obvious, the previous statement hides the belated recognition that marker-based motion capture suffers greatly from the wobble of markers placed on the skin surface. That behavior, referred to as "soft tissue artifact" among movement scientists and attributable to the deformation of tissues underneath the skin such as contracting muscles or fat, is now known to be the major obstacle to obtaining accurate skeletal kinematics [3] (109). To make matters worse, contaminated marker trajectories may be harmful in clinical contexts, potentially invalidating injury risk assessment (e.g. 110). Although a multitude of numerical approaches exists to tackle this issue, the most common, yet incomplete, solution is multi-body kinematics optimization (or "inverse kinematics" in computer graphics and robotics; 111). This procedure uses a kinematic model and searches for the body pose that minimizes in the least-squares sense the distance between the measured marker locations and the virtual ones from the model while satisfying the constraints imposed by the various joints (112). Its accuracy is, however, decisively determined by the choice of the underlying model and its fidelity to an individual's functional anatomy (111). In contrast, motion capture with deep learning elegantly circumvents the problem by learning a geometry-aware representation of the body from the data to associate keypoints to limbs (10, 18, 27), which, of course, presupposes that one can avoid the "soft tissue artifact" when labeling.

At present, deep learning-powered pose estimation can be poorly suited to evaluate rotation about a bone's longitudinal axis. From early markerless techniques based on visual hull extraction this is a known problem (113). In marker-based settings, the problem has long been addressed by rather tracking clusters of at least three non-aligned markers to fully reconstruct a rigid segment's six degrees of freedom (114). Performing the equivalent feat in a markerless case is difficult, but it is possible by labeling multiple points (for instance on either side of the wrist to get the lower-limb orientation). Still, recent hybrid, state-of-the-art approaches jointly training under both position and orientation supervision augur very well for video-based 3D joint angle computation (75, 76).

With the notable exception of approaches leveraging radio wave signals to predict body poses through walls (115), deep learning-powered motion capture requires the individuals be visible; this is impractical for kinematic measurements over wide areas. A powerful alternative is offered by Inertial Measurement Units (IMUs)—low-cost and lightweight devices typically recording linear accelerations, angular velocities and the local magnetic field. Raw inertial data can be used for coarse behavior classification across species (3, 116). They can also be integrated to track displacement with lower power consumption and higher temporal resolution than GPS (117), thereby providing a compact and portable way to investigate whole body dynamics (e.g. 118) or, indirectly, energetics (119). Recent advances in miniaturization of electronical components now also allow precise quantification of posture in small animals (120), and open new avenues for kinematic recordings in multiple animals at once at fine motor scales.

Nonetheless, IMU-based full body pose reconstruction necessitates multiple sensors over the body parts of interest;

[2] forum.image.sc

[3] Intra-cortical pins and biplane fluoroscopy give direct, uncontaminated access to joint kinematics. The first, however, is invasive (and entails careful surgical procedures; 107) whereas the second is only operated in very constrained and complex laboratory settings (108). Both are local to a specific joint, and as such do not strictly address the task of pose estimation.

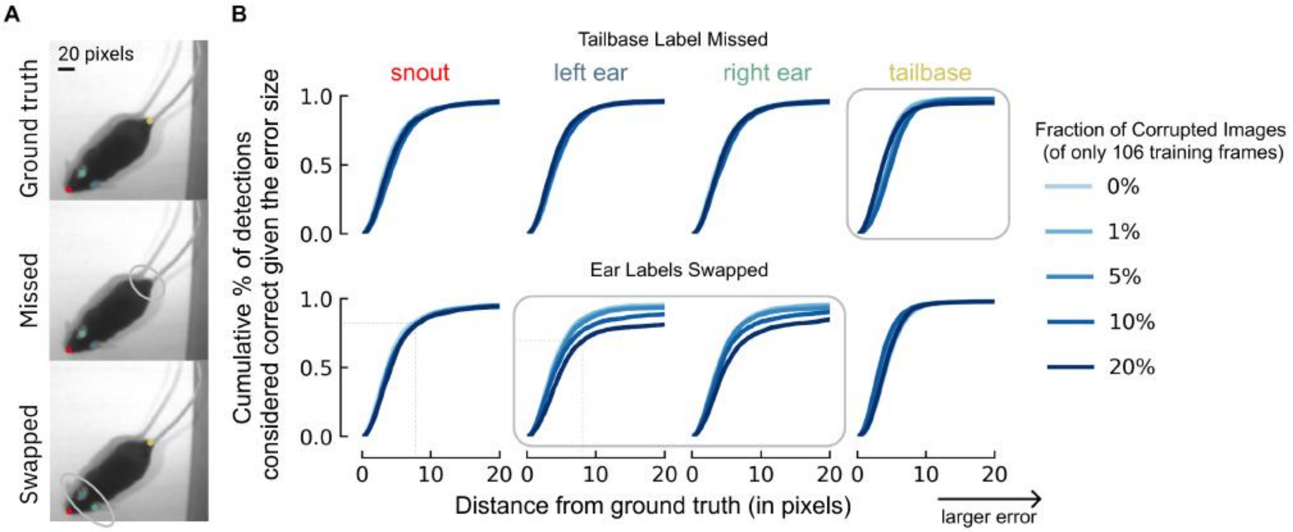


Figure 7. Labeling Pitfalls: How corruptions affect performance (A) Illustration of two types of labeling errors. Top is ground truth, middle is missing a label at the tailbase, and bottom is if the labeler swapped the ear identity (left to right, etc.). **(B)** Using a small dataset of 106 frames, how do the corruptions in A affect the percent of correct keypoints (PCK) as the distance to ground truth increases from 0 pixel (perfect prediction) to 20 pixels (larger error)? The X-axis denotes the difference in the ground truth to the predicted location (RMSE in pixels), whereas Y-axis is the fraction of frames considered accurate (e.g., ≈80% of frames fall within 9 pixels, even on this small training dataset, for points that are not corrupted, whereas for corrupted points this falls to ≈65%). The fraction of the dataset that is corrupted affects this value. Shown is when missing the tailbase label (top) or swapping the ears in $1, 5, 10$ and 20% of frames (of 106 labeled training images). Swapping vs. missing labels has a more notable adverse effect on network performance.

commercial solutions require up to 17 of them (121). That burden was recently eased by utilizing a statistical body model that incorporates anatomical constraints, together with optimizing poses over multiple frames to enforce coherence between the model orientation and IMU recordings—reducing the system down to six sensors while achieving stunning motion tracking (122). Yet, two additional difficulties remain. The first arises when fusing inertial data in order to estimate a sensor's orientation (for a comprehensive description of mathematical formalism and implementation of common fusion algorithms, see 123). The process is susceptible to magnetic disturbances that distort sensor readings and, consequently, orientation estimates (124). The second stems from the necessity to align a sensor's local coordinate system to anatomically meaningful axes, a step crucial (among others) to calculating joint angles (e.g., 125). The calibration is ordinarily carried out by having the subject perform a set of predefined movements in sequence, whose execution determines the quality of the procedure. Yet, in some pathological populations (let alone in animals), calibration may be challenging to say the least, deteriorating pose reconstruction accuracy (126).

A compromise to making the task less arduous is to combine videos and body-worn inertial sensors. Thanks to their complementary nature, incorporating both cues mitigates the limitations of each individual system; i.e., both modalities reinforce one another in that IMUs help disambiguate occlusions, whereas videos provide disturbance-free spatial information (127). The idea also applies particularly well to the tracking of multiple individuals—even without the use of appearance features, advantageously—by exploiting unique movement signatures contained within inertial signals to track identities over time (128).

Pitfalls of using deep learning-based motion capture.

Despite being trained on large scale datasets of thousands of individuals, even the best architectures fail to generalize to "atypical" postures (with respect to the training set). This is wonderfully illustrated by the errors committed by OpenPose on yoga poses (129).

These domain shifts are major challenges (also illustrated below), and while this is an active area of research with much progress, the easiest way to make sure that the algorithm generalizes well is to label data that is similar to the videos at inference time. However, due to active learning implemented for many packages, users can manually refine the labels on "outlier" frames.

Another major caveat of deep learning-powered pose estimation is arguably its intrinsic reliance on high-quality labeled images. This suggests that a labeled dataset that reflects the variability of the behavior should be used. If one – due to the quality of the video – cannot reliably identify body parts in still images (i.e., due to massive motion blur, uncertainty about body part (left/right leg crossing) or animal identity) then the video quality should be fixed, or sub-optimal results should be expected.

To give readers a concrete idea about label errors, augmentation methods, and active learning, we also provide some simple experiments with shared code and data. Code for reproducing these analyses is available at github.com/DeepLabCut/Primer-MotionCapture.

To illustrate the importance of error-free labeling, we artificially corrupted labels from the trail-tracking dataset from

Mathis et al. (20). The corruptions respectively simulate inattentive labeling (e.g., with left–right bodyparts being occasionally confounded), and missing annotation or uncertainty as to whether to label an occluded bodypart. We corrupted $1, 5, 10$ and 20% of the dataset (N=1,066 images) either by swapping two labels or removing one, and trained on 5% of the data. The effect of missing labels is barely noticeable (Figure 7A). Swapping labels, on the other hand, causes a substantial drop in performance, with an approximate 10% loss in percentage of correct keypoints (PCK) (Figure 7B). We therefore reason that careful labeling, more so than labeling a very large number of images, is the safest guard against poor ground truth annotations. We believe that explicitly modeling labeling errors, as done in Johnson and Everingham (130), will be an active area of research and integrated in some packages.

Even if labeled well, augmentation greatly improves results and should be used. For instance, when training on the example dataset of (highly)-correlated frames from one short video of one individual, the loss nicely plateaus and shows comparable train/test errors for three different augmentation methods (Figure 8A, B). The three models also give good performance and generalize to a test video of a different mouse. However, closer inspection reveals that the "scalecrop" augmentation method, which only performs cropping and scaling during training (80), leads to swaps in bodyparts with this small training set from only one different mouse (Figure 8C, D). The other two methods, which were configured to perform rotations of the training data, could robustly track the posture of the mouse. This discrepancy becomes striking when observing the PCK plots: imgaug and tensorpack outperform scalecrop by a margin of up to $\approx$ 30% (Figure 8E). One simple way to generalize to this additional case is by active learning (80), which is also available for some packages. Thereby one annotates additional frames with poor performance (outlier frames) and then trains the network from the final configuration, which thus only requires a few thousand iterations. Adding 28 annotated frames from the higher resolution camera, we get good generalization for test frames from both scenarios (Figure 8F). Generally, this illustrates, how the lack of diversity in training data leads to worse performance, but can be fixed by adding frames with poor performance (active learning).

Coping with pitfalls.

Fortunately, dealing with the most common pitfalls is relatively straightforward, and mostly demands caution and common sense. Rules of thumb and practical guidelines are given in Box 5. Video quality should be envisaged as a trade-off between storage limitations, labeling precision, and training speed; e.g., the lower the resolution of a video, the smaller the occupied disk space and the faster the training speed, but the harder it gets to consistently identify bodyparts. In practice, DeepLabCut was shown to be very robust to downsizing and video compression, with pose reconstruction degrading only after scaling videos down to a third of their original size or compression by a factor of 1000 (87).

Body parts should be labeled reliably and consistently across frames that preferably capture a variety of behaviors. Note that some packages provide the user means to automatically extract frames differing in visual content based on unsupervised clustering, which simplifies the selection of relevant images in sparse behaviors.

Utilize symmetries for training with augmentation and try to include image augmentations that are helpful. Use the strongest model (given the speed requirements). Check performance and actively grow the training set if errors are found.

Box 5: Avoiding pitfalls.

- **Video quality**: While deep learning based methods are more robust than other methods, and can even learn from blurry, low-resolution images, you will make your life easier by recording quality videos.
- **Labeling**: Label accurately and use enough data from different videos. 10 videos with 20 frames each is better than 1 video with 200 frames. Check labeling quality. If multiple people label, agree on conventions - i.e. be sure that for a larger body part (like back of mouse) the same location is labeled.
- **Dataset curation**: Collect annotation data from the full repertoire of behavior (different individuals, backgrounds, postures). Automatic methods of frame extraction exist, but the videos need to be manually selected.
- **Data Augmentation**: Are there specific features you know happen in your videos, like motion blur or contrast changes? Can rotational symmetry, or mirroring be exploited? Then use an augmentation scheme that can build this into training.
- **Optimization**: Train until loss plateaus, and do not over-train. Check that it worked by looking at performance on training images (both quantitatively and visually), ideally across “snapshots" (i.e. train iterations of the network). If that works, look at test images. Does the network generalize well? Note that, even if everything is proper, train and test performance can be different due to over-fitting on idiosyncrasies of training set. Bear in mind that the latest iterations may not be the ones yielding the smallest errors on the test set. It is therefore recommended to store and evaluate multiple snapshots.
- **Cross-validation**: You can compare different parameters (networks, augmentation, optimization) to get the best performance (see Figure 7).

Pose estimation algorithms can make different types of errors: jitter, inversion (e.g. left/right), swap (e.g. associating body part to another individual) and miss (131). Depending on the type of errors, different causes need to be addressed (i.e., check the data quality for any human-applied mistakes (20), use suitable augmentation methods). Also for some cases, post processing filters can be useful (such as Kalman filters), but also graphical models or other methods that learn the geometry of the bodyparts. We also believe that future work will explicitly model labeling errors during training.

What to do with motion capture data?

Pose estimation with deep learning is to relieve the user of the painfully slow digitization of keypoints. With markerless tracking you need to annotate a much smaller dataset and this can be applied to new videos. Pose estimation also serves as a springboard to a plethora of other techniques. Indeed, many new tools are specifically being developed to aid users of pose estimation packages to analyze movement and behavioral outputs in a high-throughput manner. Plus, many such packages existed pre-deep learning and can now be leveraged with this new technology as well. While the general topic of what to do with the data is beyond this primer, we will provide a number of pointers. These tools fall into three classes: time series analysis, supervised, and unsupervised learning tools.

A natural step ahead is the quantitative analysis of the keypoint trajectories. The computation of linear and angular displacements, as well as their time derivatives, lays the ground for detailed motor performance evaluation—a great introduction to elementary kinematics can be found in (132), and a thorough description of 151 common metrics is given in (133). These have a broad range of applications, of which we highlight a system for assessing >30 behaviors in groups of mice in an automated way (134), or an investigation of the evolution of gait invariants across animals (135). Furthermore, kinematic metrics are the basis from which to de-

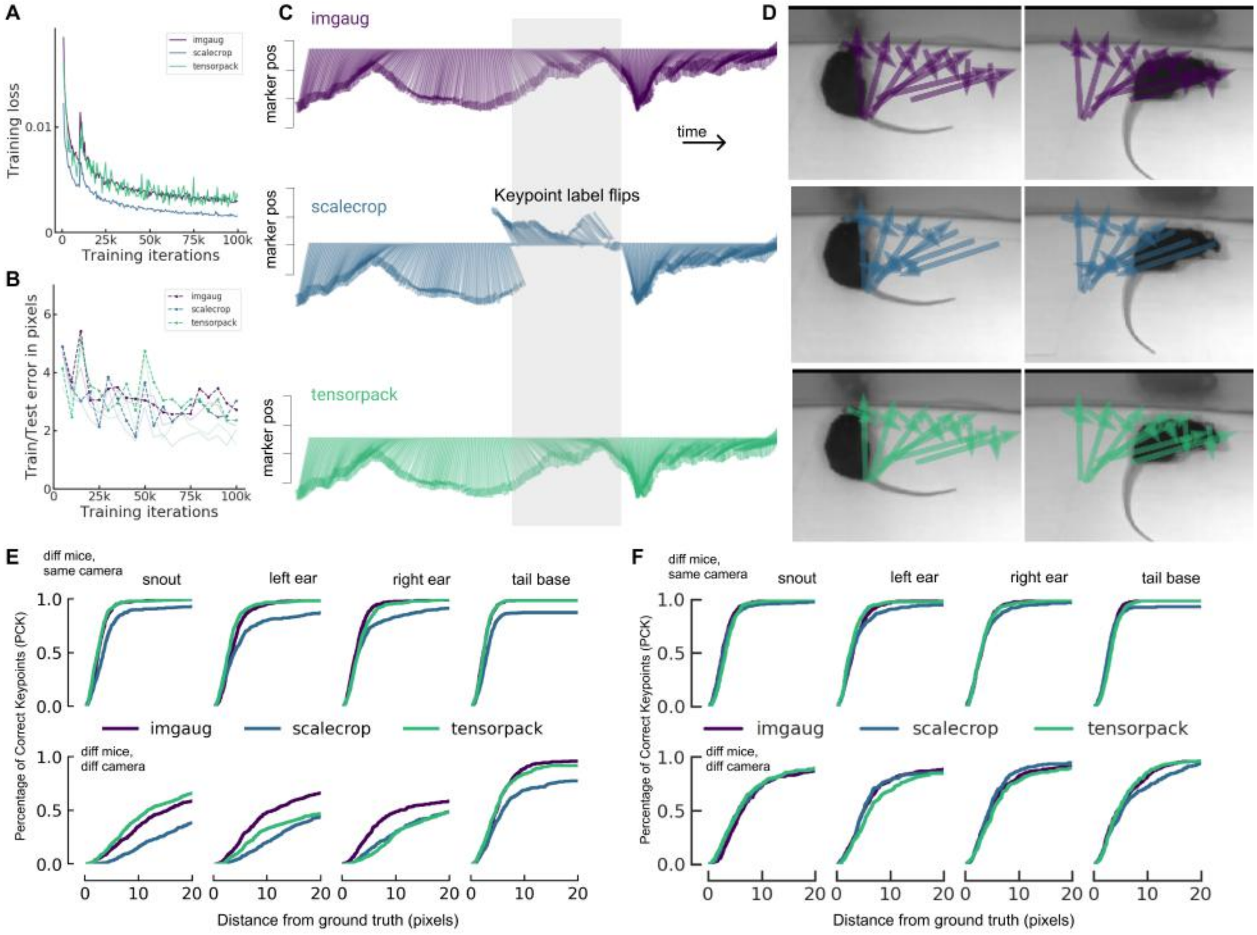


Figure 8. Data Augmentation Improves Performance Performance of three different augmentation methods on the same dataset of around 100 training images from one short video of one mouse (thus correlated). Scalecrop is configured to only change the scale, and randomly crop images; Imgaug also performs motion blur and rotation ($\pm 180°$) augmentation. Tensorpack performs Gaussian noise and rotation ($\pm 180°$) augmentation. **(A)** Loss over training iterations has plateaued, and **(B)** test errors in pixels appear comparable for all methods. **(C)** Tail base aligned skeletons across time for a video of a different mouse (displayed as a cross connecting snout to tail and left ear to right ear). Note the swap of the "T" in the shaded gray zone (and overlaid on the image to the right in **(D)**). Imgaug and tensorpack, which also included full $180°$ rotations, work perfectly). This example highlights that utilizing the rotational symmetry of the data during training can give excellent performance (without additional labeling). **(E)** Performance of the networks on different mice recorded with the same camera (top) and a different camera ($\approx$ 2.5x magnification; bottom). Networks trained with tensorpack and imgaug augmentation generalize much better, and in particular generalize very well to different mice. The generalization to the other camera is difficult, but also works better for tensorpack and imgaug augmentation. **(F)** Performance of networks on same data as in (E), but after an active learning step, adding 28 training frames from the higher resolution camera and training for a few thousand iterations. Afterwards, the network generalizes well to both scenarios.

construct complex whole-body movements into interpretable motor primitives, non-invasively probing neuromuscular control (136). Unsupervised methods such as clustering methods (137), MotionMapper (138), MoSeq (139), or variational autoencoders (140) allow the extraction of common "kinematic behaviors" such as turning, running, rearing. Supervised methods allow the prediction of human defined labels such as "attack" or "'freezing." For this, general purpose tools such as scikit-learn (137) can be ideal, or tailored solutions with integrated GUIs such as JAABA can be used (141). Sturman et al. have developed an open source package to utilize motion capture outputs together with classifiers to automate human annotations for various behavioral tests (open field, elevated plus maze, forced swim test). They showed that these open source methods outperform commercially available platforms (142).

Kinematic analysis, together with simple principles derived from physics, also allows the calculation of the energy required to move about, a methodology relevant to understanding the mechanical determinants of the metabolic cost of locomotion (e.g. 143) or informing the design of bio-inspired robots (e.g. 144, 145).

Modeling and motion understanding.

Looking forward, we also expect that the motion capture data will be used to learn task-driven and data-driven models of the sensorimotor as well as the motor pathway. We have recently provided a blueprint combining human movement data, inverse kinematics, biomechanical modeling and deep learning (146). Given the complexity of movement, as well as the highly nonlinear nature of the sensorimotor processing (145, 147), we believe that such approaches will be fruitful to leverage motion capture data to gain insight into brain function.

Perspectives

As we highlighted thus far in this primer, markerless motion capture has reached a mature state in only a few years due to the many advances in machine learning and computer vision. While there are still some challenges left (10), this is an active area of research and advances in training schemes (such as semi-supervised and self-supervised learning) and model architectures will provide further advances and even less required manual labour. Essentially, now every lab can train appropriate algorithms for their application and turn videos into accurate measurements of posture. If setups are sufficiently standardized, these algorithms already broadly generalize, even across multiple laboratories as in the case of the International Brain Lab (96). But how do we get there, and how do we make sure the needs of animal pose estimation for neuroscience applications are met?

Recent developments in deep learning.

Innovations in the field of object recognition and detection affect all aforementioned parts of the algorithm, as we discussed already in the context of using pre-trained representations. An emerging relevant research direction in machine learning is large scale semi-supervised and self-supervised representation learning (SSL). In SSL, the problem of pretraining representations is no longer dependent on large labeled datasets, as introduced above. Instead, even larger databases comprised of unlabeled examples—often multiple orders of magnitude larger than the counterparts used in supervised learning—can be leveraged. A variety of SSL algorithms are becoming increasingly popular in all areas of machine learning. Recently, representations obtained by large-scale self-supervised pre-training began to approach or even surpass performance of the best supervised methods. Various SSL methods (148–156) made strides in both image recognition (156), speech processing (157–160) and NLP (161, 162), already starting to outperform models obtained by supervised pre-training on large datasets. Considering that recent SSL models for computer vision are continued to being shared openly (e.g. 50, 156), it can be expected to impact and improve new model development in pose estimation, especially if merely replacing the backend model is required. On top, SSL methods can be leveraged in end-to-end models for estimating keypoints and poses directly from raw, unlabeled video (163–165). Approaches based on graph neural networks (166) can encode priors about the observed structure and model correlations between individual keypoints and across time (167). For some applications (like modeling soft tissue or volume) full surface reconstructions are needed and this area has seen tremendous progress in recent years (12, 14, 168). Such advances can be closely watched and incorporated in neuroscience, but we also believe our field (neuroscience) is ready to innovate in this domain too.

Pose estimation specifically for neuroscience.

The goal of human pose estimation—aside from the purely scientific advances for object detection—range from person localization in videos, self-driving cars and pedestrian safety, to socially aware AI, is related to, but does differ from, the applied goals of animal pose estimation in neuroscience. Here, we want tools that give us the highest precision, with the most rapid feedback options possible, and we want to train on small datasets but have them generalize well. This is a tall order, but so far we have seen that the glass is (arguably more than) half full. How do we meet these goals going forward? While much research is still required, there are essentially two ways forward: datasets and associated benchmarks, and algorithms.

Neuroscience needs (more) benchmarks.

In order to push the field towards innovations in areas the community finds important, setting up benchmark datasets and tasks will be crucial (i.e., the Animal version of ImageNet). The community can work towards sharing and collecting data of relevant tasks and curating it into benchmarks. This also has the opportunity of shifting the focus in computer vision research: Instead of "only" doing human pose estimation, researchers probably will start evaluating on datasets directly relevant to neuroscience community. Indeed there has been a recent interest in more animal-related work at top machine learning conferences (14, 169), and providing proper benchmarks for such approaches would be ideal.

For animals, such efforts are developing: Khan et al. recently shared a dataset comprising 22.4K annotated faces from 350 diverse species (169) and Labuguen announced a dataset of 13K annotated macaque (89). We recently released two benchmark datasets that can be evaluated for state-of-the-art performance [4] on within domain and out-of-domain data [5]. The motivation is to train on a limited number of individuals and test on held out animals (the so-called "out-of-domain" issue) (39, 44). We picked horses due to the variation in coat colors (and provide >8K labeled frames). Secondly, to directly study the inherent shift in domain between individuals, we set up a benchmark for common image corruptions, as introduced by Hendrycks et al. (170) that uses the image corruptions library proposed by Michaelis et al. (171).

Of course these aforementioned benchmarks are not sufficient to cover all the needs of the community, so we encourage consortium-style efforts to also curate data and provide additional benchmarks. Plus, making robust networks is still a major challenge, even when trained with large amounts of data (86, 172). In order to make this a possibility it will be important to develop and share common keypoint estimation benchmarks for animals as well as expand the human ones to applications of interest, such as sports (129).

Sharing Pre-trained Models.

We believe another major step forward will be sharing pre-trained pose estimation networks. If as a field we were to annotate sufficiently diverse data, we could train more robust networks that broadly generalize. This success is promised by other large scale data sets such as MS COCO (36) and MPII pose (37). In the computer vision community, sharing model weights such that models do not need to be re-trained has been critical for progress. For example, the ability to download pre-trained ImageNet weights is invaluable—training ImageNet from scratch on a standard GPU can take more than a week. Now, they are downloaded within a few seconds and fine tuned in packages like DeepLabCut. However even for custom training setups, sharing of code and easy access to cloud computing resources enables smaller labs to train and deploy models without investment in additional lab resources. Pre-training a typical object recognition model on the ILSVC is now possible on the order of minutes for less than 100 USD (173) thanks to high-end cloud computing, which is also feasible for labs lacking the necessary on-site infrastructure (Box 3).

In neuroscience, we should aim to fine tune even those models; namely, sharing of mouse-specific, primate-specific weights will drive interest and momentum from researchers without access to such data, and further drive innovations. Currently, only DeepLabCut provides model weights (albeit not at the time of the original publication) as part of the recently launched Model Zoo (modelzoo.deeplabcut.org). Currently it contains models trained on MPII pose (18), dog and cat models as well as contributed models for primate facial recognition, primate full body recognition (89) and mouse pupil detection (Figure 6). Researchers can also contribute in a citizen-science fashion by labeling data on the web (contrib.deeplabcut.org) or by submitting models.

Both datasets and models will benefit from common formatting to ease sharing and testing. Candidate formats are HDF5 (also chosen by NeuroData Without Borders (174) and DeepLabCut), TensorFlow data[6], and/or PyTorch data[7]. Specifically, for models, proto-buffer formats for weights are useful and easy to share (17, 175) for deployment to other systems. Platforms such as OSF and Zenodo allow banking of weights, and some papers (e.g. 91, 142) have also shared their trained models. We envision that having easy-to-use interfaces to such models will be possible in the future.

These pre-trained pose estimation networks hold several promises: it saves time and energy (as different labs do not need to annotate and train networks), as well as contributes to reproducibility in science. Like many other forms of biological data, such as genome sequences, functional imaging data, behavioral data is notoriously hard to analyze in standardized ways. Lack of agreement can lead to different results, as pointed out by a recent landmark study comparing the results achieved by 70 independent researchers analyzing nine hypothesis in shared imaging data (176). To increase reproducibility in behavioral science, video is a great tool (177). Analyzing behavioral data is complex, owing to its unstructured, large-scale nature, which highlights the importance of shared analysis pipelines. Thus, building robust architectures that extract the same behavioral measurements in different laboratories would be a major step forward.

Conclusions

Deep learning based markerless pose estimation has been broadly and rapidly adopted in the past two years. This impact was, in part, fueled by open-source code: by developing and sharing packages in public repositories on GitHub they could be easily accessed for free and at scale. These packages are built on advances (and code) in computer vision and AI, which has a strong open science culture. Neuroscience also has strong and growing open science culture (178), which greatly impacts the field as evidenced by tools from the Allen Institute, the UCLA Miniscope (179), OpenEphys (180), and Bonsai (175) (just to name a few).

Moreover, Neuroscience and AI have a long history of influencing each other (181), and research in Neuroscience will likely contribute to making AI more robust (181, 182). The analysis of animal motion is a highly interdisciplinary field at the intersection of biomechanics, computer vision, medicine

[4] paperswithcode.com
[5] horse10.deeplabcut.org
[6] tensorflow.org/api_docs/python/tf/data
[7] pytorch.org/docs/stable/torchvision/datasets.html

and robotics with a long tradition (1). The recent advances in deep learning have greatly simplified the measurement of animal behavior, which, as we and others believe (183), in turn will greatly advance our understanding of the brain.

Acknowledgments:

We thank Yash Sharma for discussions around future directions in self-supervised learning, Erin Diel, Maxime Vidal, Claudio Michaelis, Thomas Biasi for comments on the manuscript. Funding was provided by the Rowland Institute at Harvard University (MWM, AM), the Chan Zuckerberg Initiative (MWM, AM, JL) and the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (StS; FKZ: 01IS18039A). StS thanks the International Max Planck Research School for Intelligent Systems (IMPRS-IS) and acknowledges his membership in the European Laboratory for Learning & Intelligent Systems (ELLIS) PhD program. The authors declare no conflicts of interest. M.W.M. dedicates this work to Adam E. Max.

References

1. Reinhard Klette and Garry Tee. Understanding human motion: A historic review. In *Human motion*, pages 1–22. Springer, 2008.
2. Mary D Leakey and Richard L Hay. Pliocene footprints in the laetolil beds at laetoli, northern tanzania. *Nature*, 278(5702):317–323, 1979.
3. Roland Kays, Margaret C Crofoot, Walter Jetz, and Martin Wikelski. Terrestrial animal tracking as an eye on life and planet. *Science*, 348(6240):aaa2478, 2015.
4. Danielle D Brown, Roland Kays, Martin Wikelski, Rory Wilson, and A Peter Klimley. Observing the unwatchable through acceleration logging of animal behavior. *Animal Biotelemetry*, 1(1):20, 2013.
5. Valentina Camomilla, Elena Bergamini, Silvia Fantozzi, and Giuseppe Vannozzi. Trends supporting the in-field use of wearable inertial sensors for sport performance evaluation: A systematic review. *Sensors*, 18(3):873, 2018.
6. Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14(2):201–211, 1973.
7. Allan F O'Connell, James D Nichols, and K Ullas Karanth. *Camera traps in animal ecology: methods and analyses*. Springer Science & Business Media, 2010.
8. Ben G Weinstein. A computer vision for animal ecology. *Journal of Animal Ecology*, 87(3):533–545, 2018.
9. Xiongwei Wu, Doyen Sahoo, and Steven CH Hoi. Recent advances in deep learning for object detection. *Neurocomputing*, 2020.
10. 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.
11. 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.
12. Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7297–7306, 2018.
13. Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael Black. 3d menagerie: Modeling the 3d shape and pose of animals. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 11 2016.
14. Artsiom Sanakoyeu, Vasil Khalidov, Maureen S McCarthy, Andrea Vedaldi, and Natalia Neverova. Transferring dense pose to proximal animal classes. *arXiv preprint arXiv:2003.00080*, 2020.
15. Samsoon Inayat, Surjeet Singh, Arashk Ghasroddashti, Qandeel Qandeel, Pramuka Egodage, Ian Q Whishaw, and Majid H Mohajerani. A matlab-based toolbox for characterizing behavior of rodents engaged in string-pulling. *eLife*, 9:e54540, 2020.
16. Pablo Maceira-Elvira, Traian Popa, Anne-Christine Schmid, and Friedhelm C Hummel. Wearable technology in stroke rehabilitation: towards improved diagnosis and treatment of upper-limb motor impairment. *Journal of neuroengineering and rehabilitation*, 16 (1):142, 2019.
17. Gary Kane, Gonçalo Lopes, Jonny L. Saunders, Alexander Mathis, and Mackenzie Mathis. Real-time deeplabcut for closed-loop feedback based on posture. *bioRxiv*, 2020.
18. Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. DeeperCut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer, 2016.
19. Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. Pifpaf: Composite fields for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11977–11986, 2019.
20. 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.
21. Thomas B Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2-3):90–126, 2006.
22. Ronald Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(1):4 – 18, 2007. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2006.10.016. Special Issue on Vision for Human-Computer Interaction.
23. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
24. Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659, 2013.
25. Arjun Jain, Jonathan Tompson, Yann LeCun, and Christoph Bregler. Modeep: A deep learning framework using motion features for human pose estimation. *CoRR*, abs/1409.7963, 2014.
26. Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.
27. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018.
28. Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 466–481, 2018.
29. Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5386–5395, 2020.
30. Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.
31. David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
32. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.
33. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
34. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
35. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115 (3):211–252, 2015.
36. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

37. Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.
38. Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. *arXiv preprint arXiv:1811.08883*, 2018.
39. Alexander Mathis, Mert Yüksekgönül, Byron Rogers, Matthias Bethge, and Mackenzie W Mathis. Pretraining boosts out-of-domain robustness for pose estimation. *arXiv preprint arXiv:1909.11229*, 2019.
40. Ahmet Arac, Pingping Zhao, Bruce H Dobkin, S Thomas Carmichael, and Peyman Golshani. Deepbehavior: A deep learning toolbox for automated analysis of animal and human behavior imaging data. *Frontiers in systems neuroscience*, 13:20, 2019.
41. 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, oct 2019. ISSN 2050-084X. doi: 10.7554/eLife.47994.
42. Praneet C. Bala, Benjamin R. Eisenreich, Seng Bum Michael Yoo, Benjamin Y. Hayden, Hyun Soo Park, and Jan Zimmermann. Openmonkeystudio: Automated markerless pose estimation in freely moving macaques. *bioRxiv*, 2020. doi: 10.1101/2020.01.31.928861.
43. XiaoLe Liu, Si-yang Yu, Nico Flierman, Sebastian Loyola, Maarten Kamermans, Tycho M. Hoogland, and Chris I. De Zeeuw. Optiflex: video-based animal pose estimation using deep learning enhanced by optical flow. *bioRxiv*, 2020. doi: 10.1101/2020.04.04.025494.
44. Alexander Mathis, Thomas Biasi, Y Mert, Byron Rogers, Matthias Bethge, and Mackenzie Weygandt Mathis. Imagenet performance correlates with pose estimation robustness and generalization on out-of-domain data. *International Conference on Machine Learning 2020 Workshop on Uncertainty and Robustness in Deep Learning*, 2020.
45. Talmo D Pereira, Diego E Aldarondo, Lindsay Willmore, Mikhail Kislin, Samuel S-H Wang, Mala Murthy, and Joshua W Shaevitz. Fast animal pose estimation using deep neural networks. *Nature methods*, 16(1):117, 2019.
46. Semih Günel, Helge Rhodin, Sergei Manzhos, Joao Luiz Campagnolo, Ramdya, and Pascal Fua. Deepfly 3 d : A deep learning-based 1 approach for 3 d limb and appendage 2 tracking in tethered , adult drosophila. 2019.
47. Christian Zimmermann, Artur Schneider, Mansour Alyahyay, Thomas Brox, and Ilka Diester. Freipose: A deep learning framework for precise animal motion capture in 3d spaces. *bioRxiv*, 2020. doi: 10.1101/2020.02.27.967620.
48. Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.
49. Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
50. Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.
51. Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018.
52. Hengduo Li, Bharat Singh, Mahyar Najibi, Zuxuan Wu, and Larry S Davis. An analysis of pre-training on object detection. *arXiv preprint arXiv:1904.05871*, 2019.
53. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.
54. 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.
55. Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
56. Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.
57. Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2661–2671, 2019.
58. Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.
59. Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399, 2018.
60. Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.
61. 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. In *International conference on machine learning*, pages 647–655, 2014.
62. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
63. Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.
64. Yuxin Wu et al. Tensorpack. https://github.com/tensorpack/, 2016.
65. Alexander B. Jung, Kentaro Wada, Jon Crall, Satoshi Tanaka, Jake Graving, Christoph Reinders, Sarthak Yadav, Joy Banerjee, Gábor Vecsei, Adam Kraft, Zheng Rui, Jirka Borovec, Christian Vallentin, Semen Zhydenko, Kilian Pfeiffer, Ben Cook, Ismael Fernández, François-Michel De Rainville, Chi-Hung Weng, Abner Ayala-Acevedo, Raphael Meudec, Matias Laporte, et al. imgaug. https://github.com/aleju/imgaug, 2020. Online; accessed 01-Feb-2020.
66. Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.
67. Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.
68. Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5693–5703, 2019.
69. Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.
70. Semih Günel, Helge Rhodin, Daniel Morales, João H 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, oct 2019. ISSN 2050-084X. doi: 10.7554/eLife.48571.
71. Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4733–4742, 2016.
72. Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.
73. Eldar Insafutdinov, Mykhaylo Andriluka, Leonid Pishchulin, Siyu

Tang, Evgeny Levinkov, Bjoern Andres, and Bernt Schiele. Arttrack: Articulated multi-person tracking in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

74. Yuan Yao, Yasamin Jafarian, and Hyun Soo Park. Monet: Multiview semi-supervised keypoint detection via epipolar divergence. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 753–762, 2019.
75. Lan Xu, Weipeng Xu, Vladislav Golyanik, Marc Habermann, Lu Fang, and Christian Theobalt. Eventcap: Monocular 3d capture of high-speed human motions using an event camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4968–4978, 2020.
76. Yuxiao Zhou, Marc Habermann, Weipeng Xu, Ikhsanul Habibie, Christian Theobalt, and Feng Xu. Monocular real-time hand shape and motion capture using multi-modal data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5346–5355, 2020.
77. Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.
78. Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.
79. Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
80. 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.
81. Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2640–2649, 2017.
82. Dushyant Mehta, Helge Rhodin, Dan Casas, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation using transfer learning and improved CNN supervision. *CoRR*, abs/1611.09813, 2016.
83. Denis Tomè, Chris Russell, and Lourdes Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. *CoRR*, abs/1701.00295, 2017.
84. Ching-Hang Chen and Deva Ramanan. 3d human pose estimation= 2d pose estimation+ matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7035–7043, 2017.
85. Pierre Karashchuk, Katie L Rupp, Evyn S Dickinson, Elischa Sanders, Eiman Azim, Bingni W Brunton, and John C Tuthill. Anipose: a toolkit for robust markerless 3d pose estimation. *bioRxiv*, 2020.
86. Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *arXiv preprint arXiv:2004.07780*, 2020.
87. Alexander Mathis and Richard A. Warren. On the inference speed and video-compression robustness of deeplabcut. *bioRxiv*, 2018. doi: 10.1101/457242.
88. Michael Berger, Naubahar Shahryar Agha, and Alexander Gail. Wireless recording from unrestrained monkeys reveals motor goal encoding beyond immediate reach in frontoparietal cortex. *Elife*, 9:e51322, 2020.
89. Rollyn Labuguen, Jumpei Matsumoto, Salvador Negrete, Hiroshi Nishimaru, Hisao Nishijo, Masahiko Takada, Yasuhiro Go, Kenichi Inoue, and Tomohiro Shibata. Macaquepose: A novel 'in the wild'macaque monkey pose dataset for markerless motion capture. *bioRxiv*, 2020.
90. Teppei Ebina, Keitaro Obara, Akiya Watakabe, Yoshito Masamizu, Shin-Ichiro Terada, Ryota Matoba, Masafumi Takaji, Nobuhiko Hatanaka, Atsushi Nambu, Hiroaki Mizukami, et al. Arm movements induced by noninvasive optogenetic stimulation of the motor cortex in the common marmoset. *Proceedings of the National Academy of Sciences*, 116(45):22844–22850, 2019.
91. John M Barrett, Martinna G Raineri Tapies, and Gordon MG Shepherd. Manual dexterity of mice during food-handling involves the thumb and a set of fast basic movements. *PloS one*, 15(1): e0226774, 2020.
92. Harris S Kaplan and Manuel Zimmer. Brain-wide representations of ongoing behavior: a universal principle? *Current opinion in neurobiology*, 64:60–69, 2020.
93. Steven M Peterson, Satpreet H Singh, Nancy XR Wang, Rajesh PN Rao, and Bingni W Brunton. Behavioral and neural variability of naturalistic arm movements. *BioRxiv*, 2020.
94. Arne F Meyer, John O'Keefe, and Jasper Poort. Two distinct types of eye-head coupling in freely moving mice. *bioRxiv*, 2020.
95. Joshua H Siegle, Xiaoxuan Jia, Séverine Durand, Sam Gale, Corbett Bennett, Nile Graddis, Greggory Heller, Tamina K Ramirez, Hannah Choi, Jennifer A Luviano, et al. A survey of spiking activity reveals a functional hierarchy of mouse corticothalamic visual areas. *bioRxiv*, page 805010, 2019.
96. Kenneth D Harris, Max Hunter, Cyrille Rossant, Maho Sasaki, Shan Shen, Nicholas A Steinmetz, Edgar Y Walker, Olivier Winter, and Miles Wells. Data architecture for a large-scale neuroscience collaboration. *BioRxiv*, page 827873, 2019.
97. Irene Tracey, Clifford J Woolf, and Nick A Andrews. Composite pain biomarker signatures for objective assessment and effective treatment. *Neuron*, 101(5):783–800, 2019.
98. Silvestro Micera, Matteo Caleo, Carmelo Chisari, Friedhelm C Hummel, and Alessandra Pedrocchi. Advanced neurotechnologies for the restoration of motor function. *Neuron*, 105(4):604–620, 2020.
99. J. D. Laurence-Chasen, A. R. Manafzadeh, N. G. Hatsopoulos, C. F. Ross, and F. I. Arce-McShane. Integrating xmalab and deeplabcut for high-throughput xromm. *Journal of Experimental Biology*, 2020. ISSN 0022-0949. doi: 10.1242/jeb.226720.
100. Nidhi Seethapathi, Shaofei Wang, Rachit Saluja, Gunnar Blohm, and Konrad P Kording. Movement science needs different pose tracking algorithms. *arXiv preprint arXiv:1907.10226*, 2019.
101. Curtis T Rueden, Jeanelle Ackerman, Ellen T Arena, Jan Eglinger, Beth A Cimini, Allen Goodman, Anne E Carpenter, and Kevin W Eliceiri. Scientific community image forum: A discussion forum for scientific image software. *PLoS biology*, 17(6):e3000340, 2019.
102. Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, et al. Fiji: an open-source platform for biological-image analysis. *Nature methods*, 9(7):676–682, 2012.
103. Claire McQuin, Allen Goodman, Vasiliy S Chernyshev, Lee Kamentsky, Beth A Cimini, Kyle W. Karhohs, Minh Doan, Liya Ding, Susanne M. Rafelski, Derek J. Thirstrup, Winfried Wiegraebe, Shantanu Singh, Tim Becker, Juan C. Caicedo, and Anne E. Carpenter. Cellprofiler 3.0: Next-generation image processing for biology. *PLoS Biology*, 16, 2018.
104. Christoph Sommer, Christoph Straehle, Ullrich Koethe, and Fred A Hamprecht. Ilastik: Interactive learning and segmentation toolkit. In *2011 IEEE international symposium on biomedical imaging: From nano to macro*, pages 230–233. IEEE, 2011.
105. Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.
106. Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2, 2014.
107. DK Ramsey, PF Wretenberg, DL Benoit, M Lamontagne, and G Nemeth. Methodological concerns using intra-cortical pins to measure tibiofemoral kinematics. *Knee Surgery, Sports Traumatology, Arthroscopy*, 11(5):344–349, 2003.
108. Renate List, Barbara Postolka, Pascal Schütz, Marco Hitz, Peter Schwilch, Hans Gerber, Stephen J Ferguson, and William R Taylor. A moving fluoroscope to capture tibiofemoral kinematics during complete cycles of free level and downhill walking as well as stair descent. *PloS one*, 12(10):e0185952, 2017.
109. Valentina Camomilla, Raphaël Dumas, and Aurelio Cappozzo. Human movement analysis: The soft tissue artefact issue. *Journal of Biomechanics*, 62:1 – 4, 2017. ISSN 0021-9290. doi: https://doi.org/10.1016/j.jbiomech.2017.09.001. Human Movement Analysis: The Soft Tissue Artefact Issue.
110. Kenneth B. Smale, Brigitte M. Potvin, Mohammad S. Shourijeh, and Daniel L. Benoit. Knee joint kinematics and kinetics during the hop and cut after soft tissue artifact suppression: Time to recon-

sider acl injury mechanisms? *Journal of Biomechanics*, 62:132 – 139, 2017.
111. Mickaël Begon, Michael Skipper Andersen, and Raphaël Dumas. Multibody Kinematics Optimization for the Estimation of Upper and Lower Limb Human Joint Kinematics: A Systematized Methodological Review. *Journal of Biomechanical Engineering*, 140(3), 2018.
112. T-W Lu and JJ O'connor. Bone position estimation from skin marker co-ordinates using global optimisation with joint constraints. *Journal of biomechanics*, 32(2):129–134, 1999.
113. Elena Ceseracciu, Zimi Sawacha, and Claudio Cobelli. Comparison of markerless and marker-based motion capture technologies through simultaneous data collection during gait: proof of concept. *PloS one*, 9(3):e87640, 2014.
114. CW Spoor and FE Veldpaus. Rigid body motion calculated from spatial co-ordinates of markers. *Journal of biomechanics*, 13(4): 391–393, 1980.
115. Mingmin Zhao, Tianhong Li, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, and Dina Katabi. Through-wall human pose estimation using radio signals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7356–7365, 2018.
116. Pritish Chakravarty, Gabriele Cozzi, Arpat Ozgul, and Kamiar Aminian. A novel biomechanical approach for animal behaviour recognition using accelerometers. *Methods in Ecology and Evolution*, 10(6):802–814, 2019.
117. OR Bidder, JS Walker, MW Jones, MD Holton, P Urge, DM Scantlebury, NJ Marks, EA Magowan, IE Maguire, and RP Wilson. Step by step: reconstruction of terrestrial animal movement paths by dead-reckoning. *Movement ecology*, 3(1):1–16, 2015.
118. Alan M Wilson, Tatjana Y Hubel, Simon D Wilshin, John C Lowe, Maja Lorenc, Oliver P Dewhirst, Hattie LA Bartlam-Brooks, Rebecca Diack, Emily Bennitt, Krystyna A Golabek, et al. Biomechanics of predator–prey arms race in lion, zebra, cheetah and impala. *Nature*, 554(7691):183–188, 2018.
119. Adrian C Gleiss, Rory P Wilson, and Emily LC Shepard. Making overall dynamic body acceleration work: on the theory of acceleration as a proxy for energy expenditure. *Methods in Ecology and Evolution*, 2(1):23–33, 2011.
120. Matthieu O Pasquet, Matthieu Tihy, Aurélie Gourgeon, Marco N Pompili, Bill P Godsil, Clément Léna, and Guillaume P Dugué. Wireless inertial measurement of head kinematics in freely-moving rats. *Scientific reports*, 6:35689, 2016.
121. Daniel Roetenberg, Henk Luinge, and Per Slycke. Xsens mvn: Full 6dof human motion tracking using miniature inertial sensors. *Xsens Motion Technologies BV, Tech. Rep*, 1, 2009.
122. Timo von Marcard, Bodo Rosenhahn, Michael J Black, and Gerard Pons-Moll. Sparse inertial poser: Automatic 3d human pose estimation from sparse imus. In *Computer Graphics Forum*, volume 36, pages 349–360. Wiley Online Library, 2017.
123. Angelo Maria Sabatini. Estimating three-dimensional orientation of human body parts by inertial/magnetic sensing. *Sensors*, 11(2): 1489–1525, 2011.
124. Bingfei Fan, Qingguo Li, and Tao Liu. How magnetic disturbance influences the attitude and heading in magnetic and inertial sensor-based orientation estimation. *Sensors*, 18(1):76, 2018.
125. Julien Lebleu, Thierry Gosseye, Christine Detrembleur, Philippe Mahaudens, Olivier Cartiaux, and Massimo Penta. Lower limb kinematics using inertial sensors during locomotion: Accuracy and reproducibility of joint angle calculations with different sensor-to-segment calibrations. *Sensors*, 20(3):715, 2020.
126. Laura Susana Vargas-Valencia, Arlindo Elias, Eduardo Rocon, Teodiano Bastos-Filho, and Anselmo Frizera. An imu-to-body alignment method applied to human gait analysis. *Sensors*, 16 (12):2090, 2016.
127. Andrew Gilbert, Matthew Trumble, Charles Malleson, Adrian Hilton, and John Collomosse. Fusing visual and inertial sensors with semantics for 3d human pose estimation. *International Journal of Computer Vision*, 127(4):381–397, 2019.
128. Roberto Henschel, Timo von Marcard, and Bodo Rosenhahn. Simultaneous identification and tracking of multiple people using video and imus. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
129. Ying Huang, Bin Sun, Haipeng Kan, Jiankai Zhuang, and Zengchang Qin. Followmeup sports: New benchmark for 2d human keypoint recognition. In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pages 110–121. Springer, 2019.
130. Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. In *CVPR 2011*, pages 1465–1472. IEEE, 2011.
131. Matteo Ruggero Ronchi and Pietro Perona. Benchmarking and error diagnosis in multi-instance pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 369–378, 2017.
132. D.A. Winter. *Biomechanics and motor control of human movement*. John Wiley & Sons, 2009.
133. Anne Schwarz, Christoph M Kanzler, Olivier Lambercy, Andreas R Luft, and Janne M Veerbeek. Systematic review on kinematic assessments of upper limb movements after stroke. *Stroke*, 50(3): 718–727, 2019.
134. Fabrice de Chaumont, Elodie Ey, Nicolas Torquet, Thibault Lagache, Stéphane Dallongeville, Albane Imbert, Thierry Legou, Anne-Marie Le Sourd, Philippe Faure, Thomas Bourgeron, et al. Real-time analysis of the behaviour of groups of mice via a depth-sensing camera and machine learning. *Nature biomedical engineering*, 3(11):930–942, 2019.
135. Giovanna Catavitello, Yury Ivanenko, and Francesco Lacquaniti. A kinematic synergy for terrestrial locomotion shared by mammals and birds. *Elife*, 7:e38190, 2018.
136. Alessia Longo, Thomas Haid, Ruud Meulenbroek, and Peter Federolf. Biomechanics in posture space: Properties and relevance of principal accelerations for characterizing movement control. *Journal of Biomechanics*, 82:397–403, 2019.
137. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12 (Oct):2825–2830, 2011.
138. Gordon J. Berman, Daniel M. Choi, William Bialek, and Joshua W. Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99), 2014. ISSN 1742-5689. doi: 10.1098/rsif.2014.0672.
139. Alexander B Wiltschko, Matthew J Johnson, Giuliano Iurilli, Ralph E Peterson, Jesse M Katon, Stan L Pashkovski, Victoria E Abraira, Ryan P Adams, and Sandeep Robert Datta. Mapping sub-second structure in mouse behavior. *Neuron*, 88(6):1121–1135, 2015.
140. Kevin Luxem, Falko Fuhrmann, Johannes Kürsch, Stefan Remy, and Pavol Bauer. Identifying behavioral structure from deep variational embeddings of animal motion. *bioRxiv*, 2020.
141. Mayank Kabra, Alice A Robie, Marta Rivera-Alba, Steven Branson, and Kristin Branson. Jaaba: interactive machine learning for automatic annotation of animal behavior. *Nature methods*, 10(1):64, 2013.
142. 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*, 2020.
143. Franco Saibene and Alberto E Minetti. Biomechanical and physiological aspects of legged locomotion in humans. *European journal of applied physiology*, 88(4-5):297–316, 2003.
144. Chen Li, Chad C Kessens, Ronald S Fearing, and Robert J Full. Mechanical principles of dynamic terrestrial self-righting using wings. *Advanced Robotics*, 31(17):881–900, 2017.
145. John A Nyakatura, Kamilo Melo, Tomislav Horvat, Kostas Karakasiliotis, Vivian R Allen, Amir Andikfar, Emanuel Andrada, Patrick Arnold, Jonas Lauströer, John R Hutchinson, et al. Reverse-engineering the locomotion of a stem amniote. *Nature*, 565(7739):351, 2019.
146. Kai J Sandbrink, Pranav Mamidanna, Claudio Michaelis, Mackenzie Weygandt Mathis, Matthias Bethge, and Alexander Mathis. Task-driven hierarchical deep neural networkmodels of the proprioceptive pathway. *bioRxiv*, 2020.
147. Manu S Madhav and Noah J Cowan. The synergy between neuroscience and control theory: the nervous system as inspiration for hard control challenges. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:243–267, 2020.
148. Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Represen-

tation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

149. Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*, 2018.
150. Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.
151. Olivier J Hénaff, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.
152. Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.
153. R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.
154. Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15509–15519, 2019.
155. Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.
156. Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.
157. Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. *Proc. Interspeech 2019*, pages 3465–3469, 2019.
158. Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations (ICLR)*, 2020.
159. Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, 2020.
160. Mirco Ravanelli, Jianyuan Zhong, Santiago Pascual, Pawel Swietojanski, Joao Monteiro, Jan Trmal, and Yoshua Bengio. Multitask self-supervised learning for robust speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6989–6993. IEEE, 2020.
161. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, 2019.
162. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.
163. Rafi Umer, Andreas Doering, Bastian Leibe, and Juergen Gall. Self-supervised keypoint correspondences for multi-person pose estimation and tracking in videos. *arXiv preprint arXiv:2004.12652*, 2020.
164. Hsiao-Yu Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems*, pages 5236–5246, 2017.
165. Muhammed Kocabas, Salih Karagoz, and Emre Akbas. Self-supervised learning of 3d human pose using multi-view geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1077–1086, 2019.
166. Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.
167. Yujun Cai, Liuhao Ge, Jun Liu, Jianfei Cai, Tat-Jen Cham, Junsong Yuan, and Nadia Magnenat Thalmann. Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2272–2281, 2019.
168. Silvia Zuffi, Angjoo Kanazawa, Tanja Berger-Wolf, and Michael Black. Three-d safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *ICCV*. IEEE Computer Society, 08 2019.
169. Muhammad Haris Khan, John McDonagh, Salman Khan, Muhammad Shahabuddin, Aditya Arora, Fahad Shahbaz Khan, Ling Shao, and Georgios Tzimiropoulos. Animalweb: A large-scale hierarchical dataset of annotated animal faces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6939–6948, 2020.
170. Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *ICML*, 2019.
171. Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.
172. Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 456–473, 2018.
173. Cody Coleman, Deepak Narayanan, Daniel Kang, Tian Zhao, Jian Zhang, Luigi Nardi, Peter Bailis, Kunle Olukotun, Chris Ré, and Matei Zaharia. Dawnbench: An end-to-end deep learning benchmark and competition. *Neural Information Processing Systems Workshops*, 2017.
174. Jeffery L Teeters, Keith Godfrey, Rob Young, Chinh Dang, Claudia Friedsam, Barry Wark, Hiroki Asari, Simon Peron, Nuo Li, Adrien Peyrache, et al. Neurodata without borders: creating a common data format for neurophysiology. *Neuron*, 88(4):629–634, 2015.
175. Gonçalo Lopes, Niccolò Bonacchi, João Frazão, Joana P Neto, Bassam V Atallah, Sofia Soares, Luís Moreira, Sara Matias, Pavel M Itskov, Patrícia A Correia, et al. Bonsai: an event-based framework for processing and controlling data streams. *Frontiers in neuroinformatics*, 9:7, 2015.
176. Rotem Botvinik-Nezer, Felix Holzmeister, Colin F Camerer, Anna Dreber, Juergen Huber, Magnus Johannesson, Michael Kirchler, Roni Iwanir, Jeanette A Mumford, R Alison Adcock, et al. Variability in the analysis of a single neuroimaging dataset by many teams. *Nature*, pages 1–7, 2020.
177. Rick O Gilmore and Karen E Adolph. Video can make behavioural science more reproducible. *Nature human behaviour*, 1(7), 2017.
178. Samantha R White, Linda M Amarante, Alexxai V Kravitz, and Mark Laubach. The future is open: Open-source tools for behavioral neuroscience research. *Eneuro*, 6(4), 2019.
179. Daniel Aharoni, Baljit S Khakh, Alcino J Silva, and Peyman Golshani. All the light that we can see: a new era in miniaturized microscopy. *Nature methods*, 16(1):11–13, 2019.
180. Joshua H Siegle, Aarón Cuevas López, Yogi A Patel, Kirill Abramov, Shay Ohayon, and Jakob Voigts. Open ephys: an open-source, plugin-based platform for multichannel electrophysiology. *Journal of neural engineering*, 14(4):045003, 2017.
181. Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, 2017.
182. Fabian H. Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S. Tolias. Engineering a less artificial intelligence. *Neuron*, 103(6):967 – 979, 2019. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2019.08.034.
183. John W Krakauer, Asif A Ghazanfar, Alex Gomez-Marin, Malcolm A MacIver, and David Poeppel. Neuroscience needs behavior: correcting a reductionist bias. *Neuron*, 93(3):480–490, 2017.