A Framework for Generalizable Neural Networks for Robust Estimation of Eyelids and Pupils

Arnab Biswas and Mark Lescroart

University of Nevada Reno

Author Note

Correspondence concerning this article should be addressed to Arnab Biswas, University of Nevada, Reno, Effie Mona Mack 413, 1664 N Virginia St., Reno, NV 89557. E-mail: arnab@nevada.unr.edu

Abstract

Deep Neural Networks (DNNs) have enabled recent advances in the accuracy and robustness of video-oculography. However, to make robust predictions most DNN models require extensive and diverse training data, which is costly to collect and label. In this work, we seek to improve the cost/benefit ratio of labeling to model performance. We develop pylids, a pupil- and eyelid-estimation DNN model based on DeepLabCut. We show that performance of pylids-based pupil estimation can be related to the distance of test data from the distribution of training data. Based on this principle we explore methods for efficient data selection for training our DNN. We show that guided sampling of new data points from the training data approaches state-of-the-art pupil and eyelid estimation with fewer training data points. We also demonstrate the benefit of using an efficient sampling method to select data augmentations for training DNNs. These sampling methods aim to minimize the time and effort required to label and train DNNs while promoting model generalization on new diverse datasets.

Keywords: eyetracking, pupil estimation, eyelid estimation, deep neural networks

A Framework for Generalizable Neural Networks for Robust Estimation of Eyelids and Pupils

Introduction

Classical video-based eyetracking techniques based on glint tracking (Hennessey et al., 2006) and more recent ones using other image processing methods to segment the pupil (Kassner et al., 2014) are good at estimating the pupil position across time, but only when used in a very controlled environment with non variable lighting, fixed head positioning and a fully visible pupil. These techniques also rely on using heuristics, such as elliptical edges with certain contrast, to detect the pupil and it is often challenging to optimize the parameters used in these models. The advent of deep neural networks (DNNs) has provided the eyetracking community with a tool that circumvents these issues of parameter optimization and makes it possible to move out of controlled environments and do eyetracking in the wild. DNNs allow us to accurately detect pupils (Fuhl, Santini, Kasneci, et al., 2017; Vera-Olmos et al., 2019; Yiu et al., 2019) as well as the full anatomical shape of the eye (Biswas et al., 2021; Chaudhary et al., 2019; Kansal & Devanathan, 2019; R. S. Kothari et al., 2020; Rot et al., 2018). Despite their success, DNN-based models often perform worse on datasets with image variability very different from the one on which the DNN was trained (Kouw & Loog, 2018; Torralba & Efros, 2011).

In the domain of eyetracking, a recent study found that a neural network-based eye segmentation model had very high variability in performance depending upon the training and test data (R. S. Kothari et al., 2022). For a given test dataset, pupil center estimation error was as low as 0.638 pixels when trained on data sampled from the same distribution as the the test dataset. This error increased to as high as 4.483 pixels when the model was trained on data from a different dataset. A study by (Binaee et al., 2021) also showed that DNN models trained on indoor eye image data perform poorly when evaluated on an outdoor eye video dataset. These results demonstrate the challenges of using a pre-trained DNN model for eyetracking on new diverse datasets.

Manually labeling data for training DNNs is often very time consuming. Annotating a single frame can take up to 5 minutes based on the complexity and number of labels used. Some studies have tried to reduce the time and effort involved by using automated labeling and then refining the labels manually (Rot et al., 2018), but this method still requires manual analysis of every label in the dataset. Other studies have used semi-supervised methods to increase the number of available labels (Chaudhary et al., 2021; Fuhl et al., 2021) which although helpful does not always result in the most accurate labels. The problem of manually labeling frames is further compounded by the fact that the inclusion of every additional labeled frame does not uniformly increase DNN performance. A recent study demonstrated that up to 50% of the training data can be discarded while preserving model accuracy for a deep neural network classifier (Coleman et al., 2019). The issue of redundant frames is particularly prevalent in eye tracking videos due to the high degree of autocorrelation between frames. Hence, collecting and labeling more data with the hope of sampling the tails of the distributions of eye images is likely to be less efficient than a principled sampling strategy. In this study we develop a method to select frames to label and train a DNN to improve its performance on eye-image datasets for eyetracking.

Apart from labeling additional data, another common technique to increase the amount and diversity of training data available is data augmentation. This is a proven technique to improve DNN model performance that does not require manual labeling, for a review see (Shorten & Khoshgoftaar, 2019). Data augmentation expands the training set by using image processing techniques to add perturbations to samples and labels already part of the training data. This can increase the image variability of a dataset and help remove spurious correlations. For example, in recent studies (Chaudhary et al., 2021; Eivazi et al., 2019) authors demonstrate that using domain-specific augmentation such as addition of reflections, mock pupils, mock glints and blurring led to out-performing state-of-the-art pupil detectors when testing for generalization to new datasets.

The selection of perturbations and corresponding range of intensities to use for data

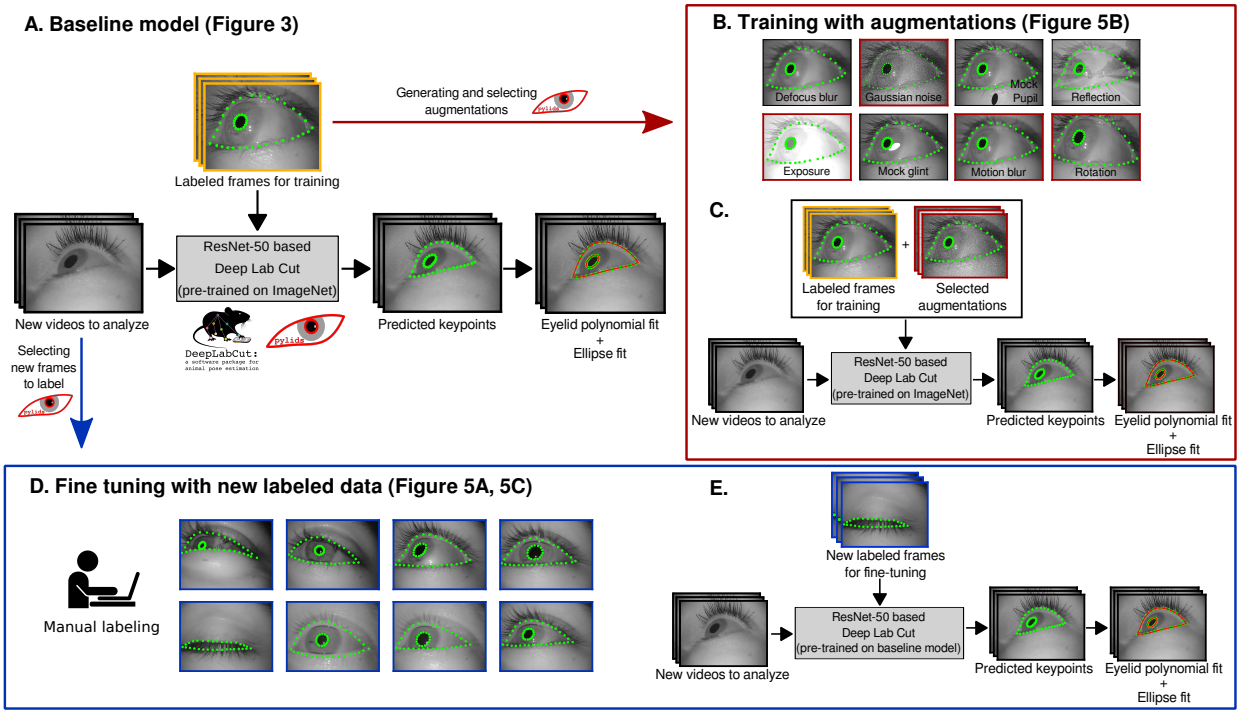

Figure 1

The pipeline for eyelid and pupil estimation in our current study. (A) We use DeepLabCut based on a ResNet50 deep neural network to train our baseline model which can detect the full shape of the eyelid and pupil. We propose methods to improve model generalization by (B) selecting specific data augmentation methods to add data variability or (D) select minimal number of frames to label from the test set; to maximize model performance. (B) We use the pylids package developed and introduced in this paper to generate new domain-specific augmentations and efficiently select augmentations (red outlined frames) for training our model. (C) We use these augmentations in combination with the initial labeled frames (yellow outlined frames) to train the new models. (D) Pylids can also be used to select a given number of frames to label and fine tune a model. (E) These new labeled frames are used to fine tune our baseline model to improve model performance while minimizing the number of frames to label.

augmentation is generally directed by subjectively estimating the additional variability of the test data with respect to the current training data. A standard process to augment training data is to randomly select a fraction of the training frames and labels, then adding a given perturbation of random intensity within a given range. Though this method is effective in improving performance of the model, because of the randomness inherent in the process, it does not ensure that the data set is augmented with frames that would lead to the most improvement in model performance. In addition to this, training time for DNN models scales with training set size. Thus, avoiding redundant frames and augmentations will speed up training with little cost —if such frames can be identified. This leads us to the second question we try to answer in this study: is it possible to select augmentations in a principled way to maximize the performance of a DNN model?

In this study, we develop a method to efficiently select eye images for training a DNN to detect the shape of the eyelids and estimate the pupil position. Our aim was to select a small number of frames that result in a relatively larger increment in model performance on disparate data sets. We show that our approach works well for both selecting data augmentations and also selecting new frames which are manually labeled.

We use DeepLabCut (DLC) (Mathis et al., 2018b), a markerless pose estimation library based on the ResNet50 DNN architecture (He et al., 2015), to train a model that estimates keypoints localizing eyelids and pupils. The DLC output is used to determine the eyelid and pupil shapes using polynomial and ellipse fits respectively (see Figure 1). Next, we devise an algorithm to guide the selection of additional training data to efficiently sample a new dataset, distinct from the exsisting training dataset. Our algorithm aims to reduce the cost of labeling new frames while improving model performance. We compare our model trained using this data selection method with additional models trained using a random subset of the datasets. Our selection method consistently outperforms models of random selection, ensuring generalization to new eye image datasets with relatively few additional labels. We use a similar approach to show that our algorithm also works well for

efficiently selecting augmentations which leads to better model performance when compared to models trained on randomly selected augmentations. Both our eyelid and pupil estimation models approach state-of-the-art performance with a relatively small number of training frames.

To disseminate this method, we introduce an open source python package called pylids which serves as a wrapper to DLC. Pylids can be used to estimate the pupil position and the shape of the eyelid from eye videos. Pylids also allows users to select a small number of frames to label and fine tune their dataset to improve generalization on new datasets. In addition to this, pylids can be used to generate domain-specific augmentations from existing training samples and efficiently select augmentations given a new test dataset. The pylids python package can be accessed at www.github.com/piecesofmindlab/pylids. All datasets used in this paper are in the public domain.

Methods

The basis for our method to detect eyelid shape and estimate pupil position is the DeepLabCut framework (DLC) (Mathis et al., 2018a), a markerless pose estimation library in python. DLC is widely used in the field of neuroscience for animal pose estimation (Lauer et al., 2022; Mathis et al., 2018a; Nath* et al., 2019). Studies have also shown that this framework is easily adaptable to accurately track pupil position in both mice (Meyer et al., 2020) and humans (Zdarsky et al., 2021). We chose DLC as it uses a dense neural network with many more parameters than other popular DNNs used for eye tracking such as EllSeg (R. S. Kothari et al., 2020), RITnet (Chaudhary et al., 2019) and DeepVOG (Yiu et al., 2019). Large networks learn task relevant representations with fewer training samples and generalize better than their smaller counterparts (Neyshabur et al., 2014; Novak et al., 2018).

In this study, we trained DLC to estimate keypoints on the edge of the pupil and on the upper and lower eyelids of participants. We then used polynomial and ellipse fits to these keypoints to localize the eyelids and pupils respectively (see Figure 1). We started by

training a baseline DNN model to estimate eyelid and pupil locations using data from a single dataset. We evaluated the baseline model's performance on a withheld subset of this dataset and in three additional and highly variable datasets. We devised an algorithm to efficiently sample new frames to label from these additional datasets. We use the new selected labeled frames to fine tune our baseline model and analyze the resulting improvement in model performance contrasted with other frame selection methods. We also use our algorithm to efficiently select augmentations for our baseline model which helps the model to generalize to new datasets. Finally, we compare the results of our best performing models with state-of-the-art pupil and eyelid estimation algorithms for different datasets. See Figure 2 for an overview of the ways we trained and evaluated our models.

Datasets

We used three publicly available labeled eye video datasets: Gaze-in-Wild (R. Kothari et al., 2020), Labeled Pupils in the Wild (Tonsen et al., 2016), and the data from Fuhl, Santini, and Kasneci (2017). Datasets and their purposes are described below.

We used part of Gaze in the Wild (GiW) for training and testing our baseline DNN model. This data consisted of eye videos from 13 (of 23 total) participants. We used data from all 13 participants that performed an indoor navigation task. These eye videos (640x480 at 120Hz) were recorded with a head-mounted Pupil Labs Pupil Core eyetracker from freely moving participants. The average length of the recording sessions from each subject was about six minutes (about 43,500 frames).

To test the generalizability of our baseline model, we tested its performance on Labeled Pupils in the Wild dataset (LPW) (Tonsen et al., 2016), which has ground truth labels for pupil centers, and the dataset introduced by Fuhl, Santini, and Kasneci (2017) (referred to as the Fuhl dataset from here on) which provides labels for eyelid outlines. While the GiW dataset was exclusively collected indoors, the LPW and Fuhl datasets both consist of both indoor and outdoor recordings. Furthermore, eye videos in these datasets commonly have dark shadows, strong corneal reflection, high exposure and low contrast,

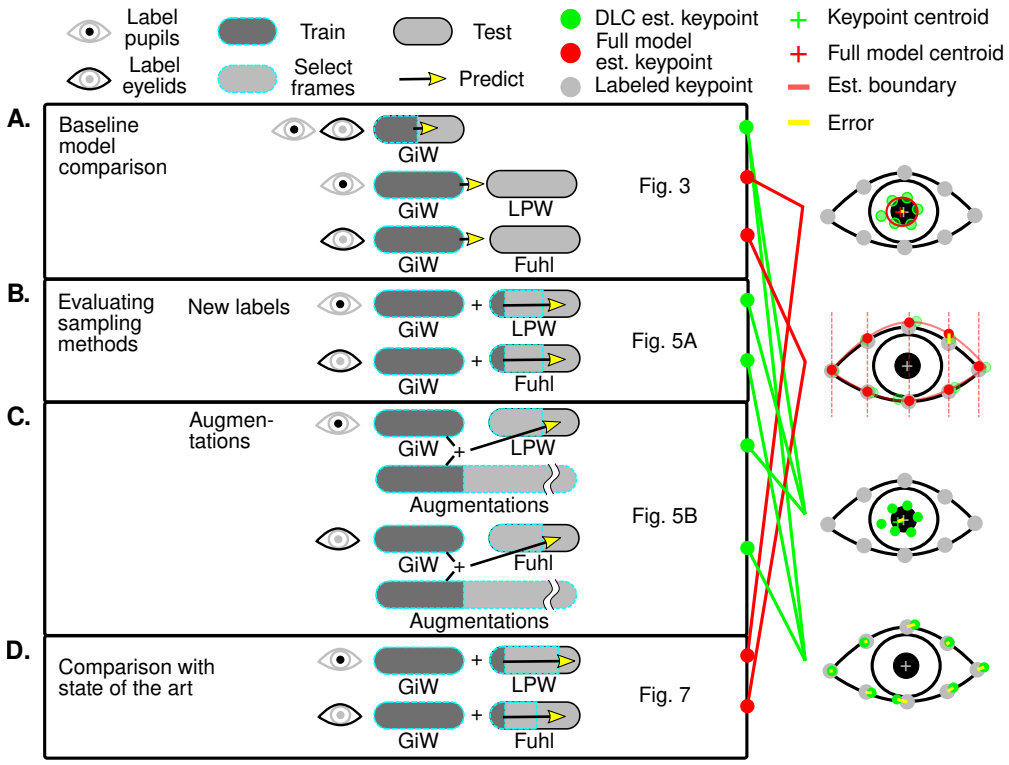

Figure 2

An overview of all analysis in our paper. Ovals indicate datasets used for each analysis; LPW is Labeled Pupils in the Wild, GiW is Gaze in the Wild, Fuhl is the eyelid label dataset from (Fuhl, Santini, & Kasneci, 2017) (A) Baseline model comparison: We train a baseline model and establish its generalizability for pupil and eyelid estimation by comparing model performance within and across datasets. (B,C) Evaluating sampling methods: Next we use guided sampling to select (B) new frames to label and (C) augmentations. We train separate new models using these new labeled frames and augmentations and compare with other sampling methods for pupil and eyelid estimation performance. (D) Comparison with state of the art: Finally, we compared our best performing pylids models with state of the art models for both pupil and eyelid estimation. Eye diagrams at the right illustrate how error was computed for each analysis. To evaluate sampling methods, we computed error for each keypoint (green dots). Keypoint error directly evaluates DNN outputs given the training data and is sensitive to differences in training. To evaluate model performance within and across datasets and to compare our model to other models, we computed error for full model estimates (red values: pupil ellipse centroids and eyelid polynomials).

heavy makeup, participants wearing eye glasses, and eyelid occlusions of pupils due to squinting. The LPW and Fuhl datasets also used different eye tracking instruments to collect the data. Hence these datasets with high eye image variability provide a strong test for generalizability of our models.

The LPW dataset consists of data collected from 22 participants across 66 recording sessions using a Pupil Labs Pupil Pro head mounted eye tracker (640 x 480 at 120Hz) (Kassner et al., 2014). The dataset also includes ground truth pupil centers for a total of 130,856 eye images. The Fuhl dataset consists of data from 11 participants with ground truth labels for the upper and lower eyelids for 5100 eye images collected using a Dikablis Essential head-mounted eye tracker (384 x 288 at 30Hz).

Subsequent to evaluating the performance of our baseline model we trained additional models using either efficiently sampled new training data or data augmentations. To avoid any test set leakage during the training process we divided the LPW and Fuhl dataset into train and test splits. For the LPW dataset each of the 22 participants have three eye videos of equal duration. For the train split we randomly selected 2 of the 3 videos and we used the remaining video as the test split. For the Fuhl dataset we randomly selected 2/3 of the data from each of the 11 participants as training data and the remaining 1/3 was denoted the test data. This resulted in a total of 3,400 train samples and 1,700 test samples for the Fuhl dataset.

Baseline model training and validation

Keypoint labeling

To select an initial set of frames for labeling and training our baseline DNN model, we used the default procedure implemented in DLC to select frames. DLC downsamples each eye video to a resolution of 30 x 30 pixels and then uses k-means clustering applied to the downsampled frames to find a pre-specified number of frames from distinct clusters. We used this procedure to extract 40 frames from each of the 13 participants of the GiW dataset for a total of 520 different frames, which we then labeled. For every eye frame, a

total of 48 keypoints were labeled —15 keypoints each for the upper and lower eyelids, two for the left and right corners of the eye, and 16 keypoints for the pupil (see Figure 1). Eyelids were labeled at the skin/sclera boundary. Each frame took about 3 to 5 minutes to label. To ensure consistent labeling of keypoints across frames we followed the protocol described in (Garbin et al., 2019). Labeling starts with annotation of the two corners of the eyelid, then for the upper eyelid each additional annotated keypoint iteratively bisects two adjacent keypoints moving from left to right until there are 15 keypoints. This is repeated for the lower eyelid to annotate another 15 keypoints on the lower eyelid and the 16 keypoints encircling the pupil. Keypoints that are not visible in a frame —for example, a partially occluded pupil or an out of frame eyelid —are skipped by the labeler, as per the DLC labeling protocol.

Model training

Our DLC-based DNN model uses a ResNet-50 deep convolutional neural network initialized with model weights derived from training on the ImageNet dataset (Deng et al., 2009). This ensures faster convergence and boosts out-of-domain performance for DLC networks (Mathis et al., 2021). For training of DLC model below, we followed the guidance described in Mathis et al. (2021). To train and evaluate our baseline DNN model we used 520 labeled frames from the GiW dataset, 40 from each of the 13 participants. 240 labeled frames from the first six participants in the GiW dataset were used as the training set and 270 labeled frames from the remaining seven participants as a test set. The model was trained with a batch size of 8 and trained for 120,000 iterations using the ADAM optimizer (Kingma & Ba, 2017) and standard cross entropy loss. Thus, we used 4,000 epochs (or cycles through the 240 labels) to train the model to detect keypoints. This is consistent with the guidance from DeepLabCut for the number of training labels that we had. We used a three stage multi-step learning rate. The learning rate was set at 10^{-4} for the first 40,000 iterations, subsequently for iterations 40,000 to 60,000 it was reduced to 5×10^{-5} and for iterations 60,000 through 120,000 the learning rate was further reduced to 10^{-5}.

All model weights were allowed to be updated during this training phase. No data augmentation techniques were used for training our baseline model. Using an NVIDIA RTX 2080 it took about 10 hours to train our network. After training, model inference ran at 82 frames per second.

The final DNN model uses the frames from the eye video as its inputs and outputs all visible keypoint locations for the eyelids and pupil. The model also assigns a likelihood metric to each keypoint, this denotes the uncertainty of the model in localizing keypoints and is correlated with prediction error. After training, we used our model to predict the eyelid and pupil keypoints for 6 participants in the held out test portion of the GiW dataset. This served as an independent test set, albeit one drawn from a similar distribution of eye videos as the training set. To evaluate the generalizability of models trained on GiW data to other datasets, we trained a DNN model using all 520 labels from the GiW dataset and tested its performance on the entirety of the LPW and Fuhl datasets.

Eyelid polynomial fitting and pupil ellipse fitting

Using the pylids package introduced in this paper we fit polynomials to the eyelid keypoints and ellipses to the pupil keypoints. To determine the full shape of the eyelids, we fit a fourth-degree polynomial separately to the upper and lower eyelid keypoints using weighted ordinary least squares. We used the likelihood value associated with the keypoints output by DLC as weights for the polynomial fits. This acts as a regularizer, ensuring a more stringent fit for points with higher likelihood as compared to those with lower likelihood. This process allows robust estimation of eyelids even when the keypoints are noisy. For the frames where the fourth degree fit to the upper and lower eyelid did not intersect at the eye corner, a third-degree polynomial was fit.

To estimate the pupil's location, we fit an ellipse to the estimated pupil keypoints for each frame. We used an ordinary least squares ellipse fit (Gander et al., 1994) as implemented in the scikit-image python package (van der Walt et al., 2014). As with eyelids, we did not estimate pupil positions for frames with average pupil key point

likelihood less than 0.6. We used a higher threshold for pupils as small errors in pupil estimates can translate to much larger gaze errors.

Comparison with alternate model

We highlight the benefit of our model over a standard non-DNN method by comparing our model's performance to pupil detection with the Pupil Labs python library (Kassner et al., 2014; Labs, 2013). The Pupil Labs library implements a standard Hough transform based on edge detectors to find the pupil, and is representative of classical approaches to pupil detection. It is open source and still widely used for pupil detection and subsequent gaze estimation (Fischer et al., 2018; Katsini et al., 2020). We evaluated the Pupil Labs model on both the held out GiW test set and the LPW dataset.

Selection of new training frames for generalization to new datasets

Recent studies suggest that DNNs are good at interpolating between training data points as opposed to extrapolating to out-of-distribution samples. This suggests that these deep networks don't learn generalizable task relevant features but rather memorize the features of the training set (Arpit et al., 2017; Zhang et al., 2021). Also, Novak et al. (2018), found that neural networks are more robust, stable and generalize well to inputs that are sampled from the neighborhood of the training data manifold. Based on these studies we hypothesized that the distance of test frames from the training distribution would determine model performance. We verified this in an earlier study (Biswas et al., 2021) and found that model performance for eye frames in the test data dropped with an increase in cosine distance from the training data in the ResNet50 feature space. Building on this, we further postulate that new training frames can be efficiently selected by sampling uniformly distributed frames closer to the test distribution. We apply this principle to design an algorithm to select both new frames to label and also to efficiently select data augmentations that help our model generalize to new datasets.

In the field of machine learning, the problem of selecting an efficient training set for models is well defined and many algorithms have been developed trying to solve this

problem (Cohn et al., 1996; Gal et al., 2017; Settles, 2009; Wang et al., 2018). One such approach is called core-set selection. Core-set selection is an instance of active learning which deals with the question of maximizing model performance from a set of samples given a fixed labeling budget. Recent work has shown that DNN activations from a network trained on a labeled subset of data can be used to determine additional training frames that would provide the most performance benefit (Coleman et al., 2019; Sener & Savarese, 2017). These studies are based on image classification rather than keypoint labeling. Additionally, these studies train DNN models from scratch from random initialization. Thus requiring multiple iterations of labeling, model training and additional labeling and training. In our study, we use an ImageNet-trained ResNet50 model to extract feature representations to use for core-set selection. By starting with a pre-trained network and then applying core-set selection procedure we hoped to reduce the number of times we label and train our models. We test whether the core-set selection principles that work for selecting training data for image classification also work for selecting training data for keypoint labeling. We note that image classification may rely on broadly distributed features in the image, whereas keypoint labeling is likely to rely more on spatially focal image features near each keypoint. Thus, the same training set selection principles are not a priori guaranteed to work for this case.

Selection of frames to label

Our aim was to efficiently select a small set of frames that reduces the cost benefit ratio of labeling additional frames and the corresponding increase in model performance on a new dataset. To this end, we attempted to sample 10 new frames to label from the train splits in the LPW and Fuhl datasets. We first mapped the frames into the ResNet50 feature space. For the Fuhl dataset train split, which has 3400 frames, we pass both the 520 labeled images from GiW dataset (the training set for our baseline model) and the 3,400 labeled eye images from the train split of the Fuhl dataset through a ResNet50 neural network pre-trained on ImageNet. The output of the final convolutional layer before the

fully connected layers gives us a 100,352-dimensional feature vector for each frame. This is considered to be the representation of each frame in ResNet50 space. Thus, at the end of this process we are left with 3,920 vectors in the ResNet50 space, one for each eye frame.

As we established earlier, an efficient sampling strategy would be to select uniformly distributed frames in this ResNet50 space. The greedy k-centers algorithm has been previously used to select a uniformly distributed core-set of training data for DNNs (Coleman et al., 2019; Sener & Savarese, 2017). We instead used a similar algorithm, k-means clustering, to select 10 new training frames from the 5100 frames in the Fuhl dataset (see discussion for motivation for using k-means rather than greedy k-centers). Running k-means on a set of data generates cluster centers which minimize the sum of the squared distances to the data, allowing for uniform sampling.

Since the new to be labeled frames are used to fine tune a model previously trained on samples from the GiW dataset, we need to ensure that new training frames selected by our algorithm provide the model with additional non-redundant information with respect to the existing training distribution. To this end, we devised an algorithm which we call guided selection (see Figure 4B for illustration). The goal of guided selection is to find a given number n of cluster centers in ResNet50 space closer to the new dataset than to to the existing training frames for our baseline model. Our algorithm uses iterative k-means to cluster the concatenated existing training set and the Fuhl dataset from which we will be selecting new frames to label. We used mini batch k-means with a batch size of 300 for clustering. This allows us to cluster larger datasets without loading the full dataset into memory. For the first clustering iteration, we use an initial guess for k based on the ratio of samples in the training and test set to help with faster convergence. For the first iteration we set k as:

$$k_1 = n + \alpha$$

here n is the number of new frames we want to label, α is used to pad n based on the ratio T. Further, $\alpha = T \times n/(1 - T)$ and T is ratio between the number of frames in the existing

training set and the new dataset defined by $T = n_{train}/(n_{train} + n_{new\,data})$. After every clustering iteration, we calculated the cosine distance between the estimated cluster centers and the mean of the 520 vectors representing the existing training data. We also calculated the cosine distance between the cluster centers and 3,400 vectors making up the train split in the Fuhl dataset. Using these two distance measurements, we determined the number of cluster centers closer to the training data as compared to samples in the Fuhl dataset. In case the number of cluster centers closer to the Fuhl dataset were less than n (the number of new frames we want to select), we updated $k_i = k_{(i-1)} - m$ and reran k-means. Here $k_{(i-1)}$ is the guess for cluster centers for the $i-1$ iteration and m is the number of cluster centers closer to the Fuhl dataset compared to the existing training set during the $i-1$ run. We repeat this process till we end up with n cluster centers closer to the Fuhl dataset than to the original training data. Next, we find n frames from the Fuhl dataset with the least cosine distance from each of these cluster centers. These n frames are spread across the Fuhl dataset in ResNet50 space and combined with the existing training dataset have the least sum of the squared distance from all other frames in the Fuhl dataset. In doing this, our algorithm minimizes the distance between the new to-be-labeled sampled frames and the new dataset of interest (in this example the Fuhl dataset).

Using the guided selection algorithm we selected n=10 frames to label from the train splits of the LPW and Fuhl datasets. Using these frames we fine tuned our baseline DNN model to derive two separate DNN models, one each for the the LPW and Fuhl datasets. As a control for our guided selection method we also trained models trained using n=10 randomly selected frames from the LPW and Fuhl datasets. For the Fuhl dataset for which complete eyelid labels were available for all eye images, we also trained 30 additional models using randomly selected non-overlapping sets of 10 training frames. We did not perform this analysis on the LPW dataset due to the absence of keypoint labels for the pupils and eyelids. This process resulted in 30 models. We compared the model trained with guided selection to these models trained with random selection to estimate the

relative accuracy of our guided selection algorithm. In addition to putting the performance benefit due to guided selection in context, this gives us an estimate of the variance in model performance based on different sets of 10 additional training frames.

Additionally, we hypothesized that since distance of the test set from the training distribution governs model performance, the worst performing model should be one fine tuned using 10 new frames *closest* to the training distribution in terms of cosine distance. We deliberately created such nearest selection models for both LPW and Fuhl datasets. We compared model performance across guided selection, random selection and nearest selection using t tests. We corrected for multiple comparisons using false discovery rate Benjamini-Hochberg procedure, with the threshold value set at 0.05.

All models were fine tuned after initializing with our baseline model weights derived from training only on the GiW dataset. During the fine tuning process we permitted the full model to be trainable, albeit with a reduced learning rate and for fewer iterations. All models were trained for 3000 iterations with a learning rate of 3×10^{-4} using the ADAM optimizer. We used cross validation to determine the appropriate learning rate and number of training iterations for fine tuning.

To visualize the effect of distance from the training set on model performance we used multi-dimensional scaling (Kruskal & Wish, 1978) which creates a low-dimensional representation of the data while preserving the relative distances between sample in the original high dimensional space (see Figure B1 in Appendix B for details). Using this we can visualize both our training and test data to confirm our hypothesis that test samples closer to the training set would have a lower average keypoint estimation error as compared to test samples farther away from the training set.

Selection of augmentations

Data augmentation is commonly used to generate additional labeled training samples for DNNs from extant labeled samples. In general, the process involves modification of an extant labeled frame —for example, a shift in the position of the image

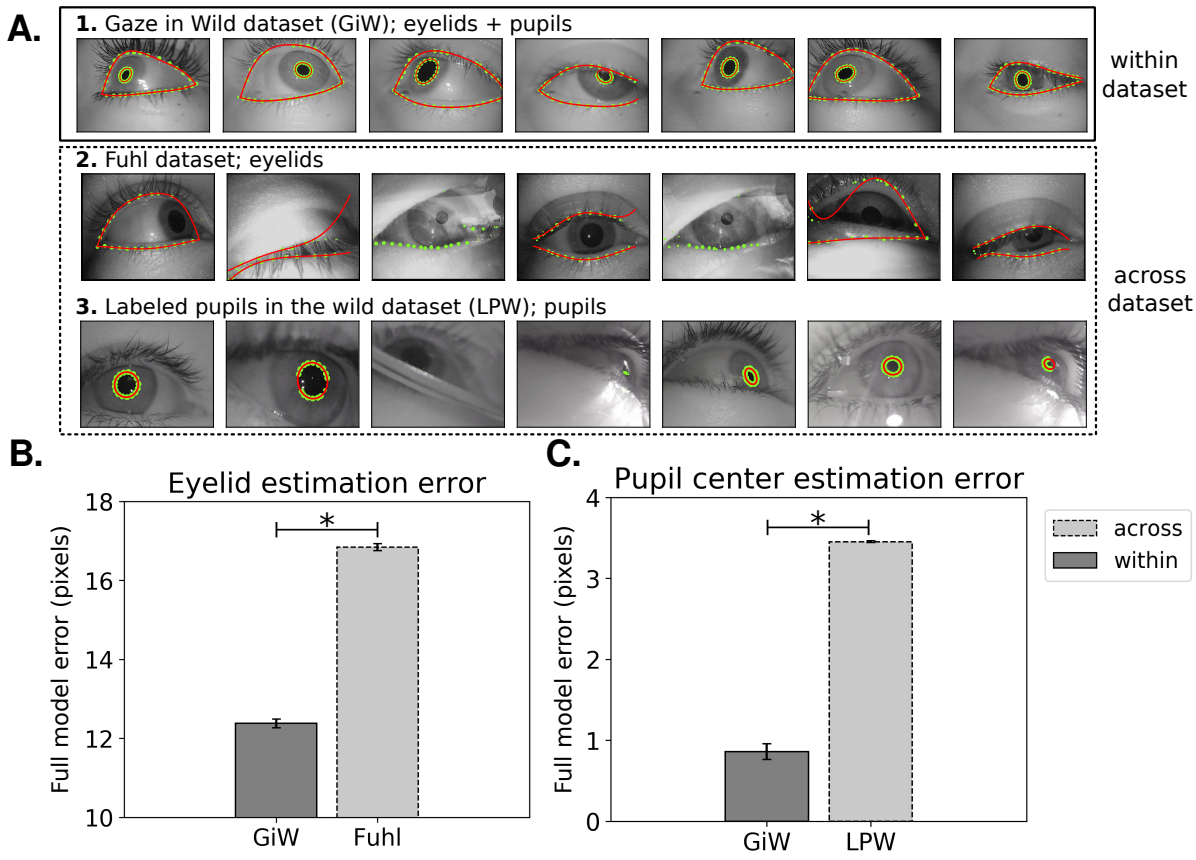

Figure 3

Assessing the generalizability of our baseline model for test samples drawn from the same dataset as the training samples (A1), or test samples draw from other datasets (A2,3) likely representing a different distribution to the training set. (A) Each row depicts outputs of our baseline model for different datasets. DLC-estimated keypoints are shown in green and pylids based pupil and eyelid estimates are shown in red. Keypoint size scales with likelihood of each estimated keypoint with uncertain keypoints visualized using smaller keypoints. Keypoint estimates are not displayed if the likelihood is below 0.4 for eyelids and 0.6 for pupils. For both eyelid (B) and pupil estimation (C) our baseline model performs significantly better when the training and test data are from the same dataset (GiW) pointing to a drop in performance for out of distribution samples.

or blurring of the image —while keeping the original label (or an appropriate transform thereof). This process increases the image variability of the set of training samples without the need for additional manual labeling and has been shown to improve model generalization performance. For this project, we augment our training data with several image perturbations that simulate ways that eye images can vary in different recording conditions. However, this yields a very large space of augmentations which may be redundant with extant training samples or otherwise unimportant for model training. In general, it is challenging to balance the number of manually labeled training samples with the number of image augmentations, the latter of which can dwarf the size of the original dataset due to combinatorial explosion. To address this issue, we apply the same data selection method to augmentations that we used to select new data frames for labeling. In this way, we aim to choose the most beneficial augmentations from a wide range of possible augmentations.

All of our baseline model training data was collected indoors on only a few people (eye images from GiW dataset). However, we aim to test our model on samples from different people, collected outdoors while performing dynamic tasks. To simulate these conditions with augmentations, we use image processing techniques to add perturbations the original frames by adding exposure, reflection, defocus blur, eye rotation, JPEG compression, motion blur, Gaussian Noise, mock pupils and mock glints (see Appendix A for details). For each perturbation we generated four additional images with increasing perturbation intensities. We generated a total of 520 (training samples) x 9 (augmentation techniques) x 4 (augmentation intensities) = 18,720 possible set of perturbed images. From these images, our goal was to augment our current training dataset by selecting uniformly distributed samples closer to the test dataset.

Based on the same principle we used for selecting new frames to label, we posited that augmentations spread across the new dataset of interest would best improve our model performance. Hence we first selected uniformly distributed frames from the new

dataset of interest and then we selected the augmentations which were closest to these uniformly distributed frames. As before, we first mapped all the data into the ResNet50 space. This included all 18,720 augmentations, the training set from our baseline model, and the training images from the new dataset to which we want our model to generalize. For this we used a ResNet50 pre-trained on the ImageNet dataset. We aimed to select n =520 augmented images (a number equal to the number of original training frames) for both the LPW and Fuhl datasets. Towards this, we used the same iterative procedure described earlier to select n cluster centers that span the combined space of existing training data and new dataset. Once we found n clusters, we selected augmentation images (from the set of 18,720 described earlier) which were nearest to the n cluster centers in terms of cosine distance. These n frames represent the selected augmentations (see Figure 4C for illustration). This algorithm ensures that we select uniformly distributed samples from the space of augmented eye images. This process selects those frames which add image variability similar to that encountered in the test set, hence aiding our DNN model to generalize. As a control for our guided selection algorithm we use the conventional method to select augmentations. We selected n =520 random augmentations equally distributed across all the types of images perturbations utilized by us. We also select n =520 augmentations which are nearest to the mean of the training data. As justified earlier, we hypothesize this would be the worst performing model as there would not be much variability in these samples with respect to the existing training dataset.

After selecting 520 different set of eye images using guided, random and nearest augmentation from both the Fuhl and LPW datasets we retrained our baseline DNN model using these three different sets of augmented images. The training set for this process included the 520 training frames used to train the baseline model and the additional 520 frames of selected augmentations. The models were trained using the same process as the baseline model starting with initialized model weights derived from an ImageNet trained ResNet50. The model was trained for 120,000 iterations using the ADAM optimizer. The

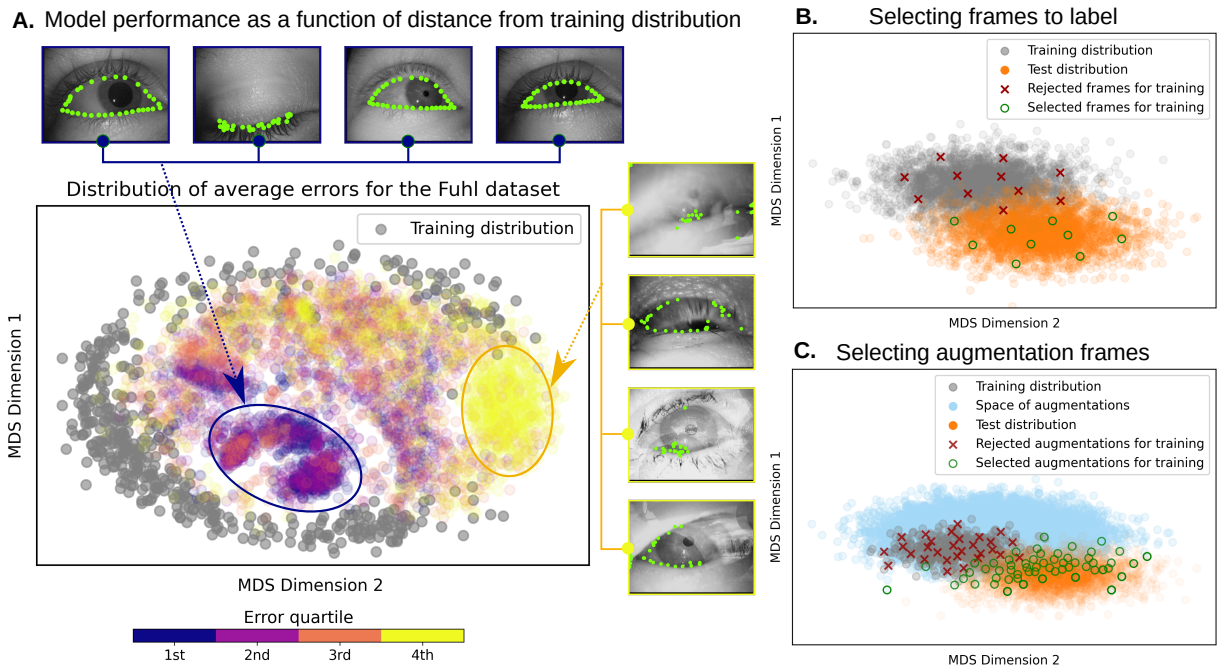

Figure 4

(A) Baseline model performance for frames in the Fuhl dataset visualized using multi-dimensional scaling (Kruskal & Wish, 1978). Baseline model training frames from the GiW dataset are visualized in grey. Cosine distance of test frames from the training distribution is correlated ($\rho = 0.62$) with average model error for each frame. This demonstrates that model performance drops with an increase in cosine distance from the training distribution. We use this principle to design our guided sampling algorithm to select new frames to label (B) or select new augmentations (C) to train our models. (B) To select additional data which minimizes the distance from the training distribution we use iterative k-means clustering to sample uniformly across the test distribution (green circles). Frames closer to the extant training distribution are rejected (red crosses). (C) Similarly, to efficiently select augmentations we use iterative k-means to sample uniformly from the test set while rejecting frames close to the existing training set. Finally from the space of augmentations (cyan) we select those which have the least cosine distance (green circles) from the samples selected from the test set.

three different augmentation selection methods resulted in separate DNN models for LPW and Fuhl datasets which were then compared to estimate change in model performance using t tests. We corrected for multiple comparisons using false discovery rate Benjamini-Hochberg procedure, with the threshold value set at 0.05.

Model evaluation

We evaluate our models on two different criteria, each useful for a slightly different purpose. To directly assess the quality of keypoint estimation by the trained deep neural network, we compute the average Euclidean distance in pixels between predicted and human labeled keypoints for all keypoints around the eyelid or pupil. We henceforth refer to this as *keypoint estimation error*. Keypoint estimation error is suitable for comparing the accuracy of the model to the inter-rater reliability of human raters, and provides a direct evaluation of the DNN output without the regularization provided by the ellipse fits to the pupil or the polynomial fits to the eyelids. However, keypoint estimation error cannot be used for datasets in which the labeled keypoints are not identical to the trained keypoints. For example, the Fuhl dataset has only 5 keypoints labeled on the eyelid. The keypoints are also not the final output of the pylids model; the ellipse and polynomial fits are, which provide arbitrarily many points around the pupil or eyelids. To evaluate the full model for eyelids, we compute the Hausdorff distance between the polynomial and the available labels. Hausdorff distance is the maximum of the minimum error between predictions (polynomials) and ground truth (labeled points on eyelids). Thus, the value for Hausdorff distance can be thought of as the worst mistake a model makes. This value may be higher than keypoint error if the model is accurate for most labeled keypoints but inaccurate for one or two. To evaluate the full model for pupils, we compute the mean squared error between the fit ellipse center and the labeled center. We refer to both the Hausdorff distance metric and the ellipse centroid mean squared error as *full model estimation error*.

Results

An overview of all analysis can be found in Figure 2. We first evaluate the generalizability of our baseline eyelid and pupil model trained on the GiW dataset. We do this by comparing baseline model performance on held out test sets from the GiW dataset (within dataset) and the Fuhl and LPW datasets (across dataset). Next, we compare guided selection, our method for selecting new frames and augmentations, to other sampling methods. We do this by training separate new models for eyelid and pupil estimation using the new labeled frames or augmentations. To evaluate these models we compare models trained using different sampling methods on held out test sets from the LPW dataset (for pupil estimation) and Fuhl dataset (for eyelid estimation). Finally, we also train and evaluate our best pylids models on LPW and Fuhl datasets to enable comparison with state of the art models for pupil and eyelid estimation.

Baseline model training and validation

Keypoint estimation error

To evaluate our baseline model trained on 240 labeled eye frames from the GiW dataset, we determined the keypoint estimation accuracy of the model on 270 held out test frames also from the GiW dataset. We first calculated the Euclidean distance between hand-labeled keypoints and the DLC model's keypoint estimates for the eyelids and pupils. The calculated average eyelid keypoint estimation error was 10.78 pixels, and for the pupil the error was 8.24 pixels. These errors value represent the keypoint to keypoint error at the edge of the eyelids and pupils. To put these values in context, an additional human labeler annotated four videos in the test set. The inter-labeler Euclidean distance for the eyelids was 12.91 pixels, and for the pupil it was 8.67 pixels. These error values may be biased as there is no unique visual cue for manually labeling each individual keypoint along the pupil and eyelid edge across frames. This results in jitter in predicted keypoints along the eyelid and pupil. For practical purposes, a precise horizontal match of keypoints is not necessary, since we ultimately fit a polynomial function to the keypoints to get a continuous estimate

of the eyelid. Still, our results suggest that the DLC model accurately determines the position of the eyelid with its keypoint estimation accuracy lying within the variance of human labelers.

Full model error estimation

To assess the generalizabilty of our baseline model, we also estimated the full model error (see Methods) for a new datasets on which our model had not been trained, the Fuhl dataset (Figure 3B). Briefly, we computed full model error by fitting a polynomial to eyelid keypoints and calculating the Hausdorff distance between the set of points on the polynomial denoting the eyelid and the set of points representing the ground truth outline of the eyelid. Mean full model eyelid estimation error for the Fuhl dataset was significantly higher as compared to error on the GiW dataset (GiW: mean = 12.38 pixels, SEM = 0.11 pixels; Fuhl: mean = 16.84 pixels, SEM = 0.08 pixels, $t = 9.84 p < 0.05$). This was in line with our expectations that model performance would be worse for frames sampled from in-distribution (GiW) versus out-of-distribution (Fuhl dataset) frames.

Pupil center error estimation

We estimated the generalizability of our baseline model by comparing the full model error for GiW with the full model error for the LPW dataset (see Methods). Briefly, for this comparison (see Figure 3C) we looked at the average Euclidean distance between pupil centers estimate using ellipse fits from our pylids model and the ground truth pupil center. Pupil center error was significantly higher for the LPW dataset as compared to the GiW dataset (GiW: mean = 0.86 pixels, SEM = 0.09 pixels, LPW: mean = 3.45 pixels , SEM = 0.01 pixels, $t = 8.67, p < 0.05$). Thus, as expected and in keeping with other recent results (R. S. Kothari et al., 2022), our model did not perform as well on a dataset on which it had not been trained. We next looked more closely at the data points on which the baseline model made errors.

In a previous study (Biswas et al., 2021) we demonstrated that cosine distance of samples from the training distribution in the ResNet50 space had an inverse correlation

with keypoint likelihood. That is, uncertainty in keypoint estimation increases with an increase in cosine distance from the training distribution. We replicated this finding on new datasets used in this study and extend it by showing that this correlation holds for keypoint error estimates along with likelihood. For the Fuhl dataset, the cosine distance from the training distribution (training samples for our baseline model) in ResNet50 space correlated with average keypoint estimation error ($\rho = 0.62, p < 0.05$) and also with drop in likelihood ($\rho = -0.81, p < 0.05$), see appendix D. We also visualized this relationship between the distance from training frames to model performance by using multi-dimensional scaling (Kruskal & Wish, 1978). There was clear evidence of frames with low error clustering closer to the training samples while frames with high error were clustered far away from the training samples (see Figure 4A). We also found anecdotal evidence that frames which clustered closer to the existing dataset were visually similar to the existing training dataset—these frames seemed to be from indoor sessions. In contrast, frames which clustered far from the existing training dataset were mostly from outdoor sessions.

Comparison with alternate model

To compare our model with a standard, non-DNN alternative, we computed pupil centers with the Pupil Labs library (PL) (see Appendix C1). For both the GiW and Fuhl datasets the PL model had a larger error than the pylids model (for PL, GiW: mean = 5.07 SEM = 0.14, LPW: mean = 42.18 SEM = 2.8). This is not surprising given that the PL model uses heuristics like edge detection and contrast between pupil and iris to detect the pupil. Such heuristics can fail in challenging conditions such as data collected outdoors.

Selection of new training frames for generalization to new datasets

In this section we discuss our results comparing different sampling methods for selecting new training frames and augmentations which we use to train our DNN models. For error estimates in this section we use keypoint estimation error (the raw keypoint estimates from the DLC model without using either the polynomial fits for eyelids or ellipse fits for pupils). This was done because the polynomial or ellipse fitting process acts

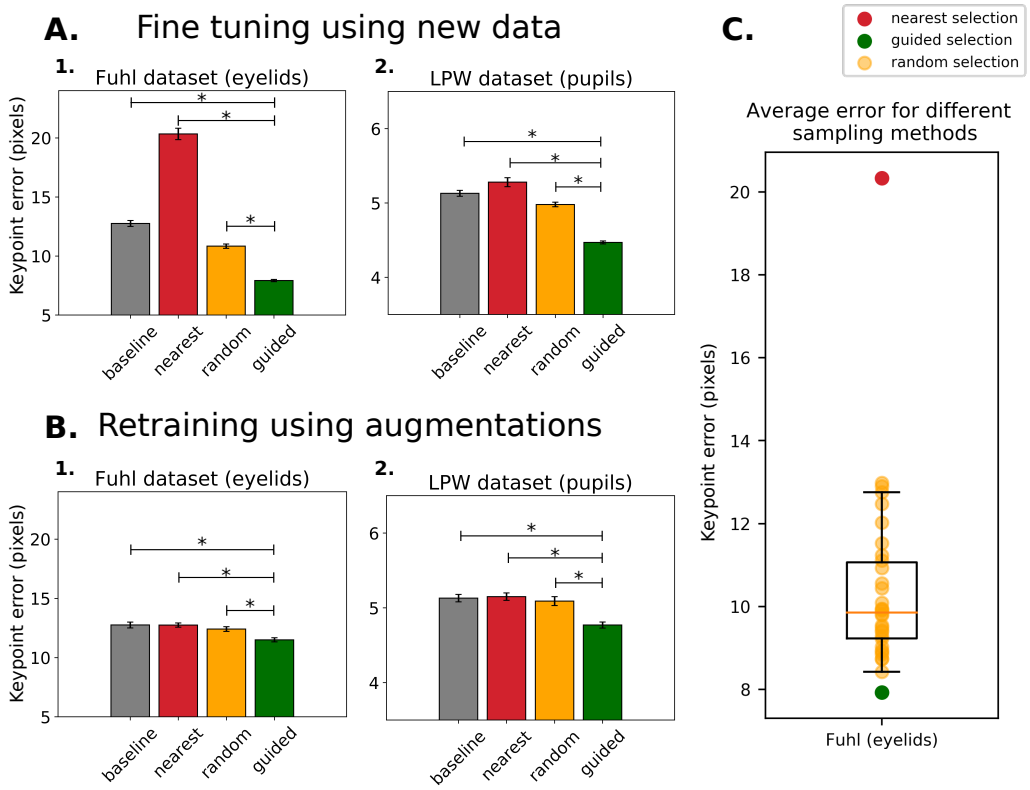

Figure 5

Comparison of average keypoint estimation error using different sampling methods for training our DNNs. (A) Guided sampling (green) outperforms random sampling (orange), nearest sampling (red) and baseline models (gray) for both eyelid estimation error (A1) and pupil estimation error (A2). (B1,2) Guided augmentation (green) also results in lower error for eyelid and pupil estimation compared to random augmentation (orange), nearest augmentation (red) and baseline model (gray). (C) Comparison of increase in model performance due to guided sampling of new training frames (green) with 30 different models trained using random sampling (orange) and nearest sampling (red). Error bars represent 95% confidence intervals.

as a regularizer operating on the output of the DNN and is not part of the DNN. For eyelid estimation error, we calculated the average symmetric surface distance based on the DLC keypoint estimates for the eyelids and the ground truth. This involves taking the mean of the minimum distance between the set of predictions (DLC keypoint estimates) and the ground truth (labeled keypoints). For comparing the pupil estimation error we compared the Euclidean distance between the centroid of all pupil keypoints and the ground truth pupil center.

Selection of frames to label

To compare the different sampling methods we use to select and label frames, we first fine tuned our baseline model based on frames sampled with each method (guided, nearest and random sampling). We then compared the model performance for these sampling methods when evaluated on the test split of the Fuhl and LPW datasets (Figure5A, C). Guided selection of new training data resulted in the maximum gain in model performance for both eyelid and pupil estimation across all different datasets. Comparing eyelid estimation error for the Fuhl dataset, guided selection (mean=7.92 pixels, SEM = 0.09 pixels) resulted in a significantly lower average keypoint error in comparison to nearest selection (mean = 20.33 pixels, SEM = 0.48 pixels, $t = 28.11, p < 0.05$), random selection (mean = 10.83 pixels, SEM = 0.18 pixels, $t = 16.11, p < 0.05$) and the baseline model error (mean = 12.75 pixels, SEM = 0.25 pixels, $t = 20.78, p < 0.05$).

The same trend was observed when we calculated the pupil center estimation errors. For the LPW dataset, the model trained using guided selection (mean = 4.47 pixels, SEM = 0.02 pixels) resulted in a significantly lower pupil estimation error in comparison to nearest selection error (mean = 5.28 pixels, SEM = 0.03 pixels, $t = 5.25, p < 0.05$), random selection error (mean = 4.98 pixels, SEM = 0.02 pixels, $t = 4.33, p < 0.05$) and baseline model error (mean = 5.13 pixels, SEM = 0.03 pixels, $t = 4.15, p < 0.05$).

To account for the variance in model performance due to random sampling we compared our guided selection model and nearest selection model with 30 different models

trained using non-overlapping randomly selected sets of 10 training frames for the Fuhl dataset (see Figure 5C). For the Fuhl dataset, guided selection was the best performing model (average error = 7.90 pixels) when compared to the 30 randomly selected ones (average error for 30 models = 10.21 pixels, minimum error = 8.42 pixels and max error = 12.98 pixels), the nearest selection was the worst performing model (average error = 20.33 pixels).

To further investigate our hypothesis that sampling uniformly across potential training frames is an efficient strategy for selecting additional training data, we used multi-dimensional scaling (MDS) to visualize model performance in the vicinity of additional training frames (see Figure 6). We transformed the ResNet50 representations for the baseline training set, the newly selected training frames, and the test set into a two dimensional MDS space. We noticed that our models perform well on frames in the neighborhood of selected training frames. We also show that for our new models the model error is lower on samples closer to the training set of the models (see Appendix D3). This suggests that the DNN models interpolate in ResNet50 space and fail to extrapolate to out-of-distribution samples farther away from extant training samples. This correspondingly also results in a drop in model performance with an increase in distance from training samples.

We also compared our best pylids models to state-of-the-art models (see Figure 7). For eyelid estimation we used a pylids model trained 50 additional frames selected using guided selection from the Fuhl dataset and evaluated our model performance on all of the Fuhl dataset. For comparing eyelid estimation performance we used the Hausdorff distance between the set of eyelid keypoints and the ground truth as suggested in Fuhl, Santini, and Kasneci, 2017. Our pylids model outperformed the original algorithm presented in Fuhl, Santini, and Kasneci (2017). Our model resulted in a cumulative detection rate (mean absolute error less than 10 pixels) for 76.92% of the Fuhl dataset as compared to the original algorithm presented in the paper which had 61.94%. For comparing pupil center

estimation performance we used a pylids model trained using 50 additional frames selected using guided selection from the LPW dataset and evaluated it on the test set specified in (R. S. Kothari et al., 2020) so that we could benchmark our model against EllSeg. This resulted in a median error of 1.2 pixels for the estimation of the pupil center. Our full model's pupil center estimation error is comparable to EllSeg's (median = 0.8 pixels) and other state of the art models DeepVOG (median = 0.9 pixels), and RITnet (median = 4.7 pixels) (Chaudhary et al., 2019; R. S. Kothari et al., 2020; Yiu et al., 2019)

Selection of augmentations

Next, we evaluated models trained on guided sampling of augmentations to estimate the gain in model performance relative to other selection methods. To this end, we compared the change in our baseline model performance after training our initial pylids model using additional augmented frames selected by guided augmentation, random augmentation and nearest augmentation (Figure 5B). These models were trained on frames and augmentations from the GiW dataset and did not include any new labeled frames for other datasets. For all datasets we compared the performance of guided augmentation with models trained on nearest and random augmentation along with the baseline model performance. For eyelid estimation errors for the Fuhl dataset, guided augmentation had a mean error of 11.50 pixels, SEM = 0.17 pixels, which was significantly lower than keypoint estimation error for models trained using random augmentations (mean = 12.41 pixels, SEM = 0.20 pixels, $t = 7.55, p < 0.05$), nearest augmentation (mean = 12.74 pixels, SEM = 0.18 pixels, $t = 11.32, p < 0.05$) and the baseline model (mean = 12.75 pixels, SEM = 0.25 pixels, $t = 6.51, p < 0.05$).

We also compared the improvement in pupil estimation error as a function of different augmentation sampling methods. For the LPW dataset guided augmentation (mean = 4.77 pixels, SEM = 0.02) performed better than random augmentation (mean = 5.09 pixels, SEM = 0.03 pixels, $t = 4.58, p > 0.05$), nearest augmentation (mean = 5.15 pixels, SEM = 0.05 pixels, $t = 7.05, p < 0.05$) and the baseline model (mean = 5.13 pixels,

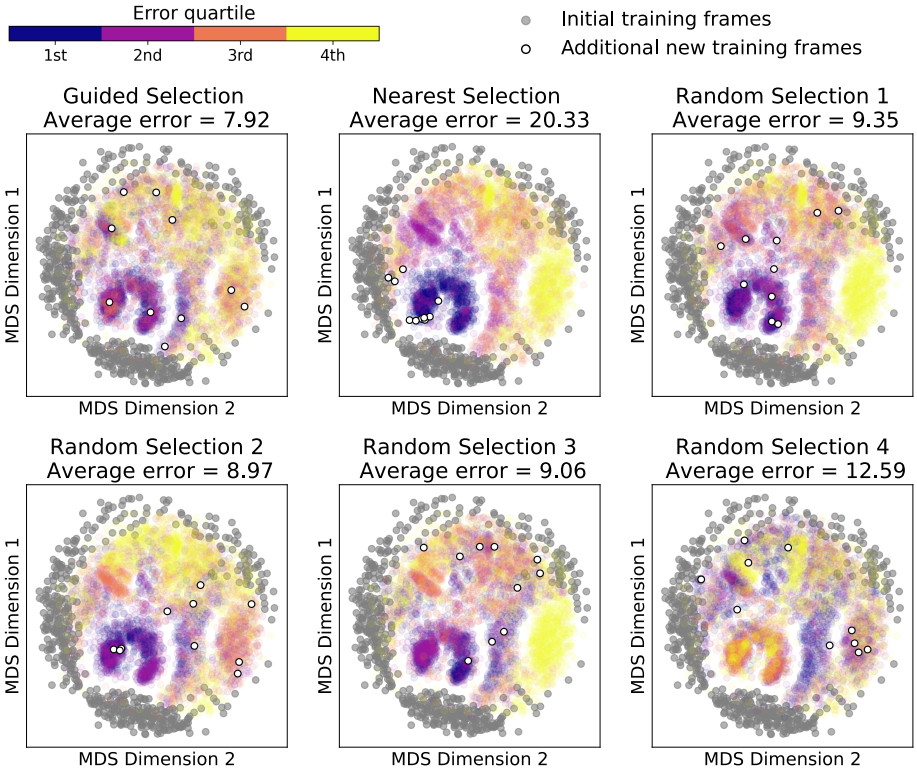

Figure 6

Model performance based on different sampling methods visualized using multi-dimensional scaling. Across all sampling methods model performance is good (error is low) in the neighborhood of training frames (white) demonstrating why guided selection which uniformly samples new training data results in the least average error. Nearest selection —in which additional training data is selected from the neighborhood of samples on which the model already performs well—results in the highest average error. Random selection performance varies based on how uniformly the newly selected training data is distributed.

SEM = 0.03 pixels, $t = 3.14, p < 0.05$).

Discussion

In this work, we develop methods to accurately track both pupils and eyelids based on a few carefully selected training frames. We share our code for this method in an open-source python package (pylids —https://github.com/piecesofmindlab/pylids). Pylids is based on DeepLabCut (Mathis et al., 2018a), a deep neural network for pose estimation. Our pylids model accurately predicts the shape of the eyelids and pupil centroids (Figures 1, C1). Our models also perform within the variance of human annotations when evaluated on a test split sampled from a distribution similar to the training set. When evaluated on dissimilar data, model performance suffers (as is true for nearly all DNNs) (Figure 3). However, fine tuning our baseline model (trained on 520 frames of eye data) with a minimal amount of carefully selected additional training data (50 frames), our model achieves performance on par with state-of-the-art algorithms on two large data sets. In general, other models leverage far more data to achieve comparable performance (e.g. EllSeg (R. S. Kothari et al., 2020) is trained on sets of 8,826 to 93,127 eye images; DeepVOG (Yiu et al., 2019) uses 3,946 training eye images and RITnet uses 8,916 training eye images (Chaudhary et al., 2019)).

Training generalizable DNNs is a challenging problem. We approach this problem by attempting to find non-redundant training data which is informative for our model. We see the efficiency of our model's training regimen as a central contribution of this work. To find an efficient set of samples to train our model, we build on previous work on core-set selection (Coleman et al., 2020; Sener & Savarese, 2017) to devise an algorithm we call *guided selection*. We use this algorithm to efficiently select new frames to label and fine tune our model. Guided selection can also be used to efficiently select augmentations for reducing generalization errors to new datasets. Models trained using guided selection consistently perform better at both eyelid and pupil estimation when compared to models trained using random selection and nearest selection methods (Figure 5A,B). Our guided

selection trained model outperforms models trained with the other sampling methods by up to 40% (Figure 5C). Our method provides an approximate algorithm to substantially increase model performance while also reducing the time and effort required to train a model using new labeled data.

In this study, we also replicate and extend findings in our previous work (Biswas et al., 2021) showing that model performance on a given eye video frame is proportional to the cosine distance from the training distribution to that frame in ResNet50 space (Figure 4A). Here, we demonstrate that guided sampling based on the distance of new frames to the training data leads to better model performance than random sampling 6. Using the same principle we also hypothesized that sampling new training frames close to the pre-existing training distribution would perform the worst. These results holds true for both selecting new frames to label and selecting new augmentations (Figure 5).

We note that guided sampling is more beneficial for selecting new frames to label as compared to selecting augmentations (Figure 5B). This is likely because the variability that image augmentations can add to a training dataset is constrained by the variability of the dataset from which the augmentations are generated. This limits the benefit that guided sampling can provide over random selection of augmentations. However, due to our constrained problem domain we can add specific augmentations that are likely to be important sources of real variation. For example, we added eye reflections, defocus blur and motion blur along with other perturbations (see Appendix A for a detailed list). This allows us to augment the training data with specific kinds of variation that we observe to be absent in the initial training data. In this same vein, guided sampling provides a principled way to balance the number of image augmentations and original frames in a training data set. This balancing can be challenging because it's possible to create a very large number of augmentations. Many augmentations will increase model training time, sometimes to little benefit or even to a cost in model performance. Guided selection chooses only the augmentations that span the relevant variability in new data.

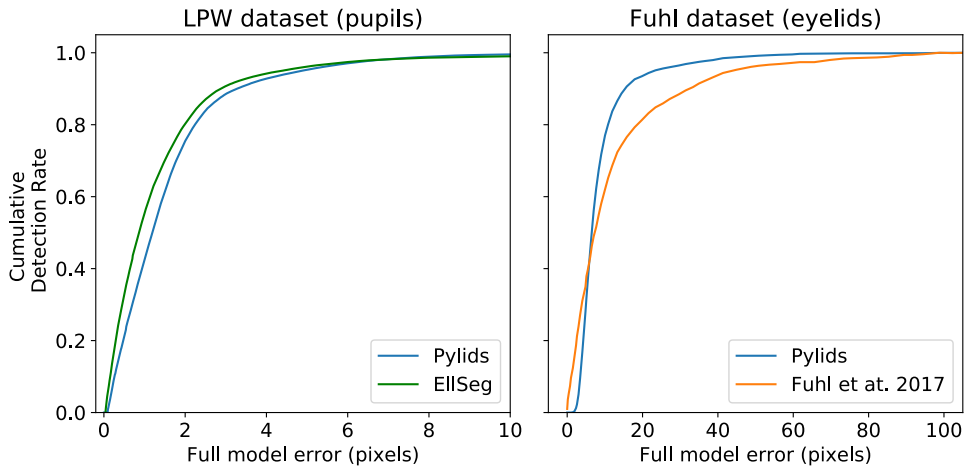

Figure 7

Comparison of model performance of pylids with state-of-the-art models for eyelid and pupil estimation. (A) For the LPW dataset we visualized the cumulative detection rate based on error in estimation of the pupil center error for pylids and EllSeg. (B) For the Fuhl dataset we compare the cumulative detection rate based on the Hausdorff distance for pylids and the model mentioned in (Fuhl, Santini, & Kasneci, 2017). The values corresponding to the Fuhl eyelids estimation model were obtained from Figure 11 (c) in (Fuhl, Santini, & Kasneci, 2017)

Several avenues for development remain which may further improve our model's performance. First, processing frames of eye videos with our model is relatively slow; it runs at approximately 82 frames per second for one eye at 400x400 pixels on an NVIDIA RTX2080 graphics card. This is slower than real time for common rates of eye tracking acquisition (often 120-200 Hz), even with desktop hardware. Our focus with this model was on post-hoc processing of gaze for research, not on real-time applications, so we chose to prioritize accuracy over speed. Second, the accuracy of our pupil model relies on estimation of an ellipse fit to the estimated keypoints around the edge of the pupil. Alternative and more computationally intensive means to exclude outliers and estimate the pupil ellipse (e.g. RANSAC) may increase accuracy of our pupil detection. However, these

might further slow down the algorithm.

Additionally, there are possible ways to improve our guided selection algorithm. A recent study comparing all core-set selection algorithms demonstrated that even though these methods provide an advantage for DNN training data selection, random selection is still a strong baseline, especially for homogeneous datasets (Guo et al., 2022). Prevalent core-set selection techniques for DNNs rely on training a new model on an initial labeled subset of the data for a given task (e.g. object classification) (Coleman et al., 2019; Sener & Savarese, 2017). This helps the model learn an approximate feature representations enabling it to optimize for the prescribed task. Following this, activations from the neural network's final hidden layer extracted using prospective training data are used to select new frames to label using the greedy k-centers algorithm. In this paper we use a slightly different approach, utilizing the activations of the final layer from a ResNet50 model pre-trained on the ImageNet dataset (Deng et al., 2009) on an object classification task and not on eyelid and pupil keypoint estimation. Our approach prioritizes using robust features derived from a large dataset (ImageNet) on a task (Image classification) under the assumption that there is an overlap between feature representations that helps both classify objects and track eyes. This has been demonstrated by Zamir et al. (2018) who showed that there is a shared latent space between a number of computer vision tasks such as image classification and keypoint estimation. Future work can explore whether using a set of features based on a limited training set for eyetracking is more beneficial compared to using a more robust set of features derived from a large dataset but on a less related task.

After extracting DNN activations many different methods exist for selecting uniformly distributed frames in a vector space. Recent core-set selection algorithms used to efficiently select a subset of frames to train DNNs have relied on greedy k-centers (Coleman et al., 2019; Sener & Savarese, 2017). Other alternatives include k-means, as used by us, and k-medioids (H.-S. Park & Jun, 2009). Greedy k-centers selects cluster centers that minimize the maximum distance to any given point. Using this sampling method we would

minimize the maximum error of a DNN model. We chose k-means which minimizes the median error on a dataset. k-means and greedy k-centers have a time complexity of $O(tkn)$, where as k-medioids has a time complexity of $O(tkn^2)$, when partitioning n elements into k clusters using t iterations. The linear run time for k-means and greedy k-centers make it more advantageous than using k-medioids for which run time scales non-linearly with dataset size. Despite being slower, k-medioids does provide the benefit of selecting actual data points that minimize sum of the squared distance. By contrast, k-means and greedy k-centers choose cluster centers that do not necessarily correspond to any sampled data point. A drawback for all these method is that they do not account for the density of the different clusters over which the distance or squared distance is minimized. Applying k-means to clusters with approximately equal samples as described in (Malinen & Fränti, 2014) can ensure more uniform and equitable distribution of the new to-be-selected training frames across the test distribution —possibly boosting model performance. Future studies can include comparison of these different uniform sampling methods to determine optimizing for which of the above discussed criteria leads to most benefit.

Kothari et al. 2022 (R. S. Kothari et al., 2022) demonstrated that eyetracking models trained on multiple different datasets generalize better to unseen data compared to dataset-specific models. Combining this result with the framework developed in this paper we suggest training a model with labels from multiple varying eye datasets. This model can then be efficiently fine tuned based on the methods provided in this paper. A similar framework exists for large DNN models used for object classification and natural language processing. Standard DNN models such as AlexNet, VGG-16, and GPT-3 trained on an initial large dataset are fine tuned using an additional smaller set of training data to help generalize to new datasets (Brown et al., 2020; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014). Although the training time for our baseline pylids model is non-trivial, our framework provides the tools for efficiently selecting and labeling frames and fine tuning the initial model. Substantial increases in model performance often require labeling

and training using only 10 additional frames, which can be done in an hour.

With the recent improvements in graphical rendering, generative modeling (Karras et al., 2020; Zhu et al., 2017) and emergence of diffusion based methods (Ramesh et al., 2022; Rombach et al., 2021)it has become possible to synthesize training data. Recent studies have demonstrated the ability to improve gaze estimation using synthesized eye images and labels (Chaudhary et al., 2022; Nair et al., 2020; S. Park et al., 2019; Swirski & Dodgson, 2013). Combining these data synthesis methods with our method opens up the possibility of real time synthesis of training data that promotes generalization by generating images that add image augmentations based on the dataset of interest.

Although current DNNs models used for eyetracking are efficient at pupil estimation (Chaudhary et al., 2019; R. S. Kothari et al., 2020) these may not generalize to datasets very different from the ones on which they were trained (R. S. Kothari et al., 2022). Current datasets span a substantial degree of variability in eye appearance (Fuhl, Santini, & Kasneci, 2017; R. Kothari et al., 2020; Tonsen et al., 2016), but we suspect even more varied data will become available as mobile eye tracking data is collected under more and more circumstances (e.g. more active tasks in outdoor environments) and by more diverse people. We expect that our efficient label selection will be of value to help current algorithms accurately generalize to new large and highly variable datasets.

The success of our pylids model in defining the full eyelid shape with high precision also paves way for detailed, quantitative studies of blink dynamics such as characterization of blinks across individuals and tasks. The robust estimate of eyelids may also be useful for detection of eye camera slippage in mobile eye tracking systems —allowing more accurate gaze estimation over long recording sessions.

Acknowledgments

We thank Kaylie Capurro for helping label the eyelid and pupil data. We would also like to thank Kamran Binaee, Matthew Shinkle and Joseph (Yu) Zhao for helpful discussions. This work was supported by NSF EPSCoR # 1920896 to Michelle R. Greene,

Mark D. Lescroart, Paul MacNeilage, and Benjamin Balas.

References

Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. (2017). A closer look at memorization in deep networks. *International conference on machine learning*, 233–242.

Binaee, K., Sinnott, C., Capurro, K. J., MacNeilage, P., & Lescroart, M. D. (2021). Pupil tracking under direct sunlight. *ACM Symposium on Eye Tracking Research and Applications*, 1–4.

Biswas, A., Binaee, K., Capurro, K. J., & Lescroart, M. D. (2021). Characterizing the performance of deep neural networks for eye-tracking. *ACM Symposium on Eye Tracking Research and Applications*, 1–4.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. *CoRR*, *abs/2005.14165*. https://arxiv.org/abs/2005.14165

Chaudhary, A. K., Kothari, R., Acharya, M., Dangi, S., Nair, N., Bailey, R., Kanan, C., Diaz, G., & Pelz, J. B. (2019). Ritnet: Real-time semantic segmentation of the eye for gaze tracking. *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 3698–3702. https://doi.org/10.1109/ICCVW.2019.00568

Chaudhary, A. K., Nair, N., Bailey, R. J., Pelz, J. B., Talathi, S. S., & Diaz, G. J. (2022). Temporal rit-eyes: From real infrared eye-images to synthetic sequences of gaze behavior. *IEEE Transactions on Visualization and Computer Graphics*, *28*(11), 3948–3958.

Chaudhary, A. K., Gyawali, P. K., Wang, L., & Pelz, J. B. (2021). Semi-supervised learning for eye image segmentation. *ACM Symposium on Eye Tracking Research and Applications*, 1–7.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of artificial intelligence research*, *4*, 129–145.

Coleman, C., Yeh, C., Mussmann, S., Mirzasoleiman, B., Bailis, P., Liang, P., Leskovec, J., & Zaharia, M. (2019). Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*.

Coleman, C., Yeh, C., Mussmann, S., Mirzasoleiman, B., Bailis, P., Liang, P., Leskovec, J., & Zaharia, M. (2020). Selection via proxy: Efficient data selection for deep learning. *International Conference on Learning Representations*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. *CVPR09*.

Eivazi, S., Santini, T., Keshavarzi, A., Kübler, T., & Mazzei, A. (2019). Improving real-time cnn-based pupil detection through domain-specific data augmentation. *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. https://doi.org/10.1145/3314111.3319914

Fischer, T., Chang, H. J., & Demiris, Y. (2018). Rt-gene: Real-time eye gaze estimation in natural environments. *Proceedings of the European conference on computer vision (ECCV)*, 334–352.

Fuhl, W., Santini, T., Kasneci, G., & Kasneci, E. (2017). Pupilnet v2.0: Convolutional neural networks for robust pupil detection. *CoRR*.

Fuhl, W., Kasneci, G., & Kasneci, E. (2021). Teyed: Over 20 million real-world eye images with pupil, eyelid, and iris 2d and 3d segmentations, 2d and 3d landmarks, 3d eyeball, gaze vector, and eye movement types. *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 367–375.

Fuhl, W., Santini, T., & Kasneci, E. (2017). Fast and robust eyelid outline and aperture detection in real-world scenarios. *2017 IEEE Winter conference on applications of computer vision (WACV)*, 1089–1097.

Gal, Y., Islam, R., & Ghahramani, Z. (2017). Deep bayesian active learning with image data. *International Conference on Machine Learning*, 1183–1192.

Gander, W., Golub, G. H., & Strebel, R. (1994). Least-squares fitting of circles and ellipses. *BIT Numerical Mathematics*, *34*(4), 558–578.

Garbin, S. J., Shen, Y., Schuetz, I., Cavin, R., Hughes, G., & Talathi, S. S. (2019). Openeds: Open eye dataset. *arXiv preprint arXiv:1905.03702*.

Guo, C., Zhao, B., & Bai, Y. (2022). Deepcore: A comprehensive library for coreset selection in deep learning. *arXiv preprint arXiv:2204.08499*.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, *abs/1512.03385*. http://arxiv.org/abs/1512.03385

Hennessey, C., Noureddin, B., & Lawrence, P. (2006). A single camera eye-gaze tracking system with free head motion. *Proceedings of the 2006 symposium on Eye tracking research & applications*, 87–94.

Jung, A. B., Wada, K., Crall, J., Tanaka, S., Graving, J., Reinders, C., Yadav, S., Banerjee, J., Vecsei, G., Kraft, A., Rui, Z., Borovec, J., Vallentin, C., Zhydenko, S., Pfeiffer, K., Cook, B., Fernández, I., De Rainville, F.-M., Weng, C.-H., ... Laporte, M., et al. (2020). imgaug [Online; accessed 01-Feb-2020].

Kansal, P., & Devanathan, S. (2019). Eyenet: Attention based convolutional encoder-decoder network for eye region segmentation. *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 3688–3693.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8110–8119.

Kassner, M., Patera, W., & Bulling, A. (2014). Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction. 2014. *arXiv preprint arXiv:1405.0006*, *10*(2638728.2641695).

Katsini, C., Abdrabou, Y., Raptis, G. E., Khamis, M., & Alt, F. (2020). The role of eye gaze in security and privacy applications: Survey and future hci research directions. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–21.

Kingma, D. P., & Ba, J. (2017). Adam: A method for stochastic optimization.

Kothari, R., Yang, Z., Kanan, C., Bailey, R., Pelz, J. B., & Diaz, G. J. (2020). Gaze-in-wild: A dataset for studying eye and head coordination in everyday activities. *Scientific reports*, *10*(1), 1–18.

Kothari, R. S., Bailey, R. J., Kanan, C., Pelz, J. B., & Diaz, G. J. (2022). Ellseg-gen, towards domain generalization for head-mounted eyetracking. *Proceedings of the ACM on Human-Computer Interaction*, *6*(ETRA), 1–17.

Kothari, R. S., Chaudhary, A. K., Bailey, R. J., Pelz, J. B., & Diaz, G. J. (2020). Ellseg: An ellipse segmentation framework for robust gaze tracking. *arXiv preprint arXiv:2007.09600*.

Kouw, W. M., & Loog, M. (2018). An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, 1097–1105.

Kruskal, J. B., & Wish, M. (1978). *Multidimensional scaling*. Sage.

Labs, P. (2013). *Pupil labs github repository*. GitHub repository. https://github.com/pupil-labs/pupil

Lauer, J., Zhou, M., Ye, S., Menegas, W., Schneider, S., Nath, T., Rahman, M. M., Di Santo, V., Soberanes, D., Feng, G., et al. (2022). Multi-animal pose estimation, identification and tracking with deeplabcut. *Nature Methods*, *19*(4), 496–504.

Malinen, M. I., & Fränti, P. (2014). Balanced k-means for clustering. *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, 32–41.

Mathis, A., Biasi, T., Schneider, S., Yuksekgonul, M., Rogers, B., Bethge, M., & Mathis, M. W. (2021). Pretraining boosts out-of-domain robustness for pose estimation. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 1859–1868.

Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018a). Deeplabcut: Markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*. https://www.nature.com/articles/s41593-018-0209-y

Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018b). Deeplabcut: Markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience*, *21*(9), 1281–1289.

Meyer, A. F., O'Keefe, J., & Poort, J. (2020). Two distinct types of eye-head coupling in freely moving mice. *Current Biology*, *30*(11), 2116–2130.

Nair, N., Kothari, R., Chaudhary, A. K., Yang, Z., Diaz, G. J., Pelz, J. B., & Bailey, R. J. (2020). Rit-eyes: Rendering of near-eye images for eye-tracking applications. *ACM Symposium on Applied Perception 2020*, 1–9.

Nath*, T., Mathis*, A., Chen, A. C., Patel, A., Bethge, M., & Mathis, M. W. (2019). Using deeplabcut for 3d markerless pose estimation across species and behaviors. *Nature Protocols*. https://doi.org/10.1038/s41596-019-0176-0

Neyshabur, B., Tomioka, R., & Srebro, N. (2014). In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., & Sohl-Dickstein, J. (2018). Sensitivity and generalization in neural networks: An empirical study. *arXiv preprint arXiv:1802.08760*.

Park, H.-S., & Jun, C.-H. (2009). A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, *36*(2), 3336–3341.

Park, S., Mello, S. D., Molchanov, P., Iqbal, U., Hilliges, O., & Kautz, J. (2019). Few-shot adaptive gaze estimation. *Proceedings of the IEEE/CVF international conference on computer vision*, 9368–9377.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Rebecq, H., Ranftl, R., Koltun, V., & Scaramuzza, D. (2019). High speed and high dynamic range video with an event camera. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01), 1–1. https://doi.org/10.1109/TPAMI.2019.2963386

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2021). High-resolution image synthesis with latent diffusion models.

Rot, P., Emeršič, Ž., Struc, V., & Peer, P. (2018). Deep multi-class eye segmentation for ocular biometrics. *2018 IEEE international work conference on bioinspired intelligence (IWOBI)*, 1–8.

Sener, O., & Savarese, S. (2017). Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Settles, B. (2009). Active learning literature survey.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*(1), 1–48.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Swirski, L., & Dodgson, N. (2013). A fully-automatic, temporal approach to single camera, glint-free 3d eye model fitting. *Proc. PETMEI*, 1–11.

Tonsen, M., Zhang, X., Sugano, Y., & Bulling, A. (2016). Labelled pupils in the wild: A dataset for studying pupil detection in unconstrained environments. *Proceedings of*

the ninth biennial ACM symposium on eye tracking research & applications, 139–142.

Torralba, A., & Efros, A. A. (2011). Unbiased look at dataset bias. *CVPR 2011*, 1521–1528.

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., Yu, T., & the scikit-image contributors. (2014). Scikit-image: Image processing in Python. *PeerJ*, *2*, e453. https://doi.org/10.7717/peerj.453

Vera-Olmos, F. J., Pardo, E., Melero, H., & Malpica, N. (2019). Deepeye: Deep convolutional network for pupil detection in real environments. *Integrated Computer-Aided Engineering*, *26*(1), 85–95.

Wang, T., Zhu, J.-Y., Torralba, A., & Efros, A. A. (2018). Dataset distillation. *arXiv preprint arXiv:1811.10959*.

Yiu, Y.-H., Aboulatta, M., Raiser, T., Ophey, L., Flanagin, V. L., Zu Eulenburg, P., & Ahmadi, S.-A. (2019). Deepvog: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning. *Journal of neuroscience methods*, *324*, 108307.

Zamir, A. R., Sax, A., Shen, W., Guibas, L. J., Malik, J., & Savarese, S. (2018). Taskonomy: Disentangling task transfer learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zdarsky, N., Treue, S., & Esghaei, M. (2021). A deep learning-based approach to video-based eye tracking for human psychophysics. *Frontiers in human neuroscience*, *15*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, *64*(3), 107–115.

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE international conference on computer vision*, 2223–2232.

Appendix A

Data augmentation

To simulate outdoor eye images, we used image processing techniques to artificially add perturbations to eye images in our training dataset by adding exposure, reflection, defocus blur, eye rotation, JPEG compression, motion blur, Gaussian noise and adding mock pupils and glints to eye images. For each perturbation we generated four additional images with increasing perturbation intensities. We detail the process of generating perturbed eye images below.

1. **Exposure:** To simulate the effect of an increase in exposure and decrease in the contrast between the pupil/eyelashes and other regions in the eye video in bright sunlight, we added four steps of luminance increments (each 35 units) to all pixels in each frame. After each increment, pixel values were clipped to a maximum of 255.

2. **Rotation** To simulate the effect of different camera angles and facial anatomy across participants, we rotated the eye videos in four five-degree increments followed by scaling and cropping to ensure uniform frame size. This rotation resulted in the eye going partially out of the frame for the 15 and 20-degree rotation conditions.

3. **Reflection:** Corneal reflection and shadows on the eye present a challenge while recording eye videos outdoors. We used the method presented in (Eivazi et al., 2019) to add reflections and shadows to the eye images. We modified the blending factor for images superimposed on the eye video in four steps. For every frame, we randomly selected the reflected image from the Driving Events Camera Dataset (Rebecq et al., 2019) which contains videos from dashboard cameras of cars driving through highways and cityscapes.

4. **JPEG artifacts:** Compressed video formats are desirable when storing eye videos as they take up less space. Thus, we tested the robustness of our DNN to compression artifacts by altering the video frames with JPEG compression. We varied the JPEG

quality parameter (which varies from 100 to 0, denoting best to worst quality) from 32 to 8 in four steps of 8.

5. **Defocus Blur:** To mimic the defocus blur from a camera we used the *imgaug* image augmentation library (Jung et al., 2020) and iteratively increased the severity parameter from 1 to 4 to create an incremental loss of focus in the eye videos.

6. **Motion Blur:** To simulate the motion blur due to saccadic eye movements or blinks we used the motion blur as implemented in (Jung et al., 2020). Motion blur was varied between intensities 20 and 80. The angle for the direction for motion blur was randomly sampled from within the range [-45, 45] degrees.

7. **Gaussian Noise:** To simulate the effect of Gaussian Noise we used the add Gaussian noise function as implemented in (Jung et al., 2020). We added Gaussian noise to eye images sampled once per pixel from a normal distribution ranging from N(0, 0.1*255) to N(0, 0.3*255).

8. **Mock Pupils:** Often pupil like structures appear on eye images due to shadows, reflections based on the environment to avoid spurious detection of pupils due to such structure we augmented our data with blacked out mock pupils inspired bey (Eivazi et al., 2019). We created blacked out ellipses to simulate mock pupils, the center of the ellipse was sampled from a 2D Gaussian distribution centered on the center of the frame and spanning half the height and width of the eye image.

9. **Mock Glints:** Using the same procedure as mock pupils we created white ellipses to simulate the appearance of glints due to reflections and environmental conditions.

Appendix B

Multi dimensional scaling

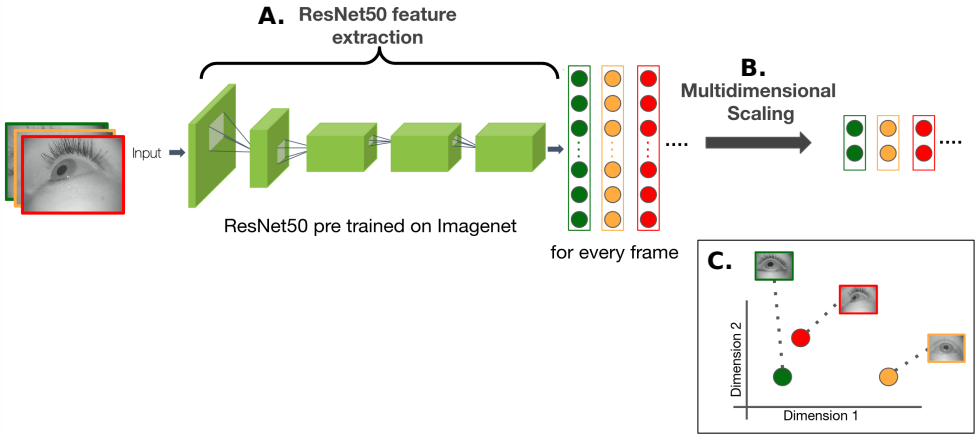

Figure B1

To visualize eye frames using multi-dimensional scaling (MDS) we first (A) pass the eye frames through a ImageNet pre-trained ResNet50 and extract the representation of the last convolutional layer before the fully connected layers. This gives us a 100,352 dimension vector for each eye frame. (B) We apply MDS to the set of ResNet50 extracted activations to end up with (C) 2 dimensional representation in MDS space which can be visualized. Each dot in MDS space represents a single eye frame and attempts to faithfully preserve the relative distances in high dimensional ResNet50 space in the low dimensional MDS space.

Appendix C

Comparison with alternate model

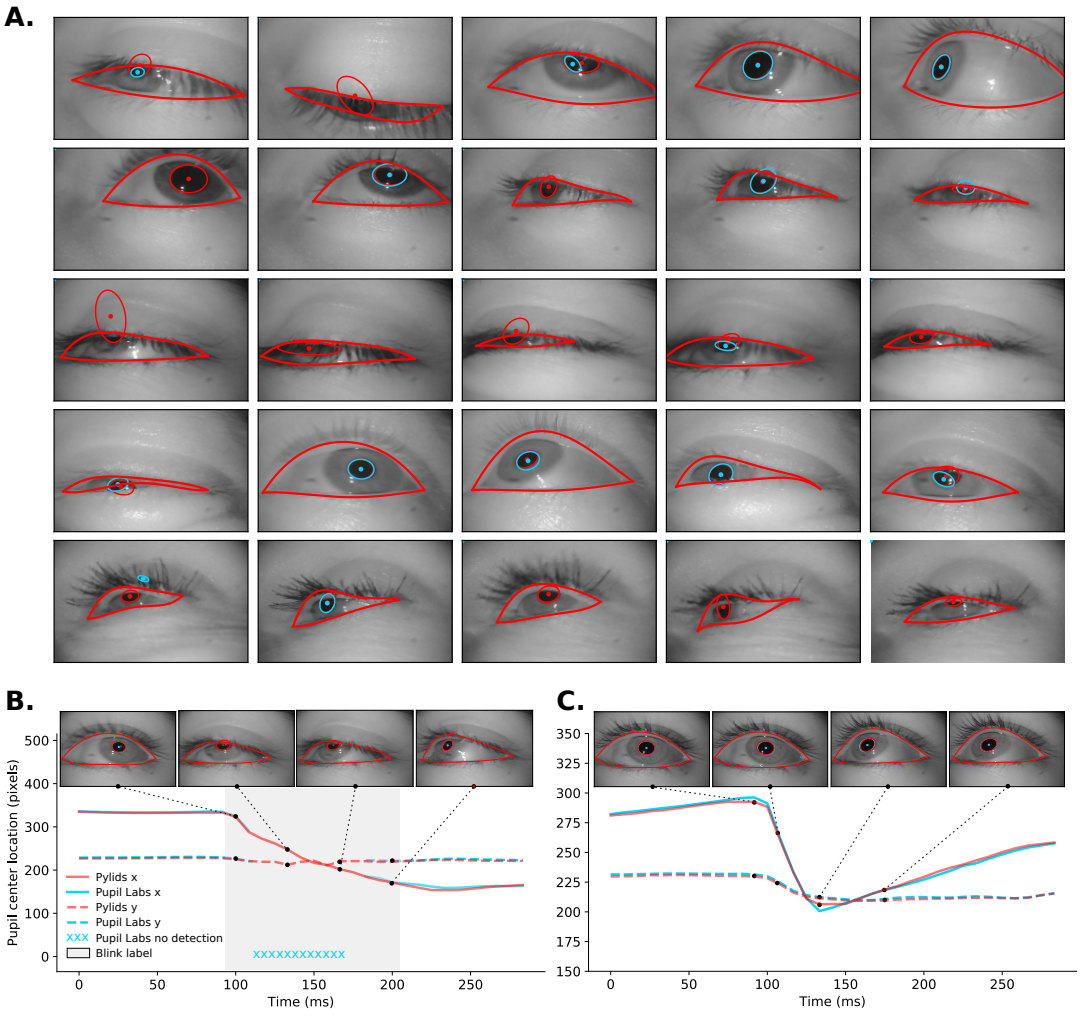

Figure C1

Eyelid polynomial fits and ellipse fits for pupils demonstrating robustness of our baseline pylids model on samples from the GiW dataset. (A) Each row represents data from a different participant from the test data. Pupil positions as estimated by our pylids model are shown in red and compared to pupil estimates using the Pupil Labs package in blue. (B) and (C) compare pylids estimated pupil positions (red) to Pupil Labs package based pupil estimation (blue) during a blink (B) and a saccade (C). Partial pupil occlusion during a blink (B) leads to loss of data when using Pupil Labs pupil detection but not with pylids.

Appendix D

Distance from training distribution is correlated with model performance

First using our baseline pylids model trained on 520 labels from the Gaze in Wild dataset we evaluated if model performance is dependent upon cosine distance from the training dataset in ResNet50 space. To test this we computed the rank order correlation between the model error on the test distribution and the Cosine distance from the training distribution in ResNet50 space (see Figure D1).

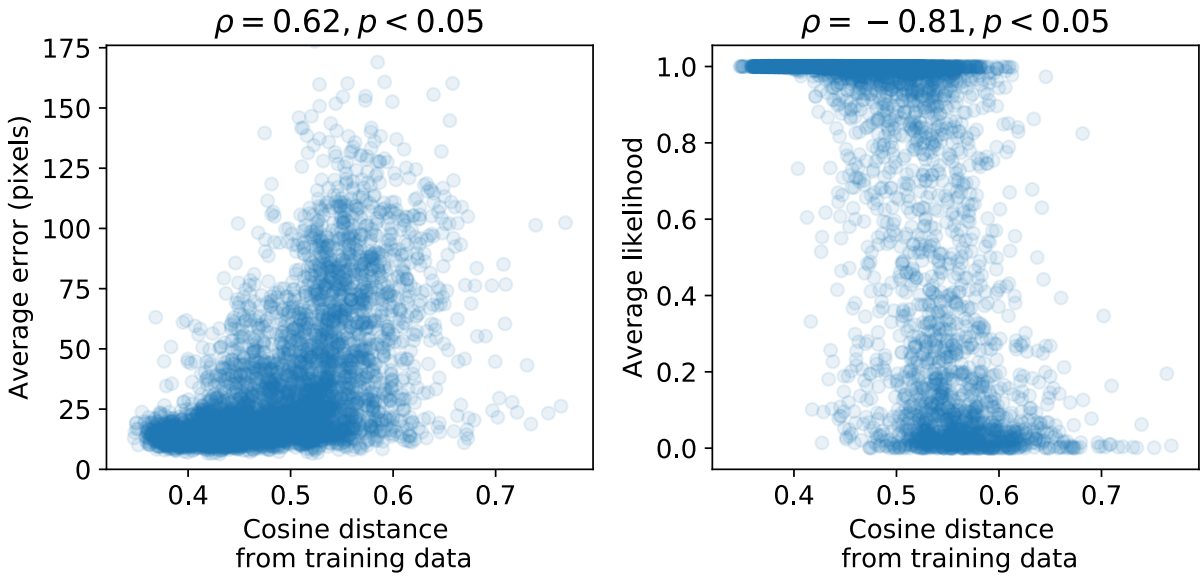

Figure D1

Correlation of cosine distance between GiW training data and test samples from the Fuhl dataset with average error and average keypoint likelihood for eyelid estimation. With an increase in Cosine distance there is monotonic increase in the average error.

Based on our finding that distance from the training distribution is correlated with model performance we used our guided sampling algorithm to select uniformly distributed samples to fine tune our model. We show that models fine tuned using guided sampling outperform models trained using other sampling methods (see Figure 6). To evaluate that the relationship between distance from the training distribution and model performance

holds for the fine tuned models we correlated the model error with distance from the training distribution (which now included new labeled frames.) We found that there is significant correlation and the relationship still holds true (see Figure D2)

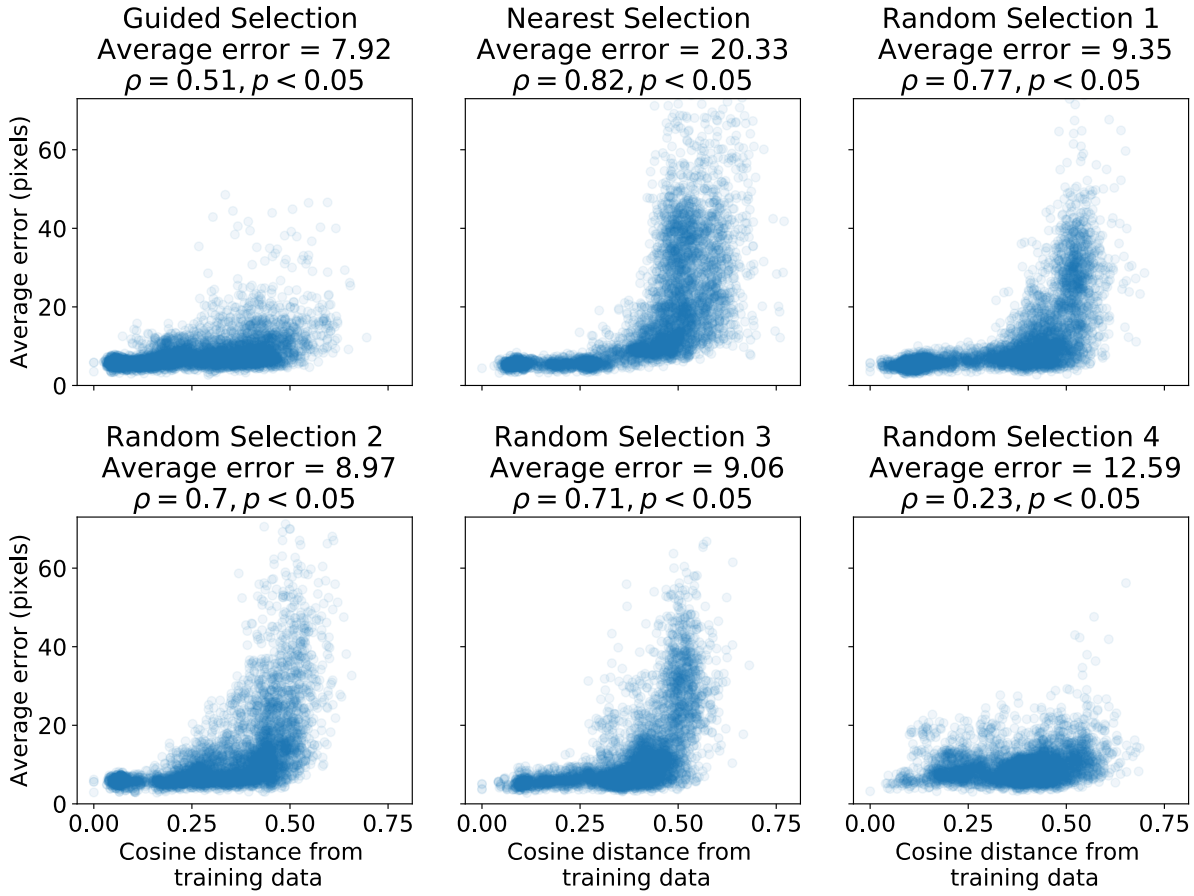

Figure D2

Correlation between model error and cosine distance from training data for models fine tuned using samples from the Fuhl dataset. The subplots are in the same order as 6. The correlation is significant across sampling methods.

To further investigate if guided sampling did indeed result in reduction in distance from the training distribution we plotted the model error as a function of both distance to the initial baseline training samples and newly labeled samples. Compared to other sampling methods guided sampling reduced the distance from the training distribution for

all test samples (see Figure D3).

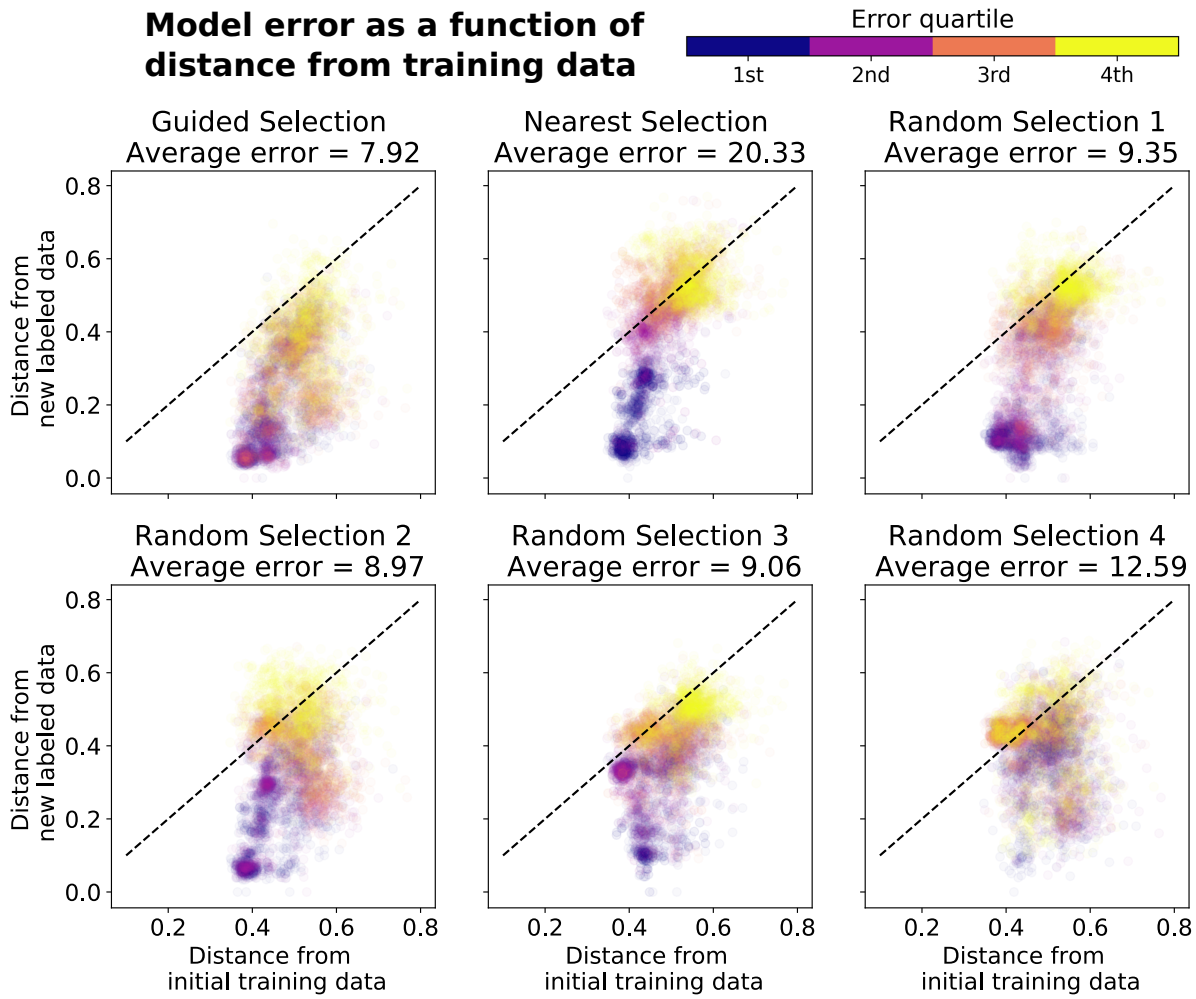

Figure D3

Model error as a function of both the cosine distance from the initial training dataset and the new selected frames for training based on the Fuhl dataset. These subplots correspond to the same subplots visualized in Figure 6 and Figure D2. The dashed diagonal line represents the identity line. More samples below the diagonal shows that samples are closer to the new labeled training frames than the initial baseline training data. This is especially true for guided sampling.