2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) | 978-1-6654-8739-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CVPRW56347.2022.00548

Unsupervised Multi-View Gaze Representation Learning

John Gideon
Toyota Research Institute
Cambridge, MA, USA
john.gideon@tri.global

Shan Su
University of Pennsylvania
Philadelphia, PA, USA
sushan@seas.upenn.edu

Simon Stent
Toyota Research Institute
Cambridge, MA, USA
simon.stent@tri.global

Abstract

We present a method for unsupervised gaze representation learning from multiple synchronized views of a person's face. The key assumption is that images of the same eye captured from different viewpoints differ in certain respects while remaining similar in others. Specifically, the absolute gaze and absolute head pose of the same subject should be different from different viewpoints, while appearance characteristics and gaze angle relative to the head coordinate frame should remain constant. To leverage this, we adopt a cross-encoder learning framework, in which our encoding space consists of head pose, relative eye gaze, eye appearance and other common features. Image pairs which are assumed to have matching subsets of features should be able to swap those subsets among themselves without any loss of information, computed by decoding the mixed features back into images and measuring reconstruction loss. We show that by applying these assumptions to an unlabelled multi-view video dataset, we can generate more powerful representations than a standard gaze cross-encoder for few-shot gaze estimation. Furthermore, we introduce a new feature-mixing method which results in higher performance, faster training, improved testing flexibility with multiple views, and added interpretability with learned confidence.

1. Introduction

Understanding where people look and why is an important aspect of understanding and interacting with humans in many different settings. Gaze estimation, in both 2D (e.g. on a screen) and 3D (e.g. in the world), has therefore received a lot of attention from computer vision and human-computer interaction researchers (see [7] for a historical survey). Appearance-based gaze estimation is an attractive solution to gaze estimation, because it is unobtrusive, places no strong requirements on hardware, and is data-driven, imposing very few assumptions on the structure of the data. Neural networks were first applied to appearance-based gaze estimation in the early 1990s [1], but in the past decade, such early efforts have been reworked and supercharged through the application of modern deep learning techniques on large labeled datasets [11, 14, 24, 30, 31].

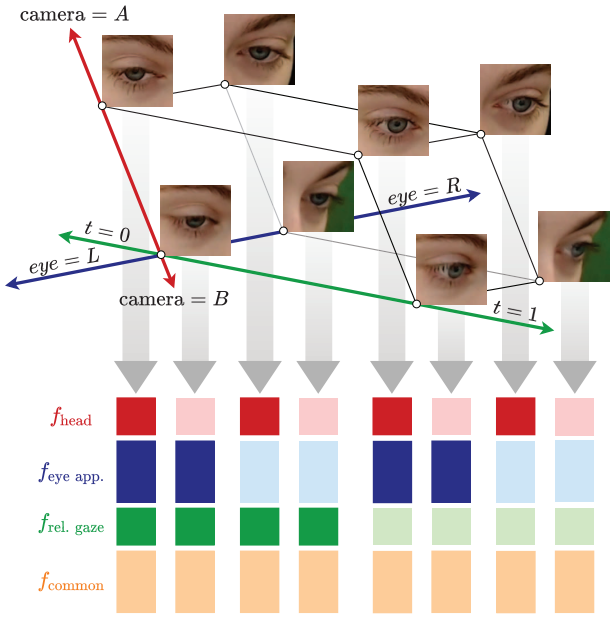


Figure 1. **Overview.** We extend recent work on single-view unsupervised gaze representation learning [27] to take advantage of synchronized multi-view gaze video datasets, such as EVE [16]. Our approach assumes an eye image can be losslessly compressed into a structured representation comprising features for head pose, eye appearance, gaze direction (relative to head pose), and common appearance features. We introduce a method to learn from unlabelled pairs of images sampled from multi-view face video sequences, forcing consistency among features depending on whether the images are sampled across camera viewpoint, left or right eye, or time.

However, the cost of creating such datasets scales linearly with size, and numerous methods have been explored to avoid the need for them altogether, including synthetic data [22, 26, 28] and other forms of freely available supervision [13, 25, 27, 29]. Here, we explore a method in the

978-1-6654-8739-9/22/$31.00 ©2022 IEEE
DOI 10.1109/CVPRW56347.2022.00548

latter category: we seek to learn gaze representations from unlabeled data in an unsupervised (or self-supervised) manner. The representations we learn are designed to encode information from four key factors of variation which govern the eye image: the underlying head pose, the gaze direction of the eye relative to the head pose, the appearance of the eye (governing details like eye lash thickness, or iris color, which are independent of gaze direction altogether), and factors common across all views (glasses, lighting, and skin color). After pre-training on an unlabelled dataset, the representations can be used for few-shot training to produce 3D gaze estimation models. We adopt a "cross-encoder" approach inspired by the recent work of Sun et al. [27]. Pairs of images are encoded into features, swap various subsets of features which are assumed to be shared, and then are decoded back into images to compute reconstruction loss. For example, comparing the two left-most columns in Figure 1, an image of the same eye taken at the same time in different cameras (A and B) can be expected to share the same appearance feature, the same relative gaze feature, but a different head pose feature (since the cameras are from different unknown viewpoints). By selecting image pairs where certain features are expected to stay constant while others are expected to differ, the model learns to both estimate and disentangle the main factors of variation behind appearance-based gaze.

Our work makes the following contributions:

- We extend the work of Sun et al. [27] to take advantage of uncalibrated multi-view image streams. This requires reformulating the structure of the gaze feature: rather than learning a feature directly for gaze direction, we learn features for head pose and relative gaze, combining them to estimate gaze direction, as described in Sec. 3.2. Although more camera views are supported, this training can be accomplished with only a stereo pair.

- We propose a new form of feature augmentation, described in Sec. 3.4, which leads to better explainability, computational efficiency, flexibility, and performance.

- Through careful experimentation with the EVE [16] multi-view video dataset in Sec. 4, we show that our method can boost the performance of the learned representation at in-domain few-shot gaze estimation, and that it successfully disentangles various factors of variation in the data and improves data explainability.

- To support further research in this direction, we make our code available upon publication.[1]

[1] https://github.com/ToyotaResearchInstitute/UnsupervisedGaze

2. Related Work

Gaze Estimation. Building large labeled datasets for gaze estimation [11, 14, 24, 30, 31] is time-consuming and expensive. It is particularly challenging to do well outside of lab settings, which is problematic since many applications of gaze tracking exist outside of lab settings, or "in the wild", where nuisance factors such as lighting, occlusions and pose may have stronger influences. Thus, various techniques have been proposed to reduce the effort required to build such datasets, and more fundamentally, to reduce the reliance on them to create performant gaze estimation systems. For example, synthetic datasets has been created or used in novel ways [22, 26, 28]. Appearance-based models have had strong inductive priors injected into their representations, such as intermediate pictorial structure [18] or a more explicit separation of head pose [4], to help train more robust models from the same data. Zheng et al. [32] introduced an encoder-decoder model for semi-supervised gaze redirection which can learn to encode extraneous factors such as head pose in a self-supervised manner, and can be used to augment training data to improve sample efficiency for gaze estimation. Gaze models have also been personalized or calibrated to individuals with minimal supervision [15, 17]. In addition, other forms of (weak) supervision have been found - particularly by trying to model the likely target of a human subject's eye gaze. This has been attempted through the estimation of scene saliency [2, 16, 20] or through direct estimation by leveraging the prior that people tend to lock gaze with one another [13].

Unsupervised Gaze Representation Learning. Our work is most closely related to recent efforts to learn gaze representations from unlabelled eye images alone. Yu and Odobez [29] recently showed the strong potential of self-supervised gaze representation learning by using gaze redirection as a pretext task. Their approach extracts a two-dimensional representation from each eye image in an input pair which are assumed to have similar head pose, and uses the difference vector to condition a gaze redirection network which attempts to reconstruct one image from the other. To solve this task well, the two-dimensional vector learns to closely resemble gaze pitch and yaw. Subsequent work by Sun et al. [27] proposed a method to take advantage of additional weak assumptions that exist in eye gaze datasets. Their method, the cross-encoder, uses a latent-code-swapping mechanism on image pairs which are assumed to be consistent in either gaze (e.g. left and right eyes of the same face), or eye appearance (e.g. left eyes of the same face at different times). Each eye patch is encoded into a gaze feature and an eye appearance feature, and the cross-encoder reconstructs images in the eye-consistent pair according to their own gaze features and the other's

eye appearance features, and in the gaze-similar pair according to their own eye features and the other's gaze features. They show that their approach improves on two recent, popular contrastive learning methods, SimCLR [3] and BYOL [6], which are not specifically designed or pre-trained for gaze representation learning, but rather for learning more general visual features. Our work builds on the cross-encoder, by introducing a further form of natural supervision: consistency across multiple views. To benefit from this supervision, we take inspiration from Deng and Zhu [4] and further sub-divide the representation space to include a camera-relative head pose feature, which is known to be different across camera views but is assumed to stay roughly consistent in nearby times (since high-speed head motions are uncommon in natural movement).

Multi-View Representation Learning. Leveraging information from other viewpoints for representation learning has been explored in numerous other works outside gaze estimation. For example, in a supervised context, multi-view bootstrapping has been successful for propagating ground truth across viewpoints in order to learn more robust keypoint detectors [23]. In the self-supervised and weakly-supervised settings, various works have shown how multi-view consistency can be used to learn geometry-aware body representations from unlabelled multi-view images [10, 21]. Our work is the first attempt to leverage unlabelled multi-view images for gaze representation learning.

3. Approach

3.1. Data Sampling Strategy

Much of our ability to learn meaningful features from eye patches without supervision is based on our ability to sample them in a structured manner. As shown in Figure 1, we select one *anchor view* and seven complementary eye patch views for each sample in the dataset. These consist of two different camera views, two different eyes (left and right), and two different times within the same sample clip – resulting in eight combinations. We assume that dataset clips are sufficiently short that they may have changing relative gaze but have relatively stable head orientation. These other seven views are selected randomly during training, but are held constant during validation and feature extraction. Because of the shortness of the clips, many pairs do not have substantially different relative gaze. Despite this, the model is able to build a disentangled representation for eye patch reconstruction. Future work could explore how to more efficiently sample for more consistently diverse pairs.

Occasionally it is not possible to have all valid views – for example, if a left eye patch is present but the right eye patch is absent due to an occlusion or missed detection. In these cases, we simply input Gaussian noise in its place to the training pipeline and treat it as any other view. However, we zero out any loss that is calculated using the invalid view. All images are converted to grayscale and are histogram-equalized, as in [27], to reduce the effect of lighting variation.

3.2. Feature Encoding and Decoding

Given an eye patch image I, we train an encoder neural network to extract four feature vectors, as seen in Figure 2:

1. Head pose feature f_h – containing information to discern head orientation relative to the camera.
2. Eye appearance feature f_a – information about eye identity within a particular eye patch (eye shape).
3. Relative gaze feature f_g – to capture gaze angle relative to the head pose.
4. Common feature f_c – general appearance information about a subject that is common across all times, cameras, and spatial regions (glasses, lighting, skin color) within a sample.

These features are then concatenated and passed through a decoder network to reconstruct the original grayscale image. While the final reconstruction is based on $\mathcal{L}_1$ distance, we explore two different losses which differ in how they alter the features before passing them through to the decoder.

3.3. Pairwise Loss

For our baseline loss, we extend the method proposed in the cross-encoder [27] for a multi-camera system. While the original cross-encoder employs eye appearance and (absolute) gaze features, we break the latter to cover head pose and relative gaze information separately, and add common features for information common across all views in an input sample. We form three sets of pairs from the eight input views shown in Figure 1. Each set of pairs compares four views to another four views where all but one feature type is held constant – either head orientation, eye appearance, or relative gaze. Figure 2 depicts the case where the input images are taken from different cameras, but at the same time instance and from the same side eye. Because of this selection of pairs, only the head features should be different between images while all others should be similar. To enforce this similarity, we swap all features except for the head features. We then reconstruct the images using the decoder and calculate $\mathcal{L}_1$ loss. We calculate this pairwise loss for each of the three dimensions and then sum them to get the overall loss.

Given eight input views, this requires eight total passes through the encoder network and 24 passes through the decoder (8 inputs $\times$ 3 pairwise losses). In practice, we find

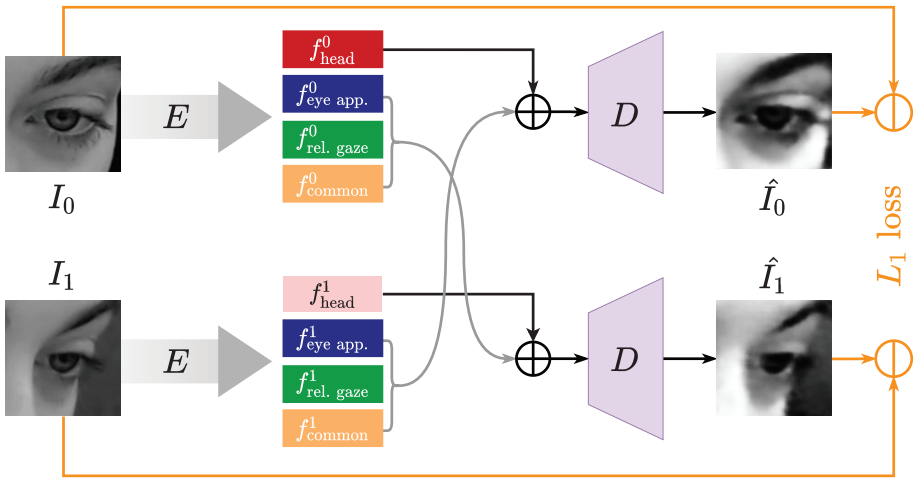


Figure 2. **Pairwise loss** requires exchanging all but one feature between image pairs. In this example, the images I_0 and I_1 are sampled from different cameras but at the same time and from the right eye, so are known to share eye appearance, relative gaze and common features. The head feature, however, is different, so it kept constant while all other features are exchanged. The input images are passed through an encoder network, E, to compute the feature representations, while input images are reconstructed using a decoder and the $\mathcal{L}_1$ loss is calculated versus the original. When using pairwise loss, we employ two other sets of input pairs – keeping all inputs except the relative eye gaze consistent and keeping all inputs except the eye appearance consistent.

that we can achieve similar performance and faster training by only passing one view from the pair through the decoder and calculating reconstruction loss. This requires 12 passes through the decoder, or 1.5 times the input number of views.

3.4. Basis Features and Loss

For our other loss, we observe that across the eight sampled input views there is much redundant information. For example, while there are four different views of a left eye, there should only be one set of features that encodes the appearance of a left eye. The same is true for the two timestamps used to elicit two different relative gazes and the two camera views used to show two different head orientations. We call these features that should be independent of other input view factors the *basis features*.

To extract the basis features, we first encode all eight input views into the feature space using an encoder, as shown in Figure 3. We use a *summary function* (S) to combine together features that share a similar source – for example, the appearance features all from the left eye. In our initial experiments, we explore two possible options for S: (i) taking the mean, and (ii) using a confidence weighted mean, similar to the multiview confidence mechanism proposed in [5]. The confidence weighted mean should allow for the pipeline to better correct for missing or obstructed views and allow for better interpretability without annotation.

The summary function produces two different basis features for each dimension – B^0_h and B^1_h for each head orientation due to camera view, B^0_a and B^1_a for each eye appearance, and B^0_g and B^1_g for each relative gaze time instance. There is only one common basis feature B_c extracted for the whole sample, as the features are defined as those traits consistent across all views. We improve the generalizability of our system to varying numbers of available views by randomly dropping out between zero and $M-1$ views from consideration for each feature, where M is the total number of available views.

Once extracted, combinations of these basis features should be able to recreate any of the input images by passing the combinations through the decoder. For example, to reconstruct a left eye taken from camera A at time 1, we concatenate the left eye appearance feature (B^0_a), the camera B head pose feature (B^0_h), the time 1 relative gaze feature (B^1_g), and the common basis feature (B_c). In this way, training can be re-framed as learning to arrange the feature space to form better basis features. Finally, we calculate the reconstruction loss on each of the eight predicted images and the originals. This means the basis loss is more memory-efficient per training batch, versus the pairwise loss, which requires 50% more passes through the decoder.

3.5. Implementation

We use a ResNet18 [8] for the encoder, following the cross-encoder work [27]. However, we append a two-layer MLP to the encoder output for each feature type. We use a feature size of 12 for the head and relative gaze features and 64 for the appearance and common features. Unlike Sun et al., which uses four DenseNet [9] deconvolution blocks, we use six transposed convolution layers as our decoder to allow for faster training. In common with their work, the nov-

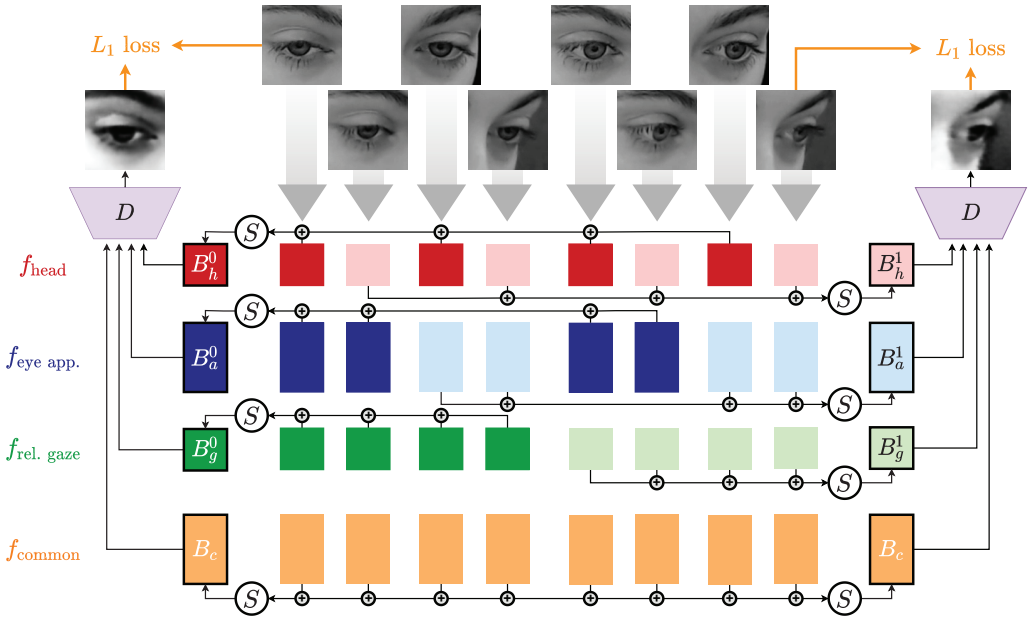


Figure 3. **Basis features** shown with $\mathcal{L}_1$ loss calculated for two views. The eight input views are first passed through the encoder, depicted by the gray arrows, to extract the four feature types – head, eye appearance, relative gaze, and common appearance. Due to the sampling strategy shown in Figure 1, each view should be similar to three other views, when considering a certain feature type (or all other views, in the case of common features). To enforce this similarity, we combine all view features from the same source using the summary function S. This results in two different basis features for each dimension, except for common features – which are consistent over the whole sample and produce only one basis feature. We then concatenate all combinations of these features back together, pass them through the decoder, and calculate $\mathcal{L}_1$ reconstruction loss with the associated input view.

elty of our method is not the network itself, but the sampling strategy and feature mixing mechanism. As such, other networks could instead be used as the encoder or decoder.

When training, we compute a held-out validation loss every 100 batches and save the model with the lowest validation loss. This best model is then used to extract features for train, validation, and test sets. When using basis features, the same summary function can be used to share information available at test time between different views. The basis feature dropout can also be used during gaze estimation to potentially improve robustness, but we default to using one camera, one time instance, and both eye patches as the available views at final gaze estimation at test time.

For final gaze estimation, we train a four-layer MLP with hidden layer sizes of 128, 64, and 32 to predict pitch and yaw. We then convert this to a vector representation and calculate angular error – both as our training loss and evaluation metric. Unless otherwise specified, we only use the head and relative gaze features for gaze estimation. We use a subset of N training samples to train the MLP and 64 validation samples to determine the best model. We then calculate our final test performance metrics on this best model.

We implement our pipeline using PyTorch [19]. We use a batch size of 96 for pairwise loss experiments and 128 for basis loss experiments when using eight views, due to the better memory-efficiency of the basis method. We scale the batch size to account for varying numbers of views in other experiments. Due to the small model and feature size during gaze estimation, we are able to fit all training samples in one batch. We optimize our models using Adam [12] with a learning rate of 10^{-4} for gaze estimation. For feature learning, we use a base learning rate of 10^{-4}, which is multiplied by the batch size. We run each experiment on a Tesla V100 SXM2 with 16GB of memory. We perform feature training for 10 epochs and gaze estimation for $2,000$ epochs. Feature training takes approximately 2 hours using the pairwise losses and 1 hour using the basis loss, when head features are used for each.

To account for variability, we train four different feature representations for each experiment using a different random seed. We then perform eight different 100-shot experiments with different data subsets for each of these feature representations. We compute the mean over these 32 runs for the final experiment performance.

When used, confidence is learned for each feature type as a two-layer MLP on top of the ResNet output. The hidden layer is set to the same dimensionality as the feature type and the output is a single neuron. A softmax function is applied to all confidence predictions from views with the same attribute (e.g. the appearance features for the four left

eye views). These confidences are then used to weight the contribution of each view to the final basis feature. In practice, we found that it was best to first multiply the input to the softmax by a scaling factor of 10^3. This makes the output confidences more sparse, which has the effect of selecting the view with the maximum estimated confidence in a differentiable way.

4. Experiments

4.1. Dataset

We adopt the EVE dataset for our experiments, as it contains video recordings synchronized over multiple views [16]. Instead of only containing samples with explicitly targeted gaze, EVE contains more naturalistic stimuli, including images, videos, and Wikipedia entries. The dataset contains a total of 54 participants, although only a subset of 44 are provided with ground truth gaze labels. Because of this, we define our own subject folds for our experiments. We use train01-train30 as the train set, train31-train39 as the validation set, and val01-val05 as the test set. Our train, validation, and test sets contain 49,404, 15,712, and 7,676 video clips, respectively.

Each EVE stimuli lasts 3 seconds and is recorded simultaneously on four cameras. We resample the videos to 10 Hz, resulting in 30 total frames per clip. Each clip includes a face frame and two eye patches, pre-extracted using facial landmarks. Each video is sampled once per epoch, selecting a random frame, eye patch, and camera as the anchor view. Other complementary views are then sampled from the same clip, as described in Sec. 3.1.

4.2. Baselines

We consider four different systems in our analysis – (1) baseline cross-encoder, using only gaze and appearance features and a single camera view with pairwise loss; (2) cross-encoder with added head features; (3) basis loss with mean summary function; (4) basis loss with confidence-weighted mean summary function. We also examine the impact of adding the common appearance features to the pipeline. The latter three systems use the head and gaze features for gaze estimation, while the baseline uses only gaze features. Note that all systems have access to the same data from all cameras – they only differ in how the data is grouped. An oracle mean baseline system performance is also shown, where the mean test gaze is calculated and used for all predictions. All quantitative results are given in mean angular error (degrees).

4.3. Quantitative Results

4.3.1 Ablation Experiment

Table 1 shows the results of gaze estimation with 100 training samples after feature learning using each system. Overall, common appearance features benefit the basis loss and hurt the pairwise loss methods. This is likely because basis features are actually calculated using all sample inputs, so the common features are more representative. Furthermore, in the case of the baseline cross-encoder, the common features may be capturing head rotation, resulting in a worse gaze feature for final estimation. If common features were also used during gaze estimation, a performance of 8.8 could be achieved for the baseline. We discuss this issue more in Sec. 4.3.2. For this reason, all further analyses use common features for basis loss methods and exclude them from pairwise loss methods. Overall, the basis loss system with confidence performs the best, with a mean angular error of 7.3 degrees. We therefore use confidence for the remainder of our basis loss analyses.

Figure 4 shows the effect of gaze estimation with varying amounts of data in the training set. The basis features outperform other methods consistently in all cases – show-

	Without Common	With Common
Mean Baseline	22.7	22.7
Cross Encoder (CE)	**9.6 (0.5)**	12.3 (1.0)
CE with Head Feature	**7.6 (0.3)**	7.8 (0.3)
Basis Loss (mean)	7.9 (0.5)	**7.5 (0.4)**
Basis Loss (confidence)	7.6 (0.5)	**7.3 (0.4)**

Table 1. **EVE results with 100-shot gaze estimation.** We show the performance with and without the use of common features, as they benefit the basis loss and hurt pairwise loss performance. Basis loss with confidence and common features performs the best of all systems.

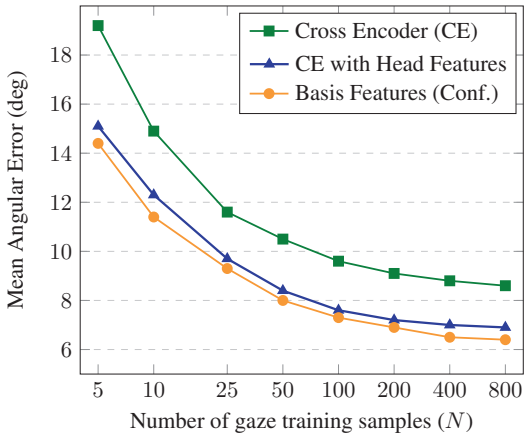


Figure 4. **Gaze estimation performance under N-shot training.** The addition of head features and multi-view self-supervision to cross-encoder training uniformly improves few-shot gaze estimation performance by a significant margin (20% on average), while using confidence-weighted basis features improves performance still further (24% on average).

Method	Features	Rel. Gaze	Head Rot.	Cam. Gaze
Mean Baseline	N/A	10.9	26.0	22.7
Cross Encoder	App	11.5	10.3	12.5
	Gaze	9.8	13.2	9.6
	All	**9.5**	**10.1**	**9.0**
Cross Encoder with Head Feat.	App	11.4	21.7	21.2
	Gaze	9.3	24.8	21.3
	Head	11.2	9.2	11.1
	Gaze+Head	9.3	**8.9**	**7.6**
	All	**9.0**	9.2	8.0
Basis Loss with Confidence	App	12.1	25.5	23.3
	Gaze	9.4	25.5	21.4
	Head	11.3	8.7	11.0
	Subject	12.3	26.1	23.0
	Gaze+Head	9.0	**8.4**	**7.3**
	All	**8.6**	8.6	7.6

Table 2. **Disentanglement analysis on EVE.** We show the ability of each feature learning method to disentangle features from their intended task. Bolded numbers show the best performance within each method and underlined numbers are the best overall.

ing the benefit of using them irrespective of the training set size. The remaining experiments focus on the N=100 case.

4.3.2 Feature Disentanglement Analysis

We next investigate how well each method disentangles its learned features for their intended task. Besides estimating absolute gaze (in the camera coordinate frame), we also show the system's performance on relative gaze estimation and head rotation estimation. Table 2 shows the result of each feature combination on each task. Overall, basis loss best disentangles the features and provides the top performance for each task. While individual features often have good performance on their own, the combination of gaze+head gives the best performance. Notably, the baseline cross-encoder method has better performance when appearance features are used alongside gaze features. This is likely because without head features being explicitly disentangled, there is no guarantee that they will be present in the gaze feature.

4.3.3 Flexibility to More Views

One benefit of the basis feature loss is that it allows for a flexible number of views at train and test time. The following experiments examine the impact of varying the amount of views during training and testing for each of the three feature types. Besides the varied dimension, we still use two cameras, two gaze instances, and two eye patches during feature learning and one camera, one gaze instance, and two eye patches during gaze estimation.

More Cameras: We first examine the impact of using more cameras per sample when training and testing. Table 3 shows the performance of each combination. While training on more than one camera allows for the system to learn head orientation, more than two cameras per sample hurts performance. This may be because more cameras does not necessarily add additional information, but "blurs" the basis features with more views. All four camera views are still present in the dataset and two views are sufficient to learn head rotation. However, more cameras at test time does yield improved performance, likely due to an improved relative gaze feature.

	Number of Test Cameras per Sample			
Train	1	2	3	4
1	9.6 (0.6)	10.1 (1.1)	10.5 (1.5)	10.6 (1.6)
2	7.3 (0.4)	7.0 (0.3)	6.7 (0.3)	**6.6 (0.2)**
3	7.5 (0.2)	7.1 (0.2)	6.8 (0.2)	6.7 (0.2)
4	7.6 (0.3)	7.2 (0.2)	6.7 (0.2)	6.9 (0.2)

Table 3. **Training and testing on different numbers of cameras used when forming each sample.** Adding more than two cameras into the sample views hurts performance, while more cameras at test time reduces error.

	Number of Test Gaze Instances per Sample			
Train	1	2	3	4
1	8.9 (0.6)	8.6 (0.7)	8.5 (0.5)	8.6 (0.7)
2	7.3 (0.4)	**7.0 (0.3)**	**7.0 (0.3)**	**7.0 (0.3)**
3	7.4 (0.3)	**7.0 (0.3)**	7.1 (0.3)	7.1 (0.3)
4	7.4 (0.3)	7.1 (0.3)	7.1 (0.3)	7.2 (0.3)

Table 4. **Training and testing on different numbers of gaze instances used when forming each sample.** Adding more than two gaze instances to training hurts performance, but increasing the number of gaze instances at test time gives better performance.

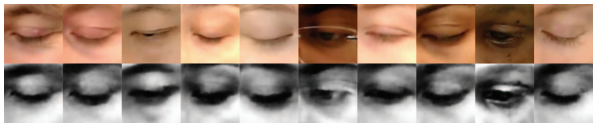
	Patches Used for Test		
Train Patches	L	L+R	L+R+F
L (1 view)	9.4 (0.7)	-	-
L+R (2 view)	7.8 (0.6)	7.3 (0.4)	-
L+R+F (2 view)	7.8 (0.5)	7.3 (0.3)	**7.2 (0.3)**
L+R+F (3 view)	8.2 (0.6)	7.7 (0.4)	7.4 (0.4)

Table 5. **Adding face patches (F) to the left (L) and right (R) eye patches.** Adding face patches to training and testing improves performance. However, the best performance is achieved when training on only two patch views per sample.

More Gaze Instances: Table 4 shows the performance of the basis loss pipeline with varying amounts of gaze instances during train and test. Similar to the previous result, more than two views at train time is harmful, but more views for test improves results. The improvement during test is likely due to better estimates of head pose, based on increased numbers of instances with similar head pose.

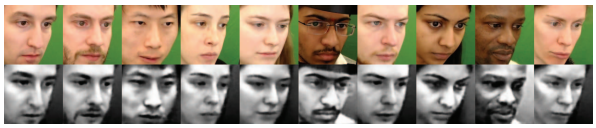
Adding Face Patch: We lastly examine the impact of

(a) High confidence appearance features.



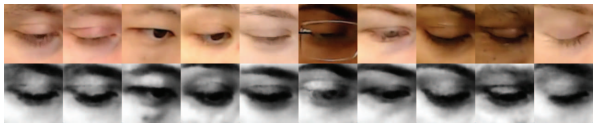
(b) Low confidence appearance features.

(c) High confidence relative gaze features.



(d) Low confidence relative gaze features.

(e) High confidence head features.



(f) Low confidence head features.

Figure 5. **Confidence analysis.** We show the most and least confident inputs and their reconstructions for each subject.

adding face patch inputs to the model in addition to eye patches. However, we only consider the left and right eye patches during final gaze estimation test metrics, to keep it comparable to the other systems. Table 5 shows the performance for increasing types of patches available during training and testing. The face patches provide a slight boost to performance to 7.2 degrees, likely due to the information they provide about head rotation. Similar to the other experiments, it is better to sample two random patches per view instead of all three when training.

4.4. Confidence Analysis

We finally examine the interpretability of the confidences estimated within the basis loss pipeline. We use the features learned during the previous face patch experiment so that we can better examine the impact of different types of information provided to the system. Figure 5 shows the views from each validation subject with the most and least confidence for appearance, relative gaze, and head features.

Appearance confidence (top row) seems to be mostly related to eye openness. Because the appearance feature is used to reconstruct the same eye in other timestamps, a closed eye would potentially need to reconstruct an open eye. Due to this missing information, the model would likely favor selecting views with open eyes.

Relative gaze confidence (middle row) is high for clearly zoomed eye patches and low for face patches. This is likely because face patches have less resolution for iris location and ambiguity about which gaze to encode between the two eyes. Notably, eye openness does not seem to impact confidence, or possibly even improves it. Because the relative gaze feature is used to reconstruct eyes in the same time instant, it is likely that other patches would also be closed except for a relatively rare wink. Therefore, gaze features must also encode openness and do not need to encode direction when closed, making them more confident. Notably the one validation subject wearing glasses had the lowest mean relative gaze confidence of all subjects.

Head orientation confidence (bottom row) behaves the opposite of relative gaze confidence. Because head position can be determined more clearly from face frames, these have high confidence. Eye patches are more difficult to determine head direction and have low confidence.

5. Discussion

We present a new approach for multi-view gaze representation learning. By explicitly disentangling the head orientation in the feature space and incorporating multi-view training, we are able to improve the mean absolute error of a baseline cross-encoder [27] 20% in few-shot learning, and by 24% through the use of our basis loss pipeline. While this is a modest improvement, it comes with many other advantages. The basis pipeline is more flexible, allowing for additional camera views, gaze instances, and cropped regions to be added at test time to improve performance even further. Basis features are also more computationally efficient – taking only half the time to train a comparable model. The pipeline with confidence can also improve interpretability, even in the absence of labels.

Although we acknowledge that 3D gaze technology can potentially be misused in harmful ways for society, we believe there are many useful applications of gaze estimation, particularly in the promotion of joint attention in human-computer interaction (HCI) and human behavioral understanding. We hope that our method will be useful to HCI practitioners who wish to fine-tune gaze models to their particular domains using simple-to-collect, synchronized, uncalibrated, multi-view face video.

References

[1] Shumeet Baluja and Dean Pomerleau. Non-intrusive gaze tracking using artificial neural networks. *NeurIPS*, 1993. 1

[2] Zhuoqing Chang, J Matias Di Martino, Qiang Qiu, Steven Espinosa, and Guillermo Sapiro. SalGaze: Personalizing gaze estimation using visual saliency. In *ICCV Workshops*, 2019. 2

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 3

[4] Haoping Deng and Wangjiang Zhu. Monocular free-head 3d gaze tracking with deep learning and geometry constraints. In *ICCV*, 2017. 2, 3

[5] John Gideon, Simon Stent, and Luke Fletcher. A multi-camera deep neural network for detecting elevated alertness in drivers. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2931–2935. IEEE, 2018. 4

[6] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in NeurIPS*, 2020. 3

[7] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *TPAMI*, 2009. 1

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[9] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 4

[10] Umar Iqbal, Pavlo Molchanov, and Jan Kautz. Weakly-supervised 3d human pose learning via multi-view images in the wild. In *CVPR*, 2020. 3

[11] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *ICCV*, 2019. 1, 2

[12] Diederik P. Kingma and Jimmy Lei Ba. Adam : A method for stochastic optimization. In *arXiv*, 2014. 5

[13] Rakshit Kothari, Shalini De Mello, Umar Iqbal, Wonmin Byeon, Seonwook Park, and Jan Kautz. Weakly-supervised physically unconstrained gaze estimation. In *CVPR*, 2021. 1, 2

[14] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *CVPR*, 2016. 1, 2

[15] Erik Lindén, Jonas Sjostrand, and Alexandre Proutiere. Learning to personalize in appearance-based gaze tracking. In *ICCV Workshops*, 2019. 2

[16] Seonwook Park, Emre Aksan, Xucong Zhang, and Otmar Hilliges. Towards end-to-end video-based eye-tracking. In *ECCV*, 2020. 1, 2, 6

[17] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. Few-shot adaptive gaze estimation. In *ICCV*, 2019. 2

[18] Seonwook Park, Adrian Spurr, and Otmar Hilliges. Deep pictorial gaze estimation. In *ECCV*, 2018. 2

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5

[20] Adria Recasens, Aditya Khosla, Carl Vondrick, and Antonio Torralba. Where are they looking? In *NeurIPS*, 2015. 2

[21] Helge Rhodin, Mathieu Salzmann, and Pascal Fua. Unsupervised geometry-aware representation for 3d human pose estimation. In *ECCV*, 2018. 3

[22] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017. 1, 2

[23] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017. 3

[24] B.A. Smith, Q. Yin, S.K. Feiner, and S.K. Nayar. Gaze Locking: Passive Eye Contact Detection for Human?Object Interaction. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2013. 1, 2

[25] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. Appearance-based gaze estimation using visual saliency. *TPAMI*, 2013. 1

[26] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. Learning-by-synthesis for appearance-based 3d gaze estimation. In *CVPR*, 2014. 1, 2

[27] Yunjia Sun, Jiabei Zeng, Shiguang Shan, and Xilin Chen. Cross-encoder for unsupervised gaze representation learning. In *ICCV*, 2021. 1, 2, 3, 4, 8

[28] Erroll Wood, Tadas Baltrusaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson, and Andreas Bulling. Rendering of eyes for eye-shape registration and gaze estimation. In *ICCV*, 2015. 1, 2

[29] Yu Yu and Jean-Marc Odobez. Unsupervised representation learning for gaze estimation. In *CVPR*, 2020. 1, 2

[30] Xucong Zhang, Seonwook Park, Thabo Beeler, Derek Bradley, Siyu Tang, and Otmar Hilliges. ETH-XGaze: A Large Scale Dataset for Gaze Estimation under Extreme Head Pose and Gaze Variation. In *ECCV*, 2020. 1, 2

[31] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Appearance-based gaze estimation in the wild. In *CVPR*, 2015. 1, 2

[32] Yufeng Zheng, Seonwook Park, Xucong Zhang, Shalini De Mello, and Otmar Hilliges. Self-learning transformations for improving gaze and head redirection. *NeurIPS*, 2020. 2