Dynamic Patch Focus for Transformer-based Gaze Estimation

Master Thesis

Dan Sochirca

Delft University of Technology

TU Delft

Dynamic Patch Focus for Transformer-based Gaze Estimation

by

Dan Sochirca

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday, July 08 2025 at 10:00 AM.

Student Number: 5295580
Project Duration: Nov 2024 – July 2025
Thesis Committee: Dr. Nergis Tömen, TU Delft, thesis advisor
Dr. Xucong Zhang, TU Delft, daily supervisor
Dr. Huijuan Wang, TU Delft, committee chair

Cover image by Voxels Republic, via Unsplash (unsplash.com).

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

Looking back, my time as a student at TU Delft feels like it passed in the blink of an eye. From challenging projects and study marathons to even the most stressful exams, I truly enjoyed the whole journey. It was a period of profound personal and academic growth, and I'm grateful to have been part of such an environment.

The journey to this thesis was marked by lots of late-night research marathons, frequent setbacks, and a lot of "experimenting around". The project originally began with a different, though related, idea: integrating deformable convolution in Vision Transformer models. Despite my best efforts, this initial direction didn't produce the results I had hoped for. It was a discouraging moment, but through the steady support and expert guidance of my supervisor, Dr. Xucong Zhang, I was able to overcome this challenge and eventually develop a new approach that not only worked but exceeded expectations. His mentorship was indispensable, helping me deepen my understanding of the field and build the confidence to pursue an original solution. I owe a great deal of this project's success to him.

I would like to extend my heartfelt gratitude to the many people who supported me throughout this journey. First and foremost, to my daily supervisor, Dr. Xucong Zhang, for your constant support, insightful feedback, and continuous encouragement. To my committee chair, Dr. Huijuan Wang, thank you for graciously taking on the role of committee chair and for your valuable time. To my thesis advisor, Dr. Nergis Tömen, thank you for agreeing to be my advisor and for your contributions to this project. A special thank you to Dr. J.C. van Gemert for the technical guidelines and writing tips that improved the quality of this report.

Finally, I could not have done this without the love and support of my family, my girlfriend, and my friends. Thank you for believing in me, your support was my foundation.

Dan Sochirca
Delft, June 2025

Contents

Preface **i**

1 Introduction **1**

2 Scientific Paper **2**

3 Supplementary material **17**
3.1 Deep Learning and Computer Vision 17
3.1.1 Neural Networks 17
3.1.2 Convolutional Neural Networks (CNNs) 19
3.1.3 Vision Transformers 22
3.2 Gaze Estimation 24
3.2.1 Task Introduction 24
3.2.2 Appearance-Based Gaze Estimation 24
3.2.3 Datasets used 25

References **26**

1

Introduction

Accurately determining where a person is looking is fundamental to modern human-computer interaction, with profound implications across various domains. From enabling more intuitive user interfaces in virtual reality and ensuring driver safety in advanced automotive systems to providing critical insights in healthcare and educational settings, 3D gaze estimation serves as a powerful, non-verbal channel for understanding human attention, intention, and cognitive state. As the demand for seamless and intelligent systems grows, so does the need for reliable and efficient methods to interpret this fundamental behavioral cue directly from visual input.

Recent advancements in computer vision have been driven by the emergence of Vision Transformers (ViTs) [3], which have demonstrated remarkable capabilities in capturing global context and long-range dependencies within images. However, applying standard ViT architectures directly to appearance-based gaze estimation presents a unique set of challenges. The task requires balancing two opposing needs: preserving fine-grained details in the ocular region while understanding the broader facial context, including head pose and orientation. Traditional ViTs often struggle on both fronts, as their rigid, fixed-grid patching mechanism can (i) split key features like the iris and pupil across multiple patches, weakening their representational power, and (ii) overload the attention mechanism with redundant information from less relevant areas like the forehead, cheeks, and background.

To address these inherent limitations, this thesis introduces **FocusViT**, a novel, lightweight, and end-to-end differentiable framework designed to enhance Vision Transformers for the specific demands of gaze estimation. Our approach improves a standard ViT by incorporating two complementary modules that operate ahead of the main encoder. First, a **Patch Translation Module**, based on Spatial Transformer Networks [8], dynamically repositions image patches to center important features, thus reducing feature fragmentation. Second, a **differentiable Top-K Selection Module** learns to identify and discard the least informative tokens, allowing the model to focus its computational resources exclusively on the most relevant facial regions.

Our experiments demonstrate that this two-part strategy yields significant improvements in both accuracy and efficiency. The complete FocusViT framework reduces the mean angular error of a strong ViT baseline from 4.98° to 4.61° on the challenging ETH-XGaze dataset, while processing 75% fewer tokens. More importantly, we uncover a key insight: leveraging token reduction to enable a finer-grained, lossless 8x8 patch grid directly addresses a critical information compression bottleneck in smaller ViT models, leading to a competitive performance of 4.42° MAE. These findings demonstrate that dynamic, task-specific patch selection can be a powerful and efficient alternative to more complex hybrid architectures.

This report is structured into three primary sections. This introduction serves to contextualize the problem and summarize our core contributions. The core of the work is presented in the scientific paper "Dynamic Patch Focus for Transformer-based Gaze Estimation". Following the paper, a supplementary chapter provides the necessary background knowledge on Vision Transformers, attention mechanisms, and other foundational concepts essential for a comprehensive understanding of our methodology.

2

Scientific Paper

Dynamic Patch Focus for Transformer-based Gaze Estimation

Dan Sochirca
Delft University of Technology
Delft, The Netherlands

Abstract

Appearance-based 3D gaze estimation must accommodate two conflicting needs: fine ocular detail and global facial context. Vanilla Vision Transformers (ViTs) struggle with both needs due to their fixed 16 × 16 patch grid that (i) fragments critical features like the eyes into multiple patches, and (ii) floods the self-attention mechanism with redundant information from the forehead, cheeks and background. We introduce FocusViT, a lightweight and end-to-end differentiable framework that enhances ViTs by first using a Patch Translation Module, based on Spatial-Transformer Networks, to dynamically translate patches to center on content, and then employing a Perturbed Top-K operator to select only the most informative tokens for processing.

Our experiments show that combining translation and selection reduces the mean angular error (MAE) of a ViT-S baseline on ETH-XGaze from 4.98° to 4.61° while using 75% fewer tokens. Furthermore, by leveraging this token reduction to enable a finer-grained, lossless 8x8 patch grid, we address a key information bottleneck in the ViT-S architecture, achieving a final MAE of 4.42°. The framework also demonstrates consistent improvements on the MPIIFaceGaze dataset, reducing the baseline error from 5.72° to 5.36°. Extensive ablation studies confirm our central finding: patch translation and token selection are complementary mechanisms that work in synergy to improve model performance.

1. Introduction

Gaze is a uniquely informative behavioural cue: it reveals attention, intent, situational awareness, and cognitive load. Estimating where a person is looking (3D gaze) has broad applications in fields such as human-computer interaction [8], healthcare [4], automotive safety [23], education [14, 16], and virtual reality [24], where it enables improved user experiences, safety, and the understanding of human behaviors. Due to low device requirements, appearance-based 3D gaze estimation, which infers gaze directly from images without requiring personal calibration, has seen significant advances with deep learning. In these settings, the camera sees the entire face, usually under unconstrained head pose and lighting, so the estimator model has to reason about both fine ocular details and global face context.

Early convolutional neural network (CNN) models either focused on cropped eye regions or used the entire face as input. Each approach has its trade-offs: using only the eyes captures fine details but ignores global context (e.g. head pose) that might be crucial in low-light or extreme pose settings, whereas using the full face provides context but dilutes the critical eye features. To address this, researchers explored hybrid strategies. For example, multi-region networks process both face and eye patches in parallel to combine local and global cues [18], learnable spatial weights can emphasize eye regions within a full-face CNN [37], and region selection networks dynamically choose the most informative face sub-regions for each image [40]. These works show that *where* the model looks in the face is as important as *how* it learns gaze from the features.

In recent years, Vision Transformers (ViTs) [13] have been applied to gaze estimation to leverage their ability to capture long-range dependencies and global context. ViTs split an image into a sequence of patch tokens and apply a self-attention mechanism to model relationships across the entire image. Cheng *et al*. introduced a hybrid transformer model - GazeTR [9], and demonstrated improved accuracy over CNN-only models by taking CNN-extracted facial features and globally integrating them using transformer encoders. Beyond single-view settings, transformers have also been used to fuse multi-view information, for example, using cross-view attention [10]. Another emerging direction is to rethink the gaze network architecture: Wang et al. proposed GazeCaps [27], which uses capsules to represent different facial properties and a self-attention routing to dynamically focus on the most relevant capsules. These works emphasize the potential of transformer-based architectures and adaptive attention mechanisms for gaze estimation.

However, applying ViTs to gaze estimation has its challenges. We identify **two key limitations** when using vanilla full-face ViTs: (1) patch fragmentation of critical eye con-

tent, and (2) redundant background and face tokens. The two limitations are as follows:

1. **Patch fragmentation of critical eye content.**
   The fixed grid patch partitioning of a ViT can split semantically important regions, predominantly the eyes, across multiple patches. If an eye falls on a patch boundary, its information will be divided into separate tokens, and this can weaken the feature representation and eventually degrade gaze prediction accuracy. CNNs, on the other hand, do not have this issue due to their overlapping receptive fields. The fragmentation also has more weight in smaller ViT models. Take, for example, the small 'ViT-S' variant with 21M parameters: this model projects each patch of size 3x16x16 to a token embedding of dimension 384. Clearly, some information has to be compressed, because $3 \times 16 \times 16 = 768$ is twice as much information as the model retains in its token embeddings. In bigger model variants such as 'ViT-B' or 'ViT-L', this issue is less present because their feature dimension is at least 768. However, the attention mechanism still has to model split-eye relationships, which is more work compared to simply integrating information from different face regions.
2. **Redundant background and face tokens.**
   Second, a full-face image yields many tokens from regions like cheeks, forehead, and background that carry little useful information for the gaze [5, 37]. These redundant tokens not only bring unnecessary computational cost but can also distract the model's attention. Indeed, recent efficient transformer studies on image classification tasks show that a substantial subset of tokens can be pruned with negligible impact on final performance [2, 19, 22]. While pruning has not been studied on gaze estimation, the eye and periocular region (including eyelids and eyebrows) have consistently been identified as the primary source of gaze information, while other facial regions have much more subtle or environmentally-subjective impact [37].
   Additionally, in the context of vision transformers, using a smaller patch size is known to improve accuracy by capturing fine-grained details [13, 26]. However, the quadratic cost of self-attention makes this resolution impractical for the whole face. By dropping irrelevant tokens first, we can afford smaller patches where they matter most, without exploding computation.

To address these shortcomings, we introduce **FocusViT**, a framework that extends a standard ViT with two lightweight, trainable modules. To combat patch fragmentation, we employ a Patch Translation Module based on a Spatial Transformer Network (STN) [15] that applies a content-aware translation to each patch, allowing them to dynamically recenter on important ocular features. To mitigate token redundancy, we use a differentiable Perturbed Top-K operator [11] that learns to select only the most informative patches for processing, pruning the rest. Our entire pipeline is trainable end-to-end and its components are optional, allowing it to be adapted for specific use cases. Compared to other token reduction techniques and Region Proposal Networks, our method is simpler and more interpretable: the model exposes exactly which face regions it keeps and why, rather than relying on implicit token mixing or token interpolation.

Our experiments demonstrate the effectiveness of this approach. On the ETH-XGaze dataset, combining patch translation with Top-K selection reduces the mean angular error (MAE) of a ViT-S baseline from 4.98° to 4.61° - while using only 25% of the original tokens. By leveraging token reduction to enable a finer-grained 8×8 patch size, our selection-only model achieves an even lower error of 4.42°. Our models also show consistent gains on the MPIIFaceGaze dataset, with the best variants reducing the baseline error from 5.72° down to 5.36°.

In summary, our contributions are as follows:

1. We introduce FocusViT, a lightweight end-to-end trainable framework that enhances vanilla/pure Vision Transformers for gaze estimation by unifying patch translation and selection. Our method includes a novel Offset-aware Positional Embedding (OPE), meant to preserve spatial coherence after patch translation.
2. Through extensive evaluation on the ETH-XGaze and MPIIFaceGaze datasets, we show a key finding: for smaller ViT architectures, using token reduction to enable a finer, lossless 8×8 patch grid can be more impactful than dynamic patch translation, as it directly addresses the information compression bottleneck at the patch-embedding layer.
3. We provide detailed ablation studies showing that patch translation and selection are complementary modules. The translation module first creates higher-quality candidate patches by centering them on relevant content, which in turn allows the selection module to prune distractive tokens more aggressively and effectively.

2. Related Work

2.1. Gaze Estimation

Before state-of-the-art models shifted to incorporating self-attention, convolutional backbones dominated appearance-based 3D gaze estimation. iTracker [18] was the first large-scale mobile gaze network, which used a four-stream CNN (face + both eyes + face grid) trained on the 2.5M frames crowdsourced GazeCapture dataset. [37] employed a Spatial-Weights CNN and showed that using the entire face with learned spatial masks instead of eye-only crops reduced the angular error to 4.8° on EYEDIAP. Later, Dilated-Net [6] preserved high spatial resolution with di-

lated convolutions and improved accuracy on MPIIGaze, while Gaze360 [17] introduced a panoramic in-the-wild dataset together with a ResNet-GRU temporal model that predicts gaze even when the eyes are off-screen. These CNNs remain strong baselines due to their efficiency and ability to capture local features, but the local receptive fields limit their ability to model long-range dependencies and global context, especially in scenarios involving extreme head poses or dual-camera setups. Transformers and self-attention mechanisms help address these shortcomings by modelling global face context and capturing multi-view or multi-region correspondences.

The transformer-based breakthroughs started with GazeTR [9], which explored both a pure Vision Transformer and a ResNet-ViT hybrid and reported that the hybrid variant already outperformed strong CNN baselines while using fewer parameters. Subsequent works injected domain knowledge into the Transformer pipeline. DV-Gaze [10] introduces Dual-View Interactive Convolution blocks plus a dual-view Transformer, cutting error by up to 30% under extreme head pose on ETH-XGaze. GazeSymCAT [41] (cross-attention between left/right eyes and face) and GazeCaps [27] (self-attention-routed capsules) explicitly model inter-eye relations, improving robustness to occlusion and extreme yaw angles. Efficiency-oriented variants like BoT2L-Net [29] insert Bottleneck-Transformer layers into a shallow ResNet and train the network with twin yaw/pitch losses, lowering mean angular error on Gaze360 without increasing model size. At the opposite end of the spectrum, TransGaze [34] shows that a plain pre-trained ViT can match hybrid models once eye-region tokens are explicitly emphasized, while reducing the training time by half, compared with deeper CNNs.

2.2. Patch and Token Reduction in Vision Transformers

The computational complexity of Vision Transformers scales quadratically with the number of tokens. To overcome this, an existing area of work tries to improve efficiency by reducing the number of image patches/tokens processed by the model. In ViTs, this can be done *pre-transformer*, or *within the transformer*. While we found no token reduction work done on gaze estimation, plenty of research has been done on tasks such as image classification and detection.

Pre-transformer selection. Work focusing on shrinking the input sequence externally to the ViT is still scarce. Differentiable Patch Selection [11] learns a perturbed-Top-K mask that crops only the most informative patches and discards the rest, yielding 3-4× FLOP savings while training end-to-end thanks to their differentiable Top-K operator. Our method adopts this operator but uses a smaller patch size and a vanilla ViT as the backbone. Other works include STTS [28], whose spatial–temporal scorer ranks tokens across both space and time and keeps only a perturbed-Top-k subset per clip, trimming about 50% of video tokens with negligible loss; AgentViT [3], which trains a reinforcement-learning agent to decide on-the-fly which patches to embed; and gViT [20], which applies Gumbel-Softmax sampling on patches in echocardiography videos. However, pure pre-transformer approaches remain rare. Most other works embed the full grid first and prune later.

Intra-layer selection. *Vision transformer: To discover the "four secrets" of image patches* [42] differentiates 3 key token selection mechanisms: token selection based on a scoring function, based on token merging, and based on convolution and pooling. Some prominent examples of scoring-based models are DynamicViT [22], which uses lightweight prediction modules that identify and prune less informative tokens hierarchically across multiple stages; AdaViT [19], with a generalized approach that can skip tokens, heads and blocks in the model with a three-layer decision network; Evo-ViT [33], which retains all tokens but uses two computational paths with different computational costs, where unimportant tokens get cheap updates in the model, while important tokens get full updates. Instead of dropping tokens, ToMe [2], a token-merging approach, greedily merges the most similar pairs with 2–3× speed-ups at 0.3% accuracy loss. Pooling approaches like HVT [21] or PSViT [25], insert pooling or 1-D convolutions between blocks to down-sample tokens in a hierarchical manner.

2.3. Patch Translation and Deformable Sampling

A complementary line of work lets the network move or resize patches instead of (or in addition to) dropping them. Spatial Transformer Networks (STNs) [15] apply a predicted affine transformation to input feature maps, allowing a network to focus on important regions by translating, scaling, or rotating the patch grid. This approach to differentiable attention to location paved the way for later deformable ViT modules. Deformable Patch-based Transformer (DPT) [7] predicts per-patch offsets + scales based on input content and generates new embeddings using bilinear interpolation from the vanilla patch embeddings. Their method improved ImageNet accuracy and COCO detection with only a slight increase in FLOPs. [32] presents DAT, a Vision Transformer with Deformable Attention at every layer, that uses learnable offset groups across all queries, which shift the keys/values to attend to important regions. In detection, Deformable DETR [43] uses a deformable attention module that acts as sparse attention with learned offsets, and leads to faster convergence and reduced complexity from quadratic to linear in the number of patches. A more relevant approach, perhaps, which combines token selection with spatial adaptation similarly to our work, is DeBiFormer [1]. They introduce Deformable Bi-level

Routing Attention (DBRA), which first finds top-k regions per query (bi-level routing) and then deforms the attended positions within those regions. The combined approach led to more semantically relevant regions being selected, which made the attention more efficient and meaningful.

In the context of 3D gaze estimation, Zhang *et al.* [40] introduce a two-stage architecture in which a Region-Selection Network (RSN) first proposes a single, content-dependent crop inside the face image and a subsequent gaze network regresses the gaze vector from that crop. The RSN is trained without any bounding-box supervision through a novel unsupervised loss that encourages the selected region to be informative for the downstream gaze loss, allowing the whole pipeline to be optimised end-to-end. Their method dynamically focuses on visible or well-lit eye regions, outperforms fixed-patch baselines, and proves especially robust under directional illumination, extreme head pose, and partial self-occlusion. Conceptually this paper is close to our work in that it selects face sub-regions; however, it experiments with just up to 3 selected rectangular crops of 68×68 resolution per image, whereas we keep a set of the most informative 16×16 patches and additionally apply per-patch translations so that each retained token can be centered or move closer to the ocular region.

3. Methodology

Our model predicts gaze by coupling a Patch Translation Module and a Top-K Selection Module, prior to a Vision-Transformer backbone for the gaze estimation task. The model architecture is presented in Figure 1. In this section, we introduce and discuss each component in detail.

3.1. Vision Transformers

A Vision Transformer [13] treats an image as a 1D token sequence: the RGB image is first partitioned into N non-overlapping patches, and each is flattened and linearly projected to a d-dimensional vector. Next, a learnable class token is prepended, positional information is added, and the resulting sequence is processed by L transformer encoder layers with h heads each. Inside the encoders, multi-head self-attention models the pairwise relations among all tokens. For the final prediction, the class token is sent to the MLP head, which, for our task, outputs yaw–pitch angles, which are then mapped to 3D unit vectors.

3.2. Patch Translation

Our Patch Translation Module is a Spatial Transformer Network (STN) [15], a learnable component designed to achieve spatial invariance against any spatial transformation. It does so by adaptively transforming its input to a specific pose of interest, with input-dependent parameters: translation, scaling, and rotation, which are part of a 2D affine transformation matrix:

$$A = \begin{pmatrix} s_x & r_x & t_x \\ r_y & s_y & t_y \end{pmatrix}, \tag{1}$$

where s, r, and t stand for scale, rotation, and translation, respectively, on horizontal and vertical axes x and y.

The STN consists of three components: the localization network, the grid generator and the sampler. Their workflow is as follows: for an input image I, the localization network predicts the parameters of matrix A; afterwards, the grid generator uses the predicted parameters to create a mapping from the transformed output image coordinates to the original input image; and finally, the sampler uses the grid generator pixel coordinates and applies bilinear interpolation to extract the output pixel values - which together form the spatially-transformed image.

In our implementation, we restrict the STN to only perform spatial translation. This was done because our datasets are already normalized (images are rotated and cropped), using the approach in [38], and the small residual head tilt can be modelled by the ViT itself. The localization network therefore, is a dedicated **Offset Predictor** which outputs a 2D translation offset (Δx,Δy). It's architecture uses the first two layers from an ImageNet pretrained ResNet-18 model (i.e., all blocks up to, and including *Layer2*). Leveraging pretrained weights has improved accuracy in our early experiments.

To predict an offset for each patch, we derived the following process. While we could simply provide each patch's content of size 3x16x16 to the Offset Predictor network, in practice, the 16x16 patch size is very small, and the network is unlikely to accurately infer the context. Therefore, we provide a larger field of view: each 16x16 patch in the input gets extended to double its size, i.e., a 32x32 patch, with the same center. This is simply achieved by unfolding the image with a kernel size of 32 and a stride of 16. This way, the STN can choose a more accurate transformation by seeing 8 extra pixels at each patch border. Due to this design decision, we have to set the scale parameter of the matrix A to 0.5, so that the STN network crops the 32x32 patch in half to the original patch size. The rotation parameters are also set to 0 to disable rotation.

However, there is one issue that remains: applying the patch translation offsets will change the patch content, and the positional encodings of the Vision Transformer are no longer accurate. To overcome this, we developed a type of conditional positional encoding that we call **Offset-aware Positional Embedding (OPE)**. It works simply by encoding the learned offset to the ViT token dimension, and is added to the standard positional vector so that the ViT is aware of the translation:

$$ope_i = pe_i + \text{MLP}(\Delta x, \Delta y), \tag{2}$$

where pe_i is the i-th standard positional vector of the ViT

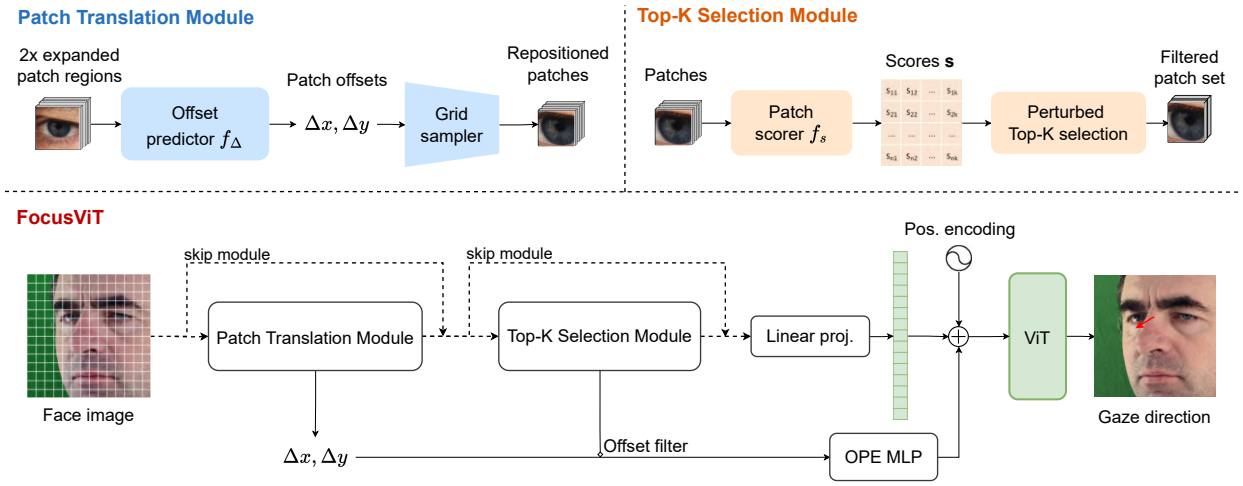

Figure 1. Overview of the proposed FocusViT pipeline. The architecture features 2 independent and optional modules: Patch Translation and Top-K Selection. The Patch Translation Module enables dynamic 2D patch offsetting ($\Delta x, \Delta y$), while the Top-K Selection Module allows for adaptive patch filtering. The aligned, selected tokens then enter an unmodified ViT encoder.

model. In our experiments, we try both a one-layer MLP as well as a two-layer variant.

3.3. Differentiable Top-K patch selection

The patch-selection task has multiple definitions, but in this paper, we use the ranking-problem formulation: a lightweight ConvNet assigns a relevance score s_i to every patch, and only the K highest-scoring patches are forwarded to the Vision Transformer. While it is a simple concept, this *hard Top-K* operation $arg\_topk(s, K)$ is non-differentiable - since the non-selected patches are discarded, they always get zero gradient, so standard back-propagation cannot tell the network how to adjust scores. We therefore adopt the **Perturbed Top-K** operator of Cordonnier et al. [11], which is designed to be differentiable based on perturbed optimizers.

The Perturbed Top-K operator works by changing the raw patch relevance scores $s = (s_1, \ldots, s_N)$ with a Monte-Carlo expectation over the hard-Top-K operation after the scores have been noised:

$$\begin{aligned} f_\sigma(\mathbf{s}) &= \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\big[\mathrm{TopK}\big(\mathbf{s}+\sigma\boldsymbol{\epsilon},\, K\big)\big] \\ &\approx \frac{1}{T}\sum_{t=1}^{T}\mathrm{TopK}\big(\mathbf{s}+\sigma\boldsymbol{\epsilon}^{(t)},\, K\big), \end{aligned} \tag{3}$$

where $\epsilon^{(t)}$ is i.i.d. Gaussian noise, σ denotes the temperature, and $TopK(\cdot, K)$ returns a 0/1 indicator vector where the K largest components have values $1/K$ so that the mask sums to 1. In other words, one-hot Top-K scores are computed after adding noise to the score vector; this operation is repeated T times, and the average is taken. This operation is differentiable because the non-smooth hard-Top-K sits inside the expectation.

In our implementation, we set the parameters similarly to the values in the [11] as $T = 500$ and $\sigma = 0.05$. In our experiments that also use the Patch Translation Module, we set T to 200 due to GPU memory constraints. During training, we follow the same temperature schedule and linearly decay σ to 0; and during inference, we set σ to 0, which is equivalent to a deterministic hard Top-K selection. Our Patch Scorer network is also based on the authors' implementation and uses two pretrained ResNet layers as our Offset Predictor network, discussed previously. The only difference with the Offset Predictor is the prediction head: the scorer uses a convolution layer followed by max-pooling, whereas the predictor uses global-pooling followed by an MLP.

4. Experiments

This section empirically validates our proposed FocusViT framework through a series of ablation studies on the ETH-XGaze dataset. We first analyze the Patch Translation and Top-K Selection modules in isolation to measure their individual contributions. We then investigate the performance gains from using a finer patch resolution, and finally, we evaluate the complete model to demonstrate the complementary benefits of combining both components. Following this analysis, we present the results of our final models on the MPIIFaceGaze dataset and compare them against

related work. We conclude with a standalone experiment investigating an alternative, implicit deformable sampling approach using Metaformer [35] architectures like Poolformer, as described in Section 4.8.

To visualize the model's regions of interest, we chose to display the selected patch boundaries directly on the input images. We used a fixed color-coding rather than a diverging palette like in a heatmap. A direct score-based heatmap proved uninformative due to the low variance among the top-K scores, while a ranking-based heatmap was misleading because ranks were often determined by arbitrary tie-breaking between patches with numerically close scores.

4.1. Setup

Datasets. We train and evaluate our models on the ETH-XGaze [39] and MPIIFaceGaze [37] datasets. **ETH-XGaze** is a high-resolution 1.1 million-image dataset of 110 subjects, with extreme head pose and 16 illumination conditions. However, since the test partition labels are not publicly released and require a time-consuming online submission, we do a 50/30 random split of the train partition of 80 subjects, and define our train and test sets respectively as such. **MPIIFaceGaze** contains 214k images of 15 subjects, collected in the wild with mostly frontal head poses. For this dataset we use the common cross-subject 15-fold evaluation procedure. Both datasets are normalized to a canonical camera space using the method in [38], which removes camera distance and head rotation variance and consistently improves learning. For MPIIFaceGaze, we initialise from ETH-XGaze weights, because this cross-dataset pre-training setup has been shown to improve model performance [9]. For training on ETH-XGaze, we initialize from ImageNet weights.

Model Variants. To concisely represent our model configurations, we adopt the following naming scheme: `FocusViT-`<*backbone*>`/`<*patch-size*>`/`<*type*>`-r`<*selection-ratio*>. In this scheme, <*backbone*> specifies the ViT architecture size (e.g., 'S' for Small). <*patch-size*> indicates the input patch resolution (e.g., 16 for 16x16 or 8 for 8x8). The <*type*> is 'D' for dynamic models that include our Patch Translation Module, and 'S' for static models that do not perform patch translation. Finally, <*selection-ratio*> denotes the fraction of tokens kept, such as $r0.5$ for 51% and $r0.25$ for 25%. For instance, FocusViT-S/16/D-r0.25 refers to the model built on a ViT-Small backbone that uses a 16x16 patch size, includes the dynamic Patch Translation Module, and selects the top 25% of patches. To maintain readability in the following sections, we will describe model configurations explicitly and reserve the compact notation for the final summary table.

Training settings. All models are implemented by extending the timm-library [30] implementation of Vision Transformers, and trained on a single NVIDIA A40 (48 GB) for 40 epochs. We use the AdamW optimiser and a cosine one-cycle schedule, with the base learning rate of 0.0005. We set the batch size to 200 - if it does not fit the model on the GPU, we set it to the highest number that does.

Evaluation metric. Performance is reported as the mean angular error (MAE) in degrees between predicted and ground-truth gaze vectors, used by most works on gaze estimation.

Baseline. Our reference is the ImageNet-pretrained ViT-S model `vit_small_patch16_224` [31] (21M parameters, 16x16 patch size, N=196, L=12, h=6, d=384). The same underlying architecture is used for our models with patch translation and Top-K selection. The finetuned baseline delivers a mean angular error of 4.98° on ETH-XGaze, and 5.72° on MPIIFaceGaze.

4.2. Contribution of Patch Translation

We begin our analysis by isolating the effect of the Patch Translation Module. In this experiment, the module is added to our ViT-S baseline, but all 196 patch tokens (for a 16x16 grid) are retained. We perform a cumulative ablation study on the ETH-XGaze dataset to evaluate different architectural choices within this module. Table 1 summarizes the results.

The key investigated components are:

- **Default translation**: A simple translation module where the offset predictor head is a single linear layer.
- **Offset-aware Positional Embedding (OPE)**: Adds the offset-dependent embedding described in Eq. 2 to the standard positional encoding.
- **Tanh offset activation**: Inserts a tanh activation to the offset predictor head to bound the predicted offsets to the range [-1, +1] - this is motivated by the STN's grid generator, which defines a regular grid in [-1, +1]×[-1, +1].
- **Hidden layer in predictor's head**: Expands the offset predictor with a hidden layer.
- **Hidden layer in OPE**: adds a 128-unit hidden layer inside the OPE's MLP.
- **Pretrained offset predictor**: Initializes the predictor with weights pretrained on ETH-XGaze.

The results in Table 1 show a clear progression. Our Default Model immediately improves performance, reducing the MAE from the baseline's 4.98° to 4.82°. However, by integrating the rest of the components, we achieve small gains that further bring the MAE down to 4.71°. This represents a 0.27° improvement over the ViT-S baseline, confirming the benefit of allowing patches to shift. From the ablation study we further see that adding OPE slightly hurts performance. While the OPE module is not beneficial here, we will show in Section 4.6 that it becomes essential when combining translation with patch selection to preserve spa-

tial coherence.

Table 1. Cumulative ablation study of the Patch Translation Module on ETH-XGaze. The baseline ViT-S achieved 4.98° MAE, and the default translation module reduces this to 4.82°.

Configuration	MAE (ETH-XGaze)
Default Patch Translation	4.82°
+ OPE	4.84°
+ Tanh Offset Activation	4.82°
+ Offset Predictor Hidden Layer	4.78°
+ Pretrained Offset Predictor	**4.71°**
+ OPE Hidden Layer	4.72°

Figure 2 shows a visualization of the best-performing model's behavior (with 4.71° MAE). The figure reveals a consistent pattern: patches located on the forehead and eyebrows shift downwards, while patches near the eye's boundary slide to better center the pupil, or the eyelids if not visible. In contrast, there is minimal movement in less informative areas, such as the cheeks and chin, suggesting the module learns to focus on relevant facial features. The visualization also shows that the patch movements, while effective, are relatively modest. This limited motion is likely the reason why the OPE module is not beneficial in this translation-only setup.

In summary, patch translation "cleans up" the input grid by pushing patches towards more valuable regions. We show later that this effect is even more pronounced when combined with patch selection, which we explore in the following sections.

4.3. Analysis of Fixed Subsampling with Translation

Before introducing a Patch Scorer, we ask: *What if we simply skip patch windows at a larger stride, while allowing them to shift their locations?* This can also be considered a form of patch selection, without requiring a scorer network. The idea is similar to the concept introduced by [36], however, they apply the shifting in multiple steps and employ feature extraction at each stage.

For this experiment, we replace the standard 14x14 grid with two sparser, fixed configurations: a 10x10 grid that keeps 100 tokens (51% of the original) and a 7x7 grid that keeps 49 tokens (25%). The layout of these grids is illustrated in Figure 3.

The results, presented in Table 2, show that both configurations underperform the full-token model (4.71°) and the baseline (4.98°). However, we observe an interesting effect: the model with fewer tokens (25% selection) achieves a better MAE of 5.33° compared to the 5.44° error of the model with more tokens (51% selection). A qualitative analysis of the selected patches in Figure 4 (rows 1-2) explains this result. The fixed stride of the subsampled grids inevitably selects patches from uninformative regions like the background, forehead, or cheeks. The sparser 7x7 grid appears to be more effective simply because it is forced to discard a larger number of these distracting tokens.

Table 2. Performance of models using patch translation on fixed, subsampled grids.

Grid stride	Selected %	MAE (ETH-XGaze)
24 px	51%	5.44°
32 px	25%	5.33°

This outcome strongly suggests that which patches are kept is as important as the number kept. This provides a clear motivation for using a learned scorer network, which can select the most informative patches in a content-aware manner, as we will explore next.

4.4. Contribution of Learned Patch Selection

We next evaluate score-based patch selection on a static grid, without patch translation. In this setup, a lightweight scorer network assigns a relevance score to each of the 196 16x16 patches, and only the Top-K are passed to the transformer backbone. We test configurations that keep the top 51% (100 patches) and 25% (49 patches) of tokens. We also tried a variant where the Patch Scorer sees a larger 32x32 context window of each patch.

The results, presented in Table 3, show that this approach does not outperform the baseline. The best configuration, which retains 51% of the patches, achieves a MAE of 5.22°, a 0.24° degradation compared to the baseline. This performance decrease is likely because patch fragmentation remains an issue - critical features like the iris can be split across multiple patches. Furthermore, by discarding tokens, the model loses some of the global context it might rely on to compensate for the fragmentation. This helps explain why retaining 51% of patches yields better results than the more aggressive 25% selection.

Table 3. Performance of learned Top-K selection on a static 16x16 grid (without translation).

Selection %	MAE (ETH-XGaze)
51%	**5.22°**
25%	5.69°
51% (32x32 scorer input)	6.09°
25% (32x32 scorer input)	5.75°

A closer inspection of the selected patches, shown in Figure 4 (rows 3-4), highlights the learned policy of our Patch

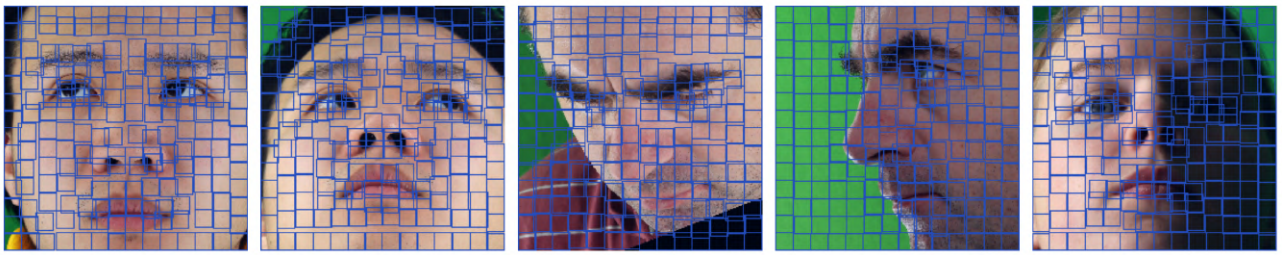

Figure 2. Visualization of the Patch Translation Module on ETH-XGaze. These examples are from our best-performing translation-only model in Table 1 (4.71° MAE). The blue squares show the final patch positions after being dynamically shifted toward more informative regions, such as the eyes and eyebrows, across various head poses.

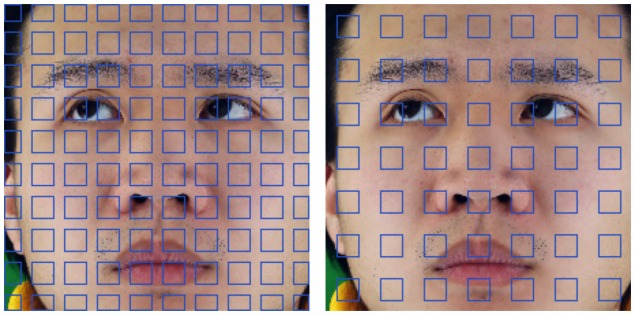

Figure 3. Illustration of the fixed 10x10 (left) and 7x7 (right) subsampling grids. While the initial placement of patches is static, each selected patch is still dynamically translated to focus on relevant content. This deterministic sampling method inevitably includes irrelevant regions like cheeks, forehead or background.

Scorer. The network consistently focuses on the periocular region (eyes, eyelids, eyebrows), while correctly ignoring large, uninformative areas like the cheeks and forehead. However, the scorer network sometimes assigns importance to non-facial artifacts like the subject's hair or background, especially when facial information is sparse.

Furthermore, we observe a bias in the scorer's behavior under uneven lighting. For example, when a face is half-lit, the model tends to select more patches from the darker side. We hypothesize that this is not a preference for darkness itself, but for the high-frequency information and sharp contrast found at the shadow boundaries. These features are rich cues for 3D facial geometry, which the simple convolutional Patch Scorer may have learned to associate with informative regions.

This general limitation of a small 16x16 patch size motivated us to experiment with providing the scorer network with a larger 32x32 context window of each patch. However, this approach significantly worsened performance. This is likely due to a misalignment between the Patch Scorer and the ViT: while the 32x32 window provides more context, the ViT still processes the central 16x16 patch. If critical information lies in the outer border of this context window, the scorer may be misled into assigning a high score even though the central patch is less informative.

4.5. Impact of Finer Patch Resolution

Top-K selection reduces the sequence length, therefore, we can afford a smaller patch size on our GPU. We explore this advantage by replacing the standard 16x16 patches with a finer 8x8 grid. Because individual 8×8 windows lack context, we reuse the trained 16×16 Patch Scorer from the previous section. The frozen 16x16 scorer assigns scores to "parent" windows, and if a parent is selected, its four corresponding 8x8 "child" tokens are passed to the ViT backbone. This process is visualized in Figure 5.

This change leads to a significant performance gain, as shown in Table 4. The model using 51% of the 8x8 tokens achieves a MAE of 4.42°, a substantial 0.56° improvement over our ViT-S baseline. This is the best result achieved in our experiments. Even when aggressively pruning 75% of the tokens (the 25% selection model), the model still surpasses the baseline with an MAE of 4.87°.

Table 4. Performance of selection-only models using a finer-grained, 8x8 patch size. Selection is guided by a frozen Patch Scorer trained on 16x16 patches. 4.4.

Model	MAE (ETH-XGaze)
25% selection	4.87°
51% selection	**4.42°**

The dramatic improvement can be attributed to a key technical detail of the ViT architecture. A single 16x16 patch contains 768 pixel values (3×16×16), which must be compressed to fit into the model's 384-dimensional embedding, causing information loss. This creates a significant information compression bottleneck, causing a loss of fine-grained detail before the data even reaches the transformer layers. In contrast, an 8x8 patch contains only 192 values (3×8×8), which fits entirely within the embedding dimen-

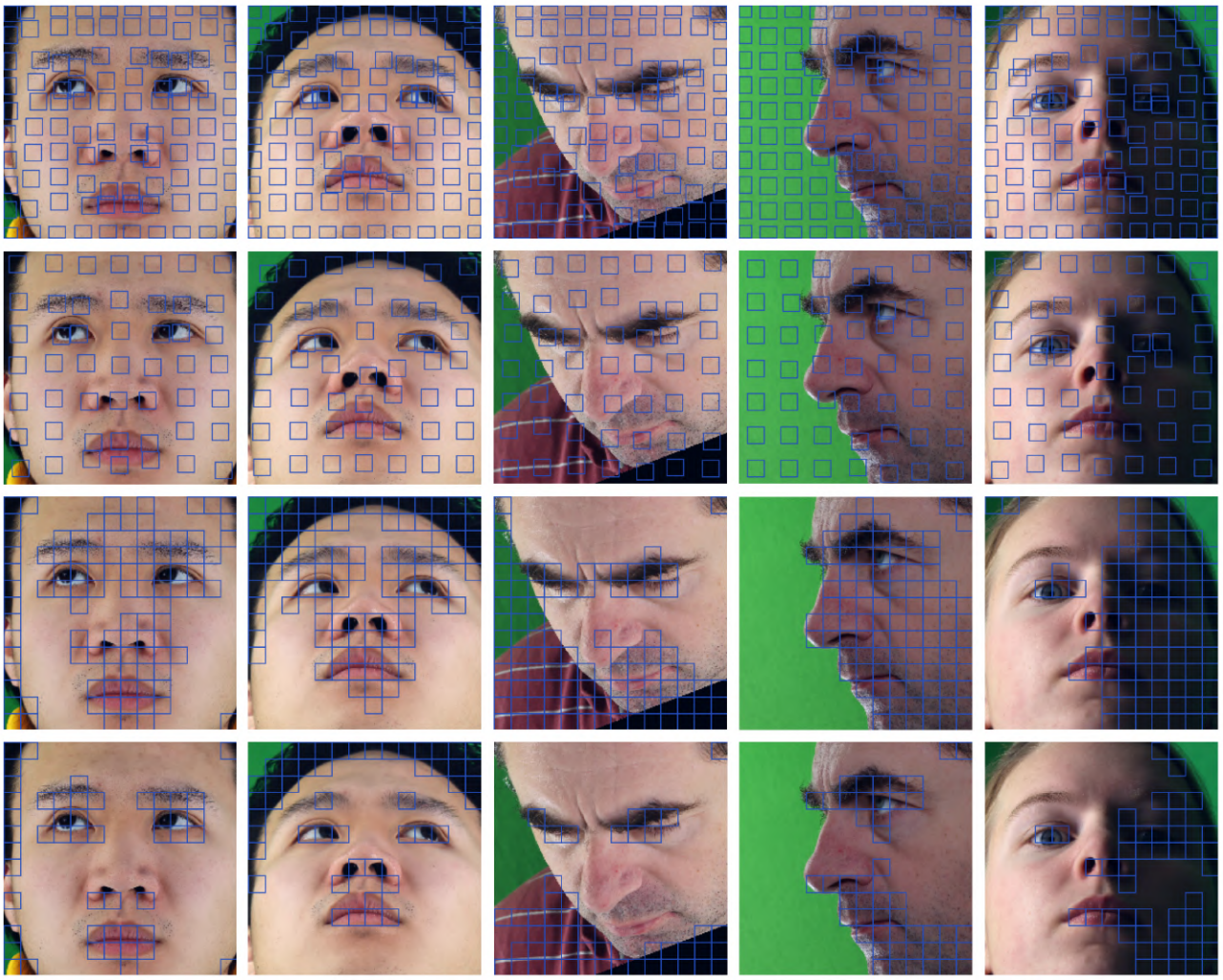

Figure 4. Patch selection by dynamic-grid subsampling (rows 1-2) vs. learned Top-K (rows 3-4). Either 51% or 25% of patches are sampled. The learned Top-K selector focuses on the eyes, nose, mouth, and head contour while consistently avoiding the cheeks and forehead. Some artefacts, such as hair strands or the person's shirt, are sometimes sampled when facial detail is weak. The dynamic-grid subsampling method accurately captures important eye features by shifting the patches, but more irrelevant tokens happen to be selected due to the grid configuration.

sion without compression. By using smaller patches, the model can access a much richer, lossless representation of fine ocular details. The finding that 51% selection still outperforms 25% selection is consistent with our previous experiment, further emphasizing that a larger amount of tokens is beneficial on a static grid.

4.6. Effect of Joint Translation and Selection

We now combine our best-performing Patch Translation Module with the Top-K Selection Module to create the full FocusViT model. This final experiment is designed to show how dynamic patch alignment and learned selection are complementary.

First, we demonstrate the critical importance of Offset-aware Positional Embedding (OPE). As shown in Table 5, activating the Patch Scorer on top of the translated patches without OPE degrades performance, resulting in an error of 6.02°. Simply enabling the OPE module to inform the model about the spatial shifts causes the error to drop to 4.78°. Adding a hidden layer to the OPE's MLP further refines the performance to 4.61° - a 0.37° improvement over the baseline. This confirms that explicitly encoding the patch offsets is essential for the transformer to maintain spatial coherence when patches are both moved and pruned.

The combined FocusViT model also reveals a major reversal in behavior. In the selection-only experiments (Sec-

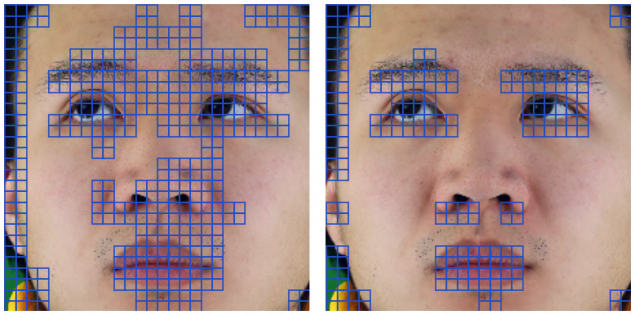

Figure 5. Visualization of 8x8 patch selection for the 51% model (left, 400 tokens) and 25% model (right, 196 tokens). Each selected 16x16 parent window is split into four 8x8 child tokens, which form the final input sequence.

Table 5. Performance of the full model combining patch translation and selection. The table ablates the selection ratio and the effect of Offset-aware Positional Embedding (OPE) for the 25% configuration.

Configuration	MAE (ETH-XGaze)
25% selection without OPE	6.02°
25% sel. + OPE	4.78°
25% sel. + OPE+hidden-layer head	**4.61**°
51% sel. + OPE+hidden-layer head	5.22°

tions 4.4 and 4.5), retaining more tokens (51%) was always better. Here, the opposite is true: the model with more aggressive 25% selection (4.61° MAE) significantly outperforms the 51% version (5.22° MAE). This improvement comes from the effective collaboration between patch translation and token selection. The translation module first "cleans up" the input by centering patches on informative content like the eyes. As a result, the Patch Scorer is presented with higher-quality candidates and can afford to be more selective, discarding a larger number of tokens without losing critical information, thereby reducing distractions.

A visual analysis in Figure 6 supports this conclusion. Compared to selection on a static grid, the final set of chosen patches is more tightly focused around the periocular region. Patches that could have been selected on the eyebrows are now shifted down to cover the eyes before being selected. This reinforces our core claim: translation and selection are complementary. The translation module creates better candidates, and this allows the selection module to filter more aggressively and effectively.

4.7. Final Results and Discussion

To conclude our analysis, we now summarize the results for our most relevant FocusViT configurations on both the ETH-XGaze and MPIIFaceGaze datasets and compare them against our baseline and related work. The complete results are presented in Table 6.

Our ablation studies on ETH-XGaze revealed clear improvements over the ViT-S baseline (4.98° MAE). Among configurations, combining patch translation and selection significantly improved performance, achieving a 4.61° MAE. However, the most impactful change came from using a finer 8×8 patch size. Specifically, our selection-only model, FocusViT-S/8/S-r0.5, attained the best result on ETH-XGaze with a 4.42° MAE. This highlights that for the ViT-S architecture, addressing the information compression bottleneck at the patch-embedding layer made the biggest difference. Furthermore, our most parameter-efficient model, FocusViT-T/8/S-r0.5, achieved a competitive 4.50° MAE with only 5.60M parameters, outperforming the full translation-and-selection ViT-S model.

On the **MPIIFaceGaze** dataset, which has less variation in head pose but features in-the-wild lighting conditions, our models also showed consistent gains over the 5.72° baseline. The best variants - FocusViT-T/8/S-r0.5 (5.36°) and FocusViT-S/16/D-r1.0 (5.37°) - reduced the error significantly. A qualitative analysis of samples from MPIIFaceGaze, shown in Figure 7, confirms that our models behave consistently across datasets. Both the patch translation and selection modules continue to perform well even under challenging low-light conditions. We again note the scorer network's tendency to select patches at high-contrast shadow boundaries, suggesting that it may use these as cues for 3D facial geometry.

Comparison with GazeTR. The GazeTR [9] model provides an excellent point of comparison, as it addresses the same issue of patch fragmentation but through a fundamentally different approach. Instead of using a simple linear projection for image patches, GazeTR starts with a ResNet18 backbone to extract rich local feature maps, which are then turned into tokens and processed by a transformer. Table 6 shows a clear performance difference between our models and GazeTR across datasets. On ETH-XGaze, our best model outperforms it (4.42° vs. 4.56°), and our small FocusViT-T model nearly matches its performance (4.50° vs. 4.56°) with about half the parameters (5.60M vs. 11.42M). This suggests that our modules are particularly effective at handling the geometric challenges associated with extreme head poses. In contrast, on MPIIFaceGaze, GazeTR's deep CNN backbone is likely more robust to complex, "in-the-wild" lighting conditions, allowing it to achieve a better 4.96° MAE compared to our 5.36°. Overall, this shows that FocusViT is a competitive and parameter-efficient alternative, especially robust in sce-

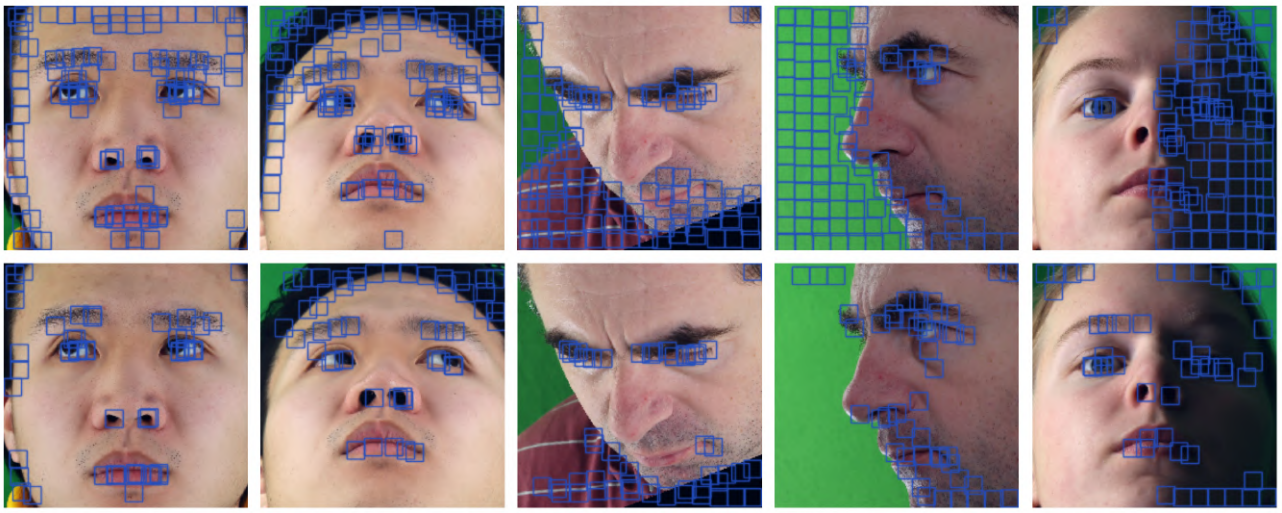

Figure 6. Visualization of the full FocusViT model (25% selection). Patches are first dynamically translated and then scored, leading to a more refined and concentrated selection on the eyes compared to models without translation.

Model	Patch Size	Patch Trans.	Selection %	Feat. Dim.	# Params	ETH-X	MPIIFace
ViT-S (baseline)	16	-	-	384	21.67M	4.98	5.72
gazeTR [9]	-	-	-	32	11.42M	4.56	4.96
FocusViT-S/16/D-r1.0	16	Y	-	384	22.48M	4.72	**5.37**
FocusViT-S/16/D-r0.5	16	Y	51%	384	23.13M	5.22	6.25
FocusViT-S/16/D-r0.25	16	Y	25%	384	23.11M	**4.61**	5.84
FocusViT-S/16/S-r0.5	16	N	51%	384	21.86M	5.22	6.22
FocusViT-S/16/S-r0.25	16	N	25%	384	21.84M	5.69	6.42
FocusViT-S/8/S-r0.5	8	N	51%	384	21.82M	**4.42**	5.92
FocusViT-S/8/S-r0.25	8	N	25%	384	21.75M	4.87	6.01
FocusViT-T/8/S-r0.5	8	N	51%	192	5.60M	4.50	**5.36**
FocusViT-T/8/S-r0.25	8	N	25%	192	5.56M	5.07	6.02

Table 6. Gaze estimation performance on MPIIFaceGaze (MPIIFace) and ETH-XGaze (ETH-X).

narios with significant geometric variation.

4.8. Alternative Approach: Implicit Deformable Sampling

To contrast with FocusViT's *explicit* input sampling, we ran an isolated experiment on *implicit* deformable sampling. Instead of shifting entire input patches, this approach allows the network to adjust its receptive fields internally based on local image content - a concept found in works on deformable attention and convolution. For this, we integrated **Deformable Convolution (DCN)** [12], an operator that learns adaptive spatial sampling offsets, into two MetaFormer-based models [35]. The first model was Poolformer-S, which uses a simple pooling operator as its token mixer. The second was a modified 12M-parameter CAFormer-S variant employing standard 2D convolutions in its first three stages and self-attention in the final stage. The CAFormer modifications involved increasing the total number of blocks to 26 and decreasing the maximum embedding dimension to 192. In both models, we replaced the standard token mixers with deformable variants: Deformable Pooling and Deformable Convolution - while keeping the architecture and parameter count nearly identical to ensure a fair comparison. Deformable Pooling is a custom operator we developed that follows the same mechanism as the DCN operator, but replaces convolution with average pooling.

Despite pretraining all models on ImageNet using the same setup as in the Metaformer paper, the results in Table 7 show no significant gains for the deformable variants. These results suggest a key limitation: while DCN is effective for object-level classification and detection tasks where they

Figure 7. Qualitative results of FocusViT models under various illumination conditions on MPIIFaceGaze. Rows 1–3 show visualizations of the translation-only model (FocusViT-S/16/D-r1.0) and selection-only models retaining 51% and 25% of patches (FocusViT-S/16/S-r0.5 and S-r0.25). Even in challenging low-light conditions, the models consistently attend to the periocular region.

model large-scale geometric transformations, they appear less effective for fine-grained regression tasks such as gaze estimation. This supports our conclusion that for such tasks, explicit, targeted spatial sampling, as used in FocusViT, is a more effective approach.

5. Conclusion and future work

We presented FocusViT, a framework that enhances Vision Transformers for gaze estimation by unifying content-adaptive patch translation with differentiable Top-K selection in a single, end-to-end trainable model. Our experiments successfully validate this approach as a proof of concept, demonstrating that the modules are complementary: translation improves patch quality by centering on ocular features, which in turn allows the selection module to prune distractive tokens more effectively.

Our objective was to improve a pure ViT baseline rather than set a new benchmark. While combining the two modules significantly improved accuracy, our most impactful finding was that token reduction enables the use of a finer-grained, lossless 8x8 patch grid. This approach resolves a key information bottleneck in the ViT-S architecture, reducing the baseline error from 4.98° down to a competitive 4.42° MAE on ETH-XGaze, and from 5.72° to 5.36° on MPIIFaceGaze. This demonstrates that for certain architectures, addressing the input representation bottleneck can be more beneficial than dynamic patch alignment alone. When compared to hybrid models like GazeTR, our framework outperforms on ETH-XGaze, while our most efficient variant nearly matches its performance with about half the parameters.

For future work, we identify several promising directions:

- **Hybrid Architectures:** Integrating the FocusViT sampling modules into a hybrid CNN-Transformer model could combine the benefits of convolutionally extracted local features with our efficient and targeted attention mechanism.
- **Multi-Scale Patch Selection:** A hierarchical approach with a larger patch size could be explored, where a scorer network evaluates coarse, large-context patches (e.g., 32×32) to select informative regions, while the ViT backbone processes finer-grained patches (e.g., 8x8) from only those areas.
- **Threshold-Based Sparsification:** Replacing the fixed-K selection with an adaptive score threshold could allow

Table 7. Performance of Metaformer-based models with and without deformable sampling. The experiment contrasts standard token mixers (Pooling, Convolution) with their deformable counterparts. MAE is reported on the ETH-XGaze dataset.

Model	Token Mixer	# Params	MAE (ETH-XGaze)
Poolformer-S	Pooling	20.9M	4.40°
Poolformer-S (Deformable)	Deformable Pooling	20.7M	4.38°
CAFormer-S (modified)	Convolution + Attention	12.0M	4.15°
CAFormer-S (Deformable)	Deformable Conv. + Attention	12.5M	4.13°

the model to dynamically adjust the number of processed tokens based on scene complexity. This could potentially improve the balance between accuracy and computational cost, and make the selected set of tokens more precise.

References

[1] NguyenHuu BaoLong, Chenyu Zhang, Yuzhi Shi, Tsubasa Hirakawa, Takayoshi Yamashita, Tohgoroh Matsui, and Hironobu Fujiyoshi. Debiformer: Vision transformer with deformable agent bi-level routing attention. In *Proceedings of the Asian Conference on Computer Vision*, pages 4455–4472, 2024. 3

[2] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022. 2, 3

[3] Francesco Cauteruccio, Michele Marchetti, Davide Traini, Domenico Ursino, and Luca Virgili. Adaptive patch selection to improve vision transformers through reinforcement learning. *Applied Intelligence*, 55(7):1–26, 2025. 3

[4] Sapana Chandel, Riju Bhattacharya, Manjushree Nayak, and Astha Pathak. Integrating eye gaze estimation with the internet of medical things (iomt) for individualized and efficient healthcare. In *2024 2nd world conference on communication & computing (WCONF)*, pages 1–6. IEEE, 2024. 1

[5] Haohan Chen, Hongjia Liu, Shiyong Lan, Wenwu Wang, Yixin Qiao, Yao Li, and Guonan Deng. Dmagaze: Gaze estimation based on feature disentanglement and multi-scale attention. *arXiv preprint arXiv:2504.11160*, 2025. 2

[6] Zhaokang Chen and Bertram E Shi. Appearance-based gaze estimation using dilated-convolutions. In *Asian Conference on Computer Vision*, pages 309–324. Springer, 2018. 2

[7] Zhiyang Chen, Yousong Zhu, Chaoyang Zhao, Guosheng Hu, Wei Zeng, Jinqiao Wang, and Ming Tang. Dpt: Deformable patch-based transformer for visual recognition. In *Proceedings of the 29th ACM international conference on multimedia*, pages 2899–2907, 2021. 3

[8] Linlin Cheng, Artem V. Belopolsky, and Koen V. Hindriks. Boundary conditions for human gaze estimation on a social robot using state-of-the-art models. In *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1486–1493, 2023. 1

[9] Yihua Cheng and Feng Lu. Gaze estimation using transformer. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 3341–3347. IEEE, 2022. 1, 3, 6, 10, 11

[10] Yihua Cheng and Feng Lu. Dvgaze: Dual-view gaze estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20632–20641, 2023. 1, 3

[11] Jean-Baptiste Cordonnier, Aravindh Mahendran, Alexey Dosovitskiy, Dirk Weissenborn, Jakob Uszkoreit, and Thomas Unterthiner. Differentiable patch selection for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2351–2360, 2021. 2, 3, 5

[12] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 11

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 4

[14] Chirag S Indi, Varun Pritham, Vasundhara Acharya, and Krishna Prakasha. Detection of malpractice in e-exams by head pose and gaze estimation. *International Journal of Emerging Technologies in Learning (Online)*, 16(8):47, 2021. 1

[15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015. 2, 3, 4

[16] Pragma Kar, Samiran Chattopadhyay, and Sandip Chakraborty. Gestatten: estimation of user's attention in mobile moocs from eye gaze and gaze gesture tracking. *Proceedings of the ACM on Human-Computer Interaction*, 4(EICS):1–32, 2020. 1

[17] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6912–6921, 2019. 3

[18] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2176–2184, 2016. 1, 2

[19] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *Proceedings of the IEEE/CVF conference on computer vi-*

sion and pattern recognition, pages 12309–12318, 2022. 2, 3

[20] Alfred Nilsson and Hossein Azizpour. Regularizing and interpreting vision transformers by patch selection on echocardiography data. *Proceedings of Machine Learning Research*, 248:155–168, 2024. 3

[21] Zizheng Pan, Bohan Zhuang, Jing Liu, Haoyu He, and Jianfei Cai. Scalable vision transformers with hierarchical pooling. In *Proceedings of the IEEE/cvf international conference on computer vision*, pages 377–386, 2021. 3

[22] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021. 2, 3

[23] Pavan Kumar Sharma and Pranamesh Chakraborty. A review of driver gaze estimation and application in gaze behavior understanding. *Engineering Applications of Artificial Intelligence*, 133:108117, 2024. 1

[24] Agata Marta Soccini. Gaze estimation based on head movements in virtual reality applications using deep learning. In *2017 IEEE Virtual Reality (VR)*, pages 413–414. IEEE, 2017. 1

[25] Yehui Tang, Kai Han, Yunhe Wang, Chang Xu, Jianyuan Guo, Chao Xu, and Dacheng Tao. Patch slimming for efficient vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12165–12174, 2022. 3

[26] Feng Wang, Yaodong Yu, Guoyizhe Wei, Wei Shao, Yuyin Zhou, Alan Yuille, and Cihang Xie. Scaling laws in patchification: An image is worth 50,176 tokens and more. *arXiv preprint arXiv:2502.03738*, 2025. 2

[27] Hengfei Wang, Jun O Oh, Hyung Jin Chang, Jin Hee Na, Minwoo Tae, Zhongqun Zhang, and Sang-Il Choi. Gazecaps: Gaze estimation with self-attention-routed capsules. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2669–2677, 2023. 1, 3

[28] Junke Wang, Xitong Yang, Hengduo Li, Li Liu, Zuxuan Wu, and Yu-Gang Jiang. Efficient video transformers with spatial-temporal token selection. In *European Conference on Computer Vision*, pages 69–86. Springer, 2022. 3

[29] Xiaohan Wang, Jian Zhou, Lin Wang, Yong Yin, Yu Wang, and Zhongjun Ding. Bot2l-net: Appearance-based gaze estimation using bottleneck transformer block and two identical losses in unconstrained environments. *Electronics*, 12(7): 1704, 2023. 3

[30] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2020. Accessed: 2025-05-13. 6

[31] Ross Wightman and the timm Contributors. `vit_small_patch16_224.augreg_in1k`: Vision Transformer (Small, 16×16) pretrained on ImageNet-1k with AugReg. `https://huggingface.co/timm/vit_small_patch16_224.augreg_in1k`, 2023. Accessed: 2025-05-13. 6

[32] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4794–4803, 2022. 3

[33] Yifan Xu, Zhijie Zhang, Mengdan Zhang, Kekai Sheng, Ke Li, Weiming Dong, Liqing Zhang, Changsheng Xu, and Xing Sun. Evo-vit: Slow-fast token evolution for dynamic vision transformer. In *Proceedings of the AAAI conference on artificial intelligence*, pages 2964–2972, 2022. 3

[34] Lang Ye, Xinggang Wang, Jingfeng Yao, and Wenyu Liu. Transgaze: exploring plain vision transformers for gaze estimation. *Machine Vision and Applications*, 35(6):128, 2024. 3

[35] Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. Metaformer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):896–912, 2023. 6, 11

[36] Xiaoyu Yue, Shuyang Sun, Zhanghui Kuang, Meng Wei, Philip HS Torr, Wayne Zhang, and Dahua Lin. Vision transformer with progressive sampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 387–396, 2021. 7

[37] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. It's written all over your face: Full-face appearance-based gaze estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 51–60, 2017. 1, 2, 6

[38] Xucong Zhang, Yusuke Sugano, and Andreas Bulling. Revisiting data normalization for appearance-based gaze estimation. In *Proceedings of the 2018 ACM symposium on eye tracking research & applications*, pages 1–9, 2018. 4, 6

[39] 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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 365–381. Springer, 2020. 6

[40] Xucong Zhang, Yusuke Sugano, Andreas Bulling, and Otmar Hilliges. Learning-based region selection for end-to-end gaze estimation. In *31st British Machine Vision Conference (BMVC 2020)*, page 86. British Machine Vision Association, 2020. 1, 4

[41] Yupeng Zhong and Sang Hun Lee. Gazesymcat: A symmetric cross-attention transformer for robust gaze estimation under extreme head poses and gaze variations. *Journal of Computational Design and Engineering*, 12(3):115–129, 2025. 3

[42] Tao Zhou, Yuxia Niu, Huiling Lu, Caiyue Peng, Yujie Guo, and Huiyu Zhou. Vision transformer: To discover the "four secrets" of image patches. *Information Fusion*, 105:102248, 2024. 3

[43] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3

3

Supplementary material

This supplementary material provides the foundational knowledge required to understand the core deep learning and computer vision concepts discussed in the main thesis. It is designed to equip readers from all backgrounds with the necessary context to fully appreciate the technical contributions and experimental results.

The material begins with the fundamental principles of Deep Learning, starting with Neural Networks, the bio-inspired models that form the basis of modern artificial intelligence. It details their structure, training procedures, and the mathematical concepts of loss functions and optimizers that drive their learning process. From there, we delve into two key architectures for computer vision. First, we explore Convolutional Neural Networks (CNNs), the established backbone of image analysis. This section explains how CNNs use learnable filters and pooling layers to build a hierarchical understanding of images, and it covers influential architectures and Residual Networks (ResNets) [7]. Next, we introduce the Vision Transformer (ViT) [3], a more recent paradigm adapted from natural language processing that treats an image as a sequence of patches and uses a powerful self-attention mechanism [22] to model global relationships within the data.

After establishing these core technologies, the focus shifts to the specific problem domain of Gaze Estimation. We introduce the task of determining where a person is looking and its applications in diverse fields from human-computer interaction to automotive safety. This section clarifies the distinction between model-based and appearance-based methods [1], with an emphasis on the latter, which leverages deep learning to function with standard cameras. Finally, we briefly describe the large-scale datasets that are crucial for training and evaluating robust gaze estimation models.

3.1. Deep Learning and Computer Vision

3.1.1. Neural Networks

Deep Learning is a specialized field within Machine Learning that draws its inspiration from the biological structure and function of the human brain. The foundational components of deep learning are Neural Networks, computational models composed of interconnected nodes known as artificial neurons. These networks have proven to be exceptionally effective at learning complex patterns and representations directly from data, making them instrumental in advancing fields like computer vision and natural language processing.

At its core, a single **artificial neuron** is a simple processing unit. It receives a vector of inputs, $x = (x_1, x_2, \ldots, x_n)$, where each input x_i is multiplied by a corresponding weight w_i. These weights signify the importance of each input. The neuron then computes a weighted sum of these inputs, adds a bias term (b), and passes the result through an activation function (g) to produce a single output, y. This operation can be expressed as:

$$y = g\left(\left(\sum_{i=1}^{n} w_i x_i\right) + b\right) \quad \text{or more compactly} \quad y = g(w^T x + b) \tag{3.1}$$

A **neural network (NN)** is constructed by organizing these neurons into layers: an input layer that receives the raw data, one or more hidden layers that perform the intermediate computations, and an output layer that produces the final prediction. Data flows directionally from the input layer through the hidden layers to the output layer in a process called forward propagation.

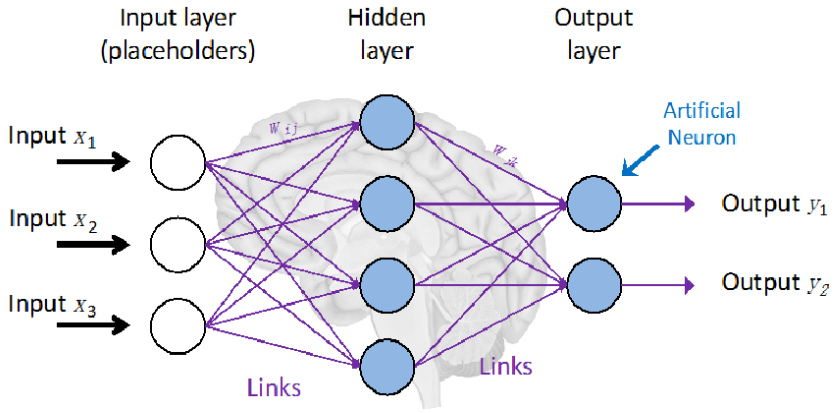

Figure 3.1: A diagram illustrating a simple neural network with an input layer, one hidden layer, and an output layer, showing the connections between neurons. [17]

Activation functions

The true power of neural networks comes from their ability to model complex, non-linear relationships, which are characteristic of most real-world data. The simple weighted sum in Equation 3.1 is a linear operation. To capture non-linearities, an **activation function** is applied at the end of each neuron's computation. Without these functions, even a deep network with many layers would behave no differently than a single, simple linear model. Common activation functions include:

- **Sigmoid:** $g(z) = \frac{1}{1+e^{-z}}$. This function maps any input value to a smooth curve between 0 and 1. While historically popular, especially for output layers in binary classification tasks, it can lead to the "vanishing gradient" problem in deep networks, where the gradients become extremely small, hindering the learning process.
- **Hyperbolic Tangent (tanh):** $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. Similar to sigmoid, but squashes values to a range between -1 and 1. Because its output is zero-centered, it often helps speed up convergence. However, like the sigmoid function, it is susceptible to the vanishing gradient problem for very large or very small input values.
- **Rectified Linear Unit (ReLU):** $g(z) = \max(0, z)$. The most popular choice in modern deep learning, ReLU is computationally efficient and simply outputs the input if it is positive, and zero otherwise. It is computationally very efficient and helps to alleviate the vanishing gradient problem. Its primary disadvantage is the "dying ReLU" problem, where a neuron can get stuck in a state where it only outputs zero and can no longer learn.

Training procedure

The objective of training a neural network is to find the optimal set of parameters (weights and biases, collectively denoted as θ) that allow it to approximate a target function. The network learns a mapping $\hat{y} = f(x; \theta)$, where the prediction $\hat{y}$ should be as close as possible to the true value y. This is achieved by:

1. Performing a **forward pass** to compute a prediction for a given input.
2. Quantifying the error between the prediction and the true value using a **loss function**.

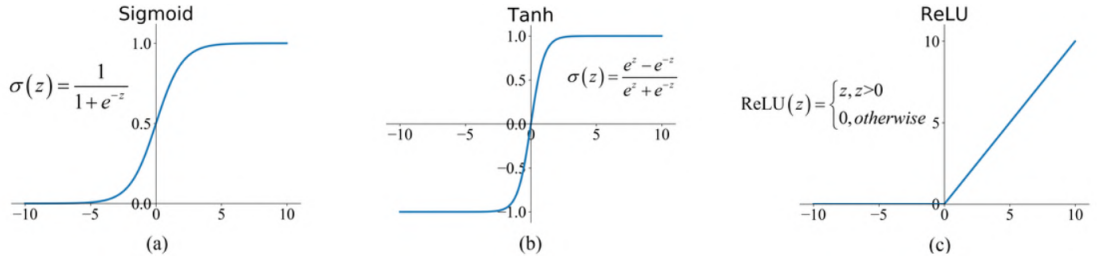

Figure 3.2: The graphs of the Sigmoid, tanh, and ReLU activation functions. [4]

3. Using an algorithm called **backpropagation**, which calculates the gradient of the loss function with respect to each network parameter.
4. Slightly adjusting the parameters in the direction that minimizes the loss, a process typically managed by an **optimization algorithm** like Stochastic Gradient Descent (SGD) or Adam.

This iterative process of forward pass, loss calculation, and backpropagation is repeated for all data in the training set, allowing the network to progressively minimize its error and improve its predictive accuracy.

To guide this minimization process, a **loss function** (or cost function) is used to formally quantify the discrepancy between the network's predicted output ($\hat{y}$) and the actual target value (y). The choice of loss function is critical and depends on the network's objective. For **regression tasks**, where the goal is to predict a continuous value like a price or an angle, a common choice is the **Mean Squared Error (MSE)**. It computes the average of the squared differences between predicted and true values, penalizing larger errors more heavily:

$$L_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.2}$$

For **classification tasks**, where the goal is to assign an input to a discrete category, **Cross-Entropy (CE) Loss** is the standard. It measures the dissimilarity between the predicted probability distribution and the true distribution: penalizing larger errors more heavily:

$$L_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{3.3}$$

For a **gaze estimation task**, which is a regression problem, the loss is often a function of the angular distance between the predicted and true gaze vectors.

Once the loss is calculated, an **optimizer** is the algorithm that applies the gradients computed during backpropagation to update the network's parameters (θ). The most fundamental optimizer is **Stochastic Gradient Descent (SGD)** [18], which updates the parameters using the gradient computed from a small, random subset of the training data, known as a mini-batch. While effective, its fixed learning rate can sometimes lead to slow or unstable convergence. To address this, more advanced **adaptive optimizers** have been developed. The most popular among these is **Adam (Adaptive Moment Estimation)** [11], which computes individual adaptive learning rates for each parameter. Adam also incorporates momentum by using moving averages of past gradients, which helps accelerate convergence and makes the training process more stable. A variant, **AdamW** [16], is frequently used in modern Transformer-based architectures for its improved weight decay implementation.

3.1.2. Convolutional Neural Networks (CNNs)

A standard NN, often called a fully connected network, treats an input as a simple list of numbers. For an image, this means flattening a 2D grid of pixels into a long 1D vector. This approach has two major drawbacks: it is computationally expensive due to the massive number of connections, and more importantly, it discards the spatial structure of the image. The information about which pixels are neighbors, which form lines, shapes, and textures, is lost. **Convolutional Neural Networks (CNNs)** are a

specialized type of neural network designed to overcome these limitations. They are the powerhouse behind many modern AI applications, particularly in computer vision tasks like image classification, object detection, and medical image analysis. CNNs are inspired by the human visual cortex and are designed to automatically and adaptively learn spatial hierarchies of features from images.

The fundamental building block of a CNN is the **convolutional kernel** (or filter). A kernel is a small matrix of learnable parameters, and the core idea is that this kernel acts as a feature detector. At each position, the kernel performs a mathematical operation called a **convolution**. This involves an element-wise multiplication of the kernel's values with the corresponding pixel values of the image patch it's currently over, and then summing up the results. This sum produces a single number in a new matrix called a **feature map**.

The formula for a 2D convolution on an image I with a kernel K to produce a feature map S is:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n)$$

This process is repeated across the entire image, creating a feature map that highlights where the specific feature detected by the kernel appears. An example is shown in Figure 3.3. The crucial point is that the network **learns** the values of these kernels during training, figuring out which features are important for the given task.

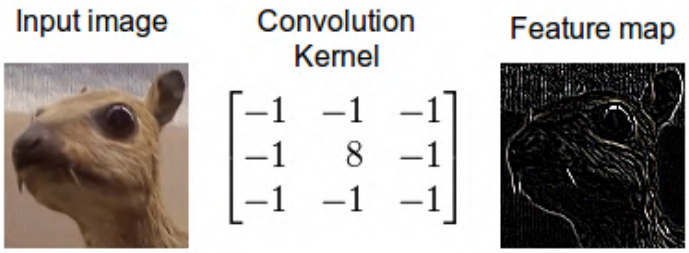

Figure 3.3: Convolution with an edge detection kernel. Image taken from `https://developer.nvidia.com/discover/convolution`

CNNs derive their efficiency and effectiveness from **three key properties**:

- **Local Connectivity:** Neurons in a convolutional layer are only connected to a small, localized region of the input layer (the "receptive field"). This is unlike a standard NN where every neuron is connected to every neuron in the previous layer. This drastically reduces the number of parameters and the computational overhead.
- **Parameter Sharing:** The same kernel is used across the entire image. This means that a feature detector (e.g., for a horizontal edge) is shared across all spatial locations. This not only further reduces the parameter count but also makes the network **translationally equivariant**.
- **Translation Equivariance:** This means that if an object in the image moves, its representation in the feature map will move by the same amount. This allows the network to recognize the same feature regardless of its position.

After a convolution operation, it's common to apply a **downsampling** (or pooling) layer. The main purpose of downsampling is to reduce the spatial dimensions of the feature maps, which has two benefits:

1. It reduces the number of parameters and computational load in the network.
2. It helps to make the feature representations more robust to small shifts and distortions.

The most common type of downsampling is **Max Pooling**. It involves sliding a window over the feature map and, for each patch, taking only the maximum value. This has the effect of summarizing the features in a region and retaining the most prominent ones. Another option is **Average Pooling**, which takes the average of the values in the patch, providing a more generalized summary. Max pooling often performs better in practice as it is more effective at highlighting salient features.

A Typical CNN Architecture

A typical CNN is composed of a sequence of layers. The most common pattern is:

Convolutional Layer $\rightarrow$ **Activation Layer (ReLU)** $\rightarrow$ **Pooling Layer**

This block of layers can be repeated multiple times.

- **Convolutional Layer:** Applies kernels to the input to create feature maps.
- **Activation Layer (ReLU):** An activation function, typically the Rectified Linear Unit (ReLU), is applied element-wise to the feature map. ReLU introduces non-linearity, allowing the network to learn more complex patterns. It simply converts all negative values to zero.
- **Pooling Layer:** Downsamples the feature map.

After several of these blocks, the final feature maps are flattened into a 1D vector and fed into one or more **Fully Connected Layers**, which are standard neural network layers that perform the final classification. As an example, we show the LeNet-5 architecture in Figure 3.4.

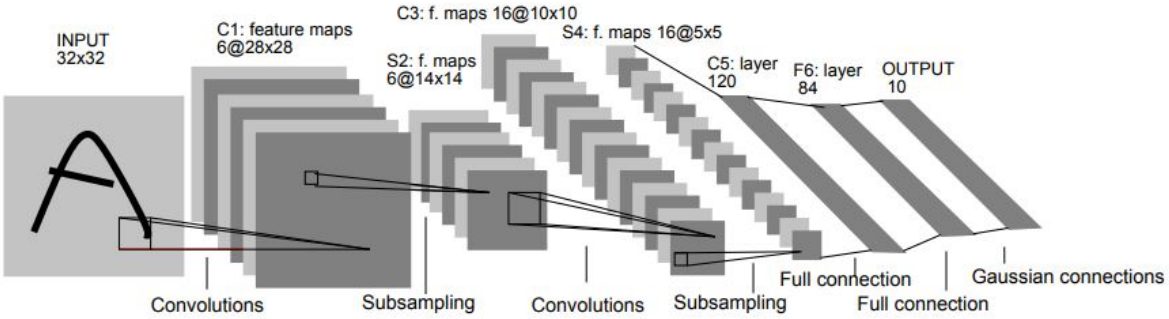

Figure 3.4: A typical CNN architecture (here, LeNet-5). [14]

Notable Architectures

- **LeNet-5 (1998):** Developed by Yann LeCun et al., it was one of the earliest CNNs and was famously used for recognizing handwritten digits on checks [14]. Its architecture of stacked convolutional and pooling layers followed by fully connected layers set the standard for years to come.
- **AlexNet (2012):** Developed by Alex Krizhevsky et al., it won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 by a huge margin [13]. It was much deeper and larger than LeNet-5 and introduced key innovations still used today: the ReLU activation function, dropout for regularization, and data augmentation. Its success kickstarted the deep learning revolution.
- **VGGNet (2014):** Developed by the Visual Geometry Group at Oxford, VGGNet demonstrated that a very deep network could achieve excellent performance by using a simple and uniform architecture [20]. It exclusively used very small 3x3 convolutional kernels, showing that depth was a critical component for performance.

Residual Networks (ResNets)

As researchers tried to build deeper networks (like VGG), they encountered the **vanishing gradient problem**. In very deep networks, the gradients (the signals used to update the network's parameters during training) can become extremely small as they are propagated backward from the output layer. This effectively stops the earlier layers from learning anything, and adding more layers can paradoxically decrease the network's performance.

Residual Networks (ResNets), introduced by Kaiming He et al. in 2015, provided an elegant solution [7]. The core idea is to use "skip connections" or "shortcuts" that allow the gradient to be directly backpropagated to earlier layers. A "residual block" learns a function $F(x)$ that is added to the original input x. This means the block is trying to learn the **residual** (the difference). It's easier for the network

to learn to push the residual to zero (if a layer is not needed) than to learn an identity mapping. This innovation allowed for the successful training of networks that were hundreds or even thousands of layers deep, leading to a new level of performance. An illustration of the residual block is provided in Figure 3.5.

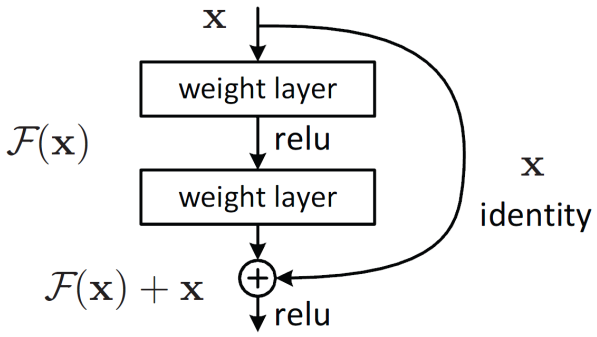

Figure 3.5: A ResNet residual block. [7]

3.1.3. Vision Transformers

For years, CNNs have been the gold standard for image-related tasks. Their architecture is inherently designed to recognize patterns locally through filters and build up a hierarchical understanding of an image. However, a 2017 paper introduced the **Transformer** architecture, which achieved state-of-the-art results in Natural Language Processing (NLP) tasks like machine translation [22]. Transformers excelled at understanding the relationships between words in a sentence, no matter how far apart they were. This prompted a question: could this same architecture be applied to images? Researchers demonstrated that it could and introduced the **Vision Transformer (ViT)** [3]. The core idea was to treat an image not as a grid of pixels, but as a sequence of smaller image patches, similar to a sequence of words in a sentence.

Unlike a CNN that processes an image pixel by pixel through sliding filters, a ViT takes a more direct, global approach.

1. **Image patching**: The input image is split into a grid of fixed-size, non-overlapping patches. For example, a 224x224 pixel image might be broken down into 196 patches, each being 16x16 pixels.
2. **Flattening & linear embedding**: Each of these 2D patches is then "flattened" into a single long vector of numbers. This vector is then projected into a fixed-dimensional space through a linear transformation, creating what are called "patch embeddings". This process is analogous to looking up a word in a dictionary to get its corresponding vector representation.
3. **Adding positional embeddings**: By breaking the image into a sequence, we lose all spatial information. The model doesn't know if a patch came from the top-left corner or the bottom-right. To solve this, a learnable **positional embedding** is added to each patch embedding. This is a vector that encodes the original (x, y) position of the patch, ensuring that the model has the spatial context necessary to understand the image structure.

Self-Attention

The sequence of embedded patches is then fed into the core of the model: the Transformer Encoder. Its key component is the **self-attention mechanism**. Self-attention allows the model to weigh the importance of all other patches in the sequence when interpreting a single patch. For instance, to understand a patch corresponding to a cat's ear, the model might "pay more attention" to other patches corresponding to the cat's face and less to a patch of the background. This is achieved by creating three vectors from each input patch embedding: a **Query (Q)**, a **Key (K)**, and a **Value (V)**.

- **Query**: Represents the current patch's "question", i.e., "what should I look for?"
- **Key**: Represents the "label" or identity of all other patches in the sequence.
- **Value**: Represents the actual content of the other patches.

The model calculates a similarity score between the *Query* of one patch and the *Key* of every other patch. These scores are then used to create a weighted sum of all the Value vectors. The formula for this "Scaled Dot-Product Attention" is:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d_k is the dimension of the *Key* vectors. This allows the ViT to build a rich, context-aware representation of each patch based on a global understanding of the entire image.

The Transformer Encoder

The self-attention mechanism is part of a larger block called a **Transformer Encoder**. This block is stacked multiple times (e.g., 12 or 24 times) to progressively refine the patch representations. Each encoder block consists of:

1. A Multi-Head Self-Attention layer (which runs the attention mechanism in parallel multiple times and concatenates the results).
2. A simple Feed-Forward Neural Network (MLP).

After passing through all the encoder layers, a special `[CLS]` (classification) token, which was added to the sequence at the beginning, is used to aggregate the information from all patches. This token's final output vector is fed into a small MLP head to produce the final classification (e.g., "cat", "dog").

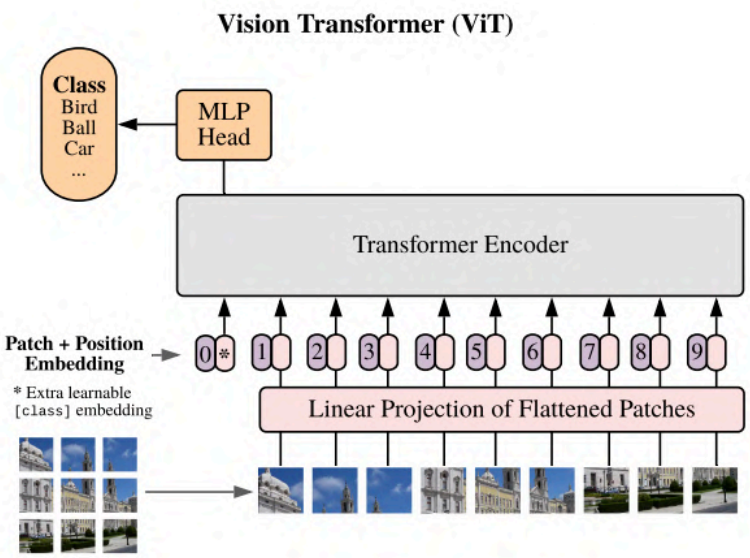

Figure 3.6: The overall architecture of the Vision Transformer for image classification. [3]

ViTs vs. CNNs

CNNs have a strong **inductive bias**. Their convolutional and pooling layers are inherently designed to respect the 2D spatial structure of images (locality and translation equivariance). This is a helpful, built-in assumption that makes them very data-efficient. They can learn well from smaller datasets.

ViTs have a much weaker inductive bias. They make no initial assumption about the image structure, treating it merely as a sequence. This flexibility is a double-edged sword:

- **Data Hungriness**: On small or medium-sized datasets, ViTs often underperform compared to CNNs because they have to learn the spatial relationships from scratch.
- **Scalability**: When trained on massive datasets (e.g., >14 million images), ViTs can learn these relationships and surpass the performance of even the best CNNs. Their ability to model long-range dependencies without the constraints of a local receptive field becomes a significant advantage.

While initially designed for image classification, the ViT architecture has been successfully adapted for many other computer vision tasks. These include object detection, image segmentation, medical imaging, and even video analysis.

3.2. Gaze Estimation

3.2.1. Task Introduction

Gaze estimation is a computer vision task that aims to determine where a person is looking. At its core, it is the technology that powers eye-tracking systems, transforming the subtle movements of the human eye into a rich source of data about cognitive states, focus, and intent. The task is generally divided into two primary objectives:

- **2D Point-of-Gaze (PoG) estimation**, which predicts the specific (x, y) coordinates where a person's line of sight intersects with a 2D plane, such as a computer or mobile device screen. This is primarily used for human-computer interaction (HCI).
- **3D Gaze Vector estimation**, which predicts a 3D unit vector representing the orientation of the eyeball in three-dimensional space. For ease of model training and to reduce the dimensionality of the regression problem, this 3D vector is often parameterized as two angles: **pitch** (vertical rotation) and **yaw** (horizontal rotation). Unlike 2D PoG, 3D gaze vector estimation is not tied to a specific screen or surface. This is a more general task, often used in applications like driver monitoring systems where the target is not confined to a screen.

The ability to understand human attention has profound implications across a multitude of domains. For instance, in Human-Computer Interaction (HCI), gaze can be used as an input modality, enabling hands-free control of devices for individuals with motor impairments or providing novel interactive experiences in virtual and augmented reality [2]. In automotive safety, driver monitoring systems employ gaze estimation to detect distraction or drowsiness, thereby preventing accidents [9]. Furthermore, gaze patterns are increasingly recognized as important biomarkers in the medical field for diagnosing and understanding neurological conditions such as Autism Spectrum Disorder (ASD) and Attention-Deficit/Hyperactivity Disorder (ADHD) [23]. In marketing and usability studies, tracking consumer gaze provides invaluable insights into attention and engagement with advertisements and products [19].

The Angular Error Evaluation Metric

The standard performance metric for 3D gaze estimation is the **mean angular error**, which measures the angle in degrees between the predicted gaze vector ($\hat{g}$) and the ground-truth gaze vector (g). For a single sample, it is calculated using the dot product:

$$\theta_{\text{error}} = \arccos(g \cdot \hat{g})$$

The final reported metric is the mean of this error across all test samples.

The interpretation of this error is highly application-dependent. State-of-the-art appearance-based methods achieve errors of approximately 3-5 degrees on challenging "in-the-wild" benchmarks. For high-precision tasks like assistive technology, an error of even 2-3 degrees can be restrictive. However, for coarse gaze zone classification, such as determining if a driver is looking at the road or a mirror, an error of 4-5 degrees is often acceptable [21]. It is also important to note that a small angular error can translate to a large error in the estimated depth (Z-axis) of the point of regard, which is a challenge for 3D interaction in VR/AR [15].

3.2.2. Appearance-Based Gaze Estimation

Gaze estimation methods are broadly divided into two paradigms: model-based and appearance-based [6].

- **Model-based (geometric) methods** create an explicit 3D geometric model of the eye. They rely on detecting specific features like the pupil center and corneal reflections (glints) from an active infrared (IR) light source. This approach can be highly accurate but typically requires specialized and expensive hardware, limiting its use to controlled lab settings. [1]
- **Appearance-based methods** reframe the task as a direct regression problem, learning a mapping function $g = f(I)$ from the raw pixel values of an image I (of the eye or full face) to the gaze

direction g. This approach leverages machine learning, particularly deep neural networks (CNNs and ViTs), to automatically learn relevant features from the data. The primary advantage is its ability to function with standard, low-cost RGB cameras found in consumer devices like webcams and smartphones, thus making the technology more accessible. [1]

The success of appearance-based methods is heavily dependent on the power of deep learning models to approximate the highly complex and non-linear mapping function. However, this learning process is complicated by significant "in-the-wild" challenges, including variations in illumination, occlusions (e.g., from hair or eyeglasses), and extreme head poses, all of which alter the input image's appearance [1]. To mitigate the difficulty of learning under free head movement, a crucial pre-processing step known as **data normalization** is often employed. This technique synthetically warps the input image to a canonical pose, relieving the network of the burden of learning head pose invariance and thereby improving accuracy and training efficiency [1].

3.2.3. Datasets used

The advancement of robust, appearance-based models is entirely linked to the development of large-scale, diverse datasets that capture "in-the-wild" conditions [1]. Two of the most influential datasets are MPIIFaceGaze and ETH-XGaze.

The **MPIIGaze** [25] dataset was a pioneering effort to collect gaze data in natural, unconstrained settings. It contains over 213,000 images gathered from 15 participants using their personal laptops over several months, resulting in significant variability in appearance and illumination. The later **MPIIFaceGaze** dataset extended this by providing the corresponding full-face images, which proved crucial for allowing models to implicitly learn and account for head pose, significantly improving performance. Together, they became a standard benchmark for person-independent gaze estimation research.

While MPIIGaze captured naturalistic data, it did not exhaustively cover the full physiological range of motion. The **ETH-XGaze** [24] dataset was created to address this by systematically sampling extreme head poses and gaze angles. Using a custom rig of 18 high-resolution DSLR cameras, researchers collected over one million images from 110 diverse participants. The dataset covers head poses up to $\pm 70^\circ$ and gaze directions up to $\pm 50^\circ$, including samples with varied illumination and subjects wearing glasses. ETH-XGaze serves as the primary benchmark for developing the next generation of highly robust gaze estimators capable of handling extreme, real-world conditions.

Other key datasets have driven progress in specific areas:

- **GazeCapture** [12]: Focused on massive scale for mobile devices, collecting nearly 2.5 million frames from over 1,450 people via a crowdsourced iOS app for 2D PoG estimation.
- **Gaze360** [10]: Targeted physically unconstrained 3D gaze in both indoor and outdoor environments, using a 360-degree camera to capture video with wide head pose variation.
- **RT-GENE** [5]: Designed for real-time applications, this dataset used mobile eye-tracking glasses to get precise ground-truth labels and then used GANs to digitally remove the glasses from the images, creating realistic training data.

References

[1] Yihua Cheng et al. “Appearance-based gaze estimation with deep learning: A review and benchmark”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[2] Piercarlo Dondi and Marco Porta. “Gaze-based human–computer interaction for museums and exhibitions: technologies, applications and future perspectives”. In: *Electronics* 12.14 (2023), p. 3064.

[3] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).

[4] Junxi Feng et al. “Reconstruction of porous media from extremely limited information using conditional generative adversarial networks”. In: *Physical Review E* 100 (Sept. 2019). DOI: 10.1103/PhysRevE.100.033308.

[5] Tobias Fischer, Hyung Jin Chang, and Yiannis Demiris. “Rt-gene: Real-time eye gaze estimation in natural environments”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 334–352.

[6] Dan Witzner Hansen and Qiang Ji. “In the eye of the beholder: A survey of models for eyes and gaze”. In: *IEEE transactions on pattern analysis and machine intelligence* 32.3 (2009), pp. 478–500.

[7] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[8] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. “Spatial transformer networks”. In: *Advances in neural information processing systems* 28 (2015).

[9] Anuradha Kar and Peter Corcoran. “A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms”. In: *IEEE Access* 5 (2017), pp. 16495–16519.

[10] Petr Kellnhofer et al. “Gaze360: Physically unconstrained gaze estimation in the wild”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6912–6921.

[11] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980.

[12] Kyle Krafka et al. “Eye tracking for everyone”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2176–2184.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[14] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE*. Vol. 86. 11. IEEE, 1998, pp. 2278–2324.

[15] Meng Liu, Youfu Li, and Hai Liu. “3D gaze estimation for head-mounted eye tracking system with auto-calibration method”. In: *IEEE Access* 8 (2020), pp. 104207–104215.

[16] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG]. URL: https://arxiv.org/abs/1711.05101.

[17] Harry McGrath et al. “Future of Artificial Intelligence in Anesthetics and Pain Management”. In: *Journal of Biosciences and Medicines* 07 (Jan. 2019), pp. 111–118. DOI: 10.4236/jbm.2019.711010.

[18] Herbert Robbins and Sutton Monro. “A stochastic approximation method”. In: *The annals of mathematical statistics* (1951), pp. 400–407.

[19] Christopher Rumpf, Felix Boronczyk, and Christoph Breuer. “Predicting consumer gaze hits: A simulation model of visual attention to dynamic marketing stimuli”. In: *Journal of Business Research* 111 (2020), pp. 208–217.

[20] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[21] Borhan Vasli, Sujitha Martin, and Mohan Manubhai Trivedi. “On driver gaze estimation: Explorations and fusion of geometric and data driven approaches”. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2016, pp. 655–660.

[22] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[23] Zheng Zeng et al. “A robust gaze estimation approach via exploring relevant electrooculogram features and optimal electrodes placements”. In: *IEEE Journal of Translational Engineering in Health and Medicine* 12 (2023), pp. 56–65.

[24] Xucong Zhang et al. “Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer. 2020, pp. 365–381.

[25] Xucong Zhang et al. “Mpiigaze: Real-world dataset and deep appearance-based gaze estimation”. In: *IEEE transactions on pattern analysis and machine intelligence* 41.1 (2017), pp. 162–175.