Generative Adversarial Framework for Eye Image Synthesis and Gaze Estimation

Final year Project Report submitted by

Prashanth Reddy Duggirala
Roll No: CED14I006

Under the Guidance of
Dr. V. Masilamani
IIITDM

in partial fulfilment for the award of the degree

B.Tech & M.Tech
in
Computer Engineering

Indian Institute of Information Technology Design and Manufacturing
Kancheepuram, Melakottaiyur, Chennai-600127
Jan - May 2019

Bonafide Certificate

This is to certify that the Project titled **"Generative Adversarial Framework for Eye Image Synthesis and Gaze Estimation"** submitted by **Prashanth Reddy Duggirala, (CED14I006)** to the Indian Institute of Information Technology Design and Manufacturing, Kancheepuram for Design Project, is a bonafide record of the project work done by him under my supervision. The contents of the project, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. V. Masilamani
Project Guide
Assistant Professor
Department of Computer Science
Indian Institute of Information Technology Design and Manufacturing
Kancheepuram

Place: Chennai
Date: April 29, 2019

Contents

List of Figures

Acknowledgement

I would like to express my deepest gratitude to my advisor Dr. V. Masilamani for his guidance, assistance and support throughout this project. His continuous inputs, comments, remarks and engagement through the learning process of M.Tech project are highly invaluable to me.

I would like to thank the dissertation committee for reviewing my proposal, addressing important issues and providing valuable suggestions. The knowledge acquired from their reviews has been very much useful in my project.

Finally, I would like to thank Indian Institute of Information Technology Design and Manufacturing, Kancheepuram for providing unrestricted access to the laboratory facilities.

Abstract

Unconstrained non-intrusive gaze tracking using off-the-shelf cameras is a challenging problem. In the recent times, promising algorithms for appearance-based gaze estimation using convolutional neural networks (CNN) have been proposed, including the ones developed by us. Improving their robustness to various confounding factors including variable head pose, subject identity, illumination and image quality remain open problems. In this report, we present a a novel multi-input CNN architecture that improves the robustness of gaze classifier to these variable head pose. We also present our current research of using Generative Adversarial Networks to generate a large dataset of high resolution images with high diversity (e.g., in subjects, head pose, camera settings) and realism, while simultaneously preserving the accuracy of gaze labels. Using this synthesized images along with real image data, we plan to improve the overall quality of the dataset in terms of diversity. In addition, the proposed models are not restricted to eye images only. It can be adapted to face images and any shape-appearance related fields.

1 Introduction

Pediatric Perimeter is a novel, first of its kind device to measure the side vision of infants. The procedure of quantifying visual field is called perimetry. This device is used on children with potential disease to cause a visual field defects such as Glaucoma, Retinitis Pigmentosa, Retinopathy of Prematurity, etc. It involves checking the eye/head movement of an infant to a visual stimulus in particular areas in the visual field using various tests in perimetry involving LEDs fitted inside the Pediatric Perimeter [1]. The device is a hemispherical dome with light emitting diodes. The LEDs are controlled using a computer program to measure Reaction Time to Gross Visual Fields (GVF) and the Visual Field Extent (VFE). Infants lie down in the dome positioned in a dark room. Eye or head movement towards the stimuli is monitored with an infrared (IR) camera.

Pediatric Perimeter is currently operated by a trained clinician, who has to acknowledge the reaction from the children during testing inside a Pediatric perimeter. This approach has an inherent issue of human errors and subjective biases in quantifying Reaction Times and judging direction of eye gaze. The idea is to incorporate gaze detection in the device's software which aids the user in performing the tests with reduced human error and help in getting more accurate readings.

Existing gaze estimation approaches fall into appearance based and model-based approaches. The appearance-based approaches estimate eye gaze directly from eye appearance or extracted eye features. The most common features are the centers of the pupil and one or more corneal reflections. The corneal reflections (first Purkinje images) are virtual images of infrared light sources that illuminate the eye, and are created by the front surface of the cornea, which acts as a convex mirror. The first Purkinje image off the corneal surface is also called the glint. In standard gaze trackers, the image of the eye is processed in three basic steps. First, the specular reflection of a stationary light source is found in the eye's image. Second, the pupil's center is found. Finally, the relative position of the light's reflection and the pupil's center is calculated [**?**]. From information about their relative positions, the gaze

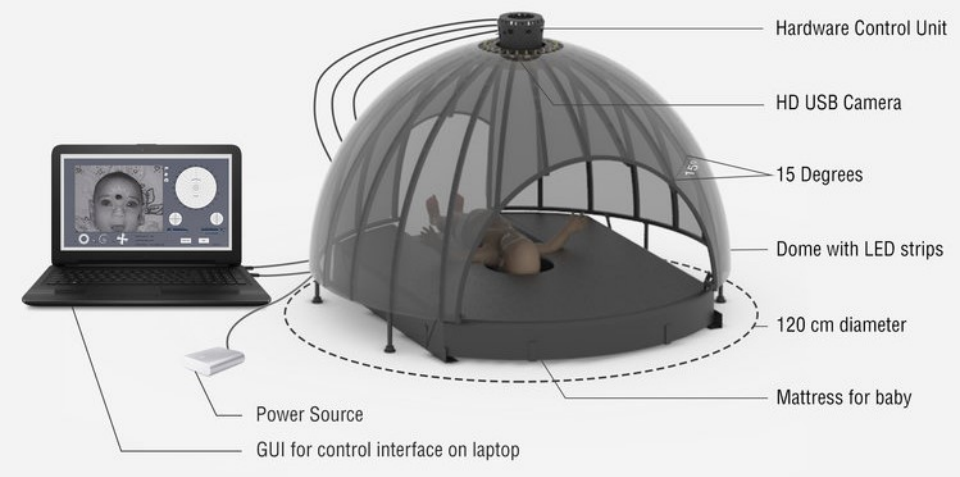

Figure 1: **Pediatric Perimeter**

direction is determined (Figure 2).

They need sufficient amount of data with ground truth labels to produce accurate results. Model-based approaches leverage on anatomical eye model knowledge to estimate eye gaze from detected eye features. They are sensitive to the variation of personal eye parameters such as eyeball radius. A subject specific personal calibration is often required for each person to estimate these personal eye parameters. They also require explicit detection of eye features such as pupil center and cornea reflections.

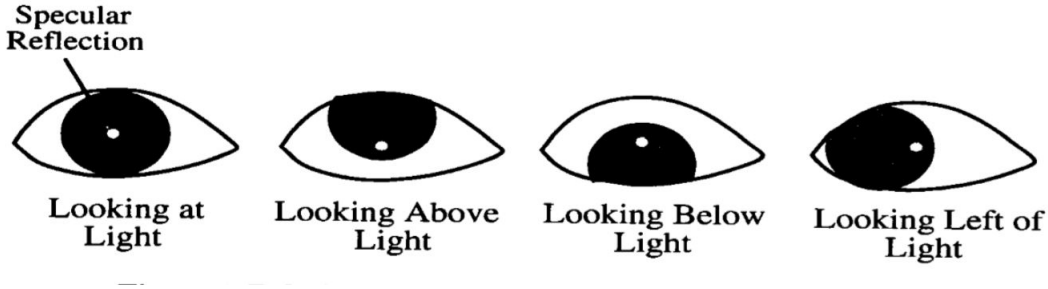

Figure 2: **Relative position of specular reflection and pupil.**

Source: Dean Pomerleau & Shumeet Baluja, School of Computer Science, Carnegie Mellon University

Unconstrained gaze tracking refers to calibration free, subject, viewpoint, and illumination-independent gaze tracking using a remotely placed off-the-shelf camera. It allows for free head motion of the subject. Recently, various promising appearance-based techniques for unconstrained remote gaze estimation have been proposed [2], [3], [4], [5], [6]. Among them, techniques based on convolutional neural networks (CNN) are the most successful [4], [6]. Convolutional Neural Networks (CNNs) have proven to be very successful frameworks for image recognition.

Researchers have also proposed computer graphics models to generate synthetic training data for gaze estimation, along with techniques to adapt them to real world data [7], [8]. Nevertheless, many open problems remain to be solved in order to increase the accuracy of gaze tracking including robustness to variability in head pose, illumination, subject identity, image quality and distribution of head pose and gaze angles [6]. In this work we address several challenges of unconstrained gaze tracking, we present a a novel multi-input CNN architecture that improves the robustness of gaze classifier to variable head pose.

The lack of sufficient amount of the training data or uneven class balance within the datasets is one of the most frequently mentioned problem in the field of machine learning. One of the ways of dealing with this problem is called data augmentation, multiple methods of data augmentation like classical image transformations like flipping, cropping, zooming, histogram based methods can be used in order to improve the training process efficiency. Further, Generative adversarial nets were recently introduced as an alternative framework for training generative models in order to sidestep the difficulty of approximating many intractable probabilistic computations. In an unconditioned generative model, there is no control on how the data is being generated. However, by conditioning the model on additional information, it is possible to direct the data generation process to generate samples belonging to the existing 9 Gaze direction classes. If this technique is successful in generating realistic sample eye images, we hope to extend this experiment to condition the models in such a way that we could generate image samples at any gaze angle with many more conditions pertaining to the background, skin-tone, illumination with a fine-grained control.

2 Related Work

There is a large amount of work present for the detection of head pose and for eye gaze estimation, but there is only few work present where both information, head pose and eye gaze are combined. For example, Matsumoto et al.[9] present a three stage system that combines head pose andeye gaze information to accurately estimate a person's point of attention. The particular point of their system is that they use a 3D head model and a 3D eye model to accurately detect head pose and eye gaze based on stereo vision. Naqvi et al.[10] developed a gaze detection system for safe driving that classified 17 gaze zones using a convolutional neural network. In order to verify the validity of the method, they designed an experiment including the images of left eye, right eye and face, respectively using facial landmarks.

Current state of the art frameworks like OpenFace claims to be the first open source tool capable of facial landmark detection, head pose estimation, facial action unit recognition, and eye-gaze estimation [11]. similarly, another proposed method called, HyperFace, fuses the intermediate layers of a deep CNN using a separate CNN followed by a multi-task learning algorithm that operates on the fused features. It exploits the synergy among the tasks which boosts up their individual performances[12].

2.1 Generative Adversarial Networks

Neural Networks are good at classifying and predicting things this class of models are called Discriminative models, and now AI Researchers are working to make the neural net more human in nature by allowing it to create rather than just letting it see things and discriminate, called as Generative models.

2.1.1 Discriminative Models

These models aim at learning $P(c|x)$ by using probabilistic approaches (e.g., logistic regression), or by mapping classes from a set of points (e.g., perceptrons and SVMs). Here c refers to a certain class and x is the data for which the class is to determined. They are relatively easier to model, given good quality data. But they are essentially good for classification tasks but not generate the data.

2.1.2 Generative Models

These models aim to model how the data is generated. From $P(x|c)P(c)$ we can obtain $P(c|x)$ (from Bayes' Theorem). - They try to learn a joint probability distribution $P(x, c)$. One advantage is that for generation task We can extract the knowledge about the data distribution. But it is very expensive owing to the existence of a lot of parameters, that too provided if we have access to huge amount of data.

In the adversarial nets framework, the generative model is pitted against an adversary: a discriminative model that learns to determine whether a sample is from the model distribution or the data distribution. The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while

the discriminative model is analogous to the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistiguishable from the genuine articles [13]. However, generating natural images of the real world have had not much success until recently. The original Generative Adversarial Networks generated images suffering from being noisy and incomprehensible, and There has been very limited published research in trying to understand and visualize what GANs learn, and the intermediate representations of multi-layer GANs.

2.2 DC GANs

Radford et al. introduced a class of CNNs called deep convolutional generative adversarial networks (DCGANs), that have certain architectural constraints, and demonstrated that they are a strong candidate for unsupervised learning [14]. Training on various image datasets, they show convincing evidence that their deep convolutional adversarial pair learns a hierarchy of representations from object parts to scenes in both the generator and discriminator.

2.3 Conditional GANs

Mirza et al. were the first to make Class-conditional models where we make the label the input, rather than the output. We can then ask the GAN to generate an example from a specific class [15]. For example, if we ask for the horse class, and it generates a picture of a horse. In an unconditioned generative model, there is no control on modes of the data being generated. However, by conditioning the model on additional information it is possible to direct the data generation process. Such conditioning could be based on class labels, on some part of data for inpainting like [16], or even on data from different modality.

2.4 Training Image Synthesis

The idea of rendering and synthesizing training images for training appearance-based gaze estimation model was introduced by Yusuke et al. in [32]. They proposed to recover the geometric structure of the face based on images taken from eight calibrated cameras. To increase the dataset size they rendered images from unseen views of the textured facial geometry. To enrich the diversity of the synthesized data, Wood et al. [33] constructed a 3D morphable model which can generate different shapes and textures of the eye region. This approach allows rendering images in an unlimited set of poses and lighting conditions while controlling the gaze direction of the 3D model. Yet, the simulated images still look synthetic and the diversity is limited to linear combinations of a few tens of subjects. Basically, this can be divided into two categories, First is to synthesize Auxiliary training samples. The synthetic samples are generated with corresponding gaze and head pose ranges matching those of the dataset and session settings. The second is Synthetic sample refinement, where an adversarial network is used for mapping from the synthetic domain into the realistic one [8].

3 Novelty

This project uses a novel hybrid model that integrates three input types, Both eyes individually and facial landmarks detected in that particular frame, both of which have proven results to some degree in estimating the gaze. In the new algorithm, instead of two different models, we present a new unified architecture which takes the two eye images and a vector containing the landmarks data.

Using Conditional Generative Adversarial Networks, it is possible to direct the data generation process to generate samples belonging to the existing 9 Gaze direction classes. If this technique is successful in generating realistic sample eye images, we can extend this experiment to condition the models in such a way that we can generate image sample at any gaze angle.

4 Tools Used

- Languages: Python, C
- libraries: TensorFlow, Keras, OpenCV
- Hardware:
  - CPU: Intel i7 7700
  - GPU: Nvidia GeForce GTX 1080Ti

5 Multi-input CNN architecture

5.1 Dataset

The data is in the form of eye images of resolution 100×60 pixels, labelled with nine classes identified with integers 0-8.

Direction	**Label**	**Label ID**
Forward	FWD	0
Right	R	1
Top-Right	TR	2
Top	T	3
Top-Left	TL	4
Left	L	5
Bottom-Left	BL	6
Bottom	B	7
Bottom-Right	BR	8

Table 1: **Class Representation Table**

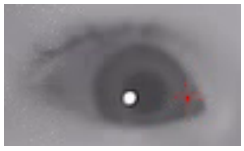

Figure 3: Image of the left eye

Figure 4: Image of the right eye

The vector representing the facial landmarks for each frame is in the shape of $(68, 2)$. There are 68 landmarks in total and each (x, y) containing the information about the location of the landmark in the picture. This is then reshaped to a 1D vector of length 136.

5.2 Test Set and Validation Set For each Eye

The Total Number of images taken from videos meant for training and validation data were in upwards of 25,000 and 2,500 respectively. In the training set, to maintain equal representation of all the classes, the data is sampled to contain 1000 data points from each class.

5.2.1 Head pose estimation from facial landmarks

In computer vision the pose of an object refers to its relative orientation and position with respect to a camera. You can change the pose by either moving the object with respect to the camera, or the camera with respect to the object.

The pose estimation problem using facial landmark information is often referred to as Perspective-n-Point problem or PNP in computer vision jargon. As we shall see in the following sections in more detail, in this problem the goal is to find the pose of an object when we have a calibrated camera, and we know the locations of n 3D points on the object and the corresponding 2D projections in the image.

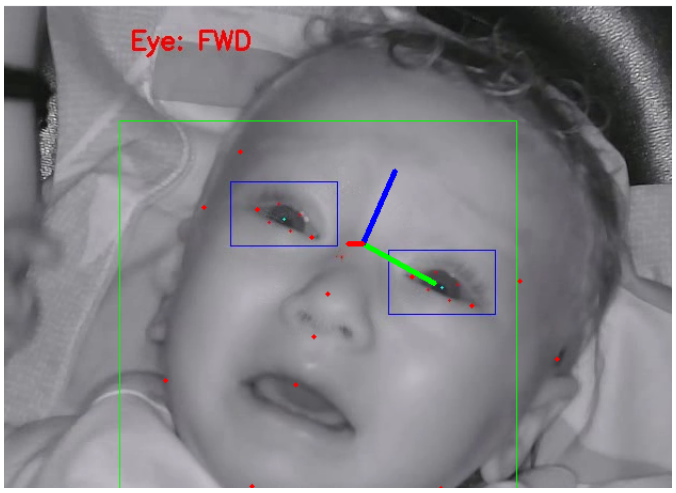

Figure 5: **Estimating Gaze using PnP algorithm for Head Pose**

5.3 CNN model

Convolutional nets are multi-layer architectures where the successive layers are designed to learn progressively higher-level features, until the last layer which produces categories. All the layers are trained simultaneously to minimize an overall objective function. the feature extraction is therefore an integral part of the classification system, rather than a separate module, and is entirely trained from data, rather than designed.

In the new algorithm, instead of two different models for each eye, we present a new unified architecture which takes the two eye images and a vector containing the landmarks data.

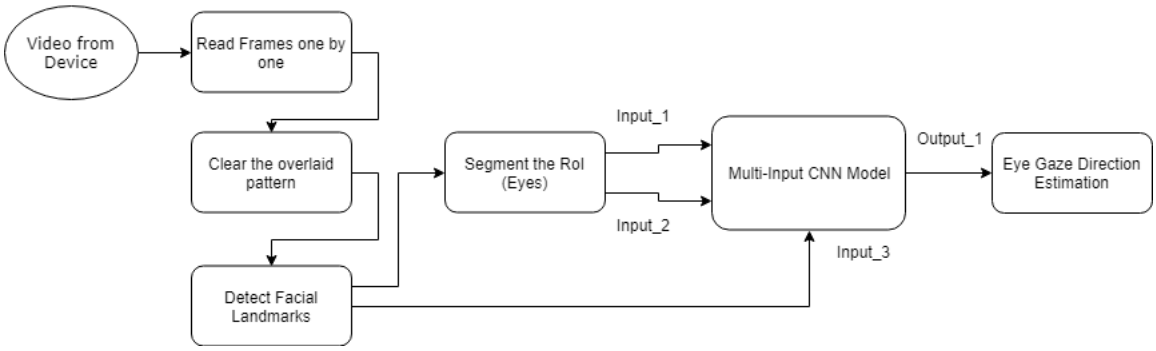

Figure 6: **Illustrating the process pipeline**

5.4 CNN Architecture

The inspiration for the convolutional network layer design is taken from mobilenet architecture. Initially designed for lower computational cost and known for its impressive performance in imagenet competition [17], it is lightweight and fast [18].

We use ReLU as an activation function which is a popular choice especially for deep networks. The activation function has been shown to speed up training time. Note that each sample image varies in scale and is not necessarily square. To use these images on CNNs, we rescale them to 128×128 pixels. The following is the schematic flowchart of the model (Figure 7).

5.5 Results

The main reason for implementing this new learning algorithm is ease of operation and training, if there is only one model which takes multiple inputs rather than multiple models for performing a single task, deployment and re-training can be significantly simpler. Additionally, there has been a marginal increase in the accuracy of the entire model. The average accuracy for the classification task is around 74.5% for the unified model where we used the hybrid model when compared with traditional CNN where accuracy was around 70%.

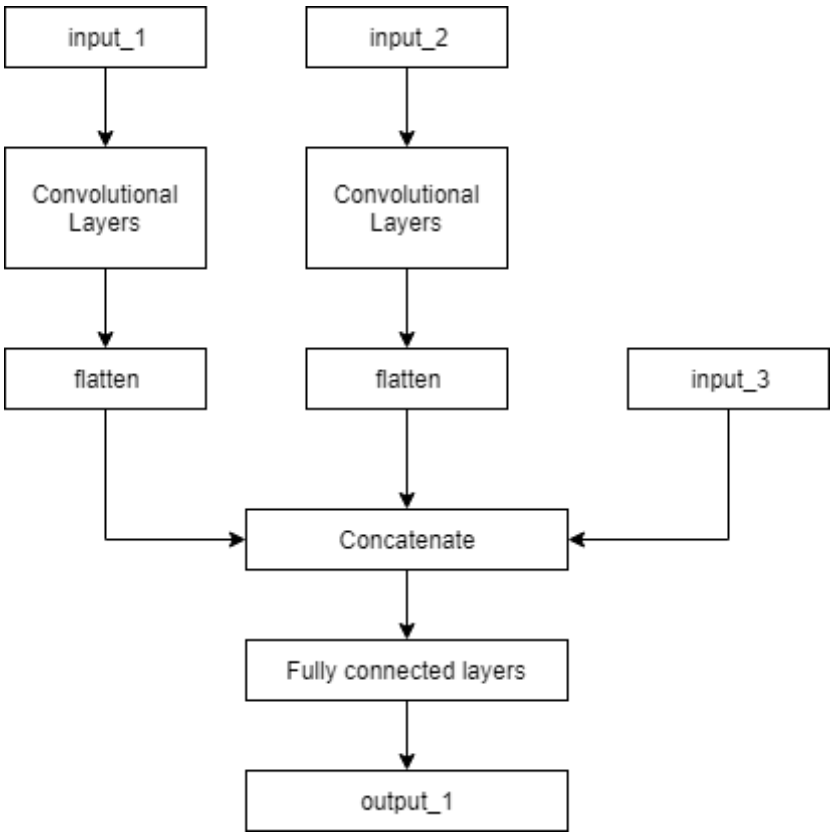

Figure 7: **Architecture of CNN**

The information about the performance of the machine learning model can be understood by a confusion matrix. In general, in a confusion matrix, the predicted classes are comparedwith the actual classes. Each row of the matrix represents the results of prediction for thecorresponding class at that row, while each column represents the actual class. The diagonalcells show the percentage of correct classifications by the trained classifier, while the offdiagonal cells represent the misclassified predictions.

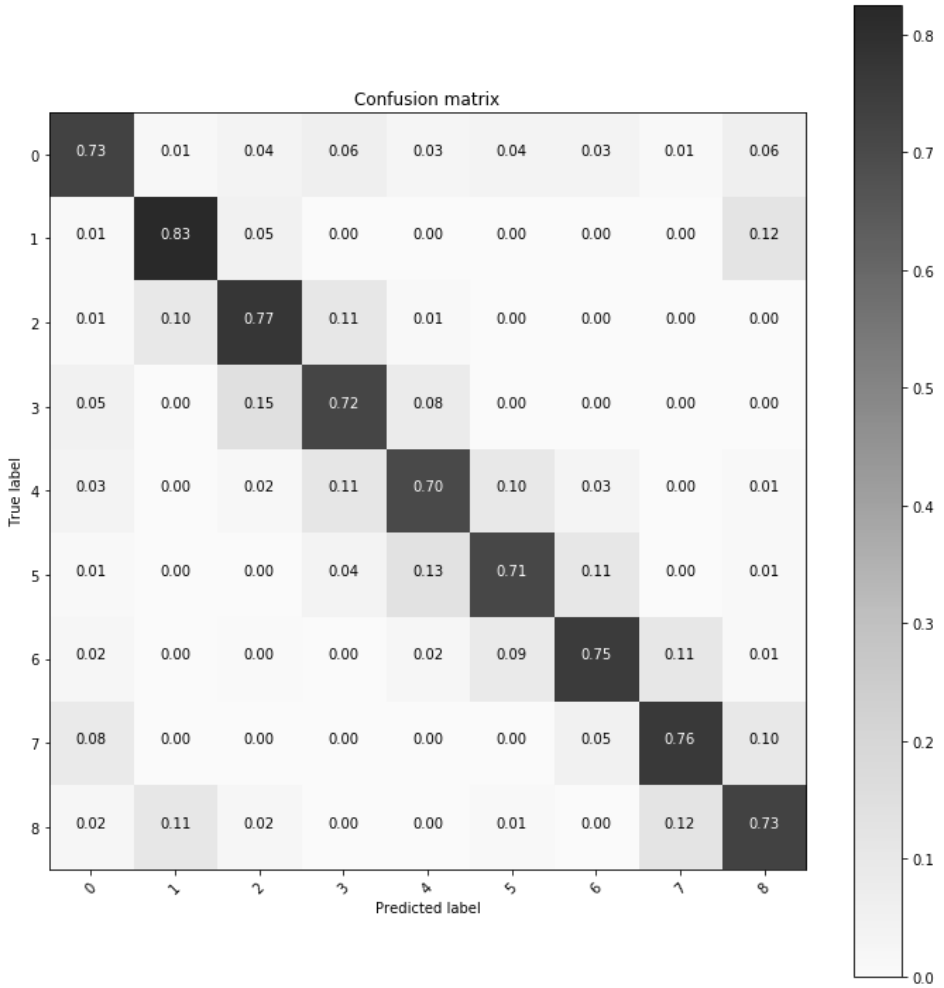

Figure 8: **Confusion Matrix**

6 Generating realistic eye image data

As aforementioned in this report, we try to generate eye images using the architectural ideas and research detailed in [14]. In this paper, they make the following Architecture guidelines for stable Deep Convolutional GANs:

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batch normalization in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

The schematic for a Deep Convolutional GAN model can be seen from the figure below;

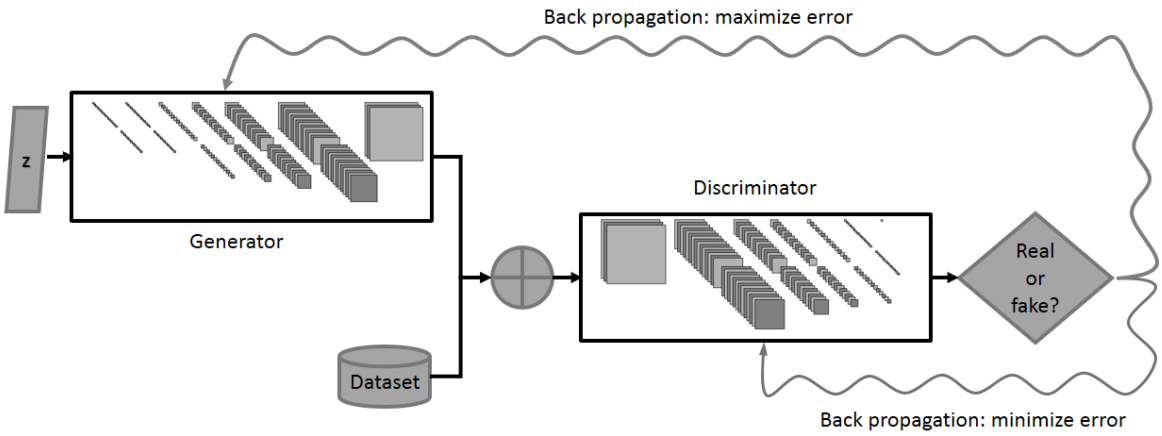

Figure 9: **Generative Adversarial Framework**

Source: Nvidia

There are four components to keep in mind about:

- X: The original, genuine data set
- Z: The random noise that goes into the generator as a source of entropy
- G: The generator which tries to copy/mimic the original data set
- D: The discriminator which tries to tell apart G's output from X

The generator network produces samples by taking an input vector z, sampled from what is called a latent distribution, and transforming it by applying the G function defined by the network, yielding G(z). The discriminator network receives alternating G(z) and x, a real data sample and outputs a probability of that input being real.

With proper hyperparameter tuning and enough training iterations, the generator and the discriminator will converge jointly performing parameter updates via a gradient descent variant called Adam [19] to a point where the distribution which described the synthesized data is the same as the distribution from which real data is sampled. For the loss function, we use binary crossentropy loss, also known as log loss.

The model is trained on 24,600 left eye images for 7000 epochs, the GAN architecture, layer-wise is detailed in the table below.

Layer (type)	Output Shape	Param #
dense_2 (Dense)	(None, 16384)	1654784
reshape_1 (Reshape)	(None, 8, 8, 256)	0
up_sampling2d_1 (UpSampling2	(None, 16, 16, 256)	0
conv2d_8 (Conv2D)	(None, 16, 16, 256)	590080
batch_normalization_7 (Batch	(None, 16, 16, 256)	1024
activation_1 (Activation)	(None, 16, 16, 256)	0
up_sampling2d_2 (UpSampling2	(None, 32, 32, 256)	0
conv2d_9 (Conv2D)	(None, 32, 32, 128)	295040
batch_normalization_8 (Batch	(None, 32, 32, 128)	512
activation_2 (Activation)	(None, 32, 32, 128)	0
up_sampling2d_3 (UpSampling2	(None, 64, 64, 128)	0
conv2d_10 (Conv2D)	(None, 64, 64, 128)	147584
batch_normalization_9 (Batch	(None, 64, 64, 128)	512
activation_3 (Activation)	(None, 64, 64, 128)	0
up_sampling2d_4 (UpSampling2	(None, 128, 128, 128)	0
conv2d_11 (Conv2D)	(None, 128, 128, 64)	73792
batch_normalization_10 (Batc	(None, 128, 128, 64)	256
activation_4 (Activation)	(None, 128, 128, 64)	0
conv2d_12 (Conv2D)	(None, 128, 128, 64)	16448
batch_normalization_11 (Batc	(None, 128, 128, 64)	256
activation_5 (Activation)	(None, 128, 128, 64)	0
conv2d_13 (Conv2D)	(None, 128, 128, 1)	257
activation_6 (Activation)	(None, 128, 128, 1)	0

Total params: 2,780,545
Trainable params: 2,779,265
Non-trainable params: 1,280

Figure 10: **Generator Model**

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 64, 64, 32)	320
leaky_re_lu_1 (LeakyReLU)	(None, 64, 64, 32)	0
dropout_1 (Dropout)	(None, 64, 64, 32)	0
conv2d_2 (Conv2D)	(None, 32, 32, 64)	18496
batch_normalization_1 (Batch	(None, 32, 32, 64)	256
leaky_re_lu_2 (LeakyReLU)	(None, 32, 32, 64)	0
dropout_2 (Dropout)	(None, 32, 32, 64)	0
conv2d_3 (Conv2D)	(None, 16, 16, 64)	36928
batch_normalization_2 (Batch	(None, 16, 16, 64)	256
leaky_re_lu_3 (LeakyReLU)	(None, 16, 16, 64)	0
dropout_3 (Dropout)	(None, 16, 16, 64)	0
conv2d_4 (Conv2D)	(None, 16, 16, 128)	73856
batch_normalization_3 (Batch	(None, 16, 16, 128)	512
leaky_re_lu_4 (LeakyReLU)	(None, 16, 16, 128)	0
dropout_4 (Dropout)	(None, 16, 16, 128)	0
conv2d_5 (Conv2D)	(None, 8, 8, 128)	147584
batch_normalization_4 (Batch	(None, 8, 8, 128)	512
leaky_re_lu_5 (LeakyReLU)	(None, 8, 8, 128)	0
dropout_5 (Dropout)	(None, 8, 8, 128)	0
conv2d_6 (Conv2D)	(None, 8, 8, 256)	295168
batch_normalization_5 (Batch	(None, 8, 8, 256)	1024
leaky_re_lu_6 (LeakyReLU)	(None, 8, 8, 256)	0
dropout_6 (Dropout)	(None, 8, 8, 256)	0
conv2d_7 (Conv2D)	(None, 4, 4, 128)	295040
batch_normalization_6 (Batch	(None, 4, 4, 128)	512
leaky_re_lu_7 (LeakyReLU)	(None, 4, 4, 128)	0
dropout_7 (Dropout)	(None, 4, 4, 128)	0
flatten_1 (Flatten)	(None, 2048)	0
dense_1 (Dense)	(None, 1)	2049

Total params: 872,513
Trainable params: 870,977
Non-trainable params: 1,536

Figure 11: **Discriminator Model**

6.1 DCGANs Generated Images

Training of the GAN itself is one of its major drawbacks, which has become quite infamous for being extremely difficult: first, training a GAN is highly hyperparameter-dependent. Second, and most importantly, the loss functions (both the generator's and discriminator's) are not informative: while the generated samples may start to closely resemble the true data, approximating significantly its distribution, This behavior can't be indexed to a trend of the losses in general, which means that we cannot just run a hyperparameter optimizer using the losses and must instead iteratively tune them manually.

The implemented model has been quite promising while generating realistic looking fake eye images, as illustrated below. We started designing models for the image size 28 X 28 pixels, and worked our way up to models generating images of 128 X 128 pixels.

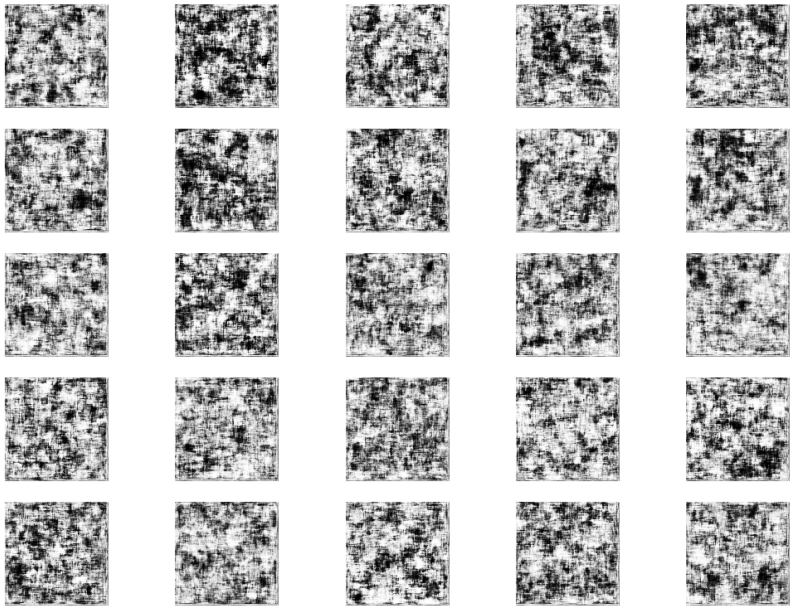

Figure 12: **Generated image samples at epoch: 0**

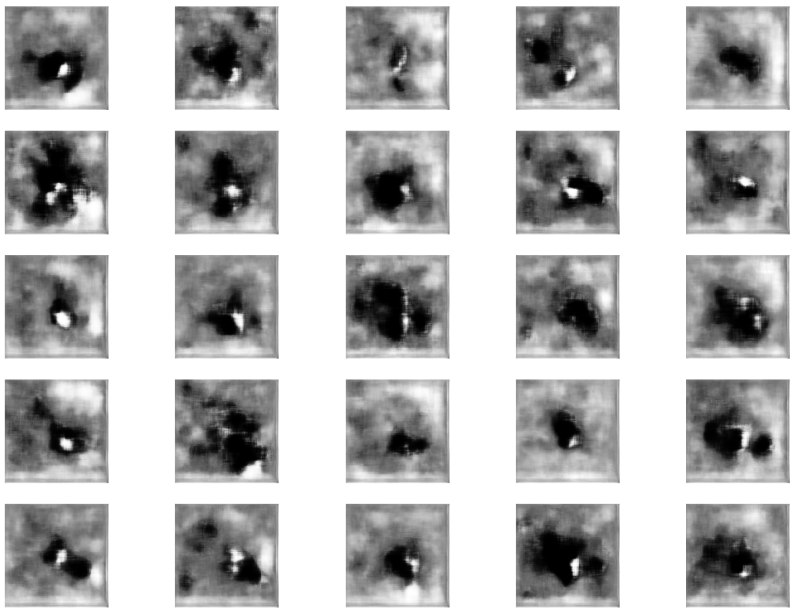

Figure 13: **Generated image samples at epoch: 100**

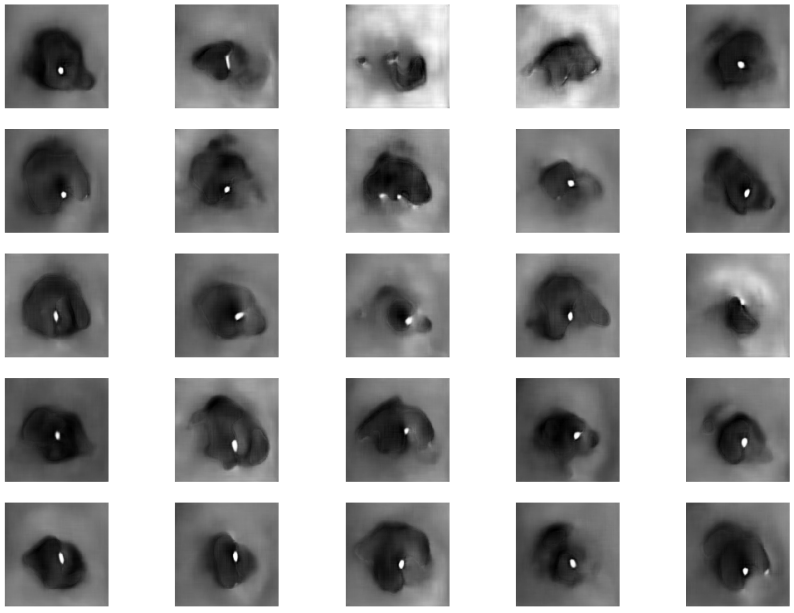

Figure 14: **Generated image samples at epoch: 1150**

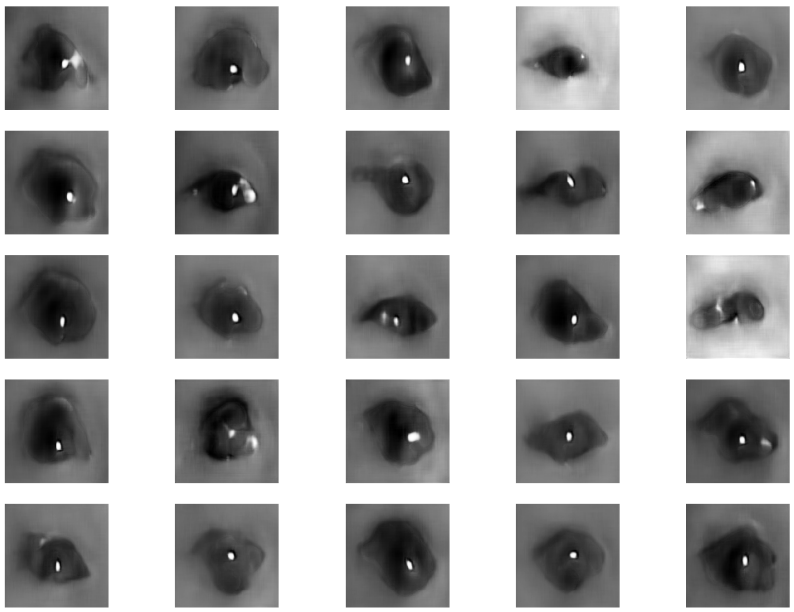

Figure 15: **Generated image samples at epoch: 2450**

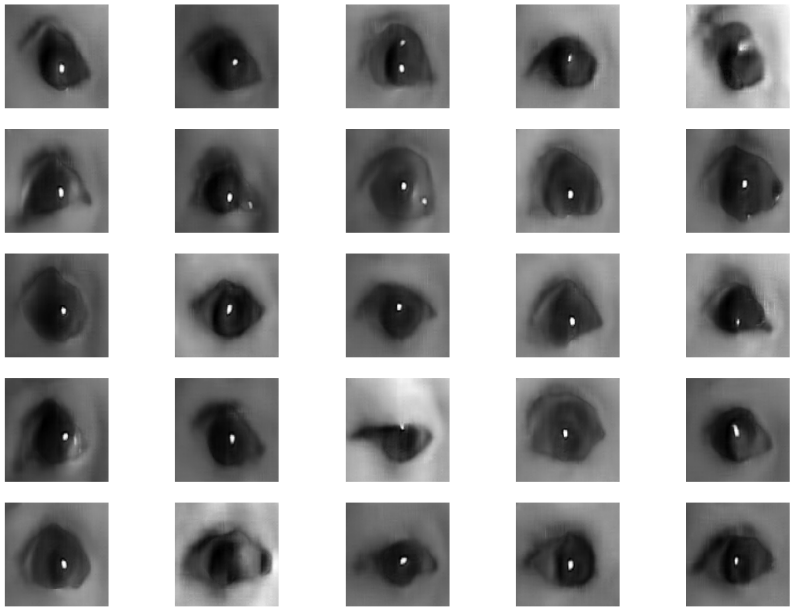

Figure 16: **Generated image samples at epoch: 4950**

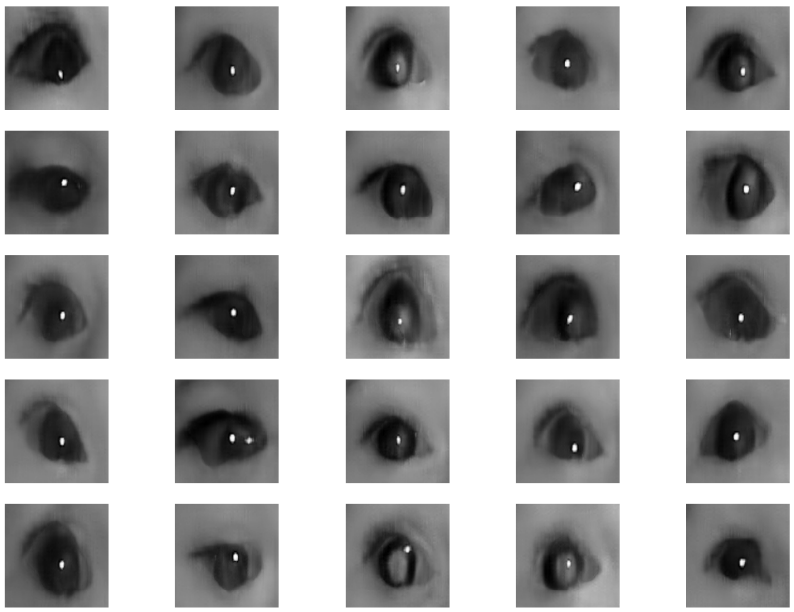

Figure 17: **Generated image samples at epoch: 6300**

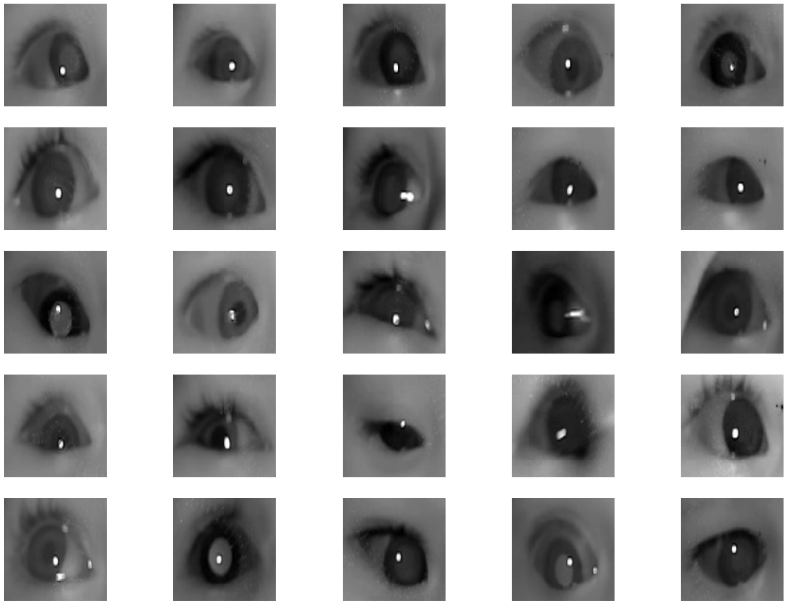

Figure 18: **Real image samples**

7 Investigating and Visualizing the internals of the Networks

We investigate the trained generators and discriminators in a variety of ways. We do not do any kind of nearest neighbor search on the training set. Nearest neighbors in pixel or feature space are trivially fooled by small image transforms. We also do not use log-likelihood metrics to quantitatively assess the model, as it is a poor metric [20].

7.1 The Latent Space

The latent vector is a a lower dimensional representation of the features of an input image. The space of all latent vectors is called the latent space. The latent vector denoted by the symbol , represents an intermediate feature space in the generator network. A generator network follows the architecture of an autoencoder which contains two networks. The first part, the encoder, encodes the input images into a lower dimensional representation (latent vector) using down-sampling. The second part, called the decoder, reconstructs the shape of the image using upsampling. The size of the latent vector is lower than the size of the input (ie, an image) of the encoder. After training a generator network we could discard the encoder part and use the latent vector to construct the generated images. This is useful because it makes the size of the model smaller.

The latent vector has a 1-dimensional shape and is usually sampled from a certain probability distribution, where nearby vectors represent similar generated images.

7.2 Interpolation

Interpolation refers to the process of finding intermediate data points between specific known data points in the space. The closer the data points the smoother the transition between these points. The simplest form of an interpolation function is a linear interpolation. Given two vectors v_1 and v_2 and N as the number of interpolated vectors, we evaluate the interpolation function as

$$F(v_1, v_2, N) = xv_2 + (1 - x)v_1, x \in \left(0, \tfrac{1}{N}, \cdots, \tfrac{N}{N}\right)$$

Note that if $x = 0$ then the first data point is v_1 and if $x = 1$ the data point which is the last is v_2. Using the value of x, we can control how the image is generated. Illustrated below are a few examples of how this works.

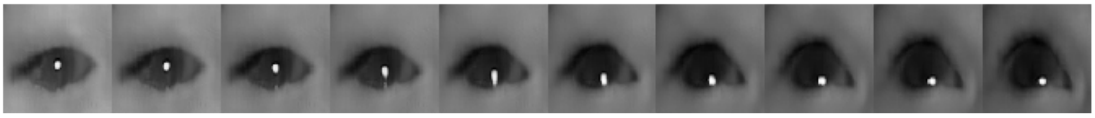

Figure 19: **Interpolated images from Forward label to Top-Left**

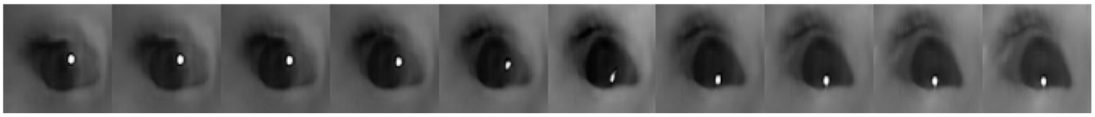

Figure 20: **Interpolated images from Bottom-Left to Top-Right**

Shape of the eye can also be manipulated using this method.

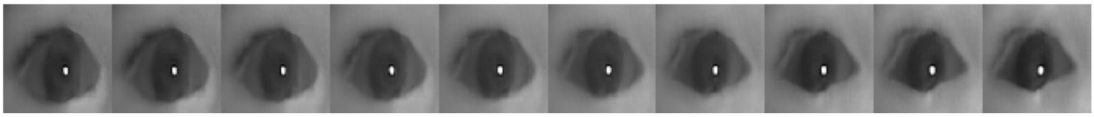

Figure 21: **Interpolated images from Forward label to Top-Left**

Similarly, Intensity, contrast and other various parameters can also be varied through latent space. This shows us that hat the space learned has smooth transitions and the eyes are modeled linearly in Z space.

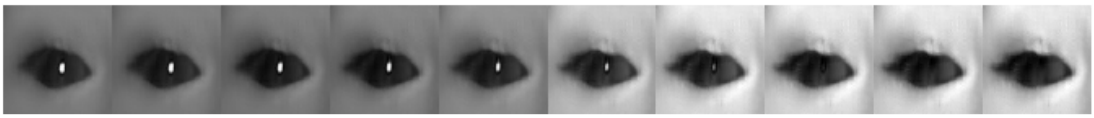

Figure 22: **Generated images with increasing contrast**

7.3 Background Hallucination

In this experiment we try to change the background of an arbitrary image while keeping the foreground the same. Note that values near zero in the latent vector mainly control the dominant class in the generated image. We can use the $f(x) = \sin(x)$ to resample different background because it preserves values near zero, so $\sin(x) \sim x$.

In the illustration below, we can see that, by varying the sinusoidal function, we can play with the background of the generated image.

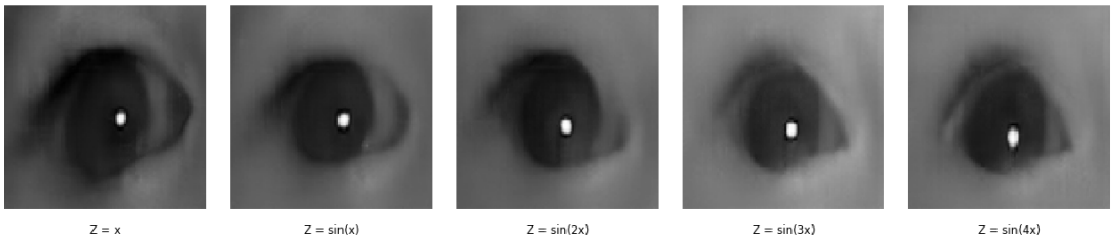

Figure 23: **Generated images with increasing contrast**

In addition to the above, we also noticed that $f(x) = \sin(x)$ can be used to refine the image in most cases.

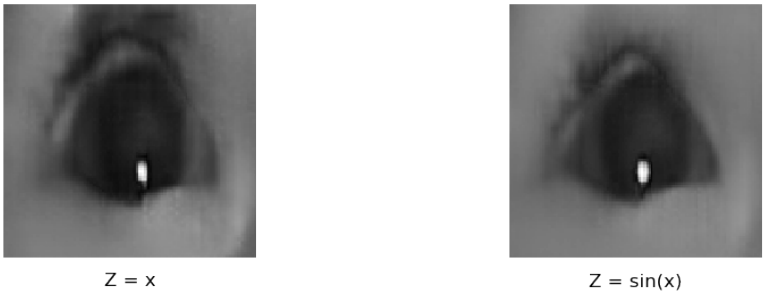

Figure 24: **Removed unintended artifacts from the generated image**

Figure 25: **Removed unintended artifacts from the generated image**

7.4 Latent Space Vector Arithmetic on generated Eye image samples

In the context of evaluating learned representations of words Mikolov et al. demonstrated that simple arithmetic operations revealed rich linear structure in representation space. One canonical example demonstrated that the vector(”King”) - vector(”Man”) + vector(”Woman”) resulted in a vector whose nearest neighbor was the vector for Queen [21]. Radford et al. investigated whether similar structure emerges in the Z representation of generators [14].

We performed similar arithmetic on the Z vectors of sets of exemplar samples for visual concepts. Experiments working on only single samples per concept were unstable, but averaging the Z vector for three exemplars showed consistent and stable generations that semantically obeyed the arithmetic.

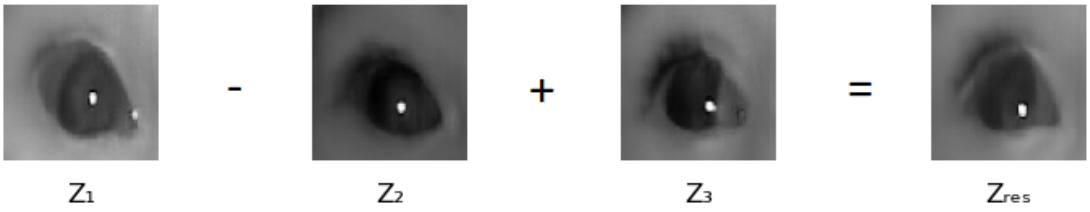

Figure 26: **Vector arithmetic for visual concepts**

From the above image, we can observe that the images generated from latent space vector Z_1 and Z_2 represent the same gaze direction (Forward), and the images generated from the vectors Z_2 and Z_3 are visually similar but represent two different gaze directions (Forward and Top-Left). Their resultant vector Z_{res} generates an eye image which is visually similar to the one generated by Z_1, but has a class value of the image generated by Z_3.

8 Conditional DCGANs for class-wise eye image generation

In GAN, there is no control over modes of the data to be generated. The conditional GAN changes that by adding the label y as an additional parameter to the generator and hopes that the corresponding images are generated. We also add the labels to the discriminator input to distinguish real images better.

This model can be modified to include additional inputs, y, on which the models can be conditioned. y can be any type of additional inputs, for example, class labels. The conditioning can be achieved by simply feeding y to both the Generator—$G(z|y)$ and the Discriminator—$D(x|y)$. The original paper on Conditional GAN used a fully connected network for both the Generator and the Discriminator and was trained on MNIST data to produce digit images [15].

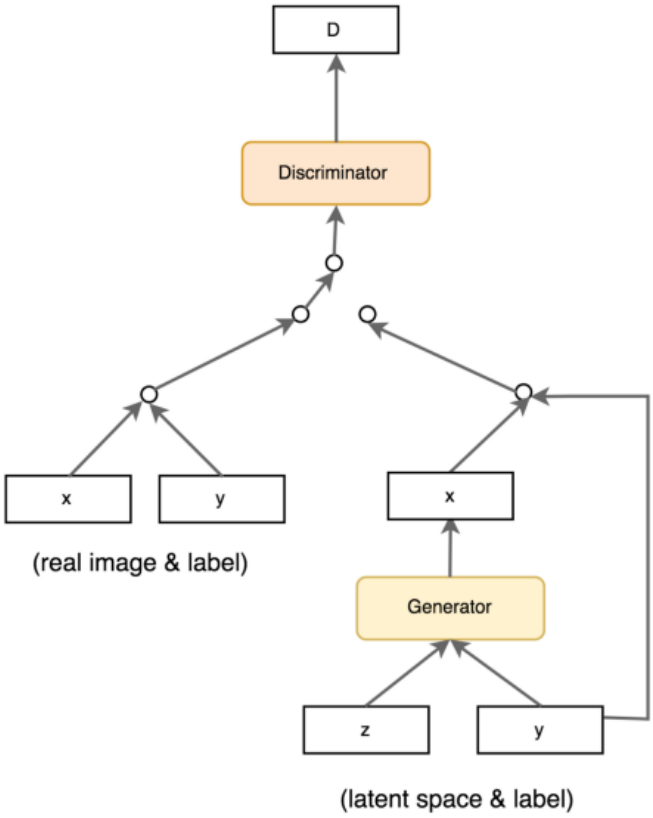

Figure 27: **Data flow for CGAN**

Source: Jonathan Hui - Towards Data Science

We trained a Conditional Deep Convolutional GAN on our eye image dataset, an upgrade over the traditional CGAN. Just like in the CGAN, we embed the class label data to the input noise/generated image. The model architectures for both the Generator and Discriminator are slight modifications over the ones we used for DCGANs before. The input data flow had to be reworked along with the training parameters. Illustrated below are few images, sampled at different epoch intervals to indicate the training process.

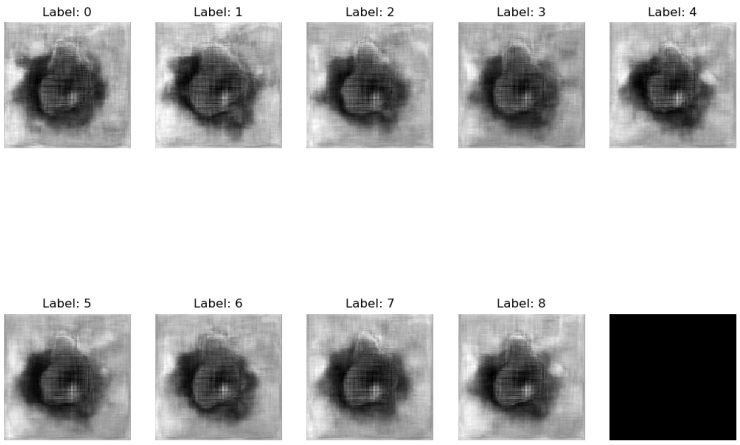

Figure 28: **Generated image samples at epoch: 200**

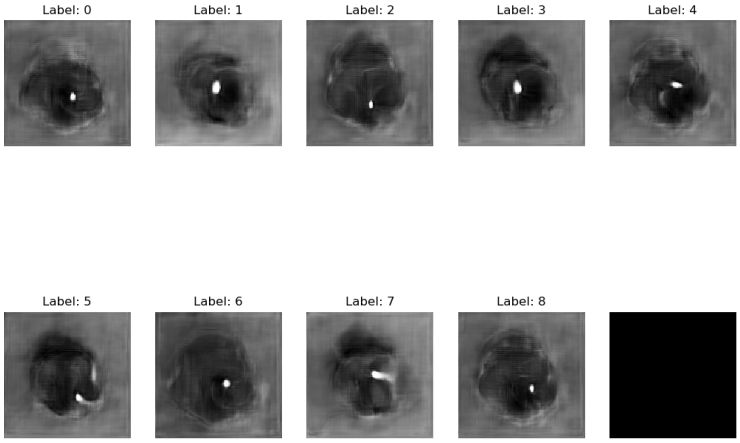

Figure 29: **Generated image samples at epoch: 5000**

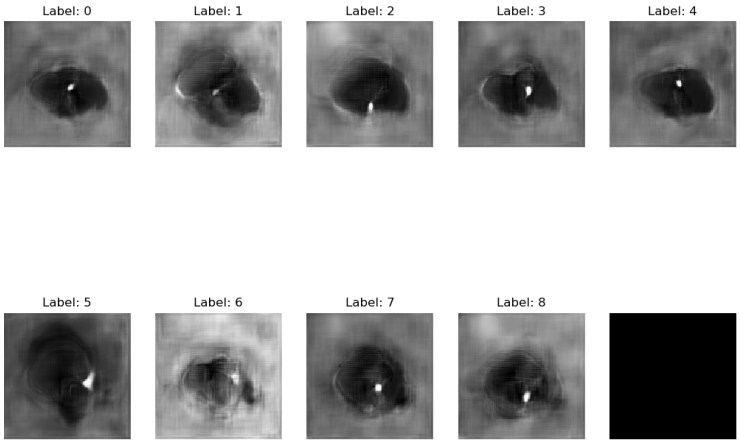

Figure 30: **Generated image samples at epoch: 10800**

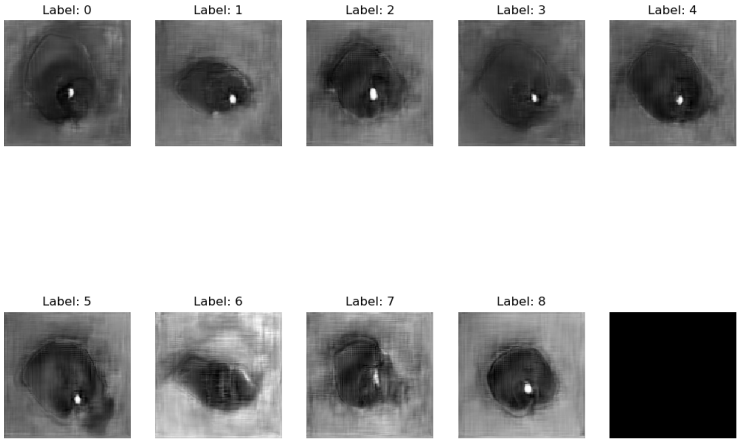

Figure 31: **Generated image samples at epoch: 19000**

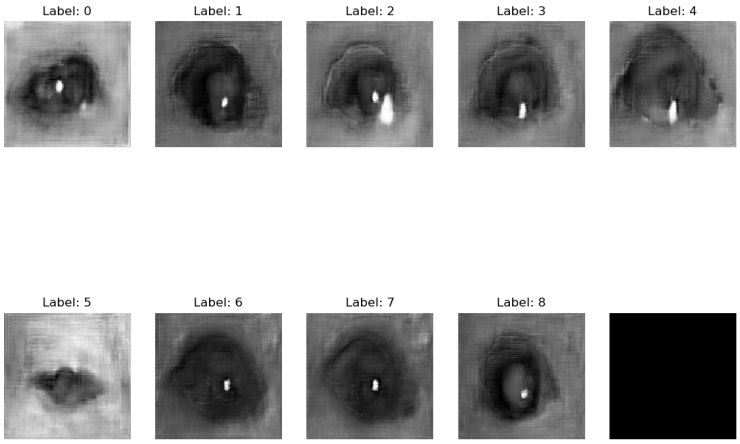

Figure 32: **Generated image samples at epoch: 26000**

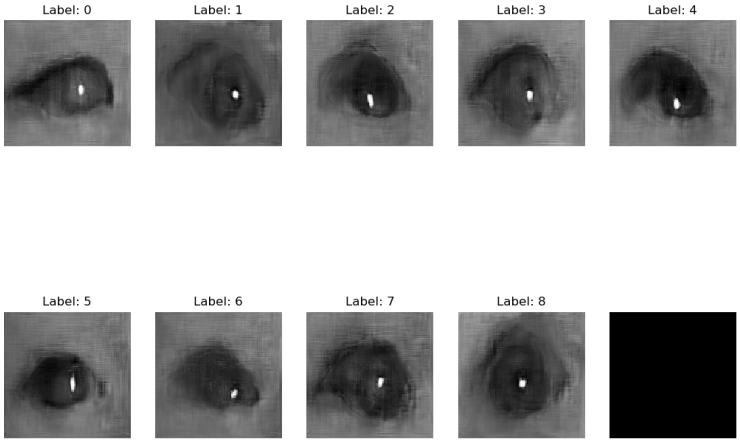

Figure 33: **Generated image samples at epoch: 31800**

9 Future Work

The course of Research and Development is detailed below

9.1 Generating realistic images of higher resolution

Development of a generative adversarial framework to generate images of higher resolution. Current model is for generating only 28 X 28 pixel images, which is quite limiting. The original images are of the resolution 100 X 60, and previous models were trained on images of resized resolution of 128 X 128.

9.2 Generating realistic images Class-Wise

Development of a generative adversarial framework to generate images representing all the 9 Gaze classes using by implementing class-conditional models like Conditional GANs or Categorical GANs.

9.3 Integration with Device Software

A new version (v4.0) of the Pediatric Perimeter desktop application is being developed on Unity in C. Integrating the Gaze prediction module into this application enables further validation of the models and moreover, helps the module to increase it's learning as more data is gathered, improving the accuracy of the models. The current python scripts have been converted to C, work remains to compile them into a DLL, without any dependencies.

References

[1] P. Satgunam, S. Datta, K. Chillakala, K. R. Bobbili, and D. Joshi, "Pediatric perimeter—a novel device to measure visual fields in infants and patients with special needs," *Translational Vision Science Technology*, vol. 6, no. 4, p. 3, 2017.

[2] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Appearance-based gaze estimation in the wild," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pp. 4511–4520, 2015.

[3] Q. Huang, A. Veeraraghavan, and A. Sabharwal, "Tabletgaze: dataset and analysis for unconstrained appearance-based gaze estimation in mobile tablets," *Mach. Vis. Appl.*, vol. 28, no. 5-6, pp. 445–461, 2017.

[4] K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. M. Bhandarkar, W. Matusik, and A. Torralba, "Eye tracking for everyone," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2176–2184, 2016.

[5] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "It's written all over your face: Full-face appearance-based gaze estimation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 2299–2308, 2017.

[6] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Mpiigaze: Real-world dataset and deep appearance-based gaze estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 1, pp. 162–175, 2019.

[7] E. Wood, T. Baltrusaitis, X. Zhang, Y. Sugano, P. Robinson, and A. Bulling, "Rendering of eyes for eye-shape registration and gaze estimation," in *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 3756–3764, 2015.

[8] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 2242–2251, 2017.

[9] R. Newman, Y. Matsumoto, S. Rougeaux, and A. Zelinsky, "Real-time stereo tracking for head pose and gaze estimation," in *4th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000), 26-30 March 2000, Grenoble, France*, pp. 122–128, 2000.

[10] R. A. Naqvi, M. Arsalan, G. Batchuluun, H. S. Yoon, and K. R. Park, "Deep learning-based gaze detection system for automobile drivers using a NIR camera sensor," *Sensors*, vol. 18, no. 2, p. 456, 2018.

[11] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L. Morency, "Openface 2.0: Facial behavior analysis toolkit," in *13th IEEE International Conference on Automatic Face & Gesture Recognition, FG 2018, Xi'an, China, May 15-19, 2018*, pp. 59–66, 2018.

[12] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 1, pp. 121–135, 2019.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial networks," *CoRR*, vol. abs/1406.2661, 2014.

[14] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[15] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014.

[16] I. J. Goodfellow, M. Mirza, A. C. Courville, and Y. Bengio, "Multi-prediction deep boltzmann machines," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 548–556, 2013.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[18] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[20] L. Theis, A. van den Oord, and M. Bethge, "A note on the evaluation of generative models," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 3111–3119, 2013.