
Latent space mapping for generation of object elements with corresponding data annotation[☆]

Shabab Bazrafkan*, Hossein Javidnia, Peter Corcoran

Department of Electronic Engineering, College of Engineering, National University of Ireland Galway, University Road, Galway, Ireland

ARTICLE INFO

Article history:

Received 14 March 2018

Available online 23 October 2018

Keywords:

Generative models

Latent space mapping

Deep neural networks

ABSTRACT

Deep neural generative models such as Variational Auto-Encoders (VAE) and Generative Adversarial Networks (GAN) give promising results in estimating the data distribution across a range of machine learning fields of application. Recent results have been especially impressive in image synthesis where learning the spatial appearance information is a key goal. This enables the generation of intermediate spatial data that corresponds to the original dataset. In the training stage, these models learn to decrease the distance of their output distribution to the actual data and, in the test phase, they map a latent space to the data space. Since these models have already learned their latent space mapping, one question is whether there is a function mapping the latent space to any aspect of the database for the given generator. In this work, it has been shown that this mapping is relatively straightforward using small neural network models and by minimizing the mean square error. As a demonstration of this technique, two example use cases have been implemented: firstly, the idea to generate facial images with corresponding landmark data and secondly, generation of low-quality iris images (as would be captured with a smartphone user-facing camera) with a corresponding ground-truth segmentation contour.

© 2018 Elsevier B.V. All rights reserved.

1. Introduction

Deep neural networks are a key driver of contemporary machine learning and artificial intelligence research and have begun to infiltrate the consumer world [1]. Advanced deep learning techniques are used to solve a wide range of long-standing problems in pattern recognition science. These approaches are famous for their power in designing and implementing regression and classification models.

Deep neural networks have shown great success when used as generative models. These models learn the distribution of a specific dataset and generate new samples from the learned Probability Distribution Function (PDF). Classical methods include Variational Bayesian and Markov Chain Monte Carlo models, which have been used to model the data distribution. Taking advantage of the neural networks non-linearity in defining the generative model goes back

to early 2000 when [2] used *tanh*, the nonlinearity of a small neural network, in modelling the data distribution. These models are usually limited to small-sized problems due to the problem complexity and they are also computationally prohibitive.

1.1. Deep neural networks as generator

With the emergence of the low-cost, high-performance hardware for training deep neural networks, it became feasible to train and test big networks and recently the generative models have also been taking advantage of deep neural networks in learning large size problems including image and sound generation.

In [3], the authors introduced two models to learn the data distribution. 1. PixelRNN which is composed of 12 2D Long Short Term Memory (LSTM) layers in which the LSTM units are applied in row and diagonal directions, namely RowLSTM and Diagonal BiLSTM respectively. 2. PixelCNN which is a fully convolutional deep neural network used to predict the conditional distribution at each pixel location. Since these models do not make use of latent space mapping, the framework proposed in this paper does not apply to them.

Variational Auto-Encoders (VAE) [4] are another approach for constructing a generative model. In this idea, the bottleneck of the auto-encoder network forms the latent space for the generative model. The encoder maps the input image to the latent space and

[☆] The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

* Corresponding author.

E-mail address: s.bazrafkan1@nuigalway.ie (S. Bazrafkan).

Table 1
Phrases used throughout the paper, their definitions and symbols.

Phrase	Definition	Symbol
Latent variable	A latent variable is a variable that is not observed but is used to describe the model or the observed data.	z
Latent space	Latent space is the space of the latent variable	\mathcal{Z} -space
Aspect of dataset	An output of any local operation on the samples of the data	$-$
Perfect generator	A generator that gives the one-to-one correspondence between samples in the \mathcal{Z} -space and every image in the database	$G()$
Perfect inverse of generator	A function that for each image in the database gives the corresponding \mathcal{Z} -space sample	$G^{-1}()$
Perfect aspect generator	Accepts the image and generates the perfect output for the specific aspect of the image	T
Aspect generator	Generates the data aspect directly from the latent space	$G_a()$
Deep neural sample generator	The generator from VAE, GAN, etc. Learns the data distribution and generates new samples. It is considered an approximation of the perfect generator.	G^*

the decoder is a generator that maps the latent space into the sample space. The main difference between VAE and ordinary auto-encoders is the constraint on the latent space distribution. In VAE, the latent space is forced to obtain the Gaussian distribution by reducing the Kullback–Leibler (KL) divergence between the latent space distribution and the Gaussian. In practice, this is done by adding a term to the loss function to optimize the KL divergence in addition to the mean square error of the auto-encoder. The downside of this approach is that the network generates blurred images due to the mean square error loss [5].

Generative Adversarial Networks (GAN) presented by [6] are another type of generative model wherein two deep neural networks (a generator and a discriminator) are engaged in a min-max game. The generator accepts samples from the latent space with a uniform distribution. A deep neural network (the generator) converts this latent sample into a signal in the shape of the original dataset. The discriminator accepts the signal from the generator and also the original samples from the database and performs a binary classification task of whether it is drawn from the database or not. Authors in [6] show that the min-max loss function of GAN decreases the Jensen–Shannon Divergence (JSD) between the generator output and the data distribution.

1.2. Proposed problem

A latent variable is a variable that is not observed but is used to describe the model or the observed data. Latent variables are said to exist in a “Latent space.” Generative models like VAE and GAN consider a latent variable for each data point and map the latent space to the data space by reducing the divergence between their output and data distribution. In this work, a new problem is defined: by knowing the generator for a database, is there a function mapping the latent space onto any aspect of the database? (For the definition of ‘aspect’, see Table 1) and if so, how can this mapping be defined? In this work, it has been shown that a deep neural network is able to accomplish this mapping and the loss function can be as simple as mean square error, i.e., there is no need to enter the divergence into the objective of the mapping anymore. Solving such a problem has several applications, including consumer electronic design and data augmentation for regression problems. To the best of our knowledge, this is the first time such a problem has been considered in the literature.

The most relevant research work is that of the Gender Preserving Generative Adversarial Network (GP-GAN) [7] where adversarial networks are exploited to synthesize faces from the landmarks. The generator sub-network in GP-GAN is based on UNet [8] and DenseNet [9] architectures while the discriminator sub-network is based on [10]. Note that the network is using a new gender-preserving loss in parallel with the perceptual loss.

Another relevant study, Age-cGAN [11], employs a GAN to generate random faces and corresponding facial metadata. The focus of Age-cGAN is identity-preserving face aging where the person’s facial attributes are altered to age his/her face while the identity is preserved. The generator and discriminator sub-networks of Age-cGAN have the same architecture as [12]. This network can be used to synthesize augmented facial datasets incorporating aging of subjects.

The rest of the paper is organized as follows. Section 2 presents the proposed method in detail. Two individual observations are illustrated in Section 3 to generate random samples and their corresponding aspects. Finally, the conclusion and future works are presented in Section 4.

2. Proposed method

2.1. Problem definition

The definitions of the phrases used in this work are given in Table 1.

Problem definition: For a given database, if there is a perfect generator mapping a latent space into the data space (since the generator is a local operation on the latent variable) by considering the data processing inequality, one can argue that the latent space includes all the information of the database. This indicates that any aspect of the dataset is extractable from the latent space alone. The problem considered in this work is to find a local operation that maps the latent space to the aspect space for a given generator and aspect. To the best of our knowledge, this is the first time such a problem has been defined. Solving this problem can facilitate high-speed implementation of regression solutions (in the presence of a real-time inverse of the generator) which is a valuable solution in consumer electronic design. Another application will be data augmentation for regression problems. More discussion on applications of the proposed idea is given in Section 4.

Naïve solutions: Suppose that there is a deep neural generator G^* mapping a uniformly distributed latent variable z_r^* to the data point x_r^* . This mapping is shown by $x_r^* = G^*(z_r^*)$. A new generator could be trained just for the specific data aspect, but training a new generator specifically for a feature set of the database would map an entirely different latent space (let’s call it \mathcal{Z}_f -space) to the feature space. It is not trivial to find the correspondence between the \mathcal{Z}_f^* -space and \mathcal{Z}_f -space. The other naïve solution would be to train a single model to learn the data and the feature distribution at the same time. This solution comes with complexities in implementation procedures. Since the feature and the data space might not be from the same class and since the dimension and the objective function for each output could be totally different, the single model solution would not converge to a reasonable output.

Proposed solution: Inverse transform sampling theorem declares that by knowing the probability distribution of a random variable x_r , there is a transformation, mapping a uniformly distributed random variable z_r into the space of x_r . This theorem requires the cumulative distribution function of x_r (shown as $F(x_r)$) to be known. In this approach, it has been shown [13] that the distribution of the output of the inverse of the cumulative distribution function for z_r is same as the distribution of x_r . In other words $p(F^{-1}(z_r)) = p(x_r)$, where p is the probability distribution function.

Since the generator network accepts a uniform random variable and transforms it to the image space, this network is learning an approximation of the data distribution. In fact, the network acts like the inverse of the cumulative distribution function of x_r . At the same time, from the signal processing inequality principle, any local operation on the data does not insert any new information into the data. This provides justification for the ability of a neural network to learn the distribution of any aspect of the data for a given generator (as far as the aspect is provided by applying local operations on the data). Our results, firstly, demonstrate that it is possible to train a network that learns the distribution of landmark annotations for a face generator and secondly, another network was trained that learns the segmentation for an iris generator. More information on these two experiments are provided in Section 3.

Suppose that there is a perfect generator, mapping \mathcal{Z}_r -space to the data space, $x_r = G(z_r)$. And the perfect inverse of the generator is shown by G^{-1} where for each image x_r in the database, the inverse of the generator gives the corresponding \mathcal{Z}_r -space sample $z_{data} = G^{-1}(x_r)$. In fact, the inverse of the generator acts as the cumulative distribution function of x_r . Therefore, the latent space will be reconstructed from applying the inverse of the generator to all the samples in the database.

The output space of the inverse of the generator is called \mathcal{Z}_{data} -space which is a subspace of \mathcal{Z}_r -space, but its distribution can be non-uniform in real life applications due to finite number of samples available in the database. This \mathcal{Z}_{data} -space and the perfect generator G contain all the information about the database, i.e., by knowing the \mathcal{Z}_{data} -space and the perfect generator G , one can recreate the database. Since all the aspects of the database can be extracted using local operations on the data itself, considering the signal processing inequality, no further information is needed to produce any aspect of the database if \mathcal{Z}_{data} -space and G are known. In fact, since G is a local operation itself, any aspect of the database is extractable solely from knowing \mathcal{Z}_{data} -space.

2.2. Mapping to the aspect-space

In this work, the aspect of a database is defined as the output of any local operation on the data samples. For example, the facial landmark detector gives an aspect (the landmark positions) of its input image or an iris segmentor returns an aspect (the binary segmentation map) of its input iris image. Signal processing inequality principle states that the information inside a database will not increase by applying any local operation on it. For example, the information inside the landmarks of a face is obviously less than the information that is carried by the corresponding face image. And as stated earlier, all the information of the database is extractable solely from knowing \mathcal{Z}_{data} -space. This means that any aspect of the data could be derived from just \mathcal{Z}_{data} -space.

Suppose that for a given database there is a perfect aspect generator, T , which accepts the image \mathbf{x}_{data} and generates the perfect output for the specific aspect of the image $T(\mathbf{x}_{data})$. This generator can be a human, drawing the facial landmarks or a super computer extracting features from an image. Note that they are local operations applied to an input sample. Suppose that these calculations

are simple enough to be estimated by a non-linear network G_a . Since all the database information is already present in the \mathcal{Z}_{data} -space, this network is able to map the \mathcal{Z}_{data} -space to the aspect space. To train the aspect generator network the mean square error loss function between G_a and the perfect aspect generator T is decreased given by:

$$loss = \mathbb{E}_{z \sim p_{\mathcal{Z}_{data}}(z)} \{ (G_a(z) - T(G(z)))^2 \}, \quad (1)$$

where $p_{\mathcal{Z}_{data}}(z)$ is the probability distribution of \mathcal{Z}_{data} . This loss function could be re-written as:

$$loss = \int_{z \sim p_{\mathcal{Z}_{data}}(z)} p_{\mathcal{Z}_{data}}(z) (G_a(z) - T(G(z)))^2 dz. \quad (2)$$

Since $p_{\mathcal{Z}_{data}}(z)$ is positive for every z in \mathcal{Z}_{data} -space and the second term in the integration is always positive as well, the minimum of this integral is zero and is achieved if and only if:

$$G_a(z) = T(G(z)), \quad \forall z \text{ in } \mathcal{Z}_{data}\text{-space}. \quad (3)$$

The interesting part of the proposed method is that while the perfect generators on the right side of Eq. 3 are potentially very complicated, the aspect generator $G_a(z)$ can be small and simple. This is understandable by considering that the perfect aspect generator T , needs to compress the information of the $G(z)$ and rule out the unnecessary information generated by the perfect generator G , in order to produce the desired aspect. But on the left side of the equation, the aspect generator G_a , bypasses all the complexity of T and G and maps the latent space \mathcal{Z}_{data} -space to the aspect space. Adding any terms to reduce the divergence between the distribution of the aspect generator G_a and perfect aspect generator T , is unnecessary since the latent space \mathcal{Z}_{data} -space where the samples are drawn from is already learnt by the inverse of the generator.

Generating random samples drawn from a specific distribution is one of the most appealing application of Deep Neural Networks which became feasible by introducing the deep generative models specially GAN. In the original GAN idea, the network learns the data distribution and is able to randomly generate samples from the same distribution. In this work, this idea is extended by learning the latent space for GAN and map it to any aspect of the data.

The main novelty of the proposed method is to take advantage of the learned latent space (known as \mathcal{Z}_{data} -space) to train a second generator G_a which learns any aspect of the database from the latent space without the presence of any intermediate processing unit. This is while in the inference stage the two generators G and G_a are blind to each other, e.g., these two generators accept the same latent space sample and the aspect generator G_a provides output matches the output of the original generator G . Being able to generate a random sample alongside with its aspect have several applications including augmentation of data and expand a database containing the images and ground truth for each sample. Another contribution of this work is the versatility of the method. It is applicable to any aspect of the database. Eqs. 1 to 3 hold for any data type. In the observations conducted in Section 3, the aspects are point sets in one of the cases and a binary map in the second one. As far as the generator converges and the aspect is extractable by a local operation, the proposed method maps the latent space into the aspect space. Training a separate network just for generating the aspect takes an essential part in the versatility of the method. To the best of our knowledge this is the first time this problem is considered and solved.

In the following section, the results of the proposed method are presented for a face generator when the aspect is the facial landmarks and another observation is done for low-quality iris generation while the aspect is considered to be the binary map of the iris.

3. Results and simulations

In this section, two different sets of data and various aspects of these datasets are presented. At first, a generator is trained on CelebA [14] database and the aspect of this database is a 49 point facial landmarks [15] generated for each image. The second simulation is done on an augmented version of the Bath800 [16] and CASIA1000 [17] iris database, while the aspect is considered to be the iris segmentation map. These observations are described in detail in the following sections.

The Boundary Equilibrium Generative Adversarial Network (BEGAN) [18] scheme has been used to train the deep neural generator. In the BEGAN framework, the generator is similar to the generator in the original GAN method, but the discriminator is a deep auto-encoder and the loss function reduces the Wasserstein distance between the error of the auto-encoder for generated and original data. The reasons for selecting BEGAN are the simplicity of implementation and the high-quality results of the generator. The BEGAN implementation is described in detail in Appendix A.

Since the generator learned the distribution of the database, it can map an N_z dimension random vector onto an interpolation point of the database distribution where N_z is the dimensionality of the latent space. The reverse applies as well. Having an image from the dataset, one can estimate the sample in the Z_r -space, which, when fed to the generator, will generate the image. The method used in this work is similar to one presented in [18] wherein the sample from the Z_r -space is approximated by optimizing the error function:

$$err = |x_r - G^*(z_r)|, \quad (4)$$

where x_r is the sample image and G^* is the generator function. In all examples, the ADAM optimizer is used to solve the problem, with learning rate, β_1 , and β_2 equal to 0.1, 0.9, and 0.999, respectively. So the inverse of the generator accepts an image and produces the latent value for the given sample. If one applies the inverse of the generator to all the samples in the database, the output is a space of latent variables, which is a subset of Z_r -space called Z_{data} -space. As described before, this new space does not need to be uniformly distributed. One can also call this space the learnt latent space since it is derived from the inverse of the generator. In our experiments, this learned latent space is used to produce the aspect of the database.

3.1. Experiment 1: face + landmarks

3.1.1. Database

The CelebA dataset [14] consisting of 202,599 original images with 40 unique attributes is used for training the GAN framework. The OpenCV frontal face cascade classifier [19] is used to detect facial regions, which are cropped and resized to 128×128 pixels. Initial landmark detection is performed using the method presented in [15] due to its ability to be effective on unconstrained faces. Authors in [15] have augmented the original cascade regression framework of [20] by proposing an incremental algorithm for cascade regression learning. This method personalizes the Supervised Descent Method (SDM) [21] for facial point localization, initializing the SDM offline on a large database of faces and using newly tracked faces to update it incrementally. The detector in [15] uses a discriminative 3D facial deformable model fitted to the 2D image. The detector was trained on the 300 W dataset [22]. It estimates a set of 49 landmarks defined by the contours of eyebrows, eyes, mouth, and the nose as shown in Fig. 1.

3.1.2. Generating data, inverse of the generator and aspect mapping

Using the BEGAN framework, a generator has been trained on the CelebA database. The latent space is considered to be 64 di-

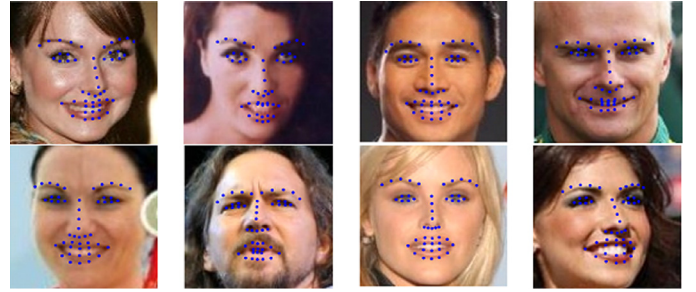


Fig. 1. Facial landmark detection from the discriminative deformable model [15].

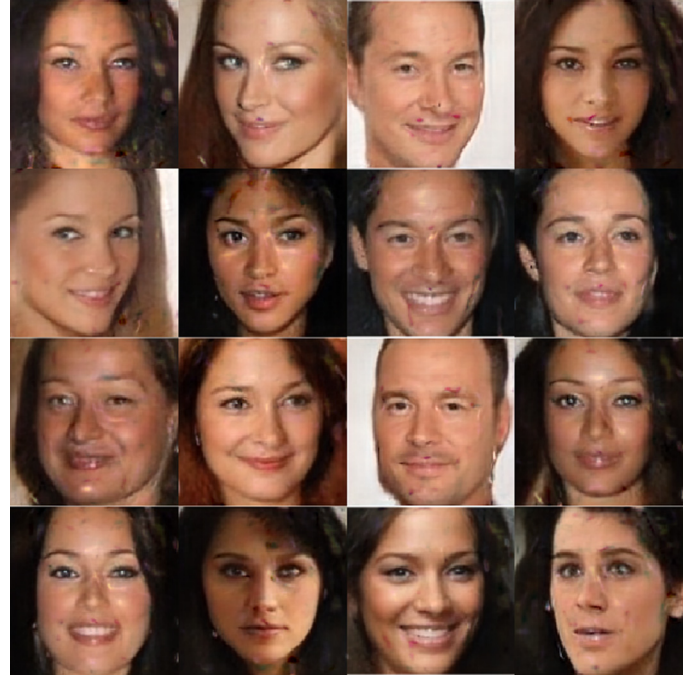


Fig. 2. Generating a random set of images using BEGAN framework on CelebA dataset.

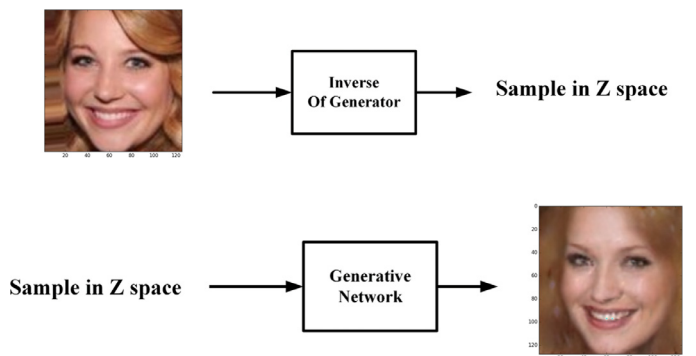


Fig. 3. The inverse of the generator can produce an accurate estimate of the latent value corresponding to each image in the database.

mensions. The ADAM optimizer was used with learning rates β_1 , and β_2 equal to 0.0001, 0.5, and 0.999, respectively. BEGAN was trained with the Lasagne library and Theano library in Python. A Geforce 1080ti desktop GPU was used to train the generator. Some randomly generated samples are shown in Fig 2.

The inverse of the generator (Eq. 4) is then applied to all images in the dataset. All the outputs of the inverse of the generator make the Z_{data} -space. Fig. 3 shows how well the inverse of the genera-

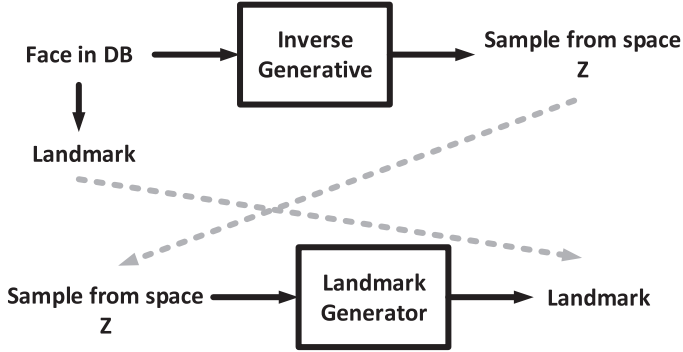


Fig. 4. Proposed method; the landmark generator maps \mathcal{Z} -space onto a “learned” landmark space.

Table 2

The architecture of the landmark generator. It is a small, fully connected deep neural network.

Layer name	Layer kind	Number of nodes	Activation
Input Layer	Input	64	—
First Hidden	Fully connected	128	RELU
Second Hidden	Fully connected	128	RELU
Third Hidden	Fully connected	128	RELU
Output Layer	Fully connected	98	Sigmoid

tor works. In this figure, the estimated latent sample is fed to the generator and a very similar image is generated at the output.

Knowing all the samples in the \mathcal{Z}_{data} -space and also landmarks for each image (the output of perfect aspect generator T described in Section 2.1), an aspect generator is trained, mapping the \mathcal{Z}_{data} -space to the landmark space. See Fig 4.

The architecture of the Landmark Generator network trained to approximate the landmarks is shown in Table 2. This network accepts 64-dimensional samples from the \mathcal{Z}_r -space and the output is a set of 98 dimensions corresponding to 49 2D landmark points.

The loss function for this network is the Mean Square Error given by:

$$loss = \frac{1}{B_N \times 98} \sum_{k=1}^{B_N} \sum_{i=1}^{98} (o_i - t_i)^2, \quad (5)$$

wherein o_i is the i 'th output of the output layer, t_i is the i 'th target value, and B_N is the batch size which is set to 16. The ADAM optimizer is used to train the network with learning rates β_1 , and β_2 equal to 0.0003, 0.9, and 0.999, respectively. The training was done for 1000 epochs using all data. No validation and test set were used in the method.

Adding validation and test sets made the results less accurate, since after reducing the number of samples in the training set, the network was blind to some examples and could not reconstruct the landmark distribution accurately. To the best of our knowledge, this is the first attempt to generate samples and their corresponding landmarks at the same time.

In the feedforward step shown in Fig 5, a uniformly distributed random vector is fed concurrently into the Generator (from BEGAN) and the Landmark Generator. The first generates a random interpolated face, and the Landmark Generator provides a 2D set of landmarks corresponding to the generated face.

Fig. 6 shows example results. Initial results show the Landmark Generator has learned to map a set of landmark points from the same \mathcal{Z}_r -space as the facial generator, with good generalization across varying pose & illumination conditions.

In [18], the authors investigate the continuity of the face distribution given by the BEGAN face generator by feeding it a random set of numbers, interpolating two samples in \mathcal{Z}_r -space and observ-

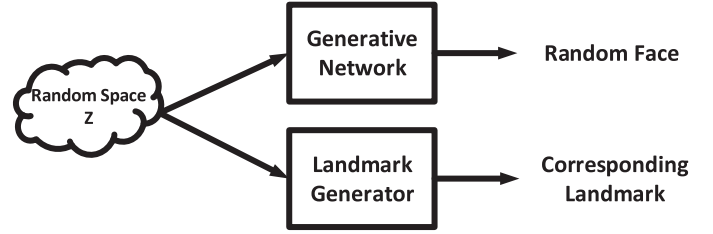


Fig. 5. Feedforward of the proposed method. The uniformly distributed random vector is fed to the Generative network given by BEGAN, and the Landmark Generator from the proposed method produces the landmark positions for the generated face image.

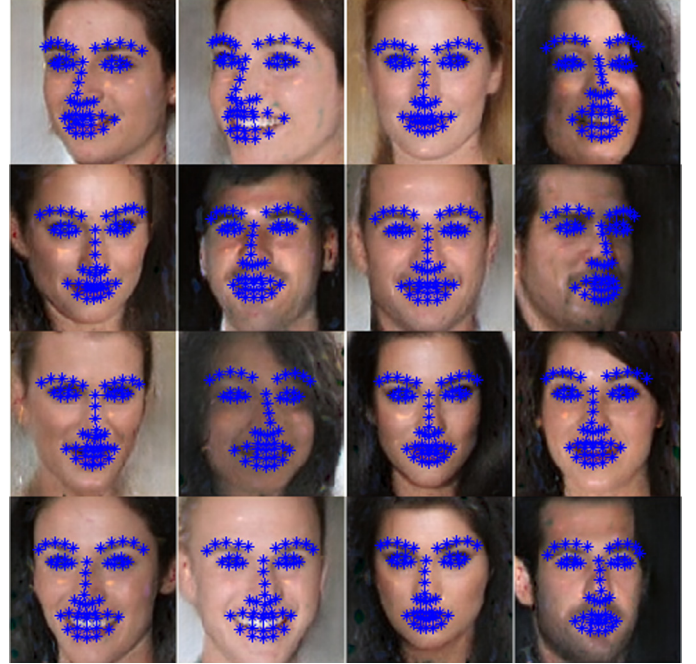


Fig. 6. Random generated faces and their corresponding landmarks.

ing the gradual changes from one face to another. To investigate the continuity of the landmark estimator distribution, the same approach is used. The z_r vectors for a given image and its mirror image are estimated using the inverse generator method and 14 interpolation points between the two \mathcal{Z}_r -space samples are fed into the face generator and Landmark Generator networks. One example result from this experiment is shown in Fig. 7. This figure shows that the landmark distribution estimated by the Landmark Generator network is smooth in the \mathcal{Z}_r -space. More results, including illumination variation and pose variation are presented in a video¹ that is generated by smoothly moving the latent variable to make the face and landmarks.

3.2. Experiment 2: Iris + segmentation

3.2.1. Dataset

In this experiment, two iris databases (Bath800 and CASIA1000) have been used to learn the generator. Bath800 is made of 31,997 iris images with a resolution of $[1280 \times 960]$ taken from 800 individuals and CASIA1000 has 20,000 Near Infrared images with a resolution of $[640 \times 480]$. All these images are resized to $[128 \times 96]$ in this experiment. None of these databases are provided with ground truth, but since they are high-quality datasets

¹ <https://youtu.be/PWdT3Q5T5U8>



Fig. 7. Interpolating in the \mathcal{Z}_f -space between a face and its mirror image indicates strong continuity for both BEGAN and the landmark generator.

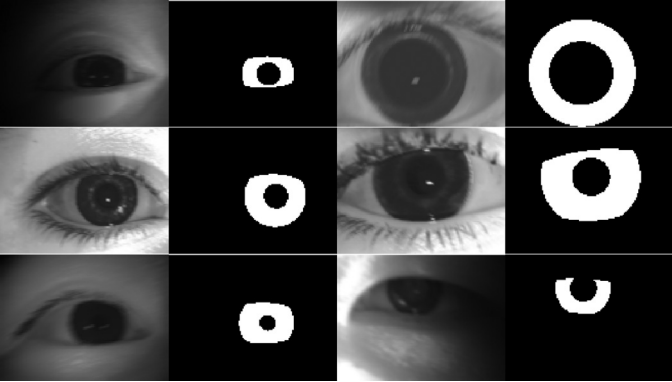


Fig. 8. Iris samples after applying augmentation and their corresponding segmentation map.

taken in highly constrained conditions, any industry standard segmentation tool will give a reasonably high-quality segmentation result. In this work, the segmentation provided by a commercial iris segmentation tool (MIRLIN [23]) is treated as the ground truth for the segmentation task. To increase the number of samples in the database, several augmentation techniques have been applied to the original dataset, including contrast reduction inside and outside of the iris region and adding shadow and motion blur. For a detailed description of the augmentation process, see [24]. Some samples of the database after applying the augmentation and their corresponding ground truth maps are shown in Fig 8.

After the augmentation process, there are 262 K samples in our training set. This database is designed initially to train a deep neural network, segmenting low-quality iris images [24]. In this section, it is used to show that the aspect of the dataset could be something more than points or features of the image. In fact, it can be the same size as the image like a binary map.

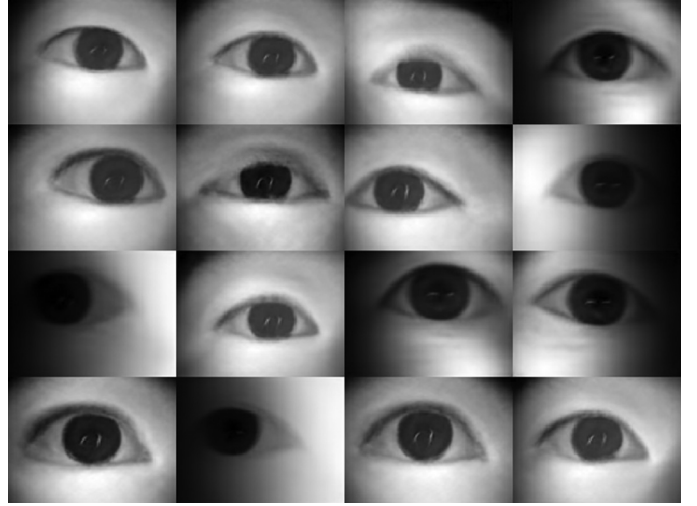


Fig. 9. The random samples drawn from the generator trained using the BEGAN method on the low-quality iris database.

3.2.2. Generating data, inverse of the generator and aspect mapping

The method to train the deep neural generator is exactly similar to the previous experiment. The BEGAN scheme has been used to learn the data distribution for the low-quality iris database. The only difference is the size of the images where in the previous experiment, the face images were 128×128 , but in this experiment, the images are 128×96 . The latent space is 64 dimensions. The ADAM optimizer was used with learning rate, β_1 , and β_2 , equal to 0.0001, 0.5, and 0.999, respectively. The BEGAN model was trained with the Lasagne library and Theano library in Python. A Geforce 1080ti desktop GPU was used to train the generator. Some randomly generated samples are shown in Fig 9. The next step is to apply the inverse of the generator to all samples in the dataset. The inverse of the generator accepts an image and estimates its corresponding latent space sample. After applying the inverse of the generator to all samples in the dataset, the \mathcal{Z}_{data} -space values for all the samples are acquired.

In this experiment, the aspect of the dataset is the binary segmentation map of the iris image. The perfect aspect generator T is the iris segmentation tool (MIRLIN [23]) described in the previous section. The architecture of the aspect generator network (mapping the latent space to the aspect space) is exactly similar to the generator in BEGAN, as described in Appendix A.

The loss function for the aspect generator is the mean square error given by:

$$loss = \frac{1}{B_N \times 96 \times 128} \sum_{k=1}^{B_N} \sum_{i=1}^{96} \sum_{j=1}^{128} (o_{ij} - t_{ij})^2, \quad (6)$$

where o_{ij} is the (i, j) 'th pixel of the output layer, t_{ij} is the (i, j) 'th pixel of the target (iris segmentation map), and B_N is the batch size, which is set to 16. The ADAM optimizer is used to train the network with the learning rate, β_1 , and β_2 , equal to 0.0003, 0.9, and 0.999, respectively. The training was done for 1,000 epochs using all data. No validation and test set were used in the method. The feedforward model is shown in Fig 10.

The results of this experiment are shown in Fig 11. In this figure, the iris image is generated by the generator trained in the BEGAN method and the segmentation maps are generated using the proposed aspect generation method.

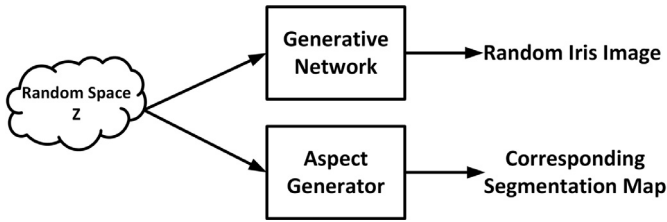


Fig. 10. The feedforward model for generating a random low-quality iris image and its corresponding segmentation map.

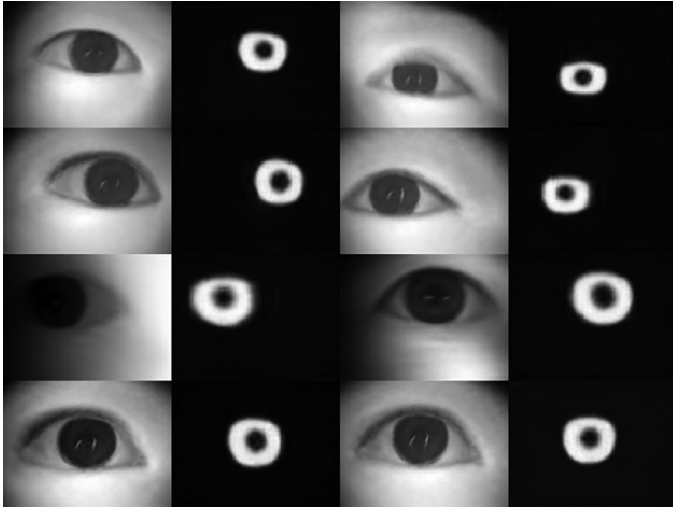


Fig. 11. Randomly generated low-quality iris images and their corresponding segmentation maps. First and third columns are randomly generated iris images; second and fourth columns are their corresponding segmentation maps created by the proposed framework.

In order to investigate the continuity of the mapping for both the iris and the segmentation, a video² illustrates the smooth changes in both the iris samples and the segmentation. The sequences are created by smoothly moving the latent sample in the \mathcal{Z}_r -space.

4. Conclusion and future works

One of the most amazing applications of the deep neural networks is to learn the data distribution and draw new samples from the learned distribution. VAE and GAN are two successful implementations of such an application. In each of these methods, the objective is to reduce the divergence of the generator output and the original data. These methods also take advantage of using a latent space that gives an opportunity to manipulate data and also learn any aspect of the database straight from the latent space.

After training the generator, it can be considered as a deterministic local nonlinear operation that maps the latent space onto the data space. Considering the data processing inequality, any local operation on the data set cannot inject extra information to the data. This means that all information of the individual samples is already in the latent space. This explains why any aspect of the database can be extracted straight from the latent space.

In this work, it has been shown that the previous statement is true and different aspects of the database, landmarks for the face and segmentation map of an iris image can be mapped from the latent space.

There are several applications for the presented framework. Since the latent space is smaller than the actual data space, this method can be used as a compression method. The feature extraction from latent space is fast, which is useful in designing fast consumer electronic devices. The other application is using the generative models as an augmentation technique. Data augmentation is a crucial step for modern machine learning frameworks, including deep learning approaches. New deep neural networks need a large number of samples to be trained in order to avoid overfitting. The augmentation process introduces a certain amount of uncertainty into the database, which helps the network prevent overfitting and generalizes the results. To the best of our knowledge, augmentations presented for regression problems are all applied in the image space. These operations include flipping, rotating, manipulating the contrast and illumination of the image, and applying distortions to the image. The framework presented in this work can utilize the data and ground truth generation in latent space. Our observations show that the mapping for both the data generator and aspect generator is continuous and smooth in the latent space. This gives the opportunity to generate a large number of samples and their corresponding ground truth, thus expanding the database by introducing more variations through the generation of multiple intermediate samples. Future work will include the investigation of the influence of the generative augmentation technique in training regression networks.

Acknowledgments

The authors would like to thank Joseph Lemley and Kimberly Sowell for their helpful comments.

The research work presented here was funded under the Strategic Partnership Program of Science Foundation Ireland (SFI) and co-funded by SFI and FotoNation Ltd. Project ID: 13/SPP/I2868 on “Next Generation Imaging for Smartphone and Embedded Platforms”.

Portions of the research in this paper use the CASIA-IrisV4 collected by the Chinese Academy of Sciences’ Institute of Automation (CASIA).

Appendices

A. Boundary Equilibrium Generative Adversarial Networks

In this work, the Boundary Equilibrium Generative Adversarial Network (BEGAN) presented in [18] is implemented to train a generator. In this approach, the discriminator network is an auto-encoder and the generator architecture is the same as the decoder part of the discriminator. The encoder and decoder parts are shown in Fig. A.1 and Fig. A.2, respectively.

In the encoder, all kernels are 3×3 and ELU [25] nonlinearity is used in all layers apart from the red layers where the kernel size is 1×1 and no nonlinearities are employed. Also no non-linearity is applied to the fully connected layers. In the decoder network, all convolutional layers have 64 channels, while in the encoder, the number of the channels is gradually increased to 128, 192, and 256 after each pooling layer.

Suppose that x is the real data coming from the database, z is a sample from the uniformly distributed random space \mathcal{Z} , \mathcal{D} is the auto-encoder function with the loss defined by:

$$\mathcal{L}(v) = |v - \mathcal{D}(v)|^2, \quad (\text{A.1})$$

where v is the input to the auto-encoder. The objectives for BEGAN given by [18] are:

² <https://youtu.be/BY9cVZPmgRU>

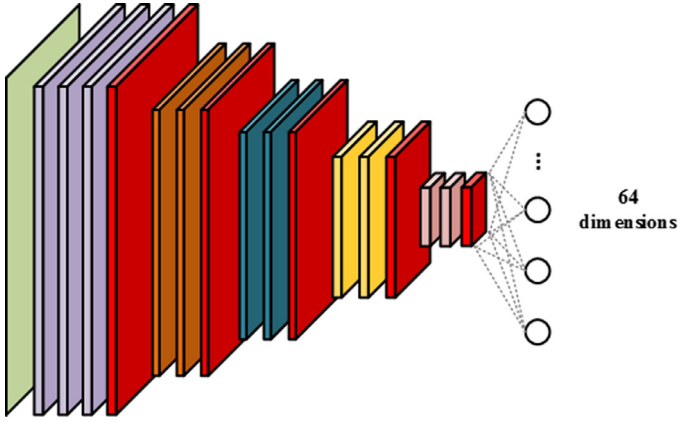


Fig. A.1. Encoder architecture for BEGAN.

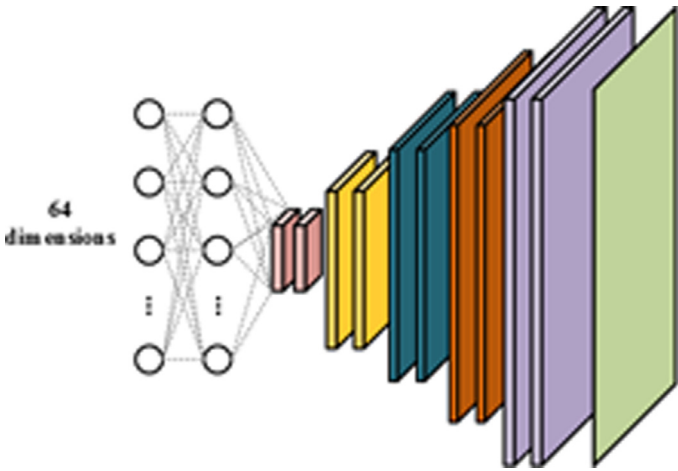


Fig. A.2. Decoder architecture for BEGAN.

$$\begin{cases} \mathcal{L}_D = L(x) - k_t \cdot L(G(z_D)), & \text{for } \theta_D \\ \mathcal{L}_G = L(G(z_G)), & \text{for } \theta_G \\ k_{t+1} = k_t + \lambda_k (\gamma \mathcal{L}(x) - \mathcal{L}(G(z_G))), & \text{for each training step } t \end{cases} \quad (\text{A.2})$$

where \mathcal{L}_D is the discriminator loss, \mathcal{L}_G is the generator loss, $G(v)$ is the output of the generator for input vector v , γ is the equilibrium hyper parameter set to 0.5 in this work, and λ_k is the learning rate for k . The ADAM optimizer is used with learning rates β_1 , and β_2 , equal to 0.0001, 0.5, and 0.999, respectively. BEGAN was trained with the Lasagne library on the top of Theano library in Python.

References

- [1] J. Lemley, S. Bazrafkan, P. Corcoran, Deep learning for consumer devices and services: Pushing the limits for machine learning, artificial intelligence, and computer vision, *IEEE Consum. Electron. Mag.* 6 (2017) 48–56, doi:[10.1109/MCE.2016.2640698](https://doi.org/10.1109/MCE.2016.2640698).
- [2] H. Valpola, J. Karhunen, An unsupervised ensemble learning method for non-linear dynamic state-space models, *Neural Comput* 14 (2002) 2647–2692.
- [3] A. van den Oord, N. Kalchbrenner, K. Kavukcuoglu, Pixel recurrent neural networks, ArXiv:1601.06759 (2016).
- [4] D.P. Kingma, M. Welling, Auto-encoding variational bayes, ArXiv:1312.6114 (2013).
- [5] K. Frans, Variational autoencoders explained, (2016).
- [6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, *Adv. Neural Inf. Process. Syst.* (2014) 2672–2680.
- [7] X. Di, V.A. Sindagi, V.M. Patel, GP-GAN: Gender preserving gan for synthesizing faces from landmarks, ArXiv:1710.00962 (2017).
- [8] O. Ronneberger, P. Fischer, T. Brox, U-Net: convolutional networks for biomedical image segmentation, in: N. Navab, J. Hornegger, W.M. Wells, A.F. Frangi (Eds.), *Med. Image Comput. Comput. Interv. – MICCAI 2015 18th Int. Conf. Munich, Ger. Oct. 5–9, 2015, Proceedings, Part III*, Springer International Publishing, Cham, 2015, pp. 234–241, doi:[10.1007/978-3-319-24574-4_28](https://doi.org/10.1007/978-3-319-24574-4_28).
- [9] G. Huang, Z. Liu, L.v.d. Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: 2017 IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 2261–2269, doi:[10.1109/CVPR.2017.243](https://doi.org/10.1109/CVPR.2017.243).
- [10] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, ArXiv:1611.07004 (2016).
- [11] G. Antipov, M. Baccouche, J.-L. Dugelay, Face aging with conditional generative adversarial networks, ArXiv:1702.01983 (2017).
- [12] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, ArXiv:1511.06434 (2015).
- [13] A. Papoulis, *Probability, random variables, and stochastic processes*, (1965).
- [14] Z. Liu, P. Luo, X. Wang, X. Tang, Deep Learning face attributes in the wild, in: 2015 IEEE Int. Conf. Comput. Vis., 2015, pp. 3730–3738, doi:[10.1109/ICCV.2015.425](https://doi.org/10.1109/ICCV.2015.425).
- [15] A. Asthana, S. Zafeiriou, S. Cheng, M. Pantic, Incremental face alignment in the wild, in: 2014 IEEE Conf. Comput. Vis. Pattern Recognit., 2014, pp. 1859–1866, doi:[10.1109/CVPR.2014.240](https://doi.org/10.1109/CVPR.2014.240).
- [16] S. Rakshit, *Novel Methods For Accurate Human Iris Recognition*, University of Bath, 2007.
- [17] CASIA Iris Image Database, 2010 (accessed July 24, 2017). <http://biometrics.idealtest.org/>.
- [18] D. Berthelot, T. Schumm, L. Metz, Began: Boundary equilibrium generative adversarial networks, ArXiv Prepr. arXiv:1703.10717 (2017).
- [19] OpenCV, Face Detection using Haar Cascades, (n.d.). https://docs.opencv.org/trunk/d7/d8b/tutorial_py_face_detection.html.
- [20] X. Cao, Y. Wei, F. Wen, J. Sun, Face alignment by explicit shape regression, in: 2012 IEEE Conf. Comput. Vis. Pattern Recognit., 2012, pp. 2887–2894, doi:[10.1109/CVPR.2012.6248015](https://doi.org/10.1109/CVPR.2012.6248015).
- [21] X. Xiong, F.D. la Torre, Supervised descent method and its applications to face alignment, in: 2013 IEEE Conf. Comput. Vis. Pattern Recognit., 2013, pp. 532–539, doi:[10.1109/CVPR.2013.75](https://doi.org/10.1109/CVPR.2013.75).
- [22] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, M. Pantic, 300 faces in-the-wild challenge: the first facial landmark localization challenge, in: 2013 IEEE Int. Conf. Comput. Vis. Work., 2013, pp. 397–403, doi:[10.1109/ICCVW.2013.59](https://doi.org/10.1109/ICCVW.2013.59).
- [23] F. Inc., MIRLIN, 2015. <https://www.fotonation.com/products/biometrics/iris-recognition/>.
- [24] S. Bazrafkan, S. Thavalengal, P. Corcoran, An end to end deep neural network for iris segmentation in unconstraint scenarios, ArXiv Prepr. ArXiv1712.02877, 2017.
- [25] D.-A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (ELUs), *CoRR*. abs/1511.0, 2015 <http://arxiv.org/abs/1511.07289>.