Deep Feature Consistent Variational Autoencoder: A Technical Report

Cem Eteke[1], Riza Arda Kirmizioglu[2] and Koray Kavakli[2]
[1]Department of Computer Engineering, [2]Department of Electrical and Electronics Engineering,
Koç University, Istanbul, Turkey
{ceteke13,rkirmizioglu,kkavakli}@ku.edu.tr

***Abstract*— Variational Autoencoder (VAE) is a method that can both learn features and generate images. VAE has some drawbacks like blurry image generation and utilization of mean square error. We re-implemented a type of Variational Autoencoder called Deep Feature Consistent Variational Autoencoder and included the results in this report. This report also shows that our implementation complies with the original results. We further test the model's performance by utilizing it in a real world application. This report also proposes that the model fails to adapt to real world.**

I. INTRODUCTION

Deep Convolutional Networks (CNN) in Computer Vision became a trending research topic in recent years. Many successful Computer Vision tasks such as image classification, object detection and captioning have been developed by using CNNs. Aim of Computer Vision by Deep CNN-based methods is to obtain a model that is trained with given data set that can store vast amount of information in limited number of parameters.

A. Image Generation by Deep Learning

Another common use of CNN is image processing by Deep Learning. For image generation problem this technique can also be adapted. Convolution kernels are also referred as filters in Deep Learning. These kernels can be trained to perform the tasks that are specified by the Convolutional Neural Networks. In image generation case, these kernels are required to be trained such a way that they can output images, hence, they become generative convolutional model.

B. Feature Learning

Feature learning is a set of techniques that are used in Machine Learning and Deep Learning. Feature learning is the practice of extracting low-dimensional, latent features from high dimensional inputs. These inputs can vary from field to field, from sensory data to images.

Computer Vision most of the time deals with images. It is important to extract high-level information from images. Therefore, feature learning approaches are common in computer vision application. The extracted features can be used to tackle both supervised and unsupervised Machine Learning problems. Learning low-dimensional features enables us to circumvent the *curse of dimensionality* and performance issues.

C. Motivation

Although there are many methods to construct generative CNN models, Variational Autoencoder has become popular since it can encode image specific features into latent variables by variational inference. [1] This implies that, by sampling from the latent space, we can generate images to fit the original data set. Pixel-by-pixel measurements provide us loss functions between original and reconstructed images, however, in terms of image quality, generated images tend to be blurry and not satisfying when they compared with the original images. [2]

Recent works on CNN-based methods can actually capture several spatial correlation of the input images. This can be achieved by simply replacing pixel-by-pixel loss with perceptual loss. In this project, a Variational Autoencoder introduced by [1] with *perceptual loss* is implemented and the results are investigated by comparing images and looking at the representative power of the features. Therefore, in this report we examine the method presented by [3] both from image generation and feature learning perspectives.

Rest of this report is structured as follows: First we introduce image generation methods in Deep Learning, then we introduce the architecture proposed by [3]. Then we continue by discussing the implementation and training details, the results and finally the conclusion.

II. BACKGROUND

A. Variational Autoencoder

Autoencoders are types of Neural Networks which learn to reconstruct its input. The output is generated by mapping the input to a latent space (encoding) and then mapping from latent space to the original space (decoding). The reconstruction is learned by minimizing the *Mean Square Error* (MSE) between the input and the output. Hence, a representation of the input data can be learned.

To also minimize the information loss during encoding, *KL-Divergence* (KLD) is also minimized for the encoder. However, to sample from the latent space and generate examples by feeding the samples to the decoder, it is required to constraint the distribution of the latent space during training i.e *Variational Autoencoder* (VAE) [4]. This sampling cannot performed during training since sampling is not differentiable. Hence, *reparameterization* is employed which is also estimating the mean and standard deviation from the latent space and forming the feature vector from those parameters.

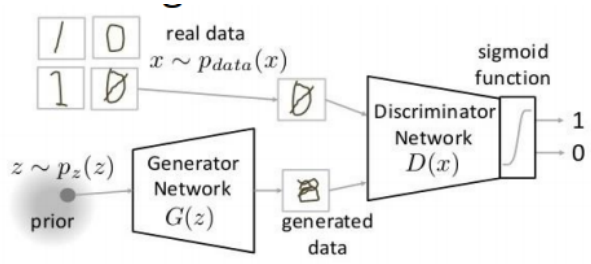


Fig. 1: GAN Architecture

$$
\begin{aligned}
h &= encoder_\theta(x) \\
\mu &= W_1 h \\
\sigma &= W_2 h \\
z &= \mu + \sigma\epsilon, \epsilon \sim \mathcal{N}(0, I) \\
y &= decoder_\phi(z)
\end{aligned} \tag{1}
$$

Since, $\mathcal{N}(\mu, \sigma)$ is desired to be as close to $\mathcal{N}(0, I)$ i.e a known distribution, KLD is employed as a loss metric. As a result, final loss to be optimized becomes

$$\mathcal{L} = MSE(x, y) + KL(\mathcal{N}(\mu, \sigma)||\mathcal{N}(0, I)) \tag{2}$$

As a result, Variational Autoencoders can both generate images from a latenet space and can learn meaningful features of images.

B. Generative Adversearial Networks

Generative Adversearial Networks (GAN) poses an architecture that uses a game like competition between two networks: a Generator and a Discriminator. They were introduced by Ian Goodfellow et al. in 2014. [5] As it is depicted in the Figure 1 Generator network trained to generate more realistic images where the discriminator network trained to differentiate whether the incoming images are original data or reconstructed solutions from the generator network.

Nevertheless, the images generated by GANs can be absurd since the Generator's only goal is to fool the Discriminator. The loss metric used by GANs do not say anything about the model's performance. To solve this issue [6] introduced Wasserstein GAN which employs Wasserstein loss metric to train GANs.

C. General Problems of VAE and GAN

Variational Autoencoders are trained over mean squarred error function. However, pixel difference is not a reliable metric for visual perception. Second and third image in Figure 2 have the same mean squared error, however, perceptually third one is better. Thus, minimizing mean squared error does not improve results always in the sense of perception.

GAN on the other hand depends on a boolean discriminator, therefore, it adapts a zero-one decision machine in order to generate images and the results may end up visually disrupted.

Fig. 2: (a) Original Image, (b) Papper & Salt, (c) Constant Difference.

III. Deep Feature Consistent Variational Autoencoder

In Deep Feature Consistent Variational Autoencoder [3] architecture minimization does not focus on mean square error. Output and input of variational autoencoder are fed to some chosen pre-trained VGG network [7] and their feature based perceptional loss is minimized.

A. System Architecture

Variational Autoencoder system consists of three main components depicted in Figure 3: an autoencoder network with encoder, latent vector, decoder and also a pre-trained loss network. The encoder encodes input image to latent space, the decoder generates an image from the latent space, then the generated image and the input image is fed into a pretrained CNN. Finally, the difference between corresponding feature of generated and real image at various CNN layers are used as a similarity metric.

Unlike standard training procedure of variational autoencoder by minimizing mean squarred error of input and output images of the network, feature consistent variational autoencoder feeds its input and output to first three layer of the pretrained VGG network. Feature consistent VAE architecture minimizes L2 norm of the outputs (input and output of VAE) of these three layers in order to train variational autoencoder network. The network is trained to minimize the following loss where $\mathcal{L}_{KL}$ is the KL-Divergence, and $\mathcal{L}^i_{rec}$ is the L2 norm between generated and input image at i^{th} VGG layer.

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{KL} + \beta\sum_{i}^{l}\mathcal{L}^i_{rec} \tag{3}$$

1) Encoder: Encoder consists of 4 Convolutional layers with kernel sizes of 4, strides of 2, and number of filters: $32, 64, 128, 256$. After each convolutional layer batch normalization is applied. Finally the output of the last convolutional layer is fed through two linear layers to form representation of size 100, each layer outputting the mean and standard deviation of the latent space. Besides the linear layers, ReLU activation function is applied to the layer outputs.

2) Decoder: Decoder first increases the dimensionality of latent space from 100 to 4096. Then the vector is passed through 4 layers of convolution. Before each layer the image is upsampled. At each layer convolution filters of sizes 3, strides of 1 and number of filters: $128, 64, 32, 3$ is applied.

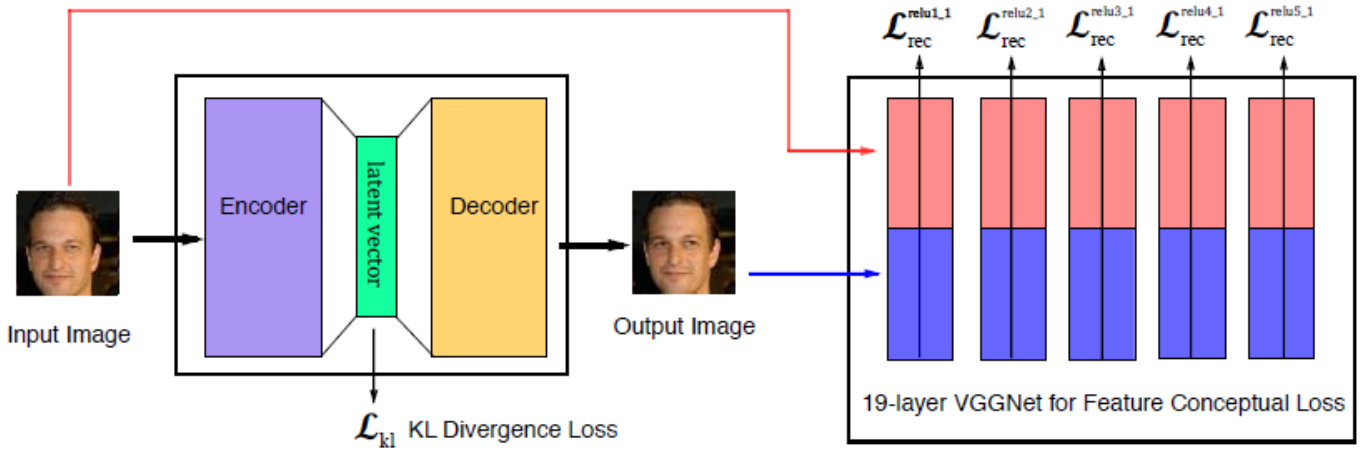


Fig. 3: Network architecture [3]

Besides the last layer, after each layer, batch normalization is applied and ReLU is used as an activation function.

B. Training Details

To train the model, CelebFaces Attributes (CelebA) Dataset [8] which contains 202,600 face images. These images are scaled and aligned to 64 x 64 pixels. The dataset also contains the 40 different facial attributes for each image. These attribute labels are used for attribute manipulation (Section V-A.1) and for attribute classification (Section V-A.2). Adam optimizer with learning rate of $1e^{-4}$ with minibatches of size 32 is used for training. The model is trained for 10 epochs. We used a predefined and pretrained Tensorflow model for VGG-19 [1]. The rest is implemented using Tensorflow in Python 3. TF Data is utilized in order to form a high performance data pipeline. The model is trained on an Nvidia GeForce GTX-1080 Ti GPU.

IV. Results

We have tested our model to see if we can observe the same results presented by the original paper [3]. In addition, we also implemented a simple real world application of this model, attribute prediction in real-time from webcam images. In this section we discuss the replication of the results in the paper [3], and the results of our small application.

KL divergance, perceptual loss, total loss results are depicted in Figure 9.

A. Image Generation

To see the generative performance of the model, we have sampled latent vectors from $\mathcal{N}(0, I)$ and fed the vectors through the decoder. The resulting images can be seen in Figure 4.

When we feed an image to trained variational autoencoder network, its encoded and decoded formats does not look like a mixture of images. However, we can say that decoding randomly sampled latent vectors contain intermediate values of multiple images. Thus, the resulting randomly generated images look like composition of multiple images where face attributes and background are disrupted.

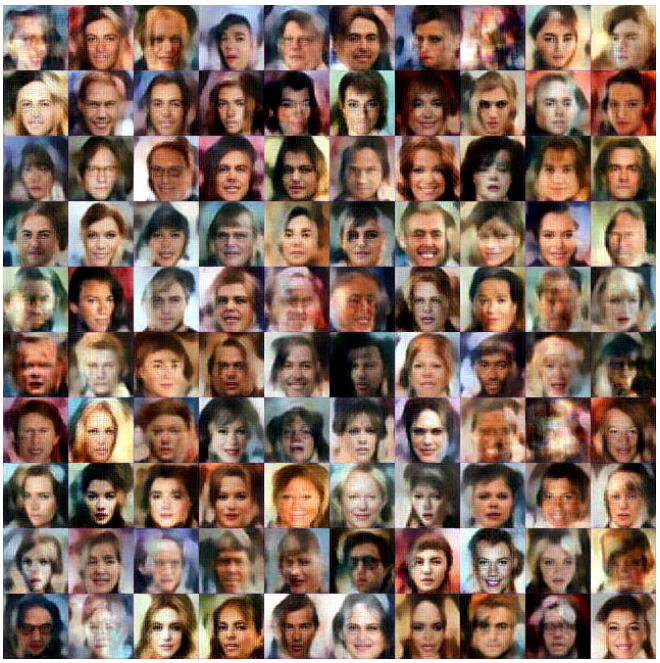


Fig. 4: Randomly generated images

B. Interpolation

Latent vector is the encoded data format of the face images. In pioneer codecs like JPEG, meaningful attribute manipulation and interpolation is not possible by changing encoded data coefficients. However, latent space vector in variational autoencoder enables such interpolation operation. The interpolation can be expressed by a linear combination

[1] https://github.com/machrisaa/tensorflow-vgg

z = (1 - α)z_{left} + αz_{right} where 0 $\leq \alpha \leq$ 1.

Fig. 5: Interpolation between latent vectors

Interpolation of two different people examples as α increases from 0 to 1 with 0.1 incerements are depicted in Figure 5. Generated mixed image is very similar to the left image where α is 0. As α increases similarity of the interpolated image resembles more to the right image and their equal mix is obtained where α = 0.5.

C. Attribute Manipulation

Unlike interpolation, attribute manipulation focuses on more regional areas rather than overall image. A specific face attribute like a smile or eyeglasses can be added or substracted to latent space. In order to gather eyeglasses vector, we randomly select 1000 face images with eyeglasses and 1000 face images without eyeglasses. Then, we subtracted the mean of latent vectors without eyeglasses from the mean of latent vectors with eyeglasses. Then, this eyeglasses vector can be added to a different face with different coefficients to manipulate the facial attributes. The same procedure can be applied for smile or other attributes.

Fig. 6: Smile Addition

Fig. 7: Eyeglasses Addition

Smile vector addition to a non-smiling woman vector where α = 0.1 ... 1 is depicted in Figure 6. Woman in the first image does not smile, and eventually, as the α grows, her smile becomes more apparent. A similar approach for eyeglasses is depicted in Figure 7.

D. Attribute Classification

In order to test the feature learning performance of the Deep Feature Consistent Variational Autoencoder, we have extracted images with a certain attribute and ones that do not contain that attribute. Then we passed those images through the encoder of the model and trained a binary classifier for each attribute: Liner SVM on the labeled, latent data. We splitted the data into 90% train and 10% test. The results for each attribute we found and the original results can be observed in Table I. It can be observed that we got nearly the same results with the original paper.

V. Real World Application

A webcam application has been developed to test attribute manipulation on the other faces that are not available within the dataset. This application extract face image by using OpenCV for face detection. Initial aim of this application was to gather the same results on attribute manipulation by applying same attribute vectors to a face obtained from webcam in real-time. However, it did not worked succesfully as the model learned only to generate face images that contain the similar features that are present in our dataset.

Due to the problems we have encountered in this application, which are discussed in the Section V-A.1, we developed an another application based on attribute classification. The main objective in this application was to classify the facial attributes from webcam images in real-time. We have tested several attributes for this application such as: eyeglasses, attractiveness and smiling. The results of this application are illustrated in Section V-A.2

A. Experimental Results

1) Real-Time Facial Attribute Manipulation: We tried to extract faces from real-time captured frames and manipulate its attributions. Extracted face is down-sampled to 64x64 image and fed to encoder. After obtaining its latent vector, we added eyeglasses vector with $\alpha = 1$. We aimed to achieve real-time facial attribute manipulation by feature consistent variational autoencoder. Yet, this operation failed and did not provide good results. The first reason for this result depends on the face position in the images. Depending on the given face position in the input image the model can create meaningful face images in terms of face shape. Although, it does not completely comply with the input image, it is observable that input and the output images share some similar structures as it can be seen in Figure 8. It is cleary seen that the model fails to regenerate the same image, it generates similar attributes like skin color, hair and a small smile.

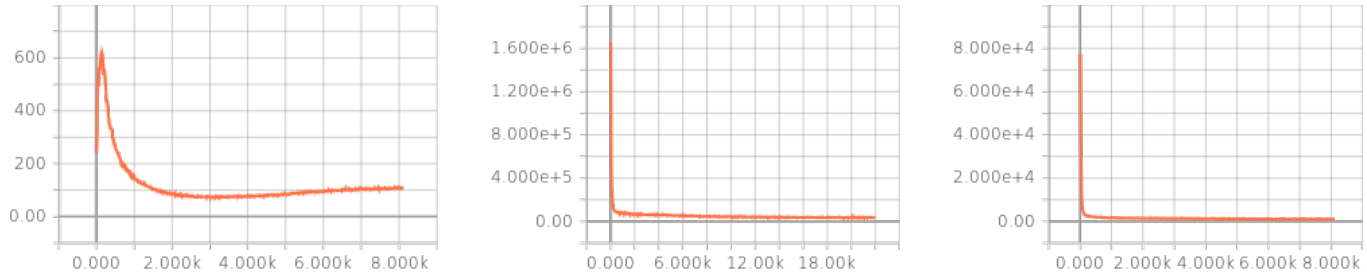
Fig. 8: Eyeglasses Addition

2) Real-Time Facial Attribute Classification: We tried to classify facial attributes in real-time webcam application with the trained SVM classifiers. We experimented that with classifiers of *Smiling*, *Attractiveness* and *Eyeglasses*. When the subject does not show the attribute (negative class), bounding box of face detector responses red. When subject shows the attribute (positive class) the bounding box responses by turning into green.

Figure 10 depicts the results of this application. As it can be seen, only the smiling classifier worked and eyeglasses, especially attractiveness failed. Like the observations in real-time attribute manipulation, this failure also showed that the model works well only when given an image from the dataset.

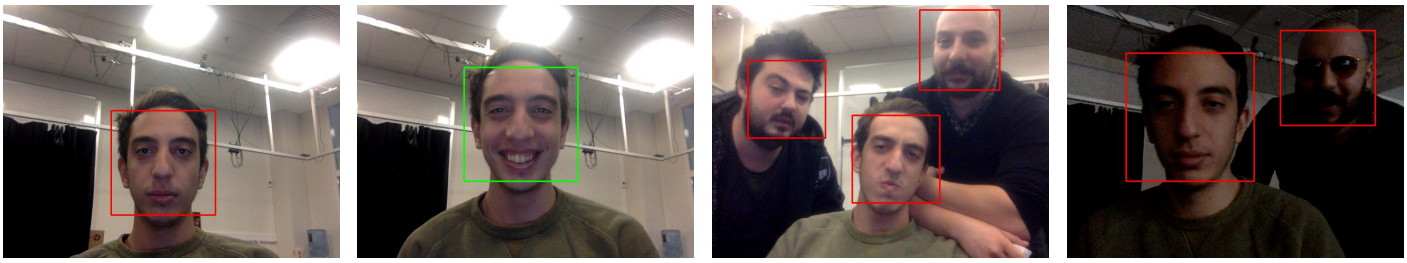
	5 Shadow	Arch. Eyebrows	Attractive	Bags Un. Eyes	Bald	Bangs	Big Lips	Big Nose	Black Hair	Blond Hair	Blurry	Brown Hair	Bushy Eyebrows	Chubby	Double Chin	Eyeglasses	Goatee	Grayy Hair	Heavy Makeup	H. Cheekbones	
VAE-123 (Paper Results)	89	77	75	81	98	91	76	79	83	92	95	80	87	94	95	96	94	96	85	81	
VAE - 123 (Our Results)	88	78	75	80	98	91	76	79	82	92	95	80	87	94	95	96	94	97	85	81	
	Mouth S. O.	Mustache	Narrow Eyes	No Beard	Oval Face	Pale Skin	Pointy Nose	Reced. Hairline	Rosy Cheeks	Sideburns	Smiling	Straight Hair	Wavy Hair	Wear. Earrings	Wear. Hat	Wear. Lipstick	Wear. Necklace	Wear. Necktie	Young	Male	Average
VAE-123 (Paper Results)	80	96	89	88	73	96	73	92	94	95	87	79	74	82	96	88	88	93	81	90	86.95
VAE - 123 (Our Results)	80	96	88	87	73	96	73	92	94	95	87	79	75	81	96	87	88	88	82	90	86.68

TABLE I: Binary classification results for each attribute



(a) KL Divergence (b) Perceptual Loss (c) Total Loss

Fig. 9: Plots of learning curves during training



(a) No smile (b) Smile (c) Attractiveness (d) Eyeglasses

Fig. 10: Real world application experimental results

VI. Conclusion

In this report, we presented the results that we obtain when the Deep Feature Consistent Variational Autoencoder introduced by [3] is replicated. The results obtained are close to the original ones. On top of that, we implemented a real world application and showed that the model does not work well when given real world images.

References

[1] D. P. Kingma and M. Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

[2] Z. Wang and A. C. Bovik. A universal image quality index. IEEE signal processing letters, 9(3):8184, 2002.

[3] Hou, Xianxu, et al. "Deep feature consistent variational autoencoder." Applications of Computer Vision (WACV), 2017

[4] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Neural Information Processing Systems Conference, 2014

[6] Martin Arjovsky, Soumith Chintala, Lon Bottou. Wasserstein GAN. arXiv preprint arXiv:1701.07875

[7] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[8] Z. Liu, P. Luo, X.Wang, and X. Tang. Deep learning face attributes in the wild. In Proceedings of the IEEE International Conference on Computer Vision, pages 37303738, 2015.