MATH7017: Probabilistic Graphical Models Project

Rizwan Karim Rafat // SID: 22061272

Autumn 2024

1 Introduction

In recent years, improved deep learning approaches have changed the field of generative modelling, resulting in the creation of Conditional Variational Autoencoders (C-VAE) [1] and Conditional Generative Adversarial Networks (C-GAN) [2]. These models include extra conditional information, allowing them to create data that not only looks real but also adheres to certain features like class labels and styles.

In this project we shall investigate and analyse C-VAE and C-GAN in generating Japanese character pictures from the Kuzushiji-49 dataset used in [3], taking into account both character class and style. The goal is to assess their performance in terms of both qualitative (visual) and quantitative measures.

2 Model Description

2.1 Conditional Variational Autoencoder (C-VAE)

The architecture includes an encoder, which compresses input data, class, and style information into a latent representation. This is achieved by concatenating the input picture (represented as a flattened vector) with one-hot encoded class and style vectors. The combined input is sent through a fully linked layer, which lowers it to a hidden dimension and induces non-linearity via ReLU activation. The encoder produces $\boldsymbol{mu}$ and $\boldsymbol{logvar}$, which represent the mean and logarithm of the variance for the latent variables, in accordance with the ideas given in [4]. The reparameterization method, which is important for enabling backpropagation over stochastic layers, is used to select a latent variable $\boldsymbol{z}$ from the Gaussian distribution described by $\boldsymbol{mu}$ and $\boldsymbol{logvar}$.

The decoder, on the other hand, reconstructs the input data from the latent representation by combining $\boldsymbol{z}$ with the class and style vectors and sending it through another fully connected layer before returning it to the original input dimension. This strategy ensures that the generated samples not only mirror the original input data but also meet the required requirements as stated in [5]. This conditional arrangement allows the C-VAE to create various and conditionally correct data, making it a useful tool for tasks such as creating representations of specified characters in specific styles. The ability to include class and style information directly into the generation process gives precise control over the properties of the generated outputs, which aligns with the goals mentioned in [2].

2.2 Conditional Generative Adversarial Network (C-GAN)

The architecture of the Conditional Generative Adversarial Network (C-GAN) has two primary components: the Generator and the Discriminator. The Generator is used to create pictures based on specified class and style labels, which is important for creating data that meets the requirements. It accepts as input a concatenation of a random noise vector ***(latent_dim)***, a one-hot encoded class label vector ***(label_dim)***, and a style vector. This combined input is sent through a sequential network that includes two linked layers. The first layer maps the input to a 256-dimensional space and uses the ReLU activation function, which provides non-linearity and helps in the learning of complicated patterns in the data. The second layer lowers this to the size of the output picture ***(img_dim)***, and a Tanh activation function is used to scale the output to the [-1, 1] range, which is appropriate for image pixel values. This arrangement is consistent with the ideas established in [2], which pioneered the concept of producing data conditioned on extra input.

The Discriminator, on the other hand, is used to assess the validity of produced pictures and how they align with the stated parameters. It accepts as input a created or actual picture together with the associated class and style designations. The input vector is subsequently sent through a set of two linked layers. The first layer maps the input to a 256-dimensional space and uses a ReLU activation function to provide non-linearity and allow the network to learn data-specific properties. The second layer uses a Sigmoid activation function to generate a single value that represents the likelihood that the input picture is real and meets the specified requirements. This design makes sure that the Discriminator examines not only that the image is real but also its consistency with the specified class and style labels, supporting the conditional generation method outlined in [6].

3 Training Details

3.1 C-VAE Training

The model's key parameters are established from the start, including an input size of 784, which corresponds to 28x28 pixel pictures compressed into a single vector, and a latent dimension of 2, suggesting a reasonably limited latent space. The hidden layer is designed to have a dimensionality of 144 (12x12), allowing for the extraction of complex characteristics from input data. The model is trained across 40 epochs at a learning rate of 0.001 using the Adam optimizer, which is noted for its ability to handle big datasets and stochastic gradients.

The training loop is designed to go over the dataset, with each epoch including the model processing batches of data, labels, and style characteristics. During each iteration, the model's parameters are adjusted to minimise the loss. The cumulative loss is averaged and presented for each epoch, providing information on the model's convergence. After training, the model's parameters are retained for further usage and assessment. The entire training time is also computed and reported, providing a reasonable estimate of the computing resources needed to train the C-VAE under these precise conditions. This thorough technique ensures that the C-VAE is properly configured to provide high-quality, conditionally correct data samples rapidly.

3.2 C-GAN Training

The hyperparameters were the basis for this training, with a latent dimension (latent_dim) of 128, an image dimension (img_dim) of 784 (representing a flattened 28x28 picture), and class and style dimensions (label_dim and style_dim) that match to the specific job. The training uses a batch size of 64 and a learning rate of 0.0001 for both the Generator and the Discriminator, with the Adam optimizer used for efficient gradient-based optimisation. The training loop continues for 40 epochs, with each epoch containing numerous iterations over batches of real data and their associated labels and styles.

The training loop includes many important stages for each epoch. First, genuine photos, together with their names and styles, are loaded and sent to the appropriate device (CPU). The Generator generates random noise vectors, which are then mixed with labels and styles to create false pictures. The Discriminator is then trained by calculating its ability to distinguish between real and fake photos. This requires computing the loss for actual and fake photos individually, with true images likely to be classed as real and fake ones as false. The combined loss is used to adjust the Discriminator's settings. Following that, the Generator is updated by minimising the loss associated with producing pictures that the Discriminator would consider real. This adversarial arrangement ensures that as epochs pass, the Generator improves its ability to produce real images based on the supplied labels and styles, while the Discriminator improves its ability to discriminate between real and fake images. The generator weights are kept at the conclusion of each period for future usage or training. The overall training time is measured giving an idea of the computational resources needed to train the C-GAN under these conditions.

4 Results

In this section, we will compare the results of the Conditional Variational Autoencoder and the Conditional Generative Adversarial Network by generating images for classes 1 to 5 with both ”thick” and ”thin” styles. This comparison aims to evaluate how each model handles the conditional generation task and to assess the quality and consistency of the generated images for the specified classes and styles.

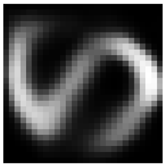
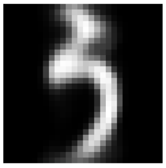
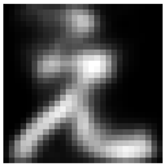
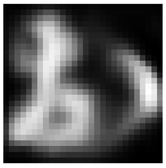
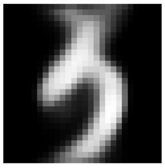
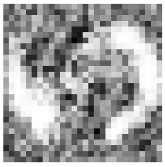
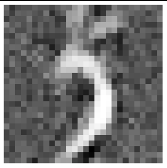
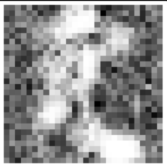
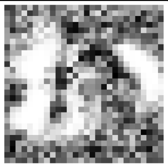
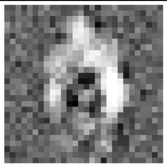
	Class: 1 **Style: Thick**	**Class: 2** **Style: Thick**	**Class: 3** **Style: Thick**	**Class: 4** **Style: Thick**	**Class: 5** **Style: Thick**
C-VAE					
C-GAN					

Table 1: Images with Style set as ”Thick”

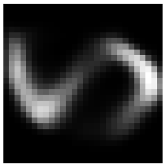
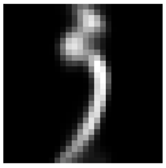
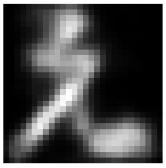
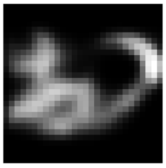
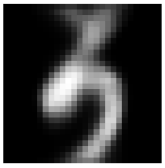
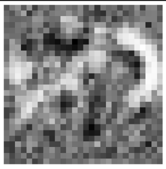
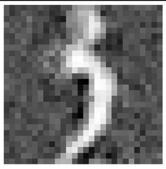
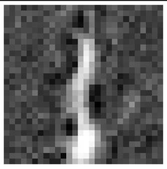
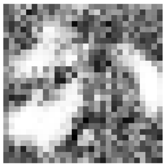
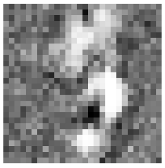
	Class: 1 **Style: Thin**	**Class: 2** **Style: Thin**	**Class: 3** **Style: Thin**	**Class: 4** **Style: Thin**	**Class: 5** **Style: Thin**
C-VAE					
C-GAN					

Table 2: Images with Style set as "Thin"

As shown in Tables 1 and 2, the C-VAE's generated images for classes 1 to 5 in both "thick" and "thin" styles show higher visual quality and consistency compared to the C-GAN. The C-VAE captures finer details and variations in style, making its outputs more visually appealing.

Average Mean Squared Error for C-VAE: 0.0670

A Mean Squared Error of 0.0670 suggests that the C-VAE is relatively good at reconstructing the input images.

Average Structural Similarity Index for C-VAE: 0.3881

A Structural Similarity Index of 0.3881 is relatively low, suggesting that while the reconstructed images might capture some structural aspects of the originals, there is a significant loss of fine details and perceptual quality.

Mean Squared Error for C-GAN: 0.2839

A Mean Squared Error of 0.2839 indicates a higher average squared difference between the generated and real images, reflecting the challenges C-GAN faces in producing exact replicas of the real images.

References

[1] Pol, A. A., Berger, V., Germain, C., Cerminara, G., & Pierini, M. (2019, December). Anomaly detection with conditional variational autoencoders. In 2019 18th IEEE international conference on machine learning and applications (ICMLA) (pp. 1651-1657). IEEE.

[2] Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[3] Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., & Ha, D. (2018). Deep learning for classical japanese literature. arXiv preprint arXiv:1812.01718.

[4] Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

[5] Sohn, K., Lee, H., & Yan, X. (2015). Learning structured output representation using deep conditional generative models. Advances in neural information processing systems, 28.

[6] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.