Probabilistic Graphic Model Project

Student ID: 22049122, Swathi Ramesh

Aim

To train Kuzushiji-49 dataset on two different models – Conditional Variational Autoencoder Model (C-VAE) and Conditional Variational Adversarial Network Model (C-GAN) and compare the model performance.

Dataset and Preprocessing

Kuzushiji-49 dataset consist of 49 classes, 270912 images of Japanese Hiragana characters. The image is 28 x 28 pixels. It consists of train image, train label, test image and test label. The images are gray scaled [1].

As a preprocessing step a style label is added for each image upon the condition using threshold. The dataset is converted into binary format using threshold value 128 and the style is added using the pixel density. Binary conversion can differentiate foreground pixels from background pixels. Median Pixel count is calculated and then it is used to identify the style thick and thin. If the pixel count is greater than the median pixel count of the label, then its style is thick otherwise the style is thin. A function to_one_hot is used to convert the class labels into one hot encoded matrix using the PyTorch so that the model can take the inputs in vector form.

Model Description

Conditional Variational Autoencoders (C-VAE):

The C-VAE consists of encoder, decoder and reparameterization layers.

Encoder: The role of the encoder is to produce the mean and log variance form the input which is the concatenated input of image, one-hot encoded label and style label.

Reparameterization Layer: The reparameterization layer produces the sample from latent space using the mean and log variance generated.

Decoder: The decoder layer reconstructs the image from the latent vector with labels and style.

The different layer are self.fc1 which is fully connected layer which takes image, class label and style as input, self.fc2_mu which is a fully connected layer in which the mean (mu) is generated as the output, self.fc2_logvar which is also a fully connected layer in which the log variance (logvar) is generated as the output, self.fc3 is a fully connected layer which takes z (latent vector), class label and style as the input, self.fc4 is a fully connected layer that is used for decoding the representation to input dimension. The forward pass method is used to pass the data through the network [2].

The loss function used is the Binary Cross Entropy (BCE) and Kullback-Leibler Divergence (KLD). The reconstruction error between the original and the generated image is measured by using the BCE and the standard normal distribution is maintained in the latent space with the help of KLD.

Conditional Generative Adversarial Network (C-GAN):

C-GAN consists of two neural networks generator and discriminator.

Generator: It is a fully connected layers with ReLU and Tanh activation function. The class labels which are one hot encoded, noise and the style labels are given as input. The output is generated image.

Discriminator: This is also a fully connected layers with ReLU and sigmoid activation functions. The input for the discriminator is the generated image from the generator, class label and style label.

The Loss function used in the model is Binary Cross Entropy (BCE). BCE is used to calculate the loss of real as well as fake images. Discriminator loss is used to measure the ability of the discriminator to distinguish between the real and fake image. Generator loss is used to measure the ability of the generator to generate images so that the images are real.

Training

C-VAE

The input dimension is 784 since the images in dataset are 28x28 pixels. The hidden dimension is 144, latent dimension is 20 and number of classes is 49 in Kuzushiji-49 dataset. The images are reshaped and normalized for inputting to the input layer of the model. The model is trained using optimizer Adam over 50 epochs. The model tends to learn the characters and the loss is minimized over each epoch. The loss in the first epoch is 239.9821 and it reduces to 184.3786 over 50 epochs. This means that the model is learning and improving to generate images. It took approximately 21 minutes for training.

C-GAN

The latent dimension is 64, the input dimension is 28 x 28 which is the dimension of Kuzushiji-49 dataset. The image dimension is 784 which is 28*28, the batch size is 32, and the learning rate is 0.001. The model is trained for 50 epochs and there are 49 classes in the dataset. The optimizer used here is Adam. During the training the discriminator generates fake images using noise sampling and the loss for real images and fake images is calculated using the BCE. The generator produces fake images using noise vectors and labels. The aim of the generator is to fool the discriminator. It also computes the loss using BCE for the fake images with real label.

Evaluation

C-VAE

The total loss over epochs is plotted for the training process (Figure 1). At the beginning of the training the loss was very high, it was around 245. It can be observed from the graph that there is a steep decrease in loss indicating its learning and improvements. The loss continues to decrease gradually but at a slower rate. The graph starts to flatten towards the end of 50 epochs indicating that it is converging.

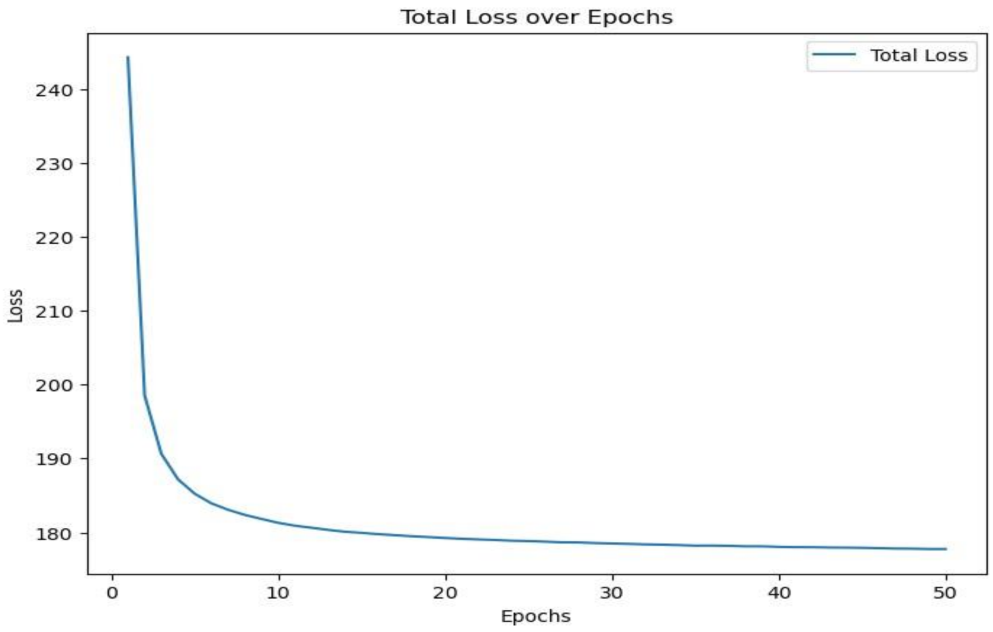

Figure 1: Total Loss Over Epochs for C-VAE

Ten sample images for a particular label are generated for the comparison (Figure 2 and Figure 3) and it can be observed that the model is recreating the label and upon training for a greater number of epochs it will generate clearer images of the label.

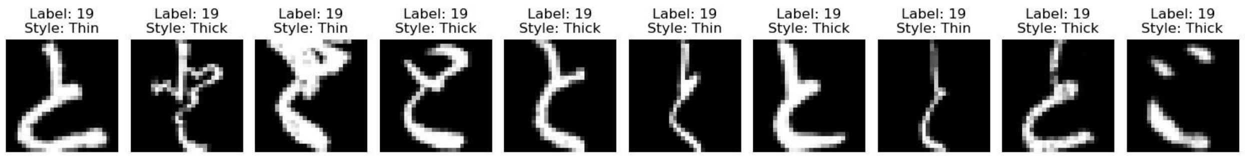

Figure 2: 10 Samples of Original Images from Kuzushiji-49 datasets.

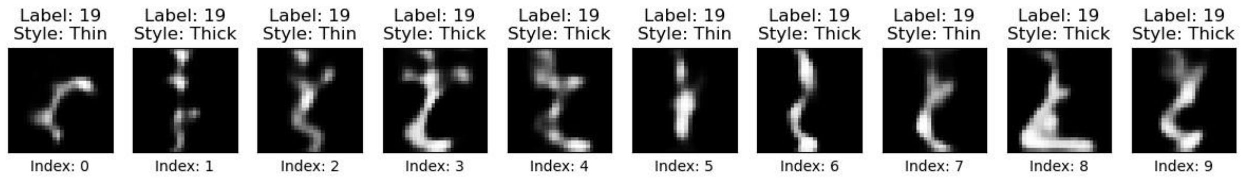
Label: 19 Style: Thin Index: 0
Label: 19 Style: Thick Index: 1
Label: 19 Style: Thin Index: 2
Label: 19 Style: Thick Index: 3
Label: 19 Style: Thick Index: 4
Label: 19 Style: Thin Index: 5
Label: 19 Style: Thick Index: 6
Label: 19 Style: Thin Index: 7
Label: 19 Style: Thick Index: 8
Label: 19 Style: Thick Index: 9

Figure 3: 10 Sample of Reconstructed Images using C-VAE

Even though the shapes are recognizable the reconstructed image is less clear than the original images, it is blurry and noisy. The clarity of the thin style is less while it was reconstructed, and the thick style is affected by noise and is blurry. This can be improved with training for more epochs and changing the encoder and decoder parameters to improve the quality and by adding more layers.

C-GAN

The loss of discriminator and generator over the epoch is plotted (Figure 4). The blue line represents the generator loss, and the orange line represents the discriminator loss. It can be observed that the generator loss seems to change a lot during the training. In the initial epochs the loss seems to be very high and then it reduces towards the end. This means that it is showing improvement over time. There is a lot of variability in the graph for the generator loss. This happens because of the ability of the discriminator to identify fake images. The discriminator loss shows less variability compared to the generator loss. The ability of the discriminator to identify fake images is very efficient. Over the training period of 50 epoch the graph does not seem to be converging, this means that there is a requirement for training the model for higher number of epochs.

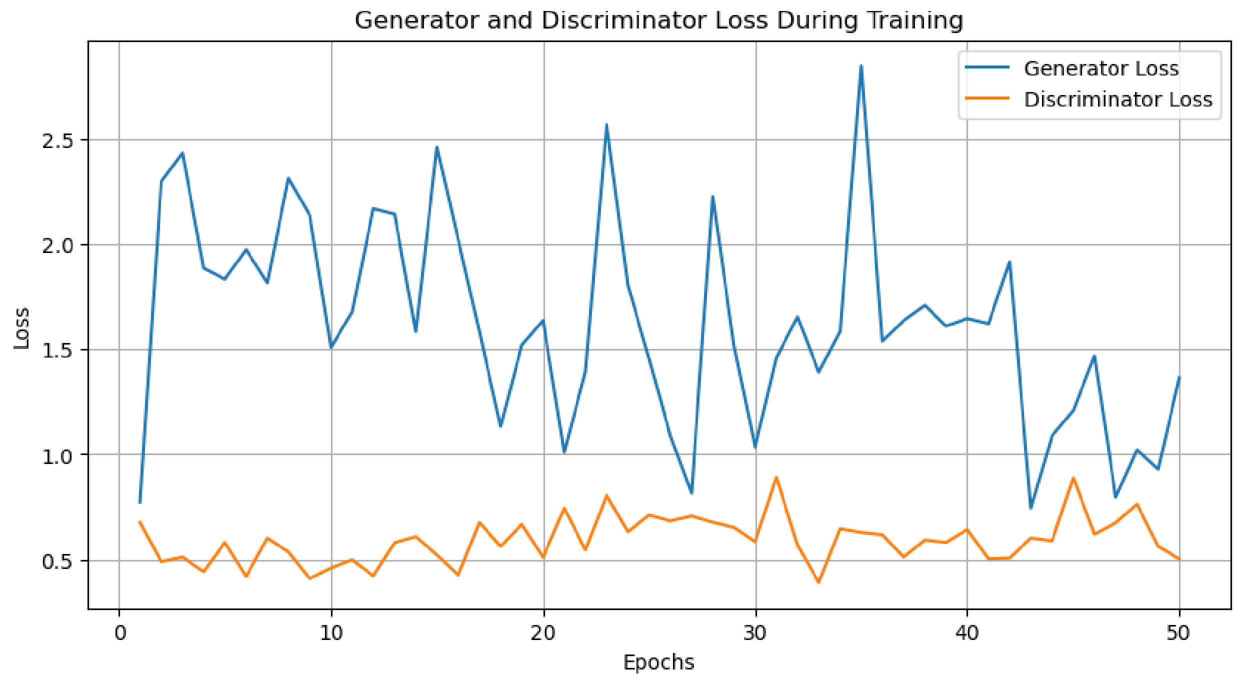

Figure 4: Generator and Discriminator loss over the epochs for C-GAN

A batch of 32 images are used for the evaluation of the model and the corresponding real and fake image for each epoch is generated.

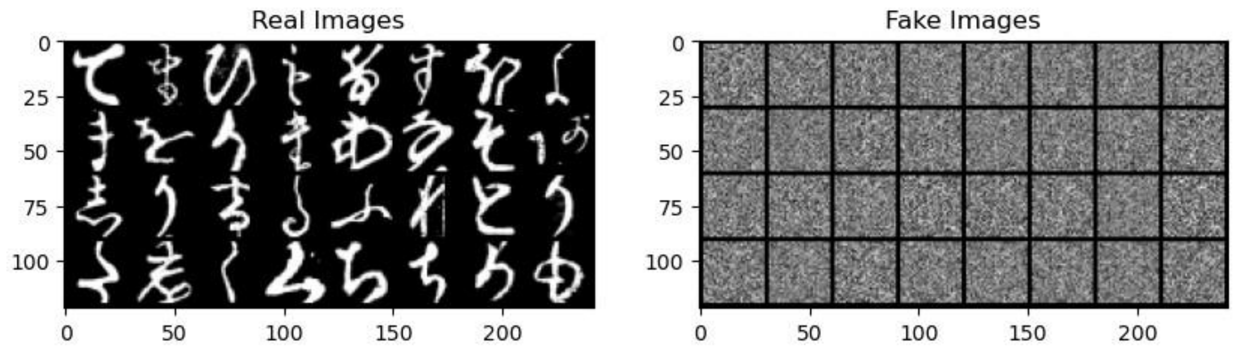

Figure 5: Real and Fake image generated using C-GAN over 1st epoch

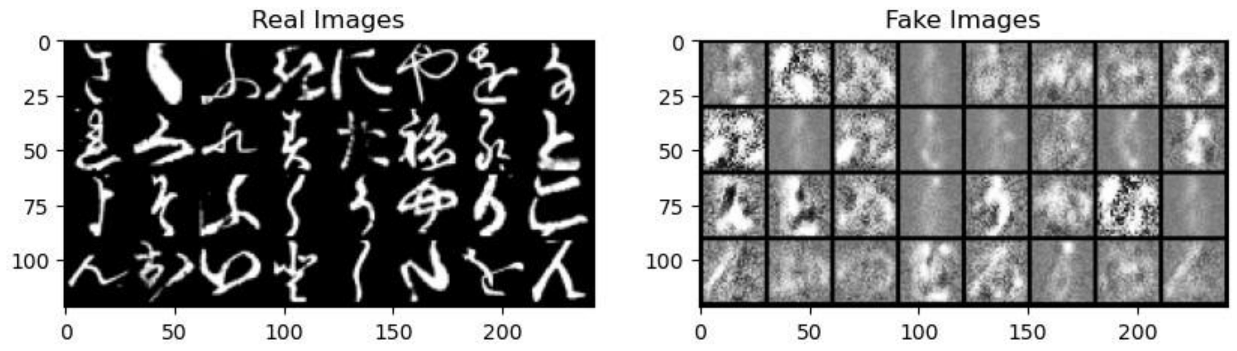

Figure 6: Real and Fake image generated using C-GAN over 25th epoch

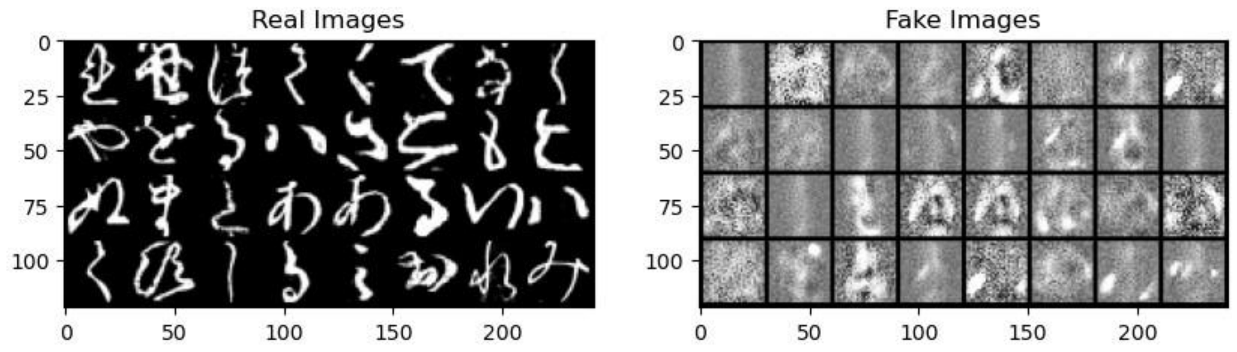

Figure 7: Real and Fake image generated using C-GAN over 50th epoch

Figure 5 shows the generated real and fake image over the 1st epoch. The real image has clear and recognizable images whereas the fake image is noisy and not distinguishable. This means that at this stage the generator has not learned enough about the characters. The generator loss was 0.7727 and the discriminator loss was 0.6760. Figure 6 shows the generated real and fake image over 25th epoch. The real image is clear and distinguishable. The fake image tends to learn and provide some structure of the character but not fully still it is not clear. The generator loss was 1.4545 and the discriminator loss was 0.7107. Still the generator loss is quite high which means that it must learn more and produce fake images which are identified as the discriminator as real. Figure 7 shows the generated real and fake image over the 50th epoch. Real images are clear and consistent whereas the fake image structure is improved and some of them are in recognizable shape. There is noise in the image, and it is still blurry. This shows improvement in the model and the model is learning to generate fake images which are labeled as real by the discriminator.

Model Comparison

Loss Functions: The loss function used in the C-VAE model measures the similarity of the generated image with the original image using BSE. The loss function used in C-GAN measures the discriminator loss and the generator loss using BSE.

Complexity: C-VAE model is less complex compared to C-GAN. C-VAE is more stable and easier to train compared to C-GAN. Careful tuning of hyperparameters and techniques are required for C-GAN compared to C-VAE.

Plots: The loss tends to decrease and converge for C-VAE (Figure 1) whereas the plot (Figure 4) has fluctuations in generator and discriminator loss and is not converging over 50 epochs.

Images: The images generated from C-VAE is more structurally visible and can be distinguished and identified (Figure 2 and Figure 3) whereas the images generated from C-GAN needs more training even after 50 epochs the images are only somewhat structured but was still blurry and has noise (Figure 5, Figure 6 and Figure 7).

Time: Time taken to train C-VAE for 50 epochs is 20 minutes and 19 seconds whereas time taken to train C-GAN for 50 epochs is 78 minutes and 22 seconds.

Challenges and Improvements

The main challenge was the images are blurry and noisy after reconstruction this means that it requires more improvement in the model parameterization and modifications in loss functions. Introducing beta weights in VAE for KL divergence can have a better impact on the reconstruction [2]. Similarly, introduction of gradient penalty can improve the training of C-GAN [3]. Fine tuning the model can also help in improving the model.

References

[1] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha, “Deep Learning for Classical Japanese Literature,” arXiv:1812.01718 [cs, stat], 9999, doi: https://doi.org/10.20676/00000341.

[2] K. Sofeikov, “Implementing Variational Autoencoders from scratch,” Medium, Apr. 25, 2023. https://medium.com/@sofeikov/implementing-variational-autoencoders-from-scratch-533782d8eb95

[3] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved Training of Wasserstein GANs,” Dec. 2017. Accessed: Jun. 17, 2024. [Online]. Available: https://arxiv.org/pdf/1704.00028v3

[4] S. Dobilas, “cGAN: Conditional Generative Adversarial Network — How to Gain Control Over GAN Outputs,” Medium, Aug. 01, 2022. https://towardsdatascience.com/cgan-conditional-generative-adversarial-network-how-to-gain-control-over-gan-outputs-b30620bd0cc8[5]T. Rose, “Conditional Variational Autoencoders with Learnable Conditional Embeddings,” Medium, Jan. 19, 2024. https://towardsdatascience.com/conditional-variational-autoencoders-with-learnable-conditional-embeddings-e22ee5359a2a (accessed Jun. 17, 2024).