IFT6135 Assignment 3 Practical

Simon Chamorro

April 12, 2022

1 Problem 1: VAEs

Question 6: Using the default parameters, we trained a VAE and evaluated it on the validation set using the ELBO. Figure 1 show the performance of the model over the training epochs. We used the function '`kl_gaussian_gaussian_analytic`' to compute the KL Divergence for the VAE loss. The model achieves a performance of -100.49.

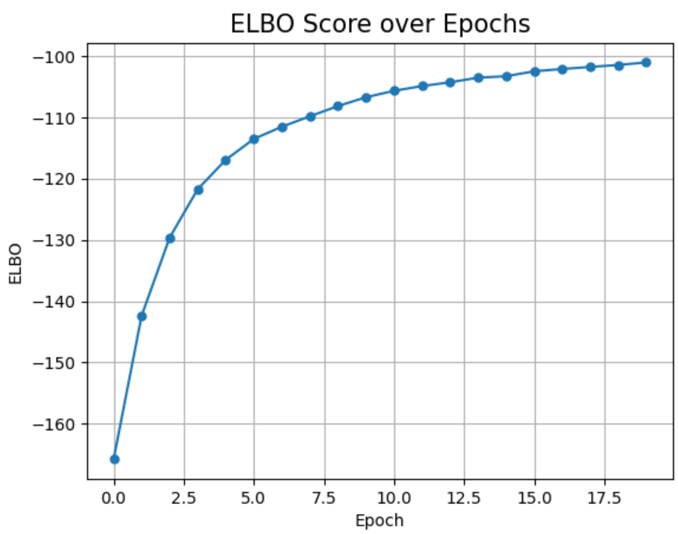


Figure 1: **ELBO.** ELBO over training epochs for our VAE model with default parameters.

Question 7: After training the model for 20 epochs, we evaluate its performance on the test set using the log-likelihood estimate. We obtain a log-likelihood of -95.75.

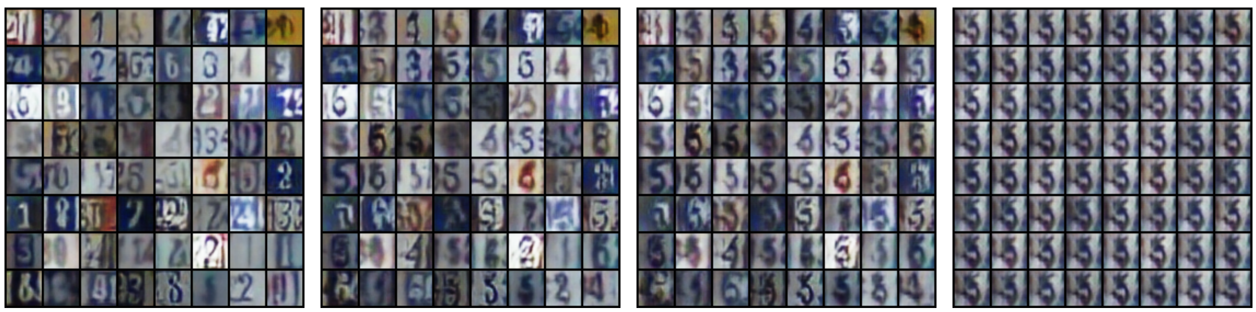
2 Problem 2: GANs

Question 2 (Training Samples): Figure 2 shows images generated by the model at different times during training. We can see that the samples are random at first and become more similar to the training distribution at every training iteration. The samples start by being pure noise, then start looking like numbers, but are very blurry. By the end of the training process, some blurriness remains, but we can see more clear numbers. At the end of training, the diversity is good, but seems to be a little bit biases, as 5s and 2s are predominant and little to no 1s, 7s and 9s can be distinguished.

(a) Iteration 0 (b) Iteration 25k (c) Iteration 50k

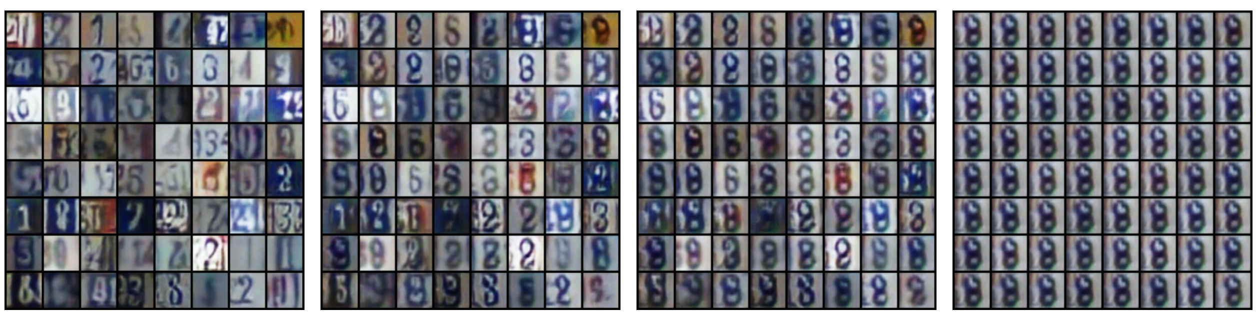
Figure 2: **GAN Training.** This figure shows samples of generated images at different moment of the training process. The GAN is trained on the SVHN dataset for 50k iterations.

Question 3 (Disentangled Representations): In this question, we study the latent representations learned by the model. For this purpose, we sample a random z from our distribution and add noise to each dimension one at a time. Figures 3 to 3 show examples of generated samples when adding noise to different dimensions of the latent space. We present different scales of noise to understand what each dimension is encoding. For example, when adding a lot of noise to dimension #6, we can see that all samples converge towards the number 5, as seen in figure 3. The same is true for dimensions 25 and 58 for the digits 8 and 2 respectively.



(a) Original (b) eps=10.0 (c) eps=20.0 (d) eps=100.0

Figure 3: **Disentangled Representations, dimension #6.**



(a) Original (b) eps=10.0 (c) eps=20.0 (d) eps=100.0

Figure 4: **Disentangled Representations, dimension #25.**

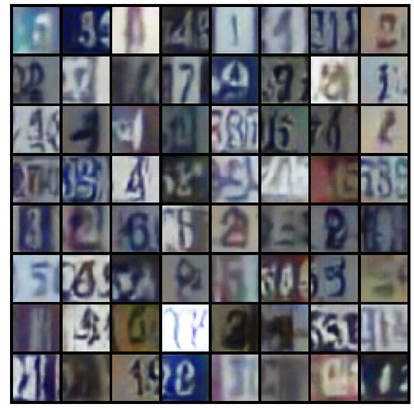
(a) Original (b) eps=10.0 (c) eps=20.0 (d) eps=100.0

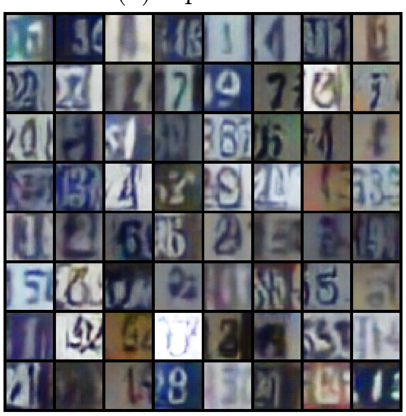
Figure 5: **Disentangled Representations, dimension #58.**

Question 4 (Interpolation): In this question, we study the difference between interpolating between two samples in latent space versus pixel space. Figure 6 shows a latent space interpolation between two sets of samples, while figure 7 shows a pixel space interpolation between the same two samples. We can see that the latent space interpolation allows us to obtain samples that seem to be similar to the initial training distribution all along the interpolation, even when alpha is set to 0.4 or 0.6. On the other hand, we do not maintain valid samples when interpolating in pixel space. For example, the images in figure 7 become very blurry when alpha is set to 0.4 or 0.6. They simply look like an overlay of samples on and 2.


(a) alpha=0.0


(b) alpha=0.2


(c) alpha=0.4


(d) alpha=0.6


(e) alpha=0.8


(f) alpha=1.0

Figure 6: **Interpolation between two samples in latent space.**

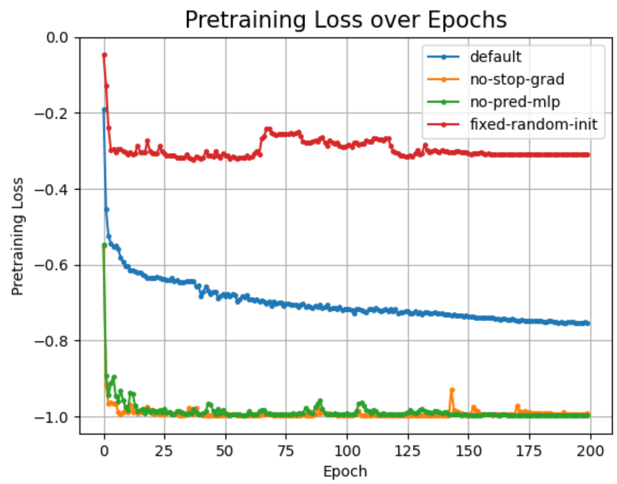
(a) alpha=0.0 (b) alpha=0.2 (c) alpha=0.4

(d) alpha=0.6 (e) alpha=0.8 (f) alpha=1.0

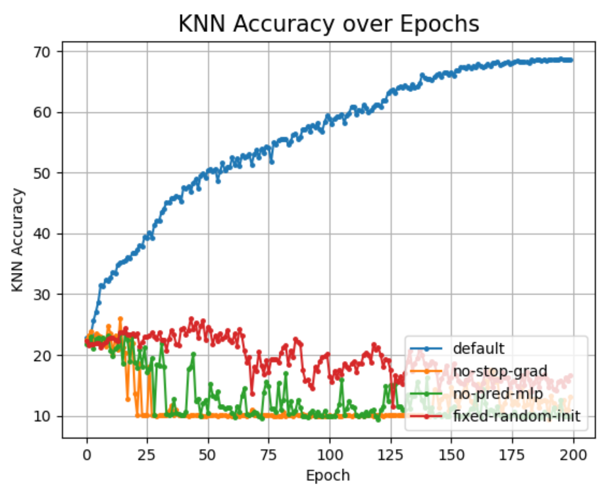
Figure 7: **Interpolation between two samples in pixel space.**

3 Problem 3: Self-Supervised Learning

Questions 4 and 5: Figure 8 shows the training results for our Self-Supervised model with default parameters as well as without gradient stopping and without an MLP predictor. We can see that gradient stopping is necessary and the model does not converge to a good solution without it. It's loss quickly falls and the model reaches a degenerate solution. We see that the model also fails to learn a good representation without an MLP predictor. We observe that the loss also reaches a minimum of -1 and the model converges to a degenerate solution. Finally, when using fixed random initialized weights for the predictor, it also fails to learn a good representation. However, in this case, it does not converge to a solution, as the loss stays high.



(a) Loss



(b) KNN Accuracy

Figure 8: **Self-Supervised Training.** This figure shows training results for our Siamese network trained with the default parameters as well as some ablations: no gradient stoping, no MLP predictor, and fixed random weights for the predictor.