HW3 report

Faezeh Pouya Mehr

May 9, 2023

1 Problem1

1.1 3

1.1.1 a

wall clock time of VAE training procedure for 30 epoch is plotted in Figure 1.

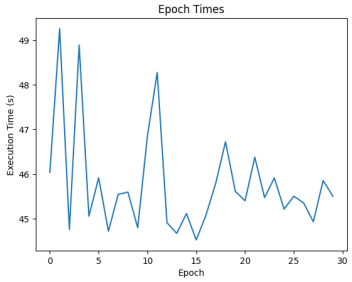

Figure 1: wall clock time of the training procedure of the VAE model

1.1.2 b

Several generated samples during training after every 5 epochs.

Figure 2: generated samples during training of the VAE model in epoch 1

Figure 3: generated samples during training of the VAE model in epoch 10

Figure 4: generated samples during training of the VAE model in epoch 15

Figure 5: generated samples during training of the VAE model in epoch 20

1.1.3 c

Figure 7: reconstructions from the test set during training of the VAE model in epoch 1

Figure 6: generated samples during training of the VAE model in epoch 25

Figure 8: reconstructions from the test set during training of the VAE model in epoch 10

Figure 9: reconstructions from the test set during training of the VAE model in epoch 15

1.1.4 d

as it can be seen in the Figures 12, 13, 14, 15, 16 generated samples are blurry and digit patterns are not obviously recognizable in almost all the generated samples it might be due to the complexity of the dataset or even choosing the not optimal hyperparameter setting.

Figure 10: reconstructions from the test set during training of the VAE model in epoch 20

Figure 11: reconstructions from the test set during training of the VAE model in epoch 25

Figure 12: first Sample from the VAE model at the end of training

Figure 13: second sample from the VAE model at the end of training

Figure 14: third sample from the VAE model at the end of training

Figure 15: forth sample from the VAE model at the end of training

Figure 16: fifth sample from the VAE model at the end of training

1.1.5 e

we can see in figure 17, that for class 0, latent variable 17 has the highest spike which means it contributed the most, then we have variables 0, 15, 16 in order and the others don't have much effect.

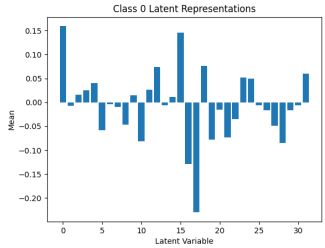

Figure 17: bar plots for class 0 to visualize the distribution of latent variables

For class 1, we see latent variables 19, 17 , 5 have the most contribution.

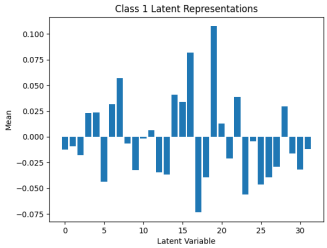

Figure 18: bar plots for class 1 to visualize the distribution of latent variables

For class 2, we see very high spikes for certain variables such as 0 and 27.

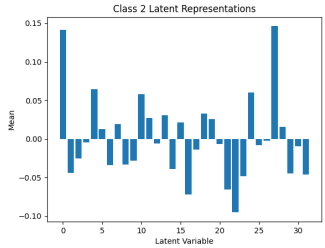

Figure 19: bar plots for class 2 to visualize the distribution of latent variables

For class 3, latent variables 18 and 11 respectively have very high spikes.

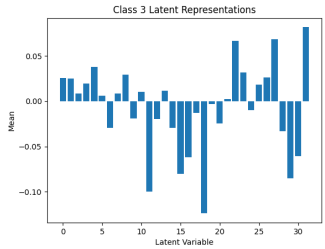

Figure 20: bar plots for class 3 to visualize the distribution of latent variables

For class 4, latent variable 21 has the highest spike and others do not contribute much.

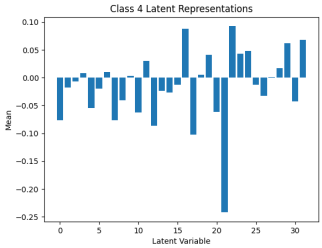

Figure 21: bar plots for class 4 to visualize the distribution of latent variables

For class 5, latent variables 8 and 15 contribute a lot.

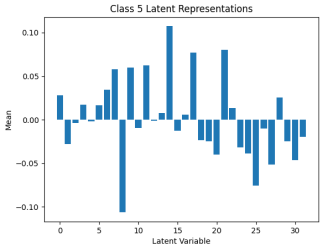

Figure 22: bar plots for class 5 to visualize the distribution of latent variables

For class 6, variables 25 and 8 contribute a lot.

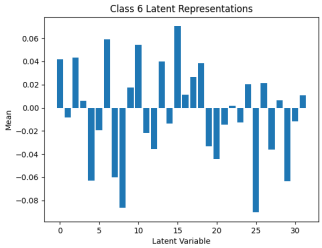

Figure 23: bar plots for class 6 to visualize the distribution of latent variables

For class 7, 29 21 and 10 th latent variables have high contributions.

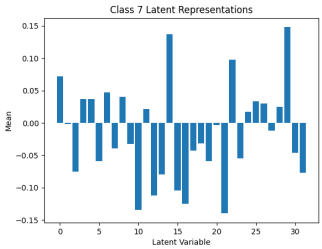

Figure 24: bar plots for class 7 to visualize the distribution of latent variables

For class 8, the 4th latent variable have the highest positive contribution follows by the 23th latent variable.

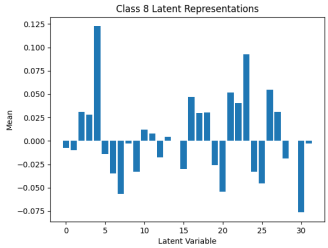

Figure 25: bar plots for class 8 to visualize the distribution of latent variables

For class 9, latent variables 19 and 21 contributed the most negatively followd by laetent variables 11, 13, and 29 which were equally contributed both in negative and positive way.

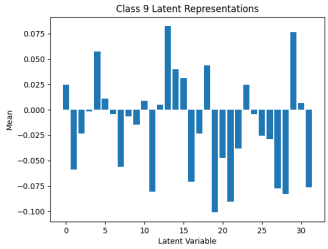

Figure 26: bar plots for class 9 to visualize the distribution of latent variables

1.1.6 f

In figure 27 we can see that digits are smoothly changing. in the first image digit 8 smoothly changes to 6 and in the second image digit 1 smoothly changes to 6, in the last one 4 changes to 6 smoothly.

Figure 27: interpolation between two points in latent space

2 problem2

2.0.1 a

Wall clock time of the DDPM training procedure for 30 epochs.

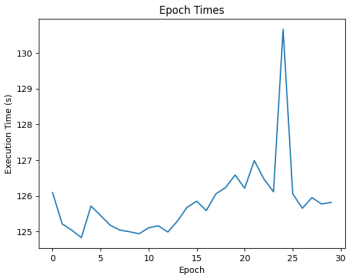

Figure 28: wall clock time of the training procedure of the DDPM

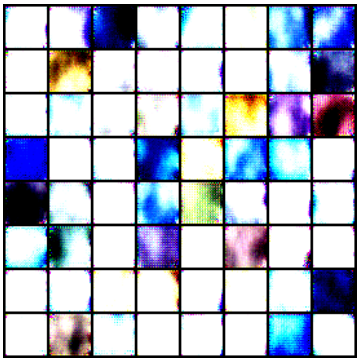

Figure 29: generated samples during training of the DDPM model in epoch 1

Figure 30: generated samples during training of the DDPM model in epoch 5

2.0.2 b

Figure 33: generated samples during training of the DDPM model in epoch 20

Figure 31: generated samples during training of the DDPM model in epoch 10

Figure 32: generated samples during training of the DDPM model in epoch 15

Figure 34: generated samples during training of the DDPM model in epoch 25

2.0.3 c

in the Figure 35 we run the forward diffusion process until timestep=30 and do the reverse process in the Figure 36. comparing these two figure, we can see that the model was able to restore the original image. while for time step= 120, as it can be seen in Figure 37 which is the forward process and Figure 38 which is the reverse process, the model was not able to restore the original image, the given image was 4 while the denoise one is not even close to 4 (may be 6).

Figure 35: add noise to samples at the timestep t = 30

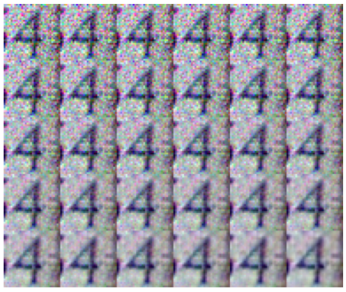

Figure 36: reconstruct the noisy sampled at timestep t = 30 using the reverse diffusion process

Figure 37: add noise to samples at the timestep t = 120

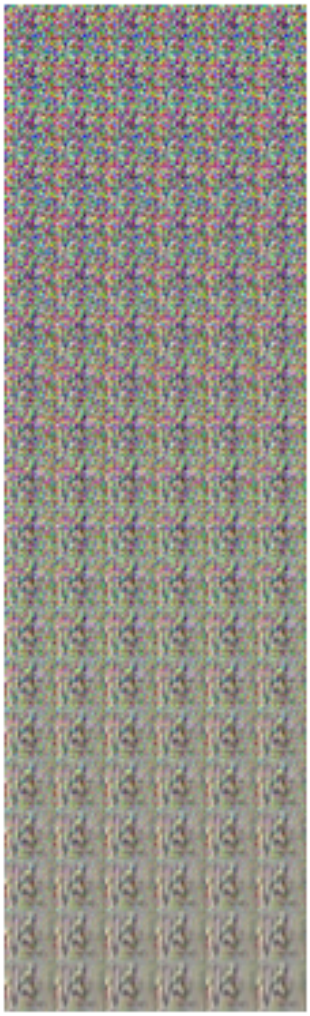

Figure 38: reconstruct the noisy sampled at timestep t = 120 using the reverse diffusion process

2.0.4 d

While most of the digits in the generated images are clear and easily identifiable, some of them are still blurry, resulting in a somewhat lower image quality overall. However, when compared to the results obtained with VAE, it can be observed that the generated samples using DDPM are slightly better.

Figure 39: first sample from the DDPM at the end of training

Figure 40: second sample from the DDPM at the end of training

Figure 41: third sample from the DDPM at the end of training

3 problem3

3.1 4

One of the key features of SimSiam is the use of a "stop-gradient" operation, where the gradients from the predictor network are stopped before they backpropagate to the projector network during

Figure 42: forth sample from the DDPM at the end of training

Figure 43: fifth sample from the DDPM at the end of training

training. This means that the predictor network does not directly update the parameters of the projector network, which is in contrast to traditional supervised learning methods where the gradients flow from the loss function to all parts of the network.

as it can be seen in the Figure 39 the accuracy of the model when using stop gradient is much better in compare to the model without using stop gradient which it shows a steadily improving accuracy. on the other hand in the model without stop gradient, the gradients from the predictor network are allowed to flow to the projector network hence the training process became unstable, which lead to poor performance or collapse of the learned representation.

in the loss plot for the model without stop gradient , the optimizer quickly finds a degenerated solution and reaches the minimum possible loss of 1 where this degeneration is caused by collapsing.

so as we expected stop gradient allows the projector network to learn a more stable and robust representation of the input data, which can lead to improved performance compared to when the gradients are allowed to flow directly between the predictor and projector networks.

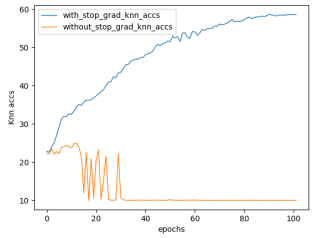

Figure 44: KNN accuracy against training epochs with and without stop gradient

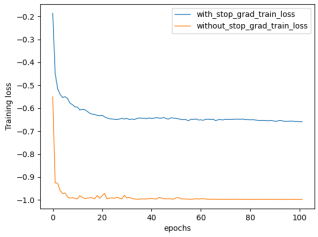

Figure 45: training loss against training epochs with and without stop gradient

3.2 5

3.2.1 a

as it can be seen in figure 41, this approach resulted in significantly worse performance compared to the original SimSiam model with a predictor network. The reason for this is that the identity mapping would effectively "turn off" the self-supervised learning component of SimSiam, as there would be no explicit encouragement for the projector network to learn a useful representation of the input.

In contrast, the predictor network in SimSiam serves as a regularizer that encourages the projector network to learn a more useful and informative representation of the input. By predicting the projection of one view of the input from the other view, the predictor network provides a signal to the projector network about what aspects of the input are important for the downstream task, which can improve the quality of the learned representation.

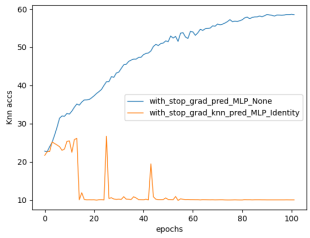

Figure 46: KNN accuracy against training epochs with stop gradient with and without pred MLP

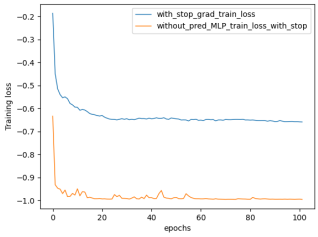

Figure 47: training loss against training epochs with stop gradient with and without pred MLP

3.2.2 b

In SimSiam, the projector network is used to map the input data to a lower-dimensional feature space that captures useful information for downstream tasks. The output of the projector network is then used to train a predictor network, which is typically a small neural network with a single output layer.

The dimensionality of the last layer of the projector network in SimSiam can then have an impact on the quality of the learned representations. By increasing the dimensionality of the last layer, the projector network may be able to capture more fine-grained details of the input data, leading to

improved representational power. This could result in better downstream task performance, as the learned representations are more informative and contain more useful information.

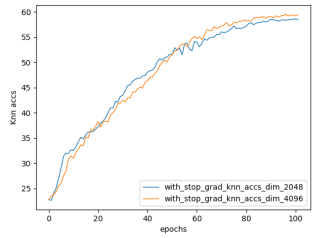

Figure 48: KNN accuracy against training epochs with stop gradient with dim=2048 and dim=4096

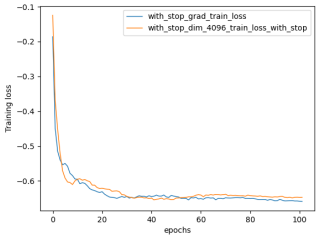

Figure 49: training loss against training epochs with stop gradient with dim=2048 and dim=4096

3.2.3 c

As compare to SimSiam, Barlow Twins does not require large batches nor asymmetry between the network twins such as a predictor network, gradient stopping, or a moving average on the weight updates. it also benefits from high-dimensional output vectors. On the other hand, based on the results presented in the previous parts SimSiam algorithm produced varying outcomes depending on certain hyperparameters, such as stop gradient, dimensionality of last layer of the projector network, or removing the predictor. Additionally, Barlow Twins is designed to work well with very high-dimensional output vectors, while SimSiam works best with smaller output dimensions. Due to these different characteristics, it is difficult to anticipate the exact behavior of the Barlow Twins method.

3.3 6

the heavy augmentations play as a "double-edged sword" for instance-wise self-supervised learning; without using negative pairs, the model would be easily misled by the heavily augmented views. Momentum encoder can increase model training stability since momentum would neutralize the misleading information from inconsistent representation. Although momentum encoder can ease heavily augmented views' side effects up to a point, DSSL is a more fundamental schema for handling various image transformations during instance-wise self-supervised learning

For each standard view (SV) augmented from the original robust image transformations, DSSL can generate various harder views (HV) derived from it by applying additional heavy augmentation policies ($\tilde{\mathcal{T}}$) beside the previous carefully selected standard data augmentation polices ($\mathcal{T}$).It means each light augmented view v would have various relevant harder augmented view v', and $\tilde{v}$ is derived from v.

These heavily augmented view has a larger d (deviation of augmented view's feature from the core-point of its relevant gold standard feature cluster)than its relevant standard view.

Given an input image I the data augmentation module in the the standard instance-wise self-supervised learning framework , produces augmented view pair set as :

$$V_{\mathcal{T}} = \{(t(I), t'(I))|t, t' \ \mathcal{T}\}$$

where t and t' are random augmentation sampled from T (augmentation policy set).

As there is no policy overlap between ($\mathcal{T}$) and ($\tilde{\mathcal{T}}$), we can construct a directed training pair collection

$$V_{\mathcal{T}<-\tilde{\mathcal{T}}} = \{(t(I), \tilde{t}(t(I)))|t\ \mathcal{T}, \tilde{t}\ \tilde{\mathcal{T}}\}$$

In this way, we can treat all augmented views of the same image as a partially ordered set (SV←SV, SV←HV) in terms of d. An asymmetric loss is introduced to encourage the representation of each heavily augmented view (HV) to be close to its relevant source standard view (SV).

In this way, the feature cluster for all augmented views can be presented as non-convex, rather than the K-means convex clustering, the whole cluster is tightened. Moreover, DSSL discards the instance-wise selfsupervised learning among RVs to bypass the issue of low mutual information among HVs(Since the mutual information among views with large d is usually low, contrasting these views leads to missing information and results in poor performance in downstream tasks). As a result, more augmentation policies can be introduced to enrich the information of the whole embedding space but keep the instances still well separated.

in the current SimSiam model we are also using data augmentation (RandomResizedCrop, RandomHorizontalFlip, RandomGrayscale) as was run in the cell below:

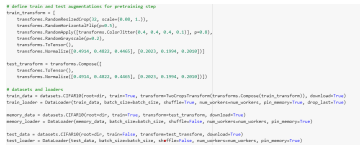

Figure 50: where SimSiam model used data augmentation in our model

3.4 7

Collapse is prevented by ensuring that the assignment of samples to clusters is as uniform as possible. for example BYOL and SimSiam exploit several tricks: batch-wise or feature-wise normalization, a "momentum encoder" in which the parameter vector of one branch is a low-pass-filtered version of the parameter vector of the other branch or a stop-gradient operation in one of the branches.
alternative class of collapse prevention methods relies on maximizing the information content of the embedding.These methods prevent informational collapse by decorrelating every pair of variables of the embedding vectors.

The basic idea is to use a loss function with three terms: **Invariance**: the mean square distance between the embedding vectors. **Variance**: a hinge loss to maintain the standard deviation (over a batch) of each variable of the embedding above a given threshold. This term forces the embedding vectors of samples within a batch to be different. **Covariance**: a term that attracts the covariances (over a batch) between every pair of (centered) embedding variables towards zero. This term decorrelates the variables of each embedding and prevents an informational collapse in which the variables would vary together or be highly correlated.

Variance and Covariance terms are applied to both branches of the architecture separately, thereby preserving the information content of each embedding at a certain level and preventing informational collapse independently for the two branches. The main contribution of this paper is the Variance preservation term, which explicitly prevents a collapse due to a shrinkage of the embedding vectors towards zero. The Covariance criterion is borrowed from the Barlow Twins method and prevents informational collapse due to redundancy between the embedding variables

3.5 8

The main difference between BYOL and SimSiam lies in their approach to representation learning.

- BYOL is a contrastive learning method that uses a "teacher" network to generate predictions about the future state of an input, and a "student" network that tries to match these predictions. The student network learns to map inputs to a latent space that preserves semantic information, allowing it to generalize to unseen data. The role of the teacher network is to provide a "moving target" for the student network to follow during training, encouraging it to learn robust representations that are invariant to certain transformations such as rotations or translations.

- In contrast, SimSiam is a simpler method that does not use a teacher network. Instead, it relies on a standard feedforward neural network that maps inputs to latent representations, which are then used to predict the original input. The network is trained to minimize the difference between the predicted and original input, while also maximizing the similarity between pairs of augmented views of the same input. The role of the latent representations is to capture meaningful information about the input that can be used to reconstruct it accurately.

In both methods, data augmentation plays a crucial role in increasing the diversity of the training data and improving the robustness of the learned representations. BYOL and SimSiam both use a similar set of data augmentations, such as random crops, flips, and color jittering, to increase the variation in the input data.

3.6 9

Exploiting asymmetry in Siamese representation learning can be relevant because it can help to capture subtle differences between input data points, which can be difficult to distinguish using symmetric models. By exploiting asymmetry, we can introduce a small amount of randomness into the process of comparing the subnetwork outputs, which can help the network to learn more robust representations that are less sensitive to variations in the input data.This can be particularly useful when trying to compare images with different backgrounds, lighting conditions, or other variations that can make it difficult to identify similarities based on symmetric models alone.s

In the context of SSL, its algorithm can benefit from this approach because they are designed to learn more diverse and informative representations of data without explicit supervision SSL algorithms leverage unlabeled data to learn representations, which can then be fine-tuned for downstream tasks. By using asymmetric Siamese networks, SSL algorithms can extract different types of information from the input data, leading to more diverse representations.

References