Performance analysis of Semi-supervised learning in the small-data regime using VAE

Abstract

Extracting the large amount of data from the biological spices involves radiation issues and the image processing in the small-data regime is one of the critical challenge in the current research. In this work, we applied an existing algorithm named Variational Auto Encoder (VAE) that pre-trains the input to represent the features in the lower-dimension for the small-data regime input. Later, we will fine-tune the network by fixing the encoder weights of the VAE. Here we will present the performance analysis of the VAE vs. conventional approach and also analyze the performance with respect to number of labeled samples in the semi-supervised learning for CIFAR10 dataset. In addition, we identified that with the minimum number of samples VAE performs better than the state-of-the-art results with small-data regime. Lastly, we will extend this idea, on the medical imaging that will give better results than the traditional methods.

I. Introduction

Artificial neural networks (ANN) become popular due to their success on image classification, feature extraction, object recognition and detection [1]. In ANN, with deep learning user can form the complex models which are composed of multiple hidden layers. Deep learning methods have enhanced the state-of-the-art performance in object recognition & detection and computer vision tasks. Convolutional neural network (CNN) is a specific deep learning architecture for processing the data that become known for handwritten numbers recognition (modern version LeNet-5). Later AlexNet with CNN reduced the error rate by half compared to its previous competitors and CNN become the most promising approach for recognition & detection and computer vision.

In the small-data regime, the test accuracy using traditional CNN is not accurate. This is the typical case that involves with the biological spices and its image processing. Exposure to the radiation for the longer time (to capture more images) is harmful to the species. On the other-side, we can simulate large set of inputs (by adding noise) but again the test accuracy is limited by the target values. In this case for the semi-supervised learning on the limited targets can't produce accurate model prediction since not enough target values are used in the model generation.

To address this problem, there exists a framework called “Auto Encoder” (AE [4]) that uses all the input data. In the AE, each input is considered and restore the same at the output using encoding and decoding operations. This step is called “pre-training”. To explain in simple words: in this framework, we represent the principle features in the latent space that is smaller in dimension than the input space. This step is called "Encoding". After encoding, the latent space image is restored using the decoding operation. In this step, the neural network (NN) learns the weights by optimising the loss function (*i.e.*, mean square error (MSE)) between the input and the restored output. At this step, we have the weights for encoder and decoder parts of the NN.

From the pre-training part it is clear that, we can represent the input features in the latent space. Now for each training image that has target function, pass through this latent space and get the lower-dimension version of the image and then pass through any other NN (it can be fully-connected or convolutional nets). In this neural network, we train the network weights to optimise the loss function between the inputs and targets. The loss function, for the classification is the cross-entropy and for the regression it is the MSE (mean square error between input image and restored image). This step is called "fine-tuning". [Note: In this step the encoder weights are fixed (which are from the pre-training step) and only the decoder weights are tuned]

In this trained model, from the steps of pre-training and fine-tuning, the weights of the encoder and decoder are found. We will pass a test image through this model and output of this model is either a class value or a restored image.

In this report, we will implement a slight different version of the AE called "Variational Auto Encoder" (VAE [4]) that randomizes the latent space dimension. In the VAE, instead of finding the mapping functions from input - latent space - output, it finds the equivalent probability distribution at the latent space. Now adding noise in the latent space with VAE can generate large amount of the restored input images. This step avoids the "Data Augmentation" used for the present data-sets. For the data-augmentation, we apply the Gaussian noise on the input images to generate large training dataset, so that predictive model is able to with stand the noise for the input test image in the test dataset.

II. Literature review

This section explains the working principle of auto-encoder [8] and variational auto-encoder [6].

A. Auto Encoder

An autoencoder is a type of ANN used to learn efficient data coding in an unsupervised manner. The aim of an autoencoder is to learn a representation for a set of data, typically for dimensionality reduction, by training the network to ignore signal noise. Along with the reduction side, a reconstructing side is learnt, where the autoencoder tries to generate from the reduced encoding a representation as close as possible to its original input. An autoencoder always consists of two parts, the encoder and the decoder, which can be defined as transitions ϕ and ψ such that [4]:

$$\phi : X \to F \tag{1}$$

$$\psi : F \to Y \tag{2}$$

$$\phi, \psi = argmin_{\phi,\psi} \| X - (\psi o \phi)(X)^2 \| \tag{3}$$

[9] where the given input X and predicted output is Y. If the feature space F has lower dimensionality than the input space X, then the feature vector $\phi(x)$ can be regarded as a compressed representation of the input X. Now freeze the weights in the encoder part and

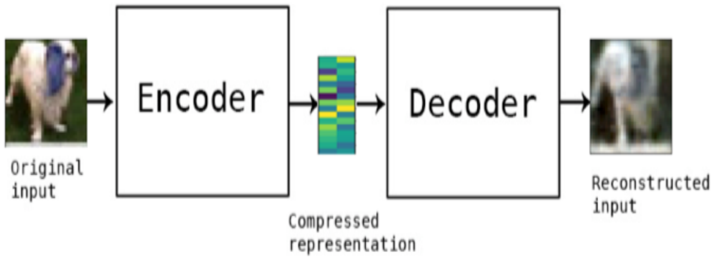

Figure 1: Autoencoder block diagram

add other simple neural network to train the network. Since the output is the class values with probability, cross-entropy is used as the loss function. From this, it is clear that using AE, we

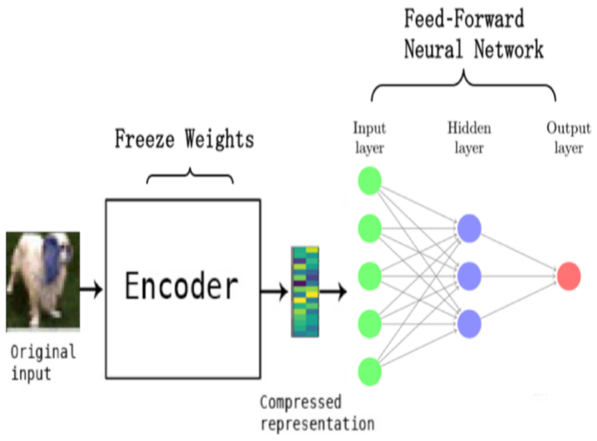

Figure 2: AE + a simple classifier using fully connected layer block diagram

can represent the input features in the lower-dimension and adding the other network is useful to predict the class for the given data.

B. VAE

To extend the idea used in the AE, there is a variation called VAE [7]which uses the "KL-divergence" between the predicted probability distribution function and actual posterior distribution in the latent space where as traditional AEs try to find the accurate mapping functions in the encoder (between input and latent space) as well as in the decoder (between the latent space and output). Using VAE, we can generate a large dataset by adding the noise in the latent space which is similar to input data augmentation (adding noise to images to increase the number of examples in the input dataset).

VAE uses a variational approach for latent representation learning, which results in an additional loss component. It assumes that the data is generated by a directed graphical model $p(\mathbf{X}|\mathbf{Z})$ and that the encoder is learning an approximation $q_\phi(\mathbf{Z}|\mathbf{X})$ to the posterior distribution $p_\theta(\mathbf{X}|\mathbf{Z})$ where ϕ and ψ denote the parameters of the encoder (recognition model) and decoder (generative model) respectively. The objective of the variational autoencoder in this case has the following form:

$$\mathcal{L}(\phi, \theta, \mathbf{x}) = D_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})\right) - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left(\log p_\theta(\mathbf{x}|\mathbf{z})\right) \tag{4}$$

where D_{KL} stands for the KL-divergence [9]. In the VAE, the principle is that minimizes the loss between input and the restored image along with the loss generated by the latent space to represent the features in the input images. In this VAE, "reparameterization trick" is used to model the uncertainty generated by the latent space.

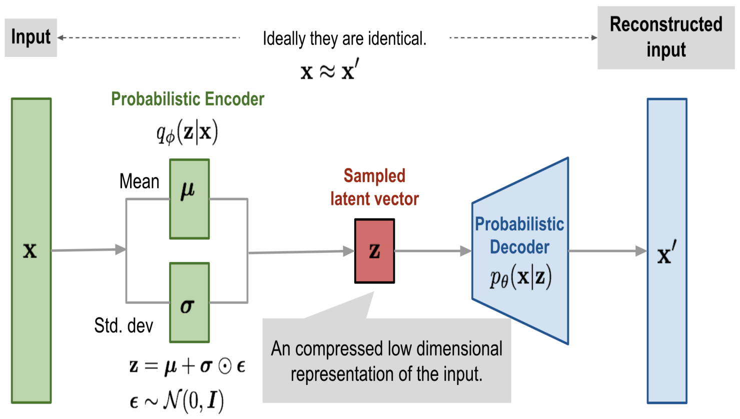
Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
μ
Std. dev
σ
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
$\mathbf{z} = \mu + \sigma \odot \epsilon$
$\epsilon \sim \mathcal{N}(0, I)$
An compressed low dimensional representation of the input.

Figure 3: VAE block diagram

The next section explains about the methodology and dataset details used in this report.

III. Methodology

In this section we enumarate the details of AE and VAE. Consider a surrogate model $y = f(x, \theta)$ which is trained using limited simulation data $\mathcal{D} = \{x^i, y^i\}_{i=1}^N, \{x^j\}_{j=N+1}^D$. Where the input data, $x^i \in \mathbb{R}^{d_x \times H \times W}$ is the input CIFAR10. Here H and W are the height and width respectively and d_x is the number of dimensions for the input x at one location. x^j is the additional data utilized for pretraining the model. $y^i \in \mathbb{R}^1$ is the classified result. θ are the model parameters and N is the total number of training data utilized during fine-tuning and D is the total number of data utilized for pre-training. Since it is semi-supervised setting. We pre-train the model with the input data $\mathbb{R}^{d_x \times H \times W}$ and the perform image classification problem $\mathbb{R}^{d_x \times H \times W} \to \mathbb{R}^1$. For both the pre-training and fine tuning, We used stochastic gradient descent with Adam to update the network weights and biases. The simulation was performed using Pytorch.

A. *AE model*

In this project, we will implement a simple auto encoder (like U-Net [11]) to pre-tune the network weights. After the encoder tuning, add a simple NN (a single or multiple fully connected layer) for fine-tuning to classify the input images after passing through the encoder part. The loss function will be the MSE for the pre-tuning and cross-entropy for the fine-tuning section.

B. *VAE*

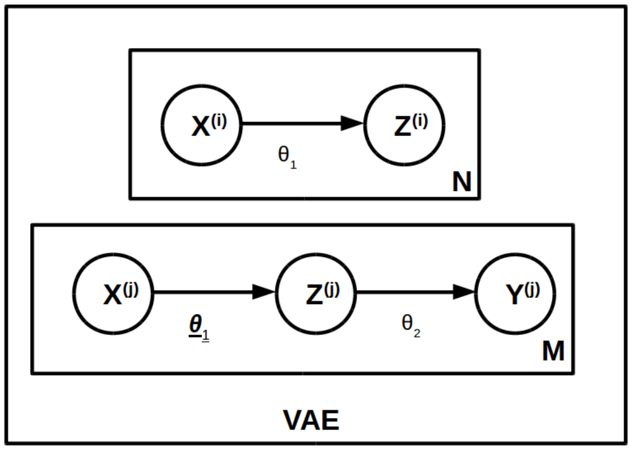

Figure 4: Network architecture: pre-training with VAE(first row) In the second row, parameter θ is underlined and bold to indicate that these parameters are freezed when the fine-tuning the network

1) VAE pre-training: In this project, we will implement dense-net [12] version of VAE for the pre-training part. Dense-net contains the encoder and decoder blocks along with the dense-layer that has the simple and complex features.

2) VAE fine-tuning: In this project, we will implement simple fully connected layer to classify the input images on the CIFAR10 [3] data. This is due to the expectation of the latent space is smaller (either 4 $\times$ 4 or 8 $\times$ 8) (image size is 32 $\times$ 32). If the number of channels are large at the latent space, we will add another one or more fully connected layers before taking the classification.

C. Dataset details

We will perform the simulations on the CIFAR10 dataset with ten image classes. CIFAR10 dataset contains ten different classification images like "airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck" with three input channels $(C = 3)$ of size 32 $\times$ 32 (W $\times$ H). CIFAR10 dataset has 50000 training images and 10000 test images.

IV. Results

In this section we enumerate the results obtained using CIFAR10 data.

A. AE vs VAE comparison

The results comparison between AE and VAE for CIFAR10 dataset is given in Figure: 5.

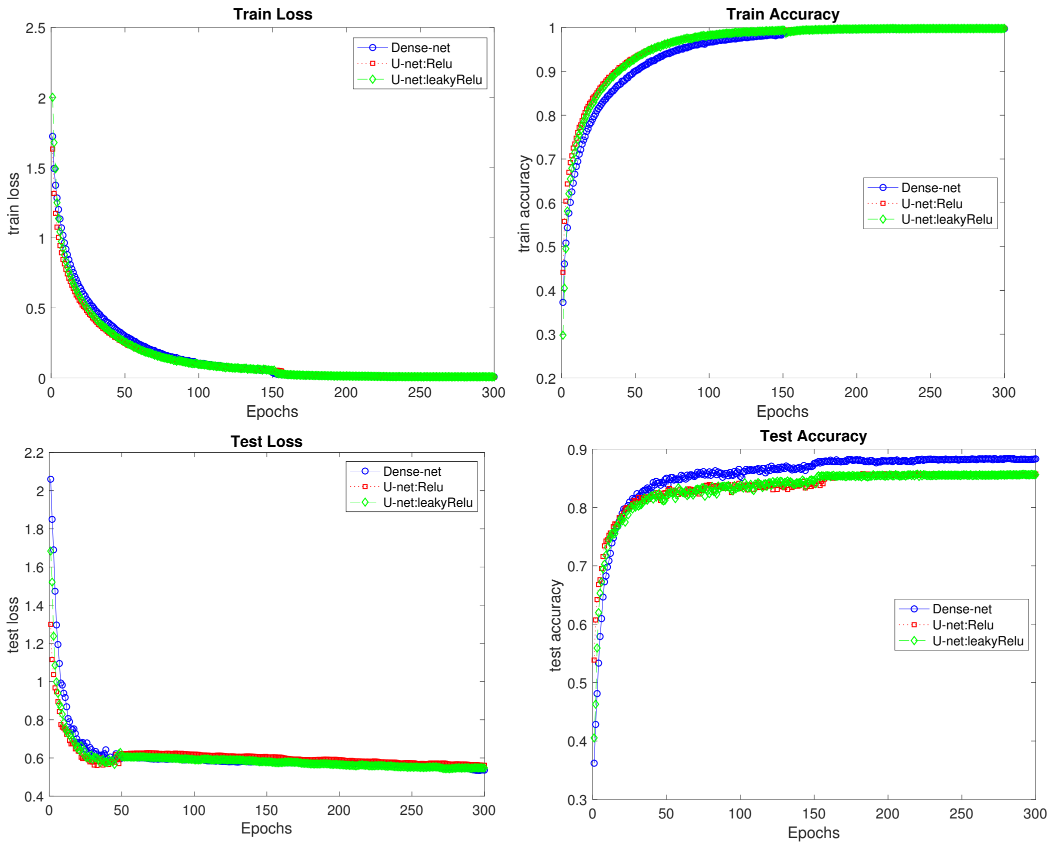

Figure 5: From top to bottom: Training loss (a), Training accuracy (b), Test loss (c) and Test accuracy (d) between Auto encoder (U-Net) and variational auto encoder models (Dense-net).

Results Discussion: Increase the number of convolution layers improves the performance. Network with Relu and Leaky-Relu activation's has similar performance and Compared to AE, VAE performs better in test accuracy. Adding batch-normalization improves the performance of the VAE.

B. VAE Performance

We only consider the latent dimension to be 6400,10000 and 14,400 in VAE. In order to evaluate the performance of the model for above three latent space, we consider the distribution estimated for the values at various pixel location.

1) VAE Pre-training: For the results presented in this section, we have implemented dataset with 50,000 $\{x^k\}_{k=1}^{50000}$ for pre-training and the test set consist of 10000, $\{x^k\}_{k=1}^{10000}$. Adam optimizer was used for training 100 epochs, with learning rate of 1e-4 and a plateau scheduler on the test RMSE. Batch size is always smaller than the number of training data. In this work, batch size of 16 for pre-training was considered. Weight decay was set to 1e-3 for pre-training.

We consider Equation: 4 loss function to evaluate the trained model on test data and also to monitor the convergence.

From figure 6, we observe that the solution is converged after 50 epochs and most importantly the loss for the three latent space is almost similar.

From figure 7, we observe that even when the latent size is small (Batch $\times$ 100 (channels) $\times$ 8 $\times$ 8) and (Batch $\times$ 100 (channels) $\times$ 10 $\times$ 10) the reconstructed density estimate is close to actual input data. The PDF with the latent size 14400 is closer to the actual input and also 6400 & 10000 latent space. Since, all the latent space yields almost similar output,we fine-tune and compare the classification accuracy in the next section.

2) VAE Fine-tuning: In this section we freeze the parameters (weights and bias) used in the pre-training stage and fine tune the parameters (weights and bias) in the classification network. For this problem, we consider small data from given in CIFAR10 dataset and use fully connected layers to perform classification. As cross entropy loss function it is commonly used for all the classification problem and hence, we implemented cross entropy as to measures the performance of a classification mode.

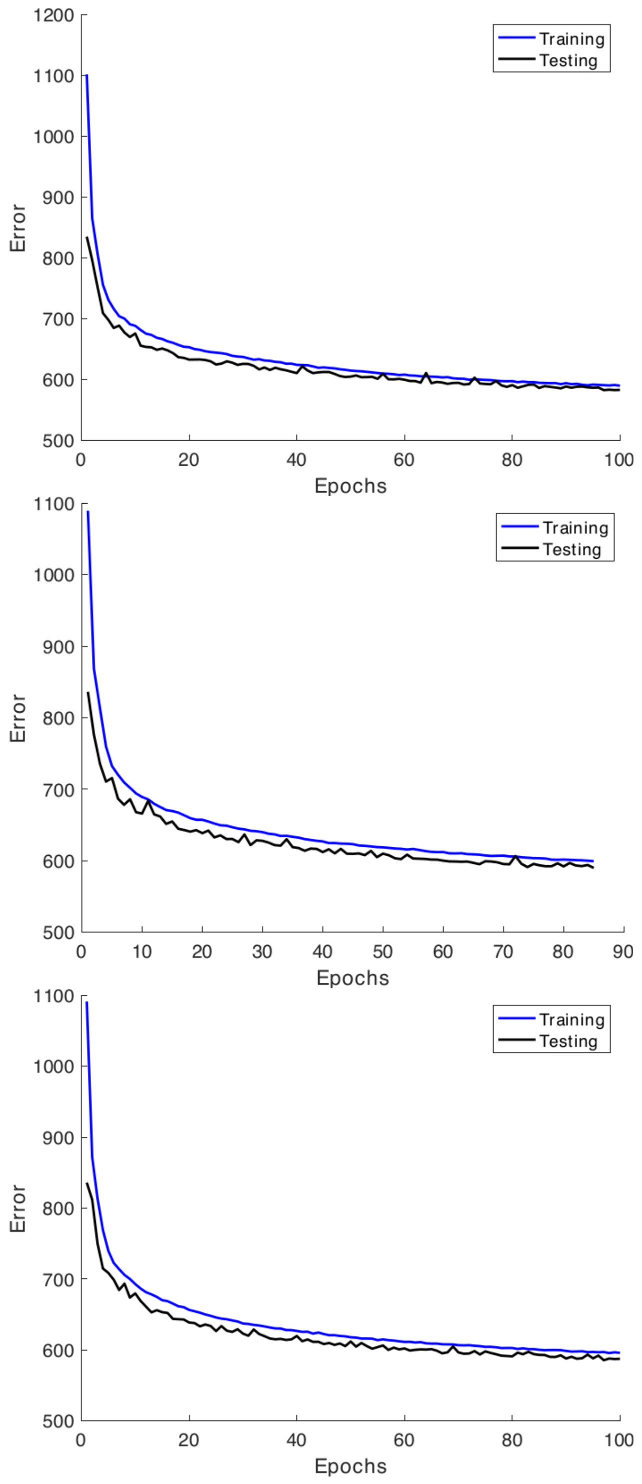

Figure 6: Error v/s epoch for 6400 latent space (top), Error v/s epoch for 10000 latent space (middle) and Error v/s epoch for 14400 latent space (bottom)

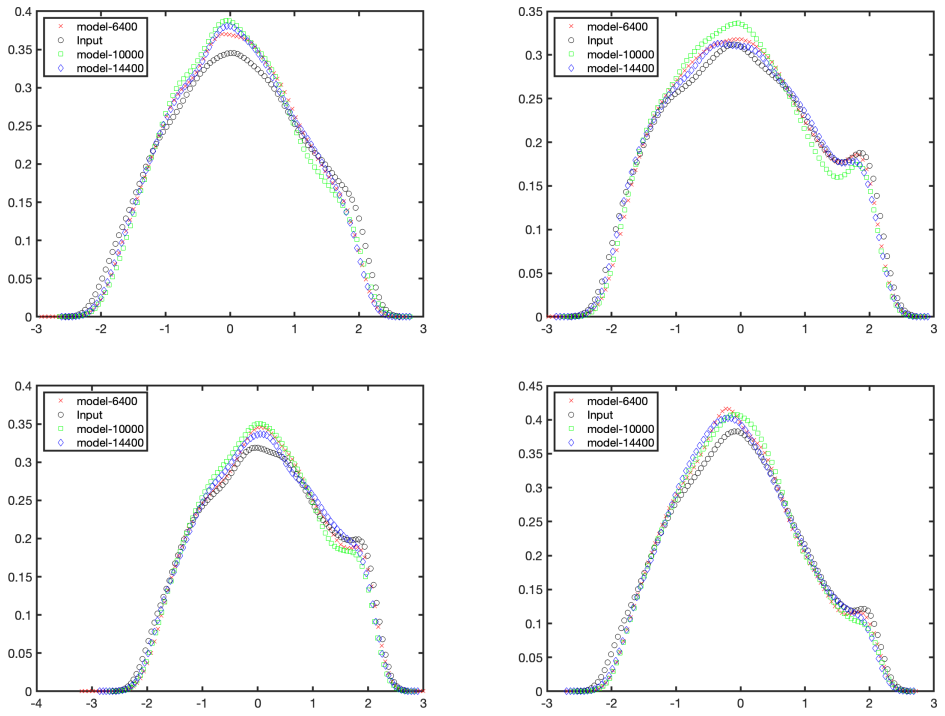

Figure 7: Distribution estimate for the values at various location of the square domain for 6400, 10000 and 14400 latent space

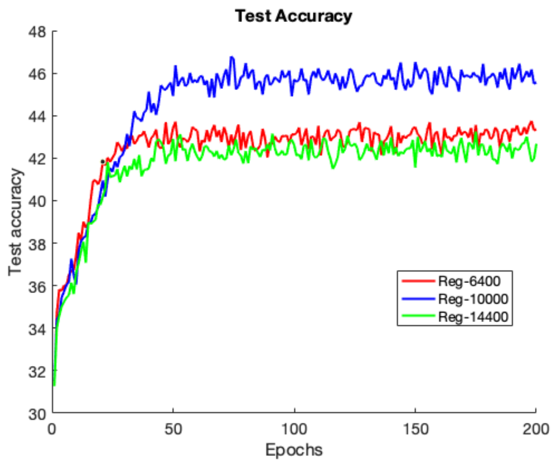

Figure 8: Fine tuning results with three different latent space models

Form figure 8, we observe the latent space 10000 yields better accuracy than other two models (6400 and 14400). This is because of model complexity [1]. The smaller the latent space the test accuracy is low due to insufficient features to classify the data whereas for the large latent space, the test accuracy is also low.

V. Mile-stones covered in the project

In this section we present how to evaluate the performance of the VAE and the comparison with the existed methods. We identified the network configuration with the following steps as part of the project on the CIFAR10 dataset using VAE.

- Identify the VAE network (number of layers, and each layer components) to represent the latent space (this is for the pre-training part). This was done until the second milestone.
- Identify the optimized latent space dimension in the VAE (in the pre-train part). This was achieved at the draft paper timeline.
- Identify the fine-tuning network (number of layers, and each layer components). This was delivered for the final paper.

We got the following results from the VAE network as part of the project on the CIFAR10 dataset.

- Plot that shows the test accuracy vs. number of samples in the semi-supervised learning
- Plot the test accuracy vs. latent space dimension for a fixed number of samples in the semi-supervised learning
- Comparison of test accuracy of the VAE vs. conventional methods (for different number of samples in the supervised learning)

VI. Resources Required

The main resource is the computational power of the equipment used for training. We used simulations to show the proof-of-concept using Python along with machine learning libraries (Pytorch) for the performance of VAE in the small-data regime. We will use GPU nodes for the training and testing of the VAE on the CIFAR10 dataset.

VII. Conclusions and Future work

The present document outlines the development of surrogate model for semi-supervised problem. In this work, we have implemented VAE as the for pre-training model and a feed forward deep learning model for the classification. The results obtained for different number of latent space are presented. It was observed that there is a slight improvement in the test accuracy when the latent space is 10000 in comparison with latent space of 6400 and 14400.

For future work, Bayesian approach can be explored. Due to a limited amount of data, it is necessary to model appropriate surrogate, since quantifying the epistemic uncertainty induced by limited data and hence, a Bayesian probabilistic approach is a natural way of addressing this challenge.

References

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. Deep Learning. The MIT Press.

[2] Zhang, Yide, Yinhao Zhu, Evan Nichols, Qingfei Wang, Siyuan Zhang, Cody Smith, and Scott Howard. "A Poisson-Gaussian Denoising Dataset with Real Fluorescence Microscopy Images." arXiv preprint arXiv:1812.10366 (2018).

[3] Alex Krizhevsky and Vinod Nair and Geoffrey Hinton. "CIFAR-10 (Canadian Institute for Advanced Research)".

[4] https://en.wikipedia.org/wiki/Autoencoder

[5] https://benchmarks.ai/cifar-10

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

[7] https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

[8] Baldi, Pierre. "Autoencoders, unsupervised learning, and deep architectures." In Proceedings of ICML workshop on unsupervised and transfer learning, pp. 37-49. 2012.

[9] https://xuan-li.github.io/post/autoencoder_review/

[10] https://www.deeplearningbook.org/contents/autoencoders.html

[11] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." In International Conference on Medical image computing and computer-assisted intervention, pp. 234-241. Springer, Cham, 2015.

[12] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708. 2017.

[13] Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M. and Aila, T., 2018. Noise2noise: Learning image restoration without clean data. arXiv preprint arXiv:1803.04189.