Exploring VAEs to Empower Self-Supervised Learning

Soham Mody
srm699@nyu.edu

Subhadarshi Panda
sp5704@nyu.edu

Tanya Nabila
tn1050@nyu.edu

Abstract

Image classification is one of the tasks where neural networks have out-performed all other machine learning techniques. However, a neural network can perform well only when the amount of labeled data available is huge (usually millions of training samples). Such an enormous amount of labeled data is difficult to get. Unlabeled data, on the other hand, is available in plenty. In this paper, we make use of a variational auto-encoder (VAE) to use unlabeled data to improve image classification accuracy. We make use of the provided supervised and unsupervised datasets to build deep learning models.

1 Introduction

Image classification is the task of assigning a class label from a given set of labels to an input image. For image classification using a deep neural network, a huge amount of labeled training data is required [4]. For some image classification tasks, the labeled data is abundantly available. However, if there is only a small amount of supervised data available, the deep neural network might not learn their distribution well, thereby misclassifying the test samples more often.

In this problem, when there are a large number of classes to predict from so, there should ideally be a large number of training samples *for each class*. But, in the supervised data given to us, there are 1000 possible classes and only 64 labeled samples are provided for each class.

Even though there is not a lot of labeled data, a huge number of images $(512,000)$ without any labels have been provided to us as is the case in the real world where a massive amount of unlabeled data is present. So, the objective is to improve classification accuracy using this unsupervised data.

We apply unsupervised and semi-supervised learning techniques in this project in an attempt to improve the performance. First, we used the unsupervised data to train a variational auto-encoder [2]. Then, we use the pretrained hidden layer weights from the encoder part of the VAE to train a convolutional neural network (CNN) [3] based classification model.

2 Related Work

There were two main lines of work which has inspired our approach in this project. We first started to seek ways to take advantage of the unsupervised data and learn their latent representations through generative models such as VAEs. It is a class of widely used generative models capable of learning latent representations of unsupervised data. The second line of work is the use of latent variable model and extend further to train a semi-supervised classification model from labeled data.
A VAE consists of an encoder which compresses the image to a low dimensional latent space, a decoder which tries to convert this latent space back to the image and a reconstruction loss which calculates how much information was lost in this reconstruction from the latent space. The VAE that was proposed consisted of a feed-forward ConvNet, coupled with a feed-back DeconvNent. Recent works using VAE also includes experiments, which addresses the possibility of using the latent

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

space to train the image classifier [1, 5, 6]. These studies mainly attempt to satisfy two objectives: (i) to learn latent representations of the unlabeled dataset and (ii) to leverage both supervised and unsupervised data to learn this representation in a combined framework[7]. These studies have raised the question of the possibility of using a trained encoding layer from the unsupervised dataset with a classifier model and fine-tuned using the labeled dataset, which was we have explored in this project.

3 Transfer Learning Using VAE

VAEs [2] are generative models which consist of an encoder, a decoder, and a loss function. The encoder is a neural network whose input is a data point x and its output is a hidden representation z and it has weights and biases θ.

The lower-dimensional space is stochastic: the encoder $q_\theta(z|x)$ outputs parameters to $q_\theta(z|x)$ which is a Gaussian probability density. We can sample from this distribution to get noisy values of the representations z. The decoder is another neural network whose input is the representation z. It outputs the parameters to the probability distribution of the data and has weights and biases ϕ. In the decoder $p_\phi(x|z)$, information is lost because the sample goes from a smaller to a larger dimensionality. We measure this information loss using the reconstruction log-likelihood $\log p_\phi(x|z)$. This measure tells us how effectively the decoder has learned to reconstruct an input image x given its latent representation z.

The loss function of the VAE is the negative log-likelihood with a regularizer. Because there are no global representations that are shared by all data points, we can decompose the loss function into only terms that depend on a single data point. The loss for a single data point is l_i. The total loss is then $\sum_{i=1}^{N} l_i$ for N total data samples.

We first train a VAE by using the unsupervised data. Then we use the trained weights of the encoding layers of the VAE to fine tune on the supervised data for classification. Since we use the pretrained weights of the VAE to do classification, this work falls under transfer learning. Specifically, we remove the decoding layers of the VAE and add layers in two different settings:

(1) We add feedforward layers to do classification.

(2) We add convolutional layers and finally feed forward layers to do classification.

For each of the above settings, we examine fine-tuning by freezing the pretrained VAE weights or by making the pretrained VAE weights learnable.

4 Experiments

We used the provided data sets containing square color images of size $96X96$. The data sizes were 64k labeled training images, 64k labeled validation images and 512k unlabelled images.

We used VAE as the latent variable model for image reconstructions in an unsupervised manner. The learned VAE is then disentangled and treated as a feature extractor on top of which a classifier is learned. This model is trained and evaluated on the training dataset as we experiment with different hyper-parameters and network structure.

The architecture of the VAE's encoder and decoder may vary, and in this project, we used a feed-forward ConvNet, coupled with a feed-back DeconvNet. Our VAE consists of two convolutional layers and a fully connected layer in the encoder followed by a fully connected layer and 2 deconvolutional layers in the decoder part of the architecture. Once the VAE encoder is trained and we have feature representation of the images, the weights of the first 3 layers is then used as a feature extractor for the image classification task.

We have experimented with 2 types of architectures for the classifier part of our model which are a feed-forward network and a convolutional network. Another parameter we focused on in our experiments was whether to freeze the weights of the encoder part of our model or not.

Our feed-forward network classifier takes the trained encoder part of the VAE and simply extends it with 2 fully connected layers. Our convolutional classifier, on the other hand, extends the chopped VAE with a convolutional layer, a max pooling layer, another convolutional layer, and finally a fully connected classifier layer.

Figure 1: **VAE reconstructions.** The first row shows the original input images. The second row shows the reconstructions after 1 epoch. The third row shows the reconstructions after 180 epochs. Reconstructions are shown for 8 different input images.

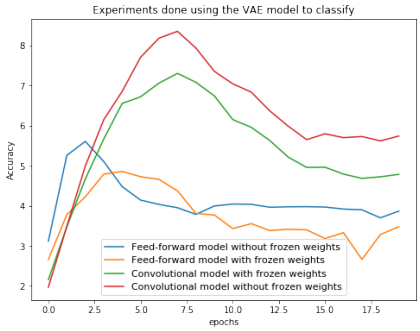

Figure 2: Validation accuracy during training

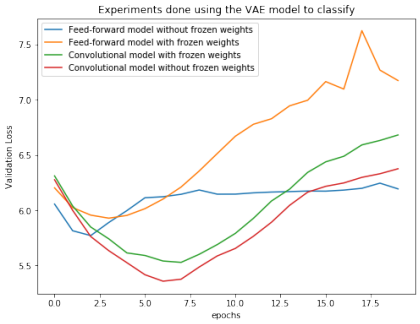

Figure 3: Validation loss during training

5 Analysis

After sufficiently training the VAE, we qualitatively evaluated the reconstructions. Figure 1 shows the reconstructions obtained after 1 and 180 epochs of training the VAE.

After training the VAE model, we used the pretrained weights for classification. See the accuracy and loss during training in Figure 2 and Figure 3. We find that the convolutional classifier without freezing weights achieves the best validation accuracy (and also the best validation loss). This is probably because the locality and compositionality of the input images still remain in the latent space (encoded vector). This property is exploited by the convolutional layers. Also having learnable VAE weights performs better than freezing VAE weights.

We compare our best model with the baseline. Figure 4 shows the validation accuracy for 50 and 100 epochs of training. We see that our best model outperforms the baseline within the first 10 epochs of training.

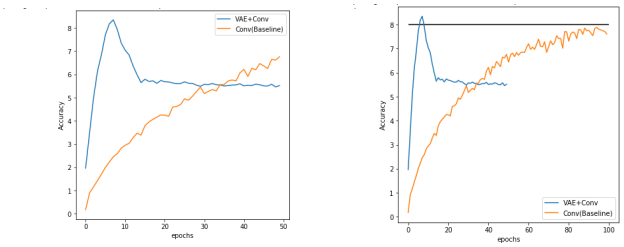

Figure 4: Validation accuracy for 50 and 100 epochs of training

Number of labeled samples per class	Validation top 1 accuracy(%)	Validation top 5 accuracy(%)
1	8.41	20.7
2	8.41	20.7
4	8.41	20.7
8	8.41	20.7
16	8.41	20.7
32	8.41	20.7
64	8.41	20.7

Table 1: Validation Performance of training with 1, 2,...,64 labeled samples per class

The validation performance of training our best-performing model (VAE+CNN) with a different number of labeled samples per class is shown in table 1. The validation top-1 and top-5 accuracy are the same for all the different number of samples. This was a very surprising result for us. It shows that the VAE model has learned the essential features required for classification. Upon using pretrained VAE for classification, it gave a top-5 accuracy of 20.7% with just 1 labeled sample per class. A probable reason for our model's accuracy not increasing further with the number of labeled samples could be that it's learning was already maxed out due to the shallow architecture of our ConvNet classifier. This probably prohibits the combined model from doing a better classification.

6 Conclusion

We have a lower final accuracy as we made the mistake of running the analysis with a relatively shallow model instead of using a deep and proven architecture like ResNET. But, even though our accuracy is not as high as some of the teams on the leaderboard, using a VAE to pretrain the classifier did help in improving the accuracy and we got an improvement of 1.55% top-5 accuracy over our baseline model. Interestingly, our model achieved higher accuracy than the baseline using just 1 labeled sample per class for supervised training. We think this is probably because the VAE learns almost all the features essential for classification.

References

[1] Marouan Belhaj and Weiwei Pan. Deep Variational Transfer: Transfer Learning through Semi-supervised Deep Generative Models.

[2] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[4] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Comput.*, 29(9):2352–2449, September 2017.

[5] Shideh Rezaeifar, Olga Taran, and Slava Voloshynovskiy. Classification by Re-generation-Towards Classification Based on Variational Inference.

[6] Chris Varano. Disentangling Variational Autoencoders for Image Classification.

[7] Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann Lecun. STACKED WHAT-WHERE AUTO-ENCODERS.