Disentangling Variational Autoencoders for Image Classification

Chris Varano
A9
101 Lytton Ave, Palo Alto
cvarano@a9.com

Abstract

In this paper, I investigate the use of a disentangled VAE for downstream image classification tasks. I train a disentangled VAE in an unsupervised manner, and use the learned encoder as a feature extractor on top of which a linear classifier is learned. The models are trained and evaluated on the MNIST handwritten digits dataset. Experiments compared the disentangled VAE with both a standard (entangled) VAE and a vanilla supervised model. Results show that the disentangled VAE significantly outperforms the other two models when the proportion of labelled data is artificially reduced, while it loses this advantage when the amount of labelled data increases, and instead matches the performance of the other models. These results suggest that the disentangled VAE may be useful in situations where labelled data is scarce but unlabelled data is abundant.

1. Introduction

Image classification is the task of assigning a class label from a fixed set of categories to a given input image. This is one of the core problems in Computer Vision that, despite its simplicity, has a large variety of practical applications; many other seemingly distinct Computer Vision tasks (such as image segmentation and object detection) can also be reduced to image classification [15]. Due to this fundamental importance, making progress on this task can have a widespread impact throughout many areas in the field.

Unsupervised and semi-supervised learning techniques have shown promise in many supervised classification tasks, including image classification. There is a wealth of unlabeled data that we can learn from, and utilizing these data effectively to improve downstream supervised tasks has been the focus of several recent research papers [7, 3, 16, 22, 10]. [7] suggest that, similar to human infants, machines should be able to learn to generalize from unlabeled data more effectively if they are able to disentangle the factors of variation. I show that such a machine is indeed better adapted to downstream image classification when it has spent time learning to disentangle factors of variation in the same domain, compared to its entangled counterpart. The machine learns these factors using a variational autoencoder (VAE) [11], and is able to learn to distinguish different classes from the MNIST hand-written digits dataset [13] using significantly less data than an its entangled counterpart.

My method is to first train a disentangled VAE on the data, and then train a linear classifier on top of the learned VAE encoder. A similar approach was also successfully used in a joint effort between the authors of the VAE and DeepMind, where they also used a second, semi-supervised method to use in conjuction with the former method [10], which is out of scope for this paper. The main difference between their method and mine is that I am using a disentangled VAE, where they use a standard VAE.

2. Related Work

Representation Learning. The performance of machine learning methods is heavily dependent on the choice of data representation on which they are applied [1]. A classic machine learning approach for a given supervised learning task takes hand-crafted features and feeds them to a learning algorithm For complex tasks like image classification, however, hand-crafted features are a major bottleneck to progress in the field. Many researchers have spent years of their lives dedicated to formulating intelligent features that capture well the underlying structure of images. This approach, while having produced many valuable features, is clearly not a scalable solution to image classification. Instead, we would like the learning algorithm to be able to construct its own features from raw sensory input (such as visual pixels) that well represenet the underlying data distribution [1].

There are many approaches to learning feature representations. In particular, deep convolutional neural networks (CNNs) used for supervised image classification tasks have been found to produce progressively more abstract feature representations as you traverse through its hidden layers [25]. That is, early layers may learn to represent simple edges, while later layers will produce composites of the lower-level features to represent more complicated concepts such as squares, honeycombs, and even human faces.

Disentangled Factors. We know that deep neural networks can learn a latent representation of the data it is trained on, but this representation is completely aribtrary and heavily dependent on the dataset used for training. We wish to learn a representation where single latent units are sensitive to changes in single generative factors, while being relatively invariant to changes in other factors [1]. With a disentangled representation, knowledge about one factor could generalize to many configurations of other factors, thus capturing the multiple explanatory factors and shared factors across tasks priors suggested by [1]. For example, for the MNIST dataset [13] we might want the model to learn that hand-written digits have a stroke width and an angle of orientation. Learning such disentangled factors of variation should allow the model to generalize much more easily to similar (but different) tasks and domains. For example, we might expect such a model to learn how to generate hand-written alphabetical characters quite quickly compared to an entangled model that has learned the idiosyncrasies of the digits in MNIST [13].

Variational Autoencoder. Variational autoencoders (VAEs) are powerful probabilistic models used for latent representation learning [11, 17]. They are comprised of a recognition network (the encoder), and a generator network (the decoder). The recognition network is an approximation $q_\phi(\mathbf{z}|\mathbf{x})$ to the intractable true posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$, where $\mathbf{z}$ is the set of latent variables we're interested in learning, and $\mathbf{x}$ is the set of input features [11]. The VAE formulation equates the marginal likelihood of the data to two terms: 1) a KL divergence term between the approximate and true posteriors, and 2) the variational lower bound on the marginal likelihood. To optimize the variational lower bound, the authors derive a Stochastic Gradient Variational Bayes (SVGB) estimator, and include a reparameterization trick to allow the SGVB estimator to be fully differentiable [11]. With a fully differentiable estimator, the parameters ϕ and θ can be jointly optimized. More recently improve versions of the VAE are importance weighted autoencoders (IWAE) [2] and ladder VAEs [20].

Unsupervised and Semi-supervised Learning. Unsupervised and semi-supervised learning is the method of learning with no or little labelled data, and instead making use primarily of unlabelled data. Though the performance of such techniques have tended to lag behind that of purely supervised models, this has begun to change with the explosion of data across the web, providing an endless stream of unlabelled data. There have been many recent advancements using these techniques, particularly with the use of deep generative models [5]. VAEs and GANs [6] are currently the most commonly used deep generative models, whereas the previously popular Deep Boltzmann Machines [19] and Deep Belief Nets [8] have since fallen out of favour. Some examples of recent applications of VAEs and GANs for unsupervised or semi-supervised learning tasks are text classification [24] and image classification [21].

3. Approach

3.1. Overview

I frame the problem as two separate steps: an unsupervised pre-training step, followed by a supervised learning step. I use the MNIST dataset [13] for my experiments, which contains 28x28 pre-processed (size-normalized and centered) grayscale images of handwritten digits. In the unsupervised learning step, I train a disentangled VAE on the dataset in order to learn a disentangled representation of the data. The learned encoder of this VAE is then used as a feature extractor on top of which a linear classifier is trained. A similar approach is taken by Kingma et al., which they refer to as the latent-feature discriminative model [10]. A key difference between their experiments and mine are that they are using a standard VAE, whereas I am training a disentangled VAE.

I first use a VAE to learn a disentangled latent representation of the domain on the MNIST dataset [13]. An additional hyperparameter, β, is added to the standard variational bound to simulate the redundancy reduction that happens in the human ventral visual system to allow learning of a disentangled latent space [7].

$$\mathcal{L}(\theta,\phi;\mathbf{x}) = \mathrm{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta \cdot D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}))||p(\mathbf{z}))$$

Tuning the hyperparameter β is critical to enabling a well disentangled latent representation. Setting $\beta = 0$ gives us standard maximum likelihood learning, while setting $\beta = 1$ gives us the Bayes solution (a standar VAE) [7], so in general $\beta > 1$ is used for disentanglement.

3.2. Variational Autoencoder

This section goes into further detail regarding the VAE architecture. As seen in Figure 1, a VAE is comprised of an encoder and a decoder. The architecture of the encoder and decoder can vary, and in my experiments I used two different architectures: MLPs and CNNs. The encoder is the piece that will be used as a feature extractor in the downstream image classification task, as its role is to map an image into a low-dimensional latent space. In Figure 2 we can see how to sample from the VAE at test time. We first sample our latent variables z from a unit Gaussian prior, then feed them through the decoder, or generator network, which will generate an "imagined" image from the VAE. We use this functionality to do the qualitative assessment of the level of disentanglement that the VAE has achieved. In order to do this we can fix all but one of the latent variables in the model, and then traverse that latent variable over three standard deviations around the unit Gaussian prior mean. Doing this for each of the latent variables will give us insights into what ”concepts” each variable has learned.

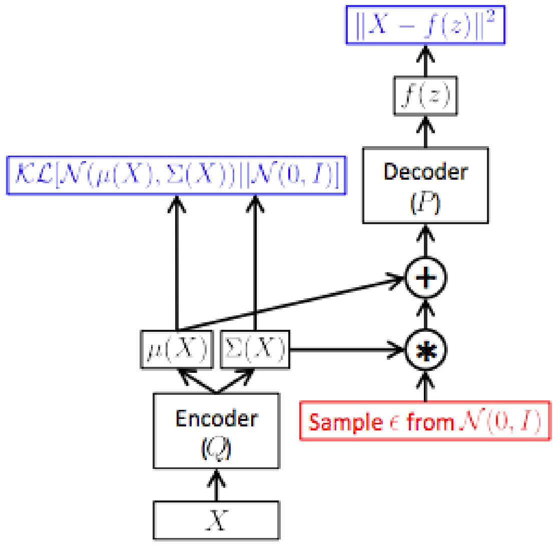
$||X - f(z)||^2$
$f(z)$
$\mathcal{KL}[\mathcal{N}(\mu(X), \Sigma(X))||\mathcal{N}(0, I)]$
Decoder
(P)
$+$
$\mu(X)$
$\Sigma(X)$
$*$
Encoder
(Q)
Sample ϵ from $\mathcal{N}(0, I)$
X

Figure 1. VAE architecture [4]

MLP Encoder/Decoder. In the MLP VAE, I parameterized both the encoder and the decoder as MLPs, each with two hidden layers of size 500. The original VAE paper uses a single hidden layer with 500 units [11], while Jan Metzen’s VAE implementation uses two hidden layers with size 500 [14], both for working with the MNIST dataset [13]. I used a β value of 4, as seen in [7].

CNN Encoder/Decoder. In the CNN VAE, I parameterized both the encoder and the decoder as CNNs. The encoder architecture is taken from the DCGAN discriminator in the CS231n Assignment 3, but replaces Leaky ReLUs with regular ReLUs. It is as follows:

- 32 filters, 5x5, Stride 1, ReLU
- Max Pool 2x2, Stride 2
- 64 Filters, 5x5, Stride 1, ReLU
- Max Pool 2x2, Stride 2
- Flatten
- Fully Connected size 4x4x64, ReLU
- Fully Connected size 10

The decoder network, also from the assignment, is taken directly from the InfoGAN paper [3], except the final tanh activation is replaced by a sigmoid activation, as is required by the VAE [11]. The architecture is as follows:

- Fully Connected size 1024, ReLU

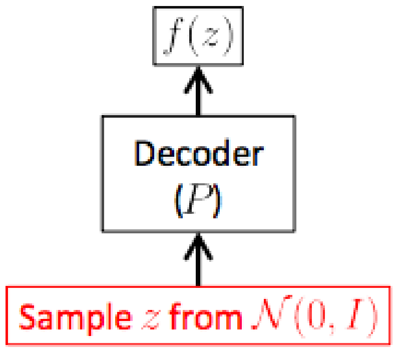

Figure 2. VAE test time sampling [4]

- BatchNorm
- Fully Connected size 7x7x128, ReLU
- BatchNorm
- Resize into Image Tensor
- 64 conv2d transpose filters of 4x4, stride 2, ReLU
- BatchNorm
- 1 conv2d transpose filter of 4x4, stride 2, sigmoid

3.3. Image Classification

Once we learn the VAE encoder that maps images into a disentangled latent space, the weights of the encoder are frozen and it is then used as a feature extractor for the downstream supervised task. A single FC layer of size TODO and a softmax layer are added on top of the encoder base, and the resulting network is trained to classify images from the MNIST dataset [13].

4. Experiment

4.1. Baselines

The first baseline is a purely supervised approach. The purpose of this baseline is to measure the effect on image classification performance of using unsupervised pre-training with a disentangling VAE. The two evaluation metrics that will be compared are number of training epochs during the supervised step, and classification accuracy.

The second baseline is unsupervised pre-training with a standard VAE. The purpose of this baseline is to compare whether using a disentangled feature extractor performs better than an entangled one for downstream image classification tasks. The same two metrics as stated above will be used for evaluation.

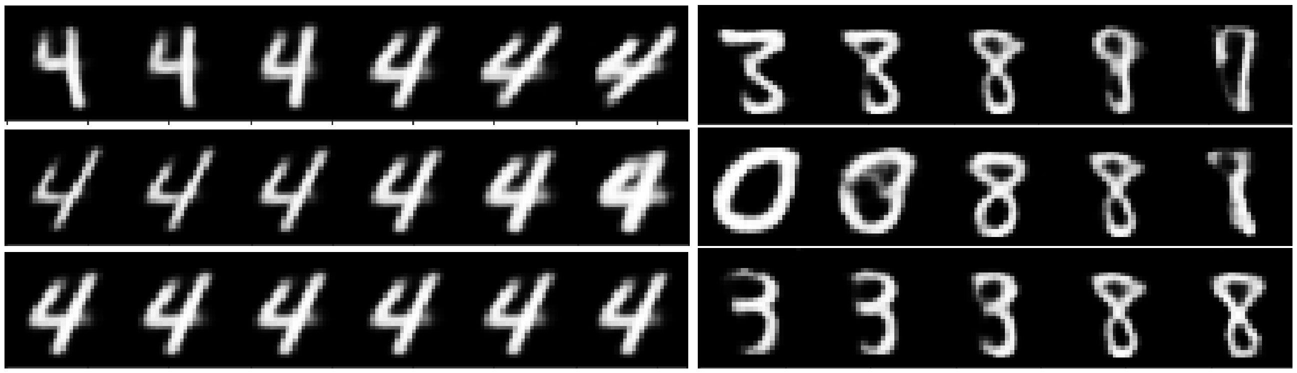

Figure 3. Left: latent variables from a disentangled VAE. (Top) Rotation, (Middle) Thickness, (Bottom) Gaussian prior, i.e. no effect. Right: Latent variables from an entangled VAE. There is no clear effect from any of the variables, and there are no variables that converged to the Gaussian prior.

4.2. Implementation Details

Unsupervised Learning. For training the VAEs I used an Adam optimizer [9] with a fixed learning rate of 0.001. Both the standard and disentangled VAEs had 10 latent units, and were trained for 50 (MLP) and 75 (CNN) epochs with batch size 100. The number of epochs was chosen based on convergence of the loss, while the batch size was chosen to be relatively small to get the benefits of stochasticity and also a divisor of the total number of training examples in the dataset. I toyed with decaying learning rates and weight decay but found no significant difference, thus chose to stick to the simplest approach. I also briefly tested using standard stochastic gradient descent as [23] suggests that SGD generalizes better than adaptive methods such as Adam [9], but did not find any significant difference in image classification performance. Since that investigation was brief, a deeper investigation could be left to future work.

Supervised Learning. For training the classifiers on top of the pre-trained feature extractors (both entangled and disentangled), I found that a fixed learning rate of 0.01 with the Adam optimizer [9] worked best. The classifiers were trained for 100 epochs. The vanilla supervised classifier was trained with a fixed learning rate of 0.001 for 100 epochs using the Adam optimizer [9]. The number of epochs were chosen based on convergence of the validation loss, while the learning rates were chosen based on the best validation accuracy at convergence using cross-validation for different learning rates on a logarithmic scale.

4.3. Experiment Setup

In order to have a measure of how well the unsupervised pre-training performs for different amounts of labelled data, I sampled several subsets of data from the full training dataset. The n-point sampled dataset contains exactly n data points from each class, giving a total size of $10n$ data points. I used the following values of n: 1, 3, 5, 10, 100, and 1000. A final test on the entire training dataset (no sub-sampling) was also done.

All three models (disentangled VAE, standard VAE, vanilla supervised model) were evaluated on all seven datasets described above (6 sub-sampled and 1 full). For n-point datasets with $n < 1000$ I ran 5 trials with different subsamples and took the average classification accuracy over them, in order to guard against noise when the datasets are small. For n-point datasets with $n >= 1000$ I found that the accuracies were stable enough across runs that I did not need to continue performing the averaging, and instead ran a single trial for each model.

As I had both MLP and CNN architectures, I repeated all experiments for each architecture individually. MLP architectures are only compared against other MLP architectures, and CNNs with CNNs.

For the MLP VAE I found $\beta = 4$ to be the best disentangling parameter, whereas for the CNN VAE I found $\beta = 6$ to be the best disentangling parameter. As I did not use a quantitative metric for disentanglement, as was done in [7], the β hyper-parameter was chosen qualitatively. I visually inspected the latent variables using the method described in Section 3.2 and chose the β that simultanesouly showed 1) at least 2 variables with very clear concepts (rotation and thickness), and 2) 40-50% of the latent variables having converged to the uninformative unit Gaussian prior.

4.4. Results

Quantitative Results. The final results were mostly against my expectations, but with an interesting exception.

The first evaluation metric, number of training epochs during supervised learning, had no significant difference between the entangled and disentangled VAEs. I omit the numbers for brevity. As an interesting aside, I did notice that the standard VAE's supervised training loss was higher than the vanilla supervised model's training loss, and

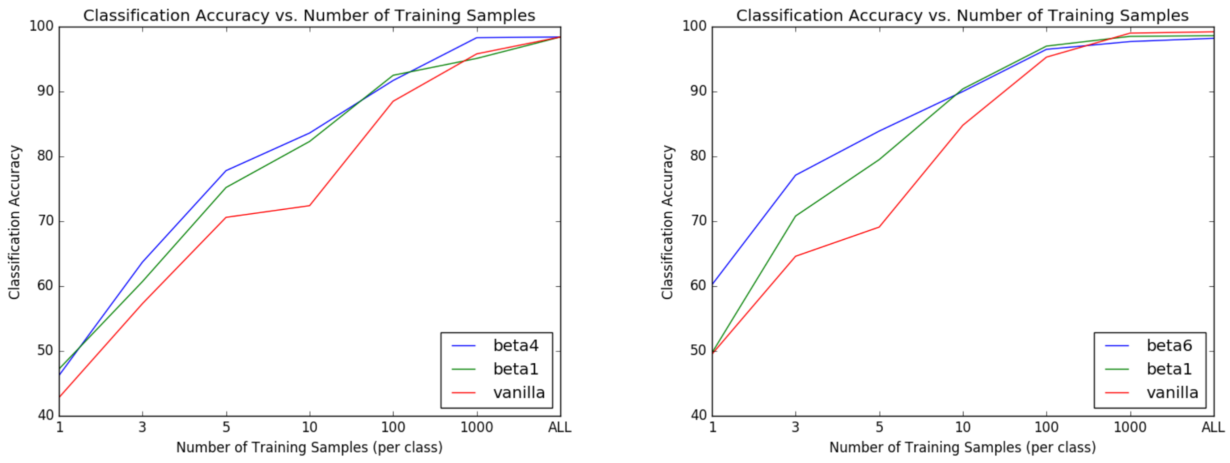

Figure 4. Quantitative results for MLP (left) and CNN (right) architectures

the disentangled VAE's supervised training loss was even higher at convergence than that of the standard VAE. I did not do a thorough investigation of this so there are no precise numbers to report, but this is likely due to the models being progressively more regularized by the KL-divergence loss term.

The results from the second evaluation metric, classification accuracy, were also against my expectations, but with an interesting twist. Originally, I expected the disentangled VAE to outperform both the vanilla supervised model and the entangled VAE across all datasets. Instead I found an interesting pattern where the disentangled VAE significantly outperforms both other models for small datasets (1-, 3-, and 5-points datasets), but was approximately the same as or slightly lower than the other two models for all of the larger datasets. These results seem to suggest that the disentangled VAE can indeed generalize much better when labelled data is scarce, but its advantage fades as the amount of labelled data available grows. It's also worth noting that the performance gaps with smaller datasets were much larger with the CNN architecture as opposed to the MLP architecture. The findings are summarized in Figure 4.

Qualitative Results.

I was able to successfully replicate the findings from [7] regarding the latent space learned from the disentangling VAE. A key difference to take note of between my work and theirs is that they used a synthetic dataset that was designed to be continuously transformed. The MNIST dataset [13] has a notable lack of this important feature, thus it is left to future work to repeat these experiments with a more suitable dataset.

Although MNIST [13] does not contain continuously transformed data, the disentangling VAE is still able to capture some of the underlying generative factors, such as rotation and thickness. In addition, about half of the latent variables have converged to the uninformative unit Gaussian prior, as seen in [7]. Figure 3 shows generated images by varying a single latent variable over three standard deviations around the unit Gaussian from both a disentangled VAE and a standard (entangled) VAE.

Another qualitative measure of the VAEs is the reconstruction quality. Good disentangled representations often lead to blurry reconstructions due to restricted capacity of the latent information channel, while entangled representations often result in the sharpest reconstructions [7]. Comparing the reconstructions of CNN VAEs against MLP VAEs we see that the CNN reconstructions for both entangled and disentangled models appear to be crisper and able to reconstruct finer details than its MLP counterpart, though these judgements are subjective. Reconstructions from both architectures are displayed in Figure 5 and Figure 6.

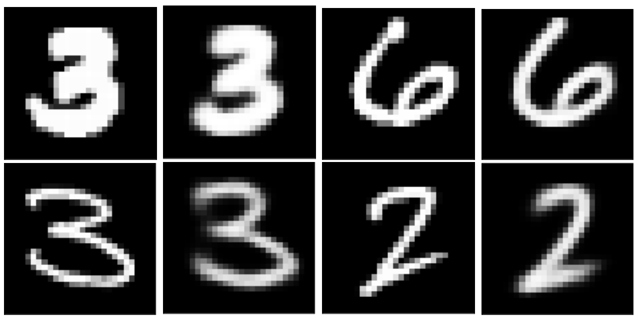

Figure 5. Pairs of MNIST test images and their VAE reconstructions. Top: standard CNN VAE. Bottom: disentangled CNN VAE

In contrast, Figure 7 and Figure 8 show examples of poor VAE reconstructions. The general (and expected) pattern is that disentangled VAEs have poorer reconstructions, in par-

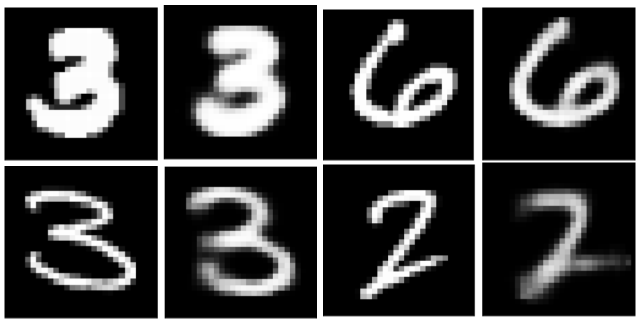

Figure 6. Pairs of MNIST test images and their VAE reconstructions. Top: standard MLP VAE. Bottom: disentangled MLP VAE

ticular they often appear much blurrier whereas the entangled VAE reconstructions are more crisp. Though I have no precise quantitative metric, I noticed a trend where the disentangled VAEs tended to ”autocomplete” images more frequently than their entangled counterparts, for example turning the digits 3, 5, or 9 into an 8. It is also worth noting that it took a much longer time to find bad reconstructions from the entangled MLP VAE.

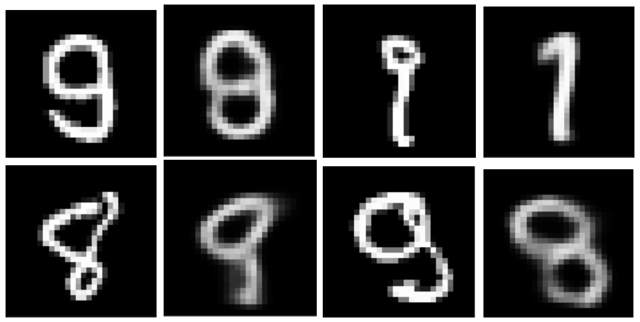

Figure 7. Pairs of MNIST test images and poor VAE reconstructions. Top: standard CNN VAE. Bottom: disentangled CNN VAE

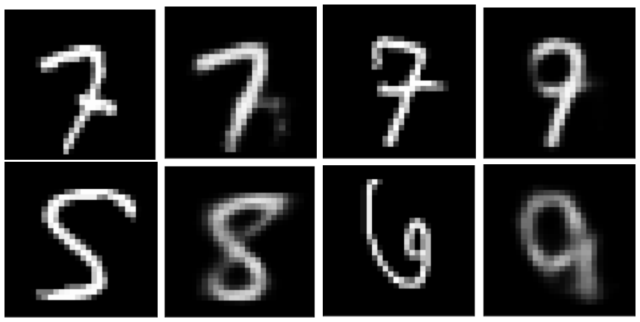

Figure 8. Pairs of MNIST test images and poor VAE reconstructions. Top: standard MLP VAE. Bottom: disentangled MLP VAE

5. Conclusion and Future Work

I compared the performance of a disentangled VAE used for downstream image classification against that of an entangled VAE and vanilla supervised model. All models were trained using the MNIST dataset of handwritten digits [13], and were trained on different sized subsamples from the training dataset in order to simulate different ratios of unlabelled to labelled data. I showed that the disentangled VAE significantly outperforms its entangled counterpart when labelled data is scarce, but approximately matches performance as the amount of labelled data grows. Both the disentangled and entangled VAEs significantly outperform the vanilla supervised model when labelled data is lacking, and approximately match performance as labelled data increases. Qualitative results show that the disentangled VAE learns visual concepts, such as rotation and thickness, while the entangled VAE shows no interpretability in its latent representation.

For future work, there are several improvements to make and directions to go. First, the authors of the disentangled VAE [7] show that having continuosly transformed data is vital for strong disentanglement of the underlying generative factors. As such, I would like to redo the experiments presented in this paper using such a dataset, where I expect that the disentangled VAE may perform even better across even larger amounts of labelled data. One way of achieving this is simply transforming MNIST [13] by rotating the images, for example. This could also be done with more complex datasets such as the CIFAR datasets [12] and ImageNet [18]. Eventually, I would like to extend this method to video datasets, where the spatial-temporal continuity of the frames should allow the disentangling VAE to learn the manifolds much more effectively, similar to the Atari dataset experiments in [7].

Second, [10] presents a semi-supervised learning objective that was shown to significantly improve performance on image classification tasks. They use a standard VAE, so I would like to redo my experiments with a disentangled VAE using their proposed semi-supervised learning objective.

Finally, there are several smaller investigations I would like to do that could improve performance or give further insights into the task. This includes testing other optimization algorithms which may provide better generalization, as proposed in [23], as well as trying other CNN architectures, as I was only able to test one. Another thing to investigate is the types of errors that the disentangled VAE makes versus the entangled VAE or vanilla supervised model. It may be the case that the disentangled VAE makes different types of errors than the other two models, which could provide insights into using the model in downstream tasks.

References

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[2] Y. Burda, R. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[3] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, 2016.

[4] C. Doersch. Tutorial on variational autoencoders. *CoRR*, abs/1606.05908, 2016.

[5] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[7] I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner. Early visual concept learning with unsupervised deep learning, 2016.

[8] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[9] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[11] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[12] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[13] Y. LeCun and C. Cortes. The mnist database of handwritten digits, 1998.

[14] J. Metzen. Variational autoencoder in tensorflow, 2015. https://jmetzen.github.io/2015-11-27/vae.html.

[15] C. C. Notes. Image classification, 2017. http://cs231n.github.io/classification/.

[16] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

[17] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[19] R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In *Artificial Intelligence and Statistics*, pages 448–455, 2009.

[20] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems*, pages 3738–3746, 2016.

[21] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

[22] W. F. Whitney, M. Chang, T. Kulkarni, and J. B. Tenenbaum. Understanding visual concepts with continuation learning. *arXiv preprint arXiv:1602.06822*, 2016.

[23] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. *arXiv preprint arXiv:1705.08292*, 2017.

[24] W. Xu, H. Sun, C. Deng, and Y. Tan. Variational autoencoder for semi-supervised text classification. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[25] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.