

Discriminative Regularization of the Latent Manifold of Variational Auto-Encoders[☆]

Ingo Kossyk^{a,b}, Zoltán-Csaba Márton^{a,*}

^a*German Aerospace Center (DLR), Robotics and Mechatronics Center,
Oberpfaffenhofen, Muenchner Str. 20, 82234 Weßling, Germany*

^b*This work was performed while employed at DLR; Present address: Hexagon Technology
Center GmbH, Heinrich-Wild-Strasse, CH-9435 Heerbrugg, Switzerland*

Abstract

We present an approach on training classifiers or regressors using the latent embedding of variational auto-encoders (VAE), an unsupervised deep learning method, as features. Usually VAEs are trained using unlabeled data and independently from the classifier, whereas we investigate and analyze the performance of a classifier or regressor that is trained jointly with the variational deep network. We found that models trained this way can improve the embedding s.t. to increase classification performance, and also can be used for semi-supervised learning, building up the information extracting latent representation in an incremental fashion.

The model was tested on two widely known computer vision benchmarks, and its generalization power was evaluated on an independent dataset. Additionally, generally applicable statistical methods are presented for evaluating similarly performing classifiers, and used to quantify the performance increase. The general applicability and ease-of-use of deep learning approaches allows for a wide applicability of the method.

Keywords: variational auto-encoder, regularization, knowledge representation, perceptual data compaction, semi-supervised learning,

[☆]Declarations of interest: none

^{*}Corresponding author

Email addresses: inkoss74@gmail.com (Ingo Kossyk), zoltan.marton@dlr.de (Zoltán-Csaba Márton)

1. Introduction

Deep learning (DL) methods became wide-spread after dataset sizes and computing power increased enough to allow for training several large layers. Most methods rely on data with ground truth labels, and pre-trained networks exist for some domains (e.g. RGB data), that can be fine-tuned or otherwise adapted to new problems. However, this is not available for sensing modalities that are novel, not widely used by the DL community, or limited/no training data is available (e.g. depth, audio, tactile, force-torque, hyperspectral, biochemical, etc.), as well as in novel application domains (e.g. space, medical, underwater, etc.).

An advantage of unsupervised methods is their capacity to learn efficient codings of unlabeled data instances or streams. One class of unsupervised methods that has recently gained a lot of interest is the variational auto-encoder (VAE), which gives more control of influencing the latent representation by incorporating a statistical prior on the underlying distributions. This prior can act as a compensation for lack of data, focusing the learning problem to adjusting the representation rather than learning it from scratch.

In this paper we investigate further how additional semantic information of the data, e.g. labels, can aid in improving the latent representation for fulfilling a functional objective. Hence, we evaluate the effect of training classifiers or regressors on the latent embedding of VAE jointly with the generative model. We investigate two different possible architectures and show empirical results of classification accuracy in comparison to a common variational model that can be used for inference. We investigate the effect of applying what we call discriminative regularization on the latent embeddings of a variational auto-encoder and introduce the Discriminatively Latent Regularized Variational Auto-Encoder (DLR-VAE) – see fig. 1. In summary, our main contributions are:

- We propose and evaluate the DLR-VAE concept in comparison with com-

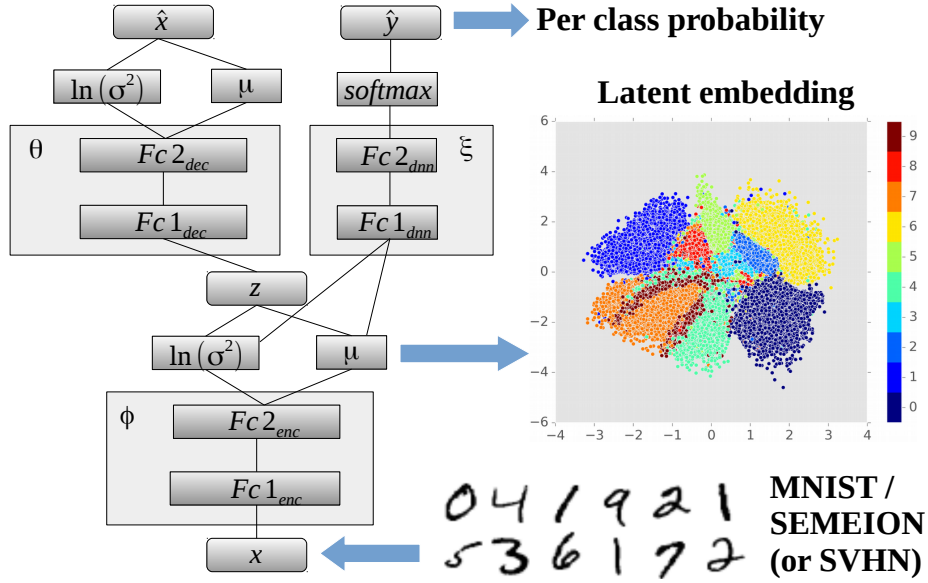


Figure 1: A variational auto-encoder with an additional classifier realized with two fully connected layers for the encoder, decoder and classifier modules.

mon classification schemes that use VAEs as feature detectors or for inference.

- We show the effects of regularization with a discriminative network on the latent embedding of variational auto-encoders in the scope of classification and regression problems. Our empirical evaluation shows performance that is comparable to state of the art results of similar models. For the MNIST [26] dataset we found that the model achieves a new state of the art result for models that only use fully connected layers and do not use data augmentation/distortions.
- We demonstrate how DLR-VAE can be used for semi-supervised learning, where knowledge from labeled and unlabeled sets of samples is acquired incrementally, and compacted in the probabilistic latent representation.
- Due to the relatively small differences between the decoupled and the jointly trained models, we employ statistical techniques to evaluate the

significance of the improvement. We hope that this will aid others in the field, since results on benchmarks and test datasets get tighter as the 100% mark is approached with more complex/fine-tuned methods.

The latent manifold of VAEs can be used efficiently for training classifiers or regressors. For this purpose the auto-encoder is usually trained separately in an unsupervised fashion and the encoder is used as a feature extractor, whereas in our approach we train the unsupervised model jointly with a supervised deep discriminative network. We draw our motivation for this approach from two main assumptions: The stochastic nature of a variational auto-encoder acts as a regularization for the classifier when learning a low dimensional embedding of the input data and the gradients of the discriminative part of the model feed back into the latent embedding of the encoder of the variational auto-encoder. We found that these models generalize better by a slight margin and can be used for efficient classification and regression on streaming data in an online fashion.

We show empirical results on the MNIST, SEMEION [4] and SVHN [31] datasets, and evaluate the statistical significance of the improvement in classification accuracy in comparison to common variational models that can be used for inference. The use of computer vision benchmarks is motivated by the fact that both fully connected and convolutional layers can be tested on them, and that the deep learning field is heavily focused on vision problems. However, MNIST and SEMEION type of data can easily come from pressure sensor, SVHN (Google’s Street-View House Numbers) type e.g. from reading product labels, and the concept can be applied to many different data sources thanks to the end-to-end trained nature of deep learning, which allows it to be easily deployed in different domains.

The presented method was already used for material state/type classification by an industrial robot, based on structure-borne sound, with different application scenarios in the area of robust manipulation for autonomous manufacturing [32]. There, the latent representation that was learned while using data

labels for a regularizing effect showed similar clustering and regressions properties to the reproducible examples on public data that is presented in this article.
75 This concise information representation enabled us to distinguish 8 materials in the audio stream generated by touching or gripping different parts, and even to draw conclusions about geometric properties.

2. Related Works

2.1. Generative Autoencoder Architectures

80 The concept of using deep neural networks (DNN) for the training of variational models has been proposed independently by two research groups. The principal contribution was the so called reparametrization trick, a mathematical trick in order to enable backpropagation through stochastic layers in deep neural networks [18, 35].

85 A different concept of a generative autoencoder based on a cost function using the Wasserstein distance has been shown to be effective by Tolstikhin et al. [36].

2.2. Semi-supervised learning with Variational Autoencoders

Kingma et al. [19] continued to show that the probabilistic nature of the proposed methods can be used for semi-supervised learning and demonstrate that
90 it is possible to untangle properties of the training data, for example class information and style. Maaløe et al. [27] carried that concept further by adding an additional auxiliary stochastic variable and propose auxiliary directed graphical models (ADGM). They show state-of-the-art results on several semi-supervised
95 classification tasks in their work. Abbasnejad et al. [1] present a method for semi-supervised learning inspired by the concept of ensembles of experts using variational autoencoders.

2.3. Flows

Kingma et al. [20] showed how to improve the ability of VAEs to model more
100 complex data distributions by applying inverse autoregressive flow in a VAE with

multiple stages of stochastic layers in combination with residual blocks [12] for feature extraction. A similar concept is reported by Tomczak and Welling [37] for applying the Householder Flow to a VAE.

2.4. *Generative Adversarial Networks*

105 Generative adversarial nets (GAN) have been proposed by Goodfellow et al. [11] and they show that two competing neural networks, a classifier and a generator that tries to fool the former into misclassification, can learn a powerful generative model. Makhzani et al. [28] apply this concept of adversarial training to auto-encoders and show that it acts as a regularization on the latent coding.

110 2.5. *Regularizing Variational Autoencoders*

A closely related idea to ours can be found in the work by Lamb et al. [25], where a pre-trained classification network is used to regularize and refine the reconstruction of an unsupervised variational model. However, in contrast to the herein presented work, the regularization is performed on the whole network.

115 2.6. *Normalization*

Exponential linear units have been proposed in [7] as an alternative to the combination of using rectified linear units (RELU) and Batch Normalization [16] for speeding up convergence during training of deep neural networks. In [21] the concept of ELUs is refined and the sELU (scaled exponential linear unit) is proposed that enable deep neural networks to provide inherent self
120 normalization.

2.7. *State of the art for classification architectures*

Residual neural networks [12] are considered as the state of the art DNN architecture for feature extraction and have been proven to set the state of the art
125 for classification purposes on common benchmark datasets like CIFAR-10/100 [22] [23] and Imagenet [8] in terms of classification error. They are based on the concept of introducing skip connections between layers that enable information flow through very deep network architectures. Densenet [14], Mobilenet

[13] or Condensenet [15] show different network architectures where the optimization criteria are either parameter count, memory footprint, computational complexity or inference time while maintaining reasonable accuracy.

In summary, the DLR-VAE idea can be combined with a wide range of advancements in the DL field, depending on the application scenario and the available data.

3. Theoretical Foundations

This section will lay out the mathematical and technical definitions of the model we are investigating in this work. We assume to be given N data pairs $(\mathbf{X}, \mathbf{Y}) = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ where $x_i \in \mathbb{R}^D$ represents a single data sample of the multidimensional input data and $y_i \in 1, \dots, L$ represents the respective class label out of L classes. For the following formulations we assume the class label to be represented as one-hot vectors. We will omit the index i whenever it is clear that we refer to the corresponding formulation regarding only a single datapoint. Similar to [18] we define the generative model as follows:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}); \quad p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{L}(\mathbf{x}; \mathbf{z}, \theta) \quad (1)$$

where $\mathcal{L}(\mathbf{x}; \mathbf{z}, \theta)$ is a suitable likelihood function depending on the nature of the underlying data model. Common models are the binary log cross entropy for Bernoulli distributions or the negative log likelihood for Gaussian distributed data. We use non-linear functions:

$$f(\mathbf{x}, \phi); \quad g(\mathbf{z}, \theta) \quad (2)$$

in order to estimate the moments of the underlying probability distributions. These non-linear functions resemble the encoder and decoder of the variational auto-encoder and are learned by deep neural networks (DNN) with parameters ϕ and θ respectively. Additionally, we define a discriminative DNN denoted by:

$$h((\mu_z, \log(\sigma_z^2)), \xi) \quad (3)$$

with parameters ξ which acts as a classifier or regressor. This discriminative network receives as an input a concatenation of the statistical moments μ_z and $\log(\sigma_z^2)$ that are estimated by the recognition model $f(\mathbf{x}, \phi)$. We train
155 $h((\mu_z, \log(\sigma_z^2)), \xi)$ in a joint fashion with the recognition and generative networks, respectively. Therefore, the discriminative loss acts as an additional regularization on the moments of the recognition DNN. For classification we use the softmax activation:

$$\sigma(\hat{\mathbf{z}})_j = \frac{e^{\hat{z}_j}}{\sum_{c=1}^L e^{\hat{z}_c}}, \text{ for } j = 1, \dots, L \quad (4)$$

for the last layer of the discriminative DNN, where $\hat{\mathbf{z}}$ represents the unscaled
160 logits output of the last layer. Hence, in the case of a classification problem the loss of the discriminative network is then defined as the cross entropy of the true class label $y \in \{0, 1\}$ to the estimated probability \hat{y} for class $k \in L$:

$$H(p_L, q_L) = - \sum_{k=1}^L y_k \log \hat{y}_k + (1 - y_k) \log (1 - \hat{y}_k) \quad (5)$$

where p_L represent the true probabilities of observing the classes and q_L the probabilities of observing the predictions. The predictions are determined by
165 the classifier, hence $\hat{y} = h((\mu_z, \log(\sigma_z^2)), \xi)$.

In the case of a regression problem we use the standard mean square error formulation as a loss function instead of (5). Note that both additional losses are not real probability measures however we found that they still act as regularizers in the loss formulation of the final model.

170 3.1. Variational Evidence Lower Bound (ELBO)

Variational models in DL are based on the idea of approximating the marginal log-likelihood of an observable random variable x which can be rewritten as sum over the joint probabilities of x with an additional non-observable, latent variable z (where θ and ϕ are trained parameters):

$$\log p_{\theta}(\mathbf{x}) = \log \sum_{\mathbf{z}} p_{\theta}(\mathbf{x}, \mathbf{z}) \quad (6)$$

175 By multiplying with $q_{\phi}(\mathbf{z}|\mathbf{x})/q_{\phi}(\mathbf{z}|\mathbf{x})$, applying the chain rule and the Jensen inequality, eq. 6 can be reformulated as follows:

$$\log \sum_{\mathbf{z}} p_{\theta}(\mathbf{x}, \mathbf{z}) \geq \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}|\mathbf{x}) \log \frac{p_{\theta}(\mathbf{z})}{q_{\phi}(\mathbf{z}|\mathbf{x})} + \log p_{\theta}(\mathbf{x}|\mathbf{z}) \quad (7)$$

Thus, per definition of the Kullback-Leibler (KL) divergence and the expectation operator, the ELBO is defined as:

$$\mathcal{L}_{ELBO}(\theta, \phi; \mathbf{x}) = -D_{KL}(q_{\phi}(\mathbf{z}|\mathbf{x})||p_{\theta}(\mathbf{z})) + \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] \quad (8)$$

This ELBO formulation can now be used as an optimization criterion where the
180 required parameters of the probability distributions are estimated with DNNs.

The ELBO as formulated above exhibits a large variance when used directly during training of a VAE, resulting in unstable or diverging training behavior. Kingma and Welling [18] and Rezende et al. [35] independently presented a differentiable procedure which they call the reparametrization trick by representing the stochastic variable \mathbf{z} with a deterministic variable $\mathbf{z} = g_{\theta}(\epsilon, \mathbf{x})$,
185 where ϵ is an auxiliary random variable usually drawn from a zero mean and unit variance normal distribution $\mathcal{N}(0, 1)$. Assuming that \mathbf{z} is Gaussian distributed $\mathbf{z} \sim p(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mu_z, \sigma_z^2)$, the reparametrization is $\mathbf{z} = \mu_z + \sigma_z \odot \epsilon$. This leads to a stochastic gradient auto-encoding variational Bayes.

190 3.2. Discriminative Latent Manifold Regularization

When applying the ELBO and the reparametrization trick, θ and ϕ are learned by DNNs. To perform classification or regression using VAEs as feature extractors one can:

1. Train the generative model with a VAE in an unsupervised fashion, learning
195 the parameters ϕ and θ of the encoder/inference network and decoder/generative network respectively.

2. Transform the training data of a labeled training dataset to the lower dimensional latent manifold which is parameterized via the moments estimated by the encoder $f(\mathbf{x}, \phi)$ as defined in eq. (2).
- 200 3. Train a classifier or regressor (for example a DNN with one or multiple layers) in a supervised fashion with labeled data on the estimated moments by $f(\mathbf{x}, \phi)$ of the embedding of the input data.

In this work we are investigating incorporating the training loss of a deep classifier into the loss function of the VAE. Therefore, the resulting deep neural
 205 network consists of a VAE and an additional classifier realized as a single- or multilayer deep neural network. The whole model is trained jointly, therefore, according to the chain rule, the cost/objective function to train the network is:

$$\begin{aligned}
 C &= \mathcal{L}_{ELBO}(\theta, \phi; \mathbf{x}) + p(\mathbf{y}, f(\mathbf{x}, \phi)) \\
 &= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] + H(p_L, q_L)
 \end{aligned}
 \tag{9}$$

Hence, $H(p_L, q_L)$ acts as an additional regularizer on the encoder together with the KL divergence term. This should, in addition to the probabilistic prior $p(\mathbf{z})$,
 210 force the latent representation to converge to a task specific optimized topology. In accordance to [18] we draw one sample from $p(\mathbf{z})$ in our experiments.

4. Model and Methods

The basic concept we propose is illustrated in fig. 1. The discriminative part of the model is trained with the output of $f(\mathbf{x}, \phi)$, which are the estimated
 215 moments μ_z and the log-variance $\log \sigma_z^2$ for the stochastic embedding variable z in case of a normally distributed posterior. Instead of fully connected layers, convolutional neural networks (CNN) can be used for the encoder and decoder, too [24, 34]. We assume a normally distributed prior and posterior with zero mean and unit variance for the latent representation, therefore, an analytical
 220 solution for the KL divergence exists:

$$-D_{KL} = \frac{1}{2} \sum_{j=1}^J (1 + \log((\sigma_j)^2) - (\mu_j)^2 - (\sigma_j)^2) \quad (10)$$

where J denotes the dimensionality of the latent manifold z . Depending on the data two different models for the reconstruction probability distributions can be applied. In the case of a multivariate Bernoulli distribution it takes the form:

$$\log p(\mathbf{x}|\mathbf{z}) = - \sum_{k=1}^D x_k \log \hat{x}_k + (1 - x_k) \cdot \log(1 - \hat{x}_k) \quad (11)$$

where \hat{x}_k are the estimated outputs of the last layer of the decoder network for
 225 one data sample transformed with the sigmoid activation function. When we
 model the data as continuous Gaussian variables two final layers are fed with the
 output of the preceding layer in parallel in order to estimate the reconstruction
 parameters μ and $\log \sigma^2$ required for the calculation of the log likelihood of a
 normal distribution with a diagonal covariance matrix:

$$\log p(\mathbf{x}|\mathbf{z}) = \log \mathcal{N}(\mathbf{x}; \mu, \sigma^2 \mathbf{I}) \quad (12)$$

230 4.1. Fully Connected Encoder and Decoder

In analogy to the work of Kingma and Welling [18] and in order to make comparisons fair our baseline model consists only of fully connected layers for the encoder and decoder network. The encoder is a deep neural network with two fully connected layers with exponential linear units (ELU) as activation func-
 235 tions [7] throughout the whole network. We found ELU's to perform equally
 well as other common state of the art normalization methods like batch normalization [16] during our experiments. We can also confirm the statement by the authors of [16] that combining ELU'S with batch normalization did not improve the training procedure in our experiments. The application of ELU's
 240 for training a variational model has also been shown to be beneficial by the authors of [20]. The last two layers of the encoder are fed in parallel with the output of the preceding layer and estimate the mean and log-variance of the

latent probability distribution with linear activation functions. In addition to using the estimated mean and log-variance for the reparametrization to form the stochastic representation z they are used to train the classifier jointly with the variational auto-encoder. The decoder also consist of two fully connected layers with the same size as the encoder layers with one final layer to estimate either the reconstruction in the form of the Bernoulli distribution in accordance to eq. 11 or with two layers estimating the parameters of the Gaussian distribution for eq. 12.

4.2. Encoder and Decoder with Convolutions

In computer vision or image recognition tasks, convolutional layers have been shown to be superior to fully connected layers in deep learning. When used in variational auto-encoders, CNNs can act as an efficient feature extractor and when used together with pooling layers generalize well over spatial variations of the data. As other existing works show the effectiveness of CNN's in variational auto-encoders we chose to investigate the effect of discriminative regularization with this architecture, too.

The convolutional variational auto-encoder (CVAE) we use consists of three convolutional layers in the encoder and four convolutional layers in the decoder. In the encoder the convolutions are followed by two fully connected layers in order to estimate the parameters of a Gaussian prior for the latent manifold z , similar to the architecture in sec. 4.1. The decoder first transforms the latent representation to an appropriate dimensionality using a fully connected layer. We use a combination of convolutional layers with 2×2 unpooling layers [33]. We assume that RGB pixel values are of continuous normally distributed nature (except in the case of the MNIST dataset where data is assumed to be binary) we use two convolutional layers in order to estimate the parameters of the reconstruction probability distribution at the top of the variational auto-encoder tower. Please see fig. 2 for a schematic of the architecture. We use ELU activation functions except at the final two layers of encoder and decoder, respectively.

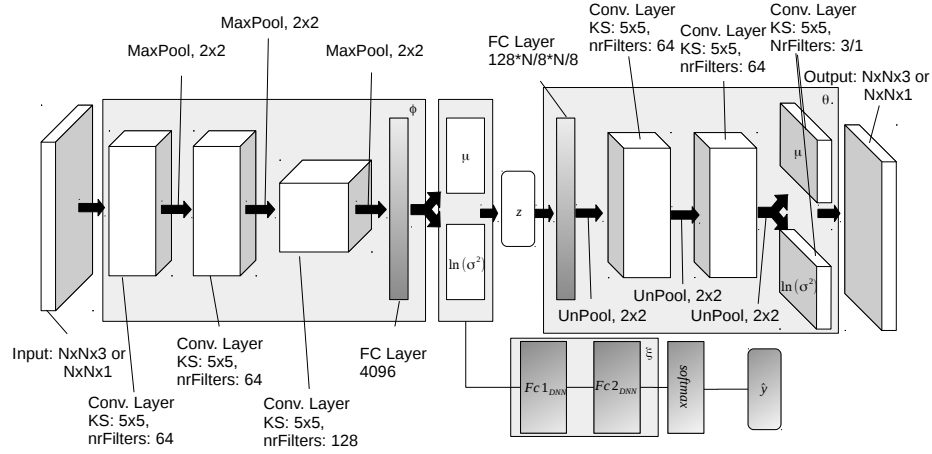


Figure 2: VAE model with convolutional layers in the encoder and decoder stages. The convolutional layers are combined with maxpooling and unpooling layers in the encoder and decoder networks, respectively. The discriminative network is realized as a two-layer DNN. Here the CVAE for continuous normally distributed data is shown. In the case of binary data that underlies a Bernoulli distribution the decoder has only one last convolution layer with the sigmoid activation function according to eq. 11.

4.3. Semi-supervised Learning

An interesting feature of the DLR-VAE is that it can be used for semi-supervised learning tasks with minor modifications when only a subset of the training data is labeled. For the purpose of description of the training procedures we define an iteration during training to be one step of optimization (for example by stochastic gradient descent or using ADAM or comparable optimizers) with one minibatch. Consequently, an epoch is defined as the count of iterations necessary so that the model sees all samples contained in the training dataset once. The following two methods are performed during training once *per epoch*. Hence, switching between labeled and unlabeled subsets of training data as well as the respective two different cost functions that are used for optimization happens once in *each epoch*. The model is trained according to the following two procedures for 1000 epochs. Please see section 5.1.3 for a detailed description of the setup we used for training in our experiments.

- Train the model by applying two different cost functions *each epoch*. In the first part of the epoch train with the labeled subset and use the cost function of eq. 9 then, in the second part, train the model with the remaining unlabeled data using the formulation of the ELBO cost function as shown in eq. 8. We call this model SS0. When investigating this training procedure we also tried to alternatively first optimize the model on the full unlabeled dataset during the first part of the epoch followed by training with the labeled part in the same epoch. We found that this did not produce different results in our experiments.
- Train the model in the first step with the labeled subset. Then, in the same epoch, use the model to predict the labels of all remaining unlabeled training data. In a third step, the model is optimized in the same epoch using the predictions as labels for the unlabeled subset and applying the cost function of eq. 9. We call this model SS1.

5. Experiments

We evaluated both approaches, the fully connected and convolutional DLR-VAE, on the common benchmark datasets MNIST and SVHN. For the experiments we trained all models for 1000 full training epochs. In order to achieve a fair comparison we kept the network fixed, therefore, the layer/neuron count was exactly the same for the VAE+DNN and the DLR-VAE models with the only difference that the former was trained in a two stage training process and the later was trained jointly. We kept the learning rates fixed for each problem domain and used the same activation function (ELU) throughout all models. As features the estimated μ_z and $\log(\sigma_z^2)$ were concatenated to form a feature vector for training the discriminator either for VAE+DNN or DLR-VAE resulting in $2n_z$ features. We used ADAM for optimization with default parameters [17].

5.1. MNIST

315 For MNIST we trained all models with the full training data of 60000 training samples. For test evaluation we used only the test dataset which consists of 10000 samples. A batch size of 100 and a learning rate of 3e-4 was used.

5.1.1. Fully Connected Layers

For the fully connected model the VAE’s encoder and decoder consisted of
320 two fully connected layers with 600 neurons each. MNIST images are binary, therefore we assume a Bernoulli distribution for reconstruction by the decoder. The latent dimensionality was chosen to be 50. The discriminative part of the model consists of two layers with 50 neurons. The last layer’s activation function is the softmax function whose output is used to calculate the logarithmic
325 softmax-binary crossentropy as a classification loss.

5.1.2. Convolutional Layers

The architecture of the CVAE follows the general model as shown in fig. 2. The encoder’s three convolutional layers were implemented with $64 \times 64 \times 128$ output filters, a kernel size 3×3 and a symmetric stride of 1. The latent manifold
330 z had 50 dimensions. In the decoder a fully connected layer increases the 50 dimensions of the latent manifold for the following convolutions. The mapping corresponds to 128 filters and 7×7 input size. The output of the fully connected layer is then used as input for three convolutional layers that were implemented with $64 \times 64 \times 1$ output filters, a kernel size of 3×3 and a stride of 1. The last
335 convolutional layer’s activation function is a sigmoid and the ELU activation function is used throughout the model.

5.1.3. Semi-supervised Learning

For an evaluation of the performance of the DLR-VAE in semi-supervised learning tasks we chose to use the fully connected model which was also used
340 for the evaluation on the fully labeled MNIST training dataset. We randomly draw class-balanced subsets of labeled data with 100, 600, 1000 and 3000 labeled

examples for the experiment and repeated the training process four times. We then calculated the mean performance of the training process. The models were trained for 1000 epochs with a learning rate of $3e-4$.

345 5.1.4. Computational Complexity

As the training time is not a critical measure that has to be kept low when thinking about complexity in the application of DNNs, we regard only the computational complexity during the test phase in the following paragraph. We will in this section limit the theoretical complexity considerations to depend on
 350 the test sample count only as the amount of neurons and activations that need to be calculated differ neglectable between the compared models. Let N_{test} be the number of test samples and L the number of classes in the problem. Then the complexity of the DLR-VAE is $\mathcal{O}(N_{test})$, thus being only dependent on the number test samples (it is not dependent on the number of classes). For the M2
 355 model it is $\mathcal{O}(L * N_{test})$ because inference is performed by maximum likelihood or maximum a posteriori estimation. For the M1 model it is $\mathcal{O}(N_{test})$. Hence, the stacked complexity for the M1+M2 model is $\mathcal{O}(N_{test}) + \mathcal{O}(L * N_{test})$. Therefore, the complexity grows linear with L for the M1+M2 model, even when taking into account that the evaluation with the M2 model is done on the low-dimensional
 360 embedding of the M1 model.

5.2. SVHN

On the SVHN dataset we used the training data in combination with the extra data which results in 604388 training samples. For testing we used the 26032 test samples. We preprocessed the training data with ZCA whitening with
 365 a regularization of $\epsilon = 0.01$ and we chose not to apply any data augmentation, therefore the input of the model was the 3072 whitened features of an SVHN RGB image.

5.2.1. Fully Connected Layers

The layers of the encoder and decoder consisted of 2000 neurons. The dimensionality of the latent manifold z was chosen to be 200. Again, we assume
 370

a Gaussian prior with zero mean and unit variance for the latent coding and that whitened pixel values of RGB images are continuous random normally distributed variables. The discriminative network has two layers with 200 neurons each.

375 5.2.2. Convolutional Layers

For the CVAE we used a similar architecture as for the MNIST CVAE experiments. However, the input now consists of RGB images, therefore, the input convolutions are applied on the 3 color channels. Two convolutional layers that output 3 color channels are used for the calculation of the reconstruction cost.
 380 The dimensionality of the latent variable z was chosen to be 200. According to the other models presented in this work we set the neuron count for the discriminative network to 200, too.

5.3. Results

The test errors of all trained models are shown in tbl. 1. We compare the
 385 test error of the vanilla VAE when used as a feature extractor for classification with a separate classifier, the DLR-VAE and the M2 model as proposed by Kingma et al. [19]. We chose the M2 model because it relies on a different evaluation scheme that follows the paradigms of Maximum Likelihood testing or Maximum a posteriori testing. According to the results we reason that the
 390 stochastic nature of the VAE leads to a regularization effect when training a DLR-VAE. Using the fully connected model trained on MNIST the DLR-VAE (1.13% test error) models generally perform better than the decoupled trained vanilla VAE+DNN counterpart (1.73% test error), which can also be seen on the evolution of the test error during training of both models in fig. 3 (left).

395 However, the DLR-VAE is outperformed on the MNIST classification task by the M2 model when only fully connected layers are used for the encoder and decoder (however, the difference is not statistically significant since the bounds overlap). We reason that this might be due to the effect of the untangling of class and style information. On the other hand, when we do not preprocess the data

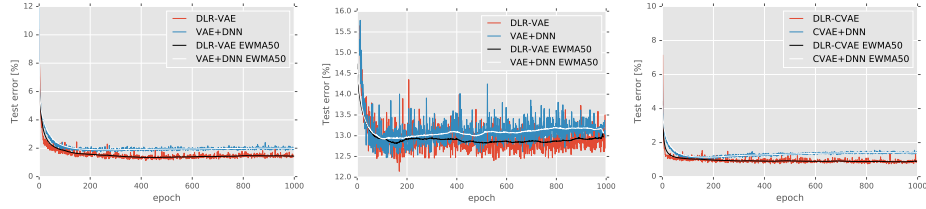


Figure 3: Test error over epochs trained (EWMA50 denotes the Exponential Weighted Moving Average over 50 Samples) on MNIST (left, CVAE right) and SVHN (middle)

(except normalization), the M2 model shows rather questionable performance on the SVHN dataset when only fully connected layers are used (55.34% error on the test data set) and is clearly outperformed by the other models, with the DLR-VAE showing the best performance in our experiments (DLR-VAE: 12.25%, VAE+DNN: 12.46%). This is also supported by the evolution of the error during training, shown in fig. 3 (middle).

We preprocessed the SVHN dataset with ZCA whitening. Additionally, we investigated a model trained with data preprocessed by a principal component analysis (PCA) and dimensional reduction. In these experiments we reduced the feature dimensionality to 600 principle components. When trained on the reduced SVHN dataset using PCA the DLR-VAE achieves 13.65% test error, the VAE+DNN performs reasonably worse with 19.50% test error and the M2 model achieved 19.06% therefore being slightly better than the former. We reason that the loss of “locality” of the features when PCA is used leads to a slightly worse performance than when ZCA is used as preprocessing step. The bad test performance of the M2 model, however, when ZCA features are used remains subject to further investigation.

We performed the experiments using CVAEs using only the vanilla VAE+DNN and the DLR-VAE models. The results show that the DLR-VAE outperforms the decoupled approach on the MNIST dataset with a test error of 0.74%, showing performance that is comparable to the state of the art. The vanilla VAE+DNN, when trained in a two stage process, shows an error that is 1.01%. Again, the evolution of the test error during training is shown in fig. 3 (right).

On the SVHN dataset the differences between both models are more minute with only 0.03% of difference. Additionally, the performance using convolutional networks is only slightly better than when fully connected layers are used. Whether this is due to the low-complexity convolutional architecture we used for the encoder and decoder or due to the assumption of a diagonal covariance matrix for the stochastic layer remains subject for further investigation. Because of architectural reasons it is difficult to compare the M2 model to the other two models in a fair manner, therefore, we omit the evaluation of the M2 model with convolutional VAE's.

In order to validate the generalization ability of the models we tested the CVAE based models that were trained on MNIST on the SEMEION dataset. SEMEION is an handwritten digits dataset similar to MNIST. The digits of the SEMEION dataset have been resized to 20×20 pixels using bilinear interpolation and the center of mass according to the pixels of each image has been centered in a 28×28 image in order to adjust the format as given by the MNIST dataset. However, during the creation of the original SEMEION dataset the aspect ratio of the images is not preserved. Moreover, the dataset consists of samples that were either written slowly and precise or quick with minimum precision. Many of the samples are also cropped. These factors make the dataset challenging to be tested with the models that were trained on MNIST, resulting in decreased performance, but in our opinion give a good indication of generalization performance. The convolutional DLR-VAE achieves 13.31% error on this independent dataset while the decoupled model 14.95%.

5.3.1. *Effects of Discriminative Regularization on the Latent Manifold*

In order to illustrate the regularization effect of training a DLR-VAE we trained a vanilla VAE and a DLR-VAE model on the MNIST dataset with a latent variable z with two dimensions. The resulting latent manifold clusters of the training dataset are shown in fig. 4 (VAE, DLR-VAE with softmax and cross entropy loss function for classification, and DLR-VAE performing regression on the class label, as an illustrative example). The latent coding of the DLR-VAE

Table 1: Test errors of the fully connected (fc) and convolutional (conv) VAE models on different datasets (with upper and lower bounds given by Jeffreys intervals having 95% Bayesian credibility, as described in subsection 5.4); best results marked with bold

Model	MNIST-fc	SVHN-fc	MNIST-conv	SVHN-conv	SEMEION /w MNIST-conv
VAE+DNN	1.73%	12.47%	1.01%	12.22%	14.95%
95% bounds	1.49–2%	12.08–12.87%	0.83–1.32%	11.83–12.63%	13.25–18.75%
DLR-VAE	1.13%	12.25%	0.74%	12.19%	13.31%
95% bounds	0.94–1.35%	11.87–12.66%	0.59–0.92%	11.8–12.59%	11.71–15.04%
M2	0.87%	55.34%	—	—	—
95% bounds	0.7–1.07%	54.36–56.31%	—	—	—

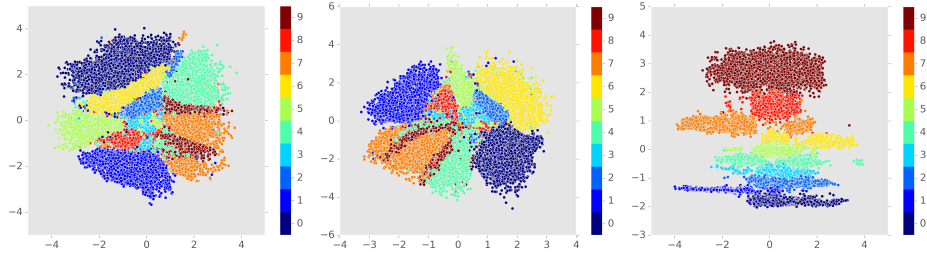


Figure 4: Comparison of the parameter μ of the latent embeddings of a VAE+DNN (left) and DLR-VAE (middle) for classification, and of a DLR-VAE, where (as an illustrative example) a regression on the class label is performed. The colors denote the class labels.

has been regularized by the classifier or regressor, respectively. We used a two-layer fully connected neural network with only one neuron per layer for the toy regression problem. The regularizing effect of this regressor forces the latent manifold to sort the points according to the label on one of the axes of the latent variable. We reason that the shape/style-dependent variance of the data is mostly represented on the first dimension of the latent variable and the class dependent variance on the second.

5.3.2. Semi-supervised Learning

The results of the experiments regarding semi-supervised learning are shown in fig. 5. We compare our proposed models SS0 and SS1 to the models M1+M2 and standalone M2 according to the results shown in Kingma et al. [19]. It is interesting to note that the performance of model SS1 is significantly better

465 than SS0 and comparable to the M2 model in the case of only 100 or 600 labels used during training. We reason that the model is able to learn additional information from the “guessing” stage were the unlabeled data is evaluated with the current learning state according to the labeled data available.

In the case of 1000 available labels the M2 model performs slightly better
470 than the SS1 model and in the case of 3000 labels the SS1 model outperforms the M2 model. The M1+M2 model shows the most consistent and accurate results, albeit being computationally most complex, and requires longer training as two models need to be trained. The results encourage us that the DLR-VAE can be feasibly used for semi-supervised learning tasks. An advantage of the
475 DLR-VAE is that in the case of a problem with a large amount of different class labels it is computationally less complex than the M2 or M1+M2 model during inference. This is an important feature for application scenarios where computational resources are limited and for time-critical applications.

5.4. Significance Evaluation

480 Since we see the same effect over different experiments, it is already a strong indication that the results are not due to chance, but we also apply statistical methods to show this. The variance of the performance can be estimated by bootstrapping, and cross-validation can simulate the effect of new data, but the most important performance measure is the performance on independently
485 acquired test data (the SEMEION dataset in our case). As classifiers get better, the offered improvements diminish, risking to be purely due to chance and not generalize. Therefore, approaches presented below can be of general use when evaluating different classifiers as well.

The most straightforward way to compare the performance of two methods,
490 is to test whether their confusion matrices differ more than it would be expected by chance. Thus, the null hypothesis is that the two methods produce the same proportion of values in the cells, which can be checked by performing Fishers’s exact test [9, 10]. Since we are dealing with multiclass prediction, it has to be applied to the contingency tables (i.e. matrices which are larger than 2×2), but

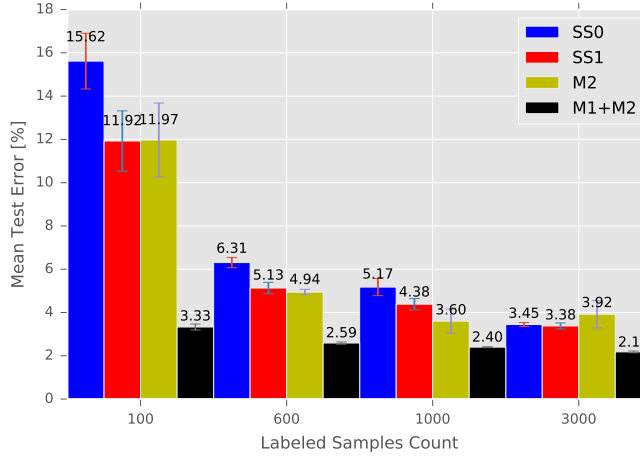


Figure 5: Performance of SS0 and SS1 in the semi-supervised training scenario on MNIST with different numbers of labeled training samples. See Kingma et al. [19] for M2 and M1+M2.

the test works for such cases as well. It assumes a (multivariate) hypergeometric distribution of the data, practically meaning that the methods predict a fixed number of samples in the different classes (but this number can differ between the classes). This is not guaranteed in our case, but since the performance is so close to each other (and to 100%), it is only a minor deviation. Due to memory limitations we had to fall back to the approximation of the test results using a Monte Carlo simulated hypergeometric test according to Mehta and Patel [30]. Under certain conditions the Pearson-Cochran chi-square test is a very good and efficient approximation both for the binary and multiclass cases [6].

Another option is to compare the values in the two contingency tables cell by cell, and check how much better they are for one method versus the other. Of course the values on the diagonal and the off-diagonal ones must be stored with different signs s.t. the pairwise differences indicate improvement in the same direction. This has some redundancy, as for example a difference in one misclassification is counted twice (once on the diagonal and once off-diagonal). However, the general trend of the differences can be checked with a Wilcoxon Rank Sum test, and even confidence intervals can be computed [2]. It is ap-

plicable here because we have paired measurements of the same cells, and this test compares the median difference to a given value. In our case, we define the null hypothesis as the median difference being 0, and try to reject it with
515 a one-sided test, in order to see how much better DLR-VAE is. This is a non-parametric test, i.e. it has no assumptions on the distribution of the data, but since the median is estimated, it is safest when the differences are symmetrically distributed around the median difference. Non-parametric tests have typically lower statistical power, meaning they are more conservative.

520 Lastly, one can also consider the true positive rate (TPR), more precisely, how many test samples were classified correctly out of the total test set. To avoid the complications of dealing with the true underlying multinomial distributions (though that is also possible, but falls outside the scope of this paper), we can simplify this case as a set of Bernoulli trials. Assuming a balanced test
525 set, the true probability of success is the real TPR, for which we obtained an estimate by applying the method to the test data. The number of true positives we obtain by classifying the test set then follows a Binomial distribution, with the proportion parameter $p = \text{TPR}$. This parameter can be estimated in a Bayesian way, including a credible interval for it, using for example the Jeffreys
530 interval¹ [3, 5]. These intervals are shown for the results in tbl. 1. If the credible intervals do not overlap, then for $\alpha = 5\%$:

$$P(p_1 \leq p_2) < 1 - P(p_1 > \text{Lower}_1) \cdot P(p_2 < \text{Upper}_2) = 1 - (1 - \alpha/2) \cdot (1 - \alpha/2) < 5\% \quad (13)$$

These methods were applied to evaluate the proposed DLR-VAE models on multiple datasets, using their standard implementation in R’s core libraries. The different mechanisms and assumptions mean that they are better suited to
535 detect certain types of differences than others. Fishers’s exact test can detect

¹for more details please see Márton and Türker [29], where the Jeffreys interval was used to derive the number of test cases needed to obtain certain bounds on the accuracy

Table 2: Statistical significance of the improvement by DLR-VAE over the VAE+DNN baseline model, see tbl. 1 (note that for the Jeffreys interval distance we took a conservative approach and reported the difference between the upper bound of VAE+DNN vs. the upper and lower bound of DLR-VAE, respectively); significant p-values marked with bold

Test	Improvement	MNIST-fc	SVHN-fc	MNIST-conv	SVHN-conv	SEMEION /w MNIST-conv
Fisher's	p-value	0.95	9.54e-3	0.98	1e-5	1.75e-2
Wilcoxon	lower 95%	1	-1	3.7e-5	-2.5	-4.5e-5
Rank	upper 95%	∞	∞	∞	∞	∞
Sum	p-value	2.1e-7	0.26	2.22e-3	0.38	0.1
Jeffreys	lower 95%	0.14%	-0.58%	-0.09%	-0.76%	-1.79%
interval	upper 95%	0.55%	0.21%	0.24%	0.03%	1.55%
	p-value	<5%	$\geq 5\%$	$\geq 5\%$	$\geq 5\%$	$\geq 5\%$
Significant at least once		yes	yes	yes	yes	yes

very small differences given a large number of samples, while the Wilcoxon Rank Sum test and the Jeffreys interval comparison are less powerful if the differences are small. Additionally, due to the bathtub shape of the non-informative Jeffreys prior for the binomial proportion p , the Jeffreys interval gets wider as the TPR approaches 50%, the prior being geared towards high, or low, TPR values (due to the symmetry of the problem, it does not change anything if the error rate is used instead of the TPR).

The results of the different tests are shown in tbl. 2. As discussed above, not all tests detected a significant difference between the models, but in all cases at least one did. Therefore, even after controlling for multiple hypothesis testing (depending on its scope and method used), we can assume that we are observing a real effect, which can be more pronounced in more complicated tasks (since the largest difference being observed is on the SEMEION dataset: 1.6%, even though not statistically significant due to the smaller sample size).

6. Conclusion

In this work we investigated the regularization effect of training a DNN classifier in combination with a variational auto-encoder. The methods are specially suitable for application when computational complexity is of concern.

Moreover, employing the DLR-VAE in semi-supervised scenarios remains an
555 interesting topic. We also presented a general approach to test the statistical
significance of classification results, and used it to evaluate our models.

In-depth analysis of the performance of the presented approach and the
influence of the regularization effect on the topology of the latent manifold in
regression scenarios gives motivation for further research. Additionally, the com-
560 bination of discriminative with adversarial regularization can provide a powerful
tool for shaping the latent representation by a Bayes prior (due to optimizing
the lower bound), functional objectives (due to discriminative regularization)
and arbitrary distributions (due to adversarial training).

Another potential application that exploits the underlying generalization
565 power of the latent manifold is incremental learning: New samples (possibly
even from a different application domain) are added to a pre-trained network,
reusing and refining the existing representation for the new task, thus reducing
the amount of training data that is required.

We believe that such approaches are highly useful for a wide range of appli-
570 cations where raw sensor data needs to be interpreted, with a limited amount of
(partially) labeled data. Such an example application was already explored in
the case of classification and regression based on structure-borne sound in the
industrial manipulation domain [32], and could prove useful when dealing with
depth images as well.

575 **Acknowledgements**

This work was partly funded by the European Union Project RobDREAM
under the H2020 framework programme grant agreement No. 645403 (no in-
volvement in study design; in the collection, analysis and interpretation of data;
in the writing of the report; and in the decision to submit the article for publi-
580 cation).

References

- [1] M Ehsan Abbasnejad, Anthony Dick, and Anton van den Hengel. Infinite variational autoencoder for semi-supervised learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 781–790. IEEE, 2017.
585
- [2] David F. Bauer. Constructing Confidence Sets Using Rank Statistics. *Journal of the American Statistical Association*, 67(339):687–690, 1972. URL <http://www.jstor.org/stable/2284469>.
- [3] Lawrence D. Brown, T. Tony Cai, and Anirban Dasgupta. Interval estimation for a binomial proportion. *Statistical Science*, 16:101–133, 2001.
590
- [4] Massimo Buscema. Metanet*: The theory of independent judges. *Substance use & misuse*, 33(2):454–455, 1998. Section: Recognition of Numbers Written by Hand.
- [5] T. Tony Cai. One-sided confidence intervals in discrete distributions. *Journal of Statistical Planning and Inference*, 131(1):63–88, 2005. ISSN 0378-3758. doi: <http://dx.doi.org/10.1016/j.jspi.2004.01.005>.
595
- [6] Ian Campbell. Chi-squared and Fisher-Irwin tests of two-by-two tables with small sample recommendations. *Stat in Med*, 26:3661–3675, 2007.
- [7] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.
600
- [8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.
605
- [9] Ronald A Fisher. On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1): 87–94, 1922.

- [10] Ronald Aylmer Fisher. *Statistical Methods for Research Workers*. Biological
610 cal Monographs and Manuals (F. A. E. Crew, D. Ward Cutler eds.) No. V.
Oliver and Boyd, Edinburgh and London, 11 edition, 1950.
- [11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-
Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative
adversarial nets. In *Advances in Neural Information Processing Systems*,
615 pages 2672–2680, 2014.
- [12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual
learning for image recognition. In *Proceedings of the IEEE conference on
computer vision and pattern recognition*, pages 770–778, 2016.
- [13] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun
620 Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets:
Efficient convolutional neural networks for mobile vision applications. *arXiv
preprint arXiv:1704.04861*, 2017.
- [14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Wein-
berger. Densely connected convolutional networks. In *CVPR*, 2017.
- [15] Gao Huang, Shichen Liu, Laurens Van der Maaten, and Kilian Q Wein-
625 berger. Condensenet: An efficient densenet using learned group convolu-
tions. *group*, 3(12):11, 2018.
- [16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating
deep network training by reducing internal covariate shift. *arXiv preprint
630 arXiv:1502.03167*, 2015.
- [17] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic opti-
mization. *arXiv preprint arXiv:1412.6980*, 2014.
- [18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes.
arXiv preprint arXiv:1312.6114, 2013.

- [19] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.
- [20] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.
- [21] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pages 971–980, 2017.
- [22] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.
- [23] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: <http://www.cs.toronto.edu/kriz/cifar.html>*, 2014.
- [24] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015.
- [25] Alex Lamb, Vincent Dumoulin, and Aaron Courville. Discriminative regularization for generative models. *arXiv preprint arXiv:1602.03220*, 2016.
- [26] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits, 1998.
- [27] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.
- [28] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

- [29] Zoltán-Csaba Márton and Serkan Türker. On bayesian inference for embodied perception of object poses. In *Workshop on Metrics of Embodied Learning Processes in Robots and Animals at IROS'13*, page 3, Tokyo, Japan, 2013.
- 665 [30] Cyrus R Mehta and Nitin R Patel. Algorithm 643: Fexact: a fortran subroutine for fisher’s exact test on unordered $r \times c$ contingency tables. *ACM Transactions on Mathematical Software (TOMS)*, 12(2):154–161, June 1986. ISSN 0098-3500. doi: 10.1145/6497.214326.
- [31] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and
670 Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 5, 2011.
- [32] Michael Neumann, Korbinian Nottensteiner, Ingo Kossyk, and Zoltan-Csaba Marton. Material classification through knocking and grasping by
675 learning of structure-borne sound under changing acoustic conditions. In *2018 14th IEEE International Conference on Automation Science and Engineering (CASE)*, 2016.
- [33] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 1520–1528, 2015.
680
- [34] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. In *Advances In Neural Information Processing Systems*, pages 2352–2360, 2016.
- 685 [35] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

- [36] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.
- ⁶⁹⁰ [37] Jakub M Tomczak and Max Welling. Improving variational auto-encoders using householder flow. *arXiv preprint arXiv:1611.09630*, 2016.