Direct Evolutionary Optimization of Variational Autoencoders With Binary Latents

Jakob Drefs[1(✉),⋆], Enrico Guiraud[2(✉),⋆], Filippos Panagiotou[1], and Jörg Lücke[1[0000−0001−9921−2529]]

[1] Machine Learning Lab, University of Oldenburg, 26129 Oldenburg, Germany
{jakob.drefs,filippos.panagiotou,joerg.luecke}@uol.de
[2] CERN, 1211 Geneva, Switzerland
enrico.guiraud@cern.ch

Abstract. Many types of data are generated at least partly by discrete causes. Deep generative models such as variational autoencoders (VAEs) with binary latents consequently became of interest. Because of discrete latents, standard VAE training is not possible, and the goal of previous approaches has therefore been to amend (i.e, typically anneal) discrete priors to allow for a training analogously to conventional VAEs. Here, we divert more strongly from conventional VAE optimization: We ask if the discrete nature of the latents can be fully maintained by applying a direct, discrete optimization for the encoding model. In doing so, we sidestep standard VAE mechanisms such as sampling approximation, reparameterization and amortization. Direct optimization of VAEs is enabled by a combination of evolutionary algorithms and truncated posteriors as variational distributions. Such a combination has recently been suggested, and we here for the first time investigate how it can be applied to a deep model. Concretely, we (A) tie the variational method into gradient ascent for network weights, and (B) show how the decoder is used for the optimization of variational parameters. Using image data, we observed the approach to result in much sparser codes compared to conventionally trained binary VAEs. Considering the for sparse codes prototypical application to image patches, we observed very competitive performance in tasks such as 'zero-shot' denoising and inpainting. The dense codes emerging from conventional VAE optimization, on the other hand, seem preferable on other data, e.g., collections of images of whole single objects (CIFAR etc), but less preferable for image patches. More generally, the realization of a very different type of optimization for binary VAEs allows for investigating advantages and disadvantages of the training method itself. And we here observed a strong influence of the method on the learned encoding with significant impact on VAE performance for different tasks.

Keywords: Variational Autoencoder · Evolutionary Optimization · Sparse Encoding · Variational Optimization · Binary Latents

⋆ Joint first authorship.

1 Introduction and Related Work

Objects or edges in images are either present or absent, which suggests the use of discrete latents for their representation. There are also typically only few objects per image (of all possible objects) or only few edges in any given image patch (of all possible edges), which suggests a sparse code (e.g., [43,61,20,56]). In order to model such and similar data, we study a novel, direct optimization approach for variational autoencoders (VAEs), which can learn discrete and potentially sparse encodings. VAEs [32,51] in their many different variations, have successfully been applied to a large number of tasks including semi-supervised learning (e.g., [40]), anomaly detection (e.g., [33]) or sentence and music interpolation [5,52] to name just a few. The success of VAEs, in these tasks, rests on a series of methods that enable the derivation of scalable training algorithms to optimize VAE parameters. These methods were originally developed for Gaussian priors [32,51]. To account for VAEs with discrete latents, novel methodology had to be introduced (we elaborate below and later in Sec. S1).

The training objective of VAEs is derived from a likelihood objective, i.e., we seek model parameters Θ of a VAE that maximize the data log-likelihood, $\mathrm{L}(\Theta) = \sum_n \log\big(p_\Theta(\vec{x}^{\,(n)})\big)$, where we denote by $\vec{x}^{(1:N)}$ a set of N observed data points, and where $p_\Theta(\vec{x})$ denotes the modeled data distribution. Like conventional autoencoders (e.g., [1]), VAEs use a deep neural network (DNN) to generate (or decode) observables $\vec{x} \in \mathbb{R}^D$, from a latent code $\vec{z}$. Unlike conventional autoencoders, however, the generation of data $\vec{x}$ is not deterministic but it takes the form of a probabilistic generative model. For VAEs with binary latents, we here consider a generative model with Bernoulli prior:

$$p_\Theta(\vec{z}) = \prod_h \big(\pi_h^{z_h}(1-\pi_h)^{(1-z_h)}\big), \;\; p_\Theta(\vec{x}\,|\,\vec{z}) = \mathcal{N}\big(\vec{x}; \vec{\mu}(\vec{z}; W), \sigma^2 \mathbb{I}\big), \qquad (1)$$

with $\vec{z} \in \{0,1\}^H$ being a binary code, $\vec{\pi} \in [0,1]^H$ being parameters of the prior on $\vec{z}$, and the non-linear function $\vec{\mu}(\vec{z}; W)$ being a DNN (that sets the mean of a Gaussian distribution). $p_\Theta(\vec{x}\,|\,\vec{z})$ is commonly referred to as *decoder*. The set of model parameters is $\Theta = \{\vec{\pi}, W, \sigma^2\}$, where W incorporates DNN weights and biases. Here, we assume homoscedasticity of the Gaussian distribution, but note that there is no obstacle to generalizing the model by inserting a DNN non-linearity that outputs a covariance matrix. Similarly, the algorithm could easily be generalized to different noise distributions should the task at hand call for it. Here, however, we will focus on the elementary VAEs given by Eq. (1).

For conventional and discrete VAEs, essentially all optimization approaches seek to approximately maximize the log-likelihood using the following series of methods (we elaborate in Sec. S1):

(A) Instead of the log-likelihood, a variational lower-bound (a.k.a. ELBO) is optimized.
(B) VAE posteriors are approximated by an *encoding model*, i.e., by a specific distribution (usually Gaussian) parameterized by one or more DNNs.
(C) The variational parameters of the encoder are optimized using gradient ascent on the lower bound, where the gradient is evaluated based on sampling

and the reparameterization trick (which allows for sufficiently low-variance and yet efficiently computable estimates).

(D) Using samples from the encoder, the parameters of the decoder are optimized using gradient ascent on the variational lower bound.

Optimization procedures for VAEs with discrete latents follow the same steps (Points A to D). However, discrete or binary latents pose substantial further obstacles for learning, mainly due to the fact that backpropagation through discrete variables is generally not possible or biased [53,2]. Widely used stochastic gradient estimators for discrete random variables typically either exploit the REINFORCE [65] estimator in combination with variance control techniques [11,13,38,34] or reparameterization of continuous relaxations of discrete distributions [29,41]; reparameterization is also combined with REINFORCE [22] or generalized to non-reparameterizable distributions [8]. Also a recent approach by Berliner et al. [3] is related to REINFORCE but uses natural evolution strategies (not to be confused with evolutionary optimization we apply here) to derive low-variance estimates for gradients (also see Related Work and Sec. S1). While accomplishing, in different senses, the goal of maintaining standard VAE training as developed for continuous latents (i.e., learning procedures and/or learning objectives that allow for gradient-based optimization of the encoder and decoder DNNs), gradient estimation methods usually apply significant amounts of methodology *additional* to the learning methods conventionally applied for VAE optimization (see Fig. S2). These additional methods, their accompanying design decisions and used hyper-parameters increase the complexity of the system. Furthermore, the additional methods usually impact the learned representations. For instance, softening of discrete distributions, e.g., by using 'Gumbel-softmax' [29] or 'tanh' approximations [18] seems to favor dense codes. While dense codes (as also used by conventional VAEs and generative adversarial networks [21]) can result in competitive performance for a subset of the above discussed tasks, other recent contributions point out advantages of sparse codes, e.g., in terms of disentanglement [63] or robustness [60,45].

In order to avoid adding methods for discrete latents to those already in place for standard VAEs, it may be reasonable to investigate more direct optimization procedures that do not require, e.g., a softening of discrete distributions or other mechanisms. Such a direct approach is challenging, however, because once DNNs are used to define the encoding model (as commonly done), we require methodologies for discrete latents to estimate gradients for the encoder (as done via sampling and reparameterization; see Points C and D). A direct optimization procedure, as we investigate here, consequently has to change VAE training substantially. For the data model of Eq. (1), we will maintain the variational setting (Point A) and a decoding model with DNNs as non-linearity. However, we will not use an encoding model parameterized by DNNs (Point B). Instead, the variational bound will be increased w.r.t. an implicitly defined encoding model which allows for an efficient discrete optimization. The procedure does not require gradients to be computed for the encoder such that discrete latents are addressed without the use of reparameterization trick and sampling approximations.

Related Work. In order to maintain the general VAE framework for encoder optimization in the case of discrete latents, different groups have suggested different possible solutions (for discussion of numerical evaluations of related approaches, see Sec. S1.3): Rolfe [53], for instance, extends VAEs with discrete latents by auxiliary continuous latents such that gradients can still be computed. Work on the concrete distribution [41] or Gumbel-softmax distribution [29] proposes newly defined continuous distributions that contain discrete distributions as limit cases. Lorberbom et al. [39] merge the Gumbel-Max reparameterization with the use of direct loss minimization for gradient estimation, enabling efficient training on structured latent spaces (also compare [49,48] for further improved Gumbel-softmax versions). Furthermore, work, e.g., by van den Oord et al. [44] combines VAEs with a vector quantization (VQ) stage in the latent layer. Latents become discrete through quantization but gradients for learning are adapted from latent values before they are processed by the VQ stage. Similarly, Tomczak & Welling [62] use, what they call, (learnable) pseudo-inputs which determine a mixture distribution as prior, and the ELBO then contains an additional regularization for consistency between prior and average posterior. Tonolini et al. [63] extend this work and introduce an additional DNN classifier which selects pseudo-inputs and whose weights are learned instead of the pseudo-inputs themselves. Tonolini et al. also argue for the benefits not only of discrete latents but of a sparse encoding in the latent layer in general. Fajtl et al. [18] base their approach on a deterministic autoencoder and use a tanh-approximation of binary latents and projections to spheres in order to treat binary values. Targeting not only the optimization of discrete latent VAEs but also more general approaches such as probabilistic programming or general stochastic automatic differentiation, Bingham et al. [4] and van Krieken et al. [35] apply gradient estimators for discrete random variables which optimize surrogate losses [54] derived based on the score function [19] or other methods [35].

2 Direct Variational Optimization

Let us consider the variational lower bound of the likelihood. If we denote by $q^{(n)}_{\Phi}(\vec{z})$ the variational distributions with parameters $\Phi = (\Phi^{(1)}, \ldots, \Phi^{(N)})$, then the lower bound is given by:

$$\mathcal{F}(\Phi,\Theta) = \textstyle\sum_n \mathbb{E}_{q^{(n)}_{\Phi}}\big[\log\big(p_{\Theta}(\vec{x}^{\,(n)}\,|\,\vec{z})\,p_{\Theta}(\vec{z})\,\big)\big] - \textstyle\sum_n \mathbb{E}_{q^{(n)}_{\Phi}}\big[\log\big(q^{(n)}_{\Phi}(\vec{z})\big)\big], \quad (2)$$

where we sum over all data points $\vec{x}^{(1:N)}$, and where $\mathbb{E}_{q^{(n)}_{\Phi}}\big[h(\vec{z})\big]$ denotes the expectation value of a function $h(\vec{z})$ w.r.t. $q^{(n)}_{\Phi}(\vec{z})$. The general challenge for the maximization of $\mathcal{F}(\Phi,\Theta)$ is the optimization of the encoding model $q^{(n)}_{\Phi}$. VAEs with discrete latents, as an additional challenge, have to address the question how gradients w.r.t. discrete latents can be computed. Seeking to avoid the problem of gradients w.r.t. discrete variables, we do *not* use a DNN for the encoding model. Consequently, we need to define an *alternative* encoding model $q_\Phi^{(n)}$, which has to remain sufficiently efficient. Considering prior work on generative models with discrete latents, variational distributions based on truncated posteriors offer themselves as such an alternative. Truncated posteriors have previously been considered to be functionally competitive (e.g., [56,27,58]). Most relevant for our purposes are very efficient and fully variational approaches that allow mixture models [26,17] and shallow generative approaches [14] to be very efficiently scaled to large model sizes. In all these previous applications, optimization of truncated variational distributions used standard expectation maximization based on closed-form or pseudo-closed form M-steps available for the shallow decoder models considered. In the context of VAEs with discrete latents, the important question arising is if or how efficient optimization with truncated variational distributions can be performed for deep generative models.

Optimization of the Encoding Model. Encoder optimization is usually based on a reformulation of the variational bound of Eq. (2) given by:

$$\mathcal{F}(\Phi,\Theta) = \textstyle\sum_n \mathbb{E}_{q_\Phi^{(n)}}\big[\log\big(p_\Theta(\vec{x}^{\,(n)}\,|\,\vec{z})\big)\big] - \textstyle\sum_n D_{\mathrm{KL}}\big[q_\Phi^{(n)}(\vec{z});p_\Theta(\vec{z})\big]. \qquad (3)$$

For discrete latent VAEs, the variational distributions in Eq. (3) are commonly replaced by an amortized encoding model $q_\Phi(\vec{z})$ with a DNN-based parameterization. When expectations w.r.t. $q_\Phi(\vec{z})$ are approximated (as usual) via sampling, the encoder optimization requires gradient estimation methods for discrete random variables (cf. Related Work and Sec. S1). At this point truncated posteriors represent alternative variational distributions which avoid gradients w.r.t. discrete latents. Given a data point $\vec{x}^{\,(n)}$, a truncated posterior is the posterior itself truncated to a subset $\Phi^{(n)}$ of the latent space, i.e., for $\vec{z}\in\Phi^{(n)}$ applies:

$$q_\Phi^{(n)}(\vec{z}) := \frac{p_\Theta(\vec{z}\,|\,\vec{x}^{\,(n)})}{\displaystyle\sum_{\vec{z}\,'\in\Phi^{(n)}} p_\Theta(\vec{z}\,'\,|\,\vec{x}^{\,(n)})} = \frac{p_\Theta(\vec{x}^{\,(n)}\,|\,\vec{z})\,p_\Theta(\vec{z})}{\displaystyle\sum_{\vec{z}\,'\in\Phi^{(n)}} p_\Theta(\vec{x}^{\,(n)}\,|\,\vec{z}\,')\,p_\Theta(\vec{z}\,')} \qquad (4)$$

while $q_\Phi^{(n)}(\vec{z}) = 0$ for $\vec{z}\notin\Phi^{(n)}$. The subsets $\Phi = \{\Phi^{(n)}\}_{n=1}^N$ are the variational parameters. Centrally for this work, truncated posteriors allow for a specific alternative reformulation of the bound. The reformulation recombines the entropy term of the original form (Eq. (2)) with the first expectation value into a single term, and is given by (see [14,26,17] for details):

$$\mathcal{F}(\Phi,\Theta) = \sum_n \log\Big(\sum_{\vec{z}\in\Phi^{(n)}} p_\Theta(\vec{x}^{\,(n)}\,|\,\vec{z})\,p_\Theta(\vec{z})\Big)\,. \qquad (5)$$

Thanks to the simplified form of the bound, the variational parameters $\Phi^{(n)}$ of the encoding model can now be sought using direct discrete optimization procedures. More concretely, because of the specific form of Eq. (5), pairwise comparisons of joint probabilities are sufficient to maximize the lower bound: if we update the set $\Phi^{(n)}$ for a given $\vec{x}^{\,(n)}$ by replacing a state $\vec{z}^{\,\mathrm{old}}\in\Phi^{(n)}$ with a state $\vec{z}^{\,\mathrm{new}} \notin \Phi^{(n)}$, then $\mathcal{F}(\Phi, \Theta)$ increases if and only if:

$$\log\big(p_{\Theta}(\vec{x}^{\,(n)}, \vec{z}^{\,\mathrm{new}})\big) > \log\big(p_{\Theta}(\vec{x}^{\,(n)}, \vec{z}^{\,\mathrm{old}})\big). \tag{6}$$

To obtain intuition for the pairwise comparison, consider the form of $\log(p_{\Theta}(\vec{x}, \vec{z}))$ when inserting the binary VAE defined by Eq. (1). Eliding terms that do not depend on $\vec{z}$ we obtain:

$$\widetilde{\log p}_{\Theta}(\vec{x}, \vec{z}) = -\|\vec{x} - \vec{\mu}(\vec{z}, W)\|^2 \;-\; 2\,\sigma^2 \textstyle\sum_h \tilde{\pi}_h\, z_h, \tag{7}$$

where $\tilde{\pi}_h = \log\big((1 - \pi_h)/\pi_h\big)$. The expression assumes an even more familiar form if we restrict ourselves for a moment to sparse priors with $\pi_h = \pi < \frac{1}{2}$, i.e., $\tilde{\pi}_h = \tilde{\pi} > 0$. The criterion defined by Eq. (6) then becomes:

$$\|\vec{x}^{\,(n)} - \vec{\mu}(\vec{z}^{\,\mathrm{new}}, W)\|^2 \;+\; 2\,\sigma^2\tilde{\pi}\;|\vec{z}^{\,\mathrm{new}}| < \;\; \|\vec{x}^{\,(n)} - \vec{\mu}(\vec{z}^{\,\mathrm{old}}, W)\|^2 \;+\; 2\,\sigma^2\tilde{\pi}\;|\vec{z}^{\,\mathrm{old}}|\,, \tag{8}$$

where $|\vec{z}\,| = \sum_{h=1}^{H} z_h$ and $2\,\sigma^2\tilde{\pi} > 0$. Such functions are routinely encountered in sparse coding or compressive sensing [16]: for each set $\Phi^{(n)}$, we seek those states $\vec{z}$ that are reconstructing $\vec{x}^{\,(n)}$ well while being sparse ($\vec{z}$ with few non-zero bits). For VAEs, $\vec{\mu}(\vec{z}, W)$ is a DNN and as such much more flexible in matching the distribution of observables $\vec{x}$ than can be expected from linear mappings. Furthermore, criteria like Eq. (8) usually emerge for maximum a-posteriori (MAP) training in sparse coding [43]. In contrast to MAP, however, here we seek a *population* of states $\vec{z}$ in $\Phi^{(n)}$ for each data point. It is a consequence of the reformulated lower bound defined by Eq. (5) that it remains optimal to evaluate joint probabilities (as for MAP) although the constructed population of states $\Phi^{(n)}$ can capture (unlike MAP training) rich posterior structures.

Evolutionary Search. But how can new states $\vec{z}^{\,\mathrm{new}}$ that optimize $\Phi^{(n)}$ be found efficiently in high-dimensional latent spaces? While blind random search for states $\vec{z}$ can in principle be used, it is not efficient; and adaptive search space approaches [26,17] are only defined for mixture models. However, a recently suggested combination of truncated variational optimization with evolutionary optimization (EVO; [14]) is more generally defined for models with discrete latents, and does only require the efficient computation of joint probabilities $p_{\Theta}(\vec{x}, \vec{z})$. It can consequently be adapted to the VAEs considered here.

EVO optimization interprets the sets $\Phi^{(n)}$ of Eq. 4 as populations of binary genomes $\vec{z}$, and we can here adapt it by using Eq. (7) in order to assign to each $\vec{z} \in \Phi^{(n)}$ a fitness for evolutionary optimization. For the concrete updates, we use for each EVO iteration $\Phi^{(n)}$ as initial parent pool. We then apply the following genetic operators in sequence to suggest candidate states $\vec{z}^{\,\mathrm{new}}$ to update the $\Phi^{(n)}$ based on Eq. (6) (see Fig. S3 for an illustration and Sec. S1.2 and [14] for further details): Firstly, *parent selection* stochastically picks states from the parent pool. Subsequently, each of these states undergoes *mutation* which flips one or more entries of the bit vectors. Offspring diversity can be further increased by crossover operations. Using the *children* generated this way as the new parent pool, the procedure is repeated giving birth to multiple *generations* of candidate states. Finally, we update $\Phi^{(n)}$ by substituting individuals with low fitness with candidates with higher fitness according to Eq. (6). The whole procedure can be seen as an evolutionary algorithm (EA) with perfect memory or very strong elitism (individuals with higher fitness never drop out of the gene pool). Note that the improvement of the variational lower bound depends on generating as many as possible *different* children with high fitness over the course of training.

We point out that the EAs optimize each $\Phi^{(n)}$ independently, which allows for distributed execution s.t. the technique can be efficiently applied to large datasets in conjunction with stochastic or batch gradient descent on the model parameters Θ. The approach is, at the same time, memory intensive, i.e., all sets $\Phi^{(n)}$ need to be kept in memory (details in Sec. S1.1). Furthermore, we point out the we here optimize variational parameters $\Phi^{(n)}$ of the encoding model which is fundamentally different from the approach of Hajewski & Oliveira [24] (who use EAs to optimize DNN architectures of otherwise conventionally optimized VAEs with continuous latents).

Optimization of the Decoding Model. Using the previously described encoding model $q^{(n)}_{\Phi}(\vec{z})$, we can compute the gradient of Eq. (2) w.r.t. the decoder weights W which results in (see Sec. S1 for details):

$$\vec{\nabla}_W \mathcal{F}(\Phi,\Theta) = -\tfrac{1}{2\sigma^2}\sum_n \sum_{\vec{z}\in\Phi^{(n)}} q^{(n)}_{\Phi}(\vec{z})\ \vec{\nabla}_W \|\vec{x}^{\,(n)} - \vec{\mu}(\vec{z},W)\|^2 \tag{9}$$

The right-hand-side has salient similarities to standard gradient ascent for VAE decoders. Especially the familiar gradient of the mean squared error (MSE) shows that, e.g., standard automatic differentiation tools can be applied. However, the decisive difference is represented by the weighting factors $q^{(n)}_{\Phi}(\vec{z})$. Considering Eq. (4), we require all $\vec{z}\in\Phi^{(n)}$ to be passed through the decoder DNN in order to compute the $q^{(n)}_{\Phi}(\vec{z})$. As all states of $\Phi^{(n)}$ anyway have to be passed through the decoder for the MSE term of Eq. (9), the overall computational complexity is not higher than an estimation of the gradient with samples instead of states in $\Phi^{(n)}$ (but we use many states per $\Phi^{(n)}$, compare Tab. S1).

To complete the decoder optimization, update equations for variance σ^2 and prior parameters $\vec{\pi}$ can be computed in closed-form (compare, e.g., [57]) and are given by

$$\begin{aligned}\sigma^2 &= \tfrac{1}{DN}\sum_n \sum_{\vec{z}\in\Phi^{(n)}} q^{(n)}_{\Phi}(\vec{z})\ \|\vec{x}^{\,(n)} - \vec{\mu}(\vec{z},W)\|^2,\\ \vec{\pi} &= \tfrac{1}{N}\sum_n \sum_{\vec{z}\in\Phi^{(n)}} q^{(n)}_{\Phi}(\vec{z})\ \vec{z}\,.\end{aligned} \tag{10}$$

The full training procedure for binary VAEs is summarized in Alg. 1. We refer to the binary VAE trained with this procedure as *Truncated Variational Autoencoder* (TVAE) because of the applied truncated posteriors[1].

[1] Source code available at https://github.com/tvlearn.

Algorithm 1 Training Truncated Variational Autoencoders (TVAE)

```
Initialize model parameters $\Theta = (\vec{\pi}, W, \sigma^2)$
Initialize each $\Phi^{(n)}$ with $S$ distinct latent states
repeat
  for all batches in dataset do
    for sample $n$ in batch do
      $\Phi^{\mathrm{new}} = \Phi^{(n)}$
      for all generations do
        $\Phi^{\mathrm{new}} = \mathrm{mutation}\,(\mathrm{selection}\,(\Phi^{\mathrm{new}}))$
        $\Phi^{(n)} = \Phi^{(n)} \cup \Phi^{\mathrm{new}}$
      end for
      Define new $\Phi^{(n)}$ by selecting the $S$ fittest elements in $\Phi^{(n)}$ using Eq. (6)
    end for
    Use Adam to update $W$ using Eq. (9)
  end for
  Use Eq. (10) to update $\vec{\pi}$, $\sigma^2$
until parameters $\Theta$ have sufficiently converged
```

3 Numerical Experiments

TVAE can flexibly learn prior parameters $\vec{\pi}$, and if low values for the π_h are obtained (which will be the case), the code is sparse. The prototypical application domain to study sparse codes is image patch data [43,20]. We consequently use such data to investigate sparsity, scalability and efficiency on benchmarks. For all numerical experiments, we employ fully connected DNNs $\vec{\mu}(\vec{z}; W)$ for the decoder (compare Fig. S4); the exact network architectures and activations used are listed in Tab. S1. The DNN parameters are optimized based on Eq. (9) using mini-batches and the Adam optimizer (details in Sec. S2.1).

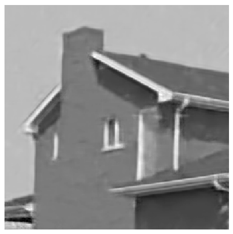
Verification and Scalability. After first verifying that the procedure can recover generating parameters using ground-truth data (see Sec. S2.2), we trained TVAE on $N = 100,000$ whitened image patches of $D = 16 \times 16$ pixels [25] using two different decoder architectures, namely a shallow, linear decoder with $H = 300$ binary latents, and second, a deep non-linear decoder with a 300-300-256 architecture (i.e., $H = 300$ binary latents and two hidden layers with 300 and 256 units, respectively; details in Sec. S2.3). For both linear and non-linear TVAE, we observed a sparse encoding with on average $\frac{\sum_h \pi_h}{H} = \frac{20.3}{300}$ and $\frac{\sum_h \pi_h}{H} = \frac{28.5}{300}$ active latents across data points, respectively. We observed sparse codes also when we varied the parameter initialization and further modified the decoder DNN architecture. As long as decoder DNNs were of small to intermediate size, we observed efficient scalability to large latent spaces (we went up to $H = 1,000$). Compared to linear decoders, the main additional computational cost is given by passing the latent states in the $\Phi^{(n)}$ sets through the decoder DNN instead of just through a linear mapping. The sets of states (i.e., the bitvectors in $\Phi^{(n)}$) could be kept small, at size $S = |\Phi^{(n)}| = 64$, such that $N \times (|\Phi^{(n)}| + |\Phi^{(n)}_{\mathrm{new}}|)$ states had to be evaluated per epoch. This compares to $N \times M$ states that would be used for standard VAE training (given M samples are drawn per data point). In contrast to standard VAE training, the sets $\Phi^{(n)}$ have to be remembered across iterations. For very large datasets, the additional $\mathcal{O}(N \times |\Phi^{(n)}| \times H)$ memory demand can be distributed over compute nodes, however.

Denoising - Controlled Conditions. Due to its non-amortized encoding model, the computational load of TVAE increases more strongly with data points compared to amortized training. Consequently, tasks such as disentanglement of features using high-dimensional input data, large DNNs, and small latent spaces are not a regime where the approach can be applied efficiently. With this in mind, we focused on tasks with relatively few data for which an as effective as possible optimization is required, and for which advantages of a direct optimization can be expected. As one such task, we here considered 'zero-shot' image denoising. To apply TVAE in a 'zero-shot' setting (in which no additional information besides the noisy image is available, e.g., [59,28]), we trained the model on overlapping patches extracted from a given noisy image and subsequently applied the learned encoding to estimate non-noisy image pixels (details in Sec. S2.4). In general, denoising represents a canonical benchmark for evaluating image patch models, and approaches exploiting sparse encodings have shown to be particularly well suited (compare, e.g., [42,68,56]). The 'zero-shot' setting has recently become popular also because the application of conventional DNN-based approaches has shown to be challenging (see discussion in Sec. S2.4).

One denoising benchmark, which allows for an extensive comparison to other methods is the House image. Standard benchmark settings for this image make use of additive Gaussian white noise with standard deviations $\sigma \in \{15, 25, 50\}$ (Fig. 1 A). First, consider the comparison in Fig. 1 C where all models used the same patch size of $D = 8 \times 8$ pixels and $H = 64$ latent variables (details in Sec. S2.4). Fig. 1 C lists the different approaches in terms of the standard measure of peak signal-to-noise ratio (PSNR). Values for MTMKL [61] and GSC [56] were taken from the respective original publications (which both established new state-of-the-art results when first published); for EBSC [14], we produced PSNRs ourselves by running publicly available source code (cf. Sec. S2.4). As can be observed, TVAE significantly improves performance for high noise levels; the approach is able to learn the best data representation for denoising and establishes new state-of-the-art results in this controlled setting (i.e., fixed D and H). The decoder DNN of TVAE provides the decisive performance advantage: TVAE significantly improves performance compared to EBSC (which can be considered as an approach with a shallow, linear decoding model), confirming that the high lower bounds of TVAE on natural images (compare Fig. S9) translate into improved performance on a concrete benchmark. For $\sigma = 25$ and $\sigma = 50$, TVAE also significantly improves on MTMKL, and GSC, which are both based on a spike-and-slab sparse coding (SSSC) model (also compare [20]). Despite the less

A Noisy (σ=50)

B TVAE Denoising Result



C Controlled Conditions

	σ=15	σ=25	σ=50
MTMKL	**34.29**	31.88	28.08
GSC	32.68	31.10	28.02
EBSC	32.20 ± .08	31.14 ± .13	28.50 ± .12
TVAE	34.27 ± .02	**32.65** ± .06	**29.61** ± .02

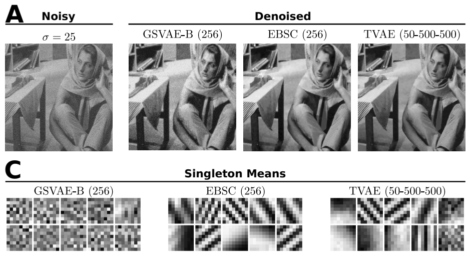
D Different Optimized Hyperparameters

	σ=15	σ=25	σ=50
N2V*	31.86 ± .37	29.34 ± .37	24.24 ± .57
S2S	34.86 ± .13	32.87 ± .08	29.95 ± .26
GSVAE-B	29.39 ± .02	28.32 ± .28	25.73 ± .14
GSVAE-C	30.94 ± .17	29.83 ± .04	26.75 ± .02
VLAE	34.47 ± .05	32.44 ± .05	29.16 ± .07
MTMKL	34.29	31.88	28.08
GSC	33.78	32.01	28.35
S5C	33.50	32.08	28.35
EBSC	33.66 ± .04	32.40 ± .07	28.98 ± .06
ES3C	**34.90** ± .04	**33.15** ± .06	29.83 ± .07
TVAE	34.27 ± .02	32.65 ± .06	**29.98** ± .05
N2V†	33.91	32.10	28.94
KSVD	34.32	32.15	27.95
WNNM	**35.13**	**33.22**	**30.33**
BM3D	34.94	32.86	29.37
EPLL	34.17	32.17	29.12
BDGAN	34.57	33.28	30.61
DPDNN	**35.40**	**33.54**	**31.04**

Fig. 1. Denoising results for House. **C** compares PSNRs (in dB) obtained with different 'zero-shot' models using a fixed patch size and number of latents (means and standard deviations were computed over three runs with independent noise realizations, see text for details). **D** lists PSNRs for different algorithms with different optimized hyper-parameters. The top category only requires the noisy image ('zero-shot' setting). The middle uses additional information such as noise level (KSVD, WNNM, BM3D) or additional noisy images with matched noise level (N2V†). The bottom three algorithms use large clean datasets. The highest PSNR per category is marked bold, and the overall highest PSNR is bold and underlined. **B** depicts the denoised image obtained with TVAE for $\sigma = 50$ in the best run (PSNR=30.03 dB).

flexible Bernoulli prior, the decoder DNN of TVAE provides the highest PSNR values for high noise levels.

Denoising - Uncontrolled Conditions. To extend the comparison, we next evaluated denoising performance without controlling for equal conditions, i.e., we also included approaches in our comparison that use large image datasets and/or different patch sizes for training (including multi-scale and whole image processing). Note that different approaches may employ very different sets of hyper-parameters that can be optimized for denoising performance (e.g., patch and dictionary sizes for sparse coding approaches, or network and training scheme hyper-parameters for DNN approaches). By allowing for comparison in this less controlled setting, we can compare to a number of recent approaches including large DNNs trained on clean data and training schemes specifically targeted to noisy training data. See Fig. 1 D for an extensive PSNR overview with results for other algorithms cited from their corresponding original publications if not stated otherwise. PSNRs for S5C originate from [55], GSVAE-B, EBSC and ES3C from [14], and WNNM and EPLL from [67]. For Noise2Void (N2V; [36]), Self2Self (S2S; [50]), GSVAE-C [29], and VLAE [47], we produced results ourselves by applying publicly available source code (details in Sec. S2.4). Note that the best performing approaches in Fig. 1 D were trained on noiseless data:



B

Model	Sparsity	Lower Bound	PSNR
GSVAE-B (256)	$\frac{128}{256}$	39.02	25.06
GSVAE-B (256-256-512)	$\frac{128}{256}$	38.68	25.06
GSVAE-C ((32×8)-256-512)	n/a	45.66	26.18
EBSC (256)	$\frac{1.5}{256}$	43.85	28.37
TVAE (256-256-512)	$\frac{2.3}{256}$	46.86	29.96
TVAE (50-500-500)	$\frac{2.6}{50}$	**47.98**	**30.03**
VLAE (50-500-500)	n/a	45.93*	28.83
ES3C (256)	$\frac{2.0}{256}$	**46.09**	**29.69**

Fig. 2. Data encodings and denoising results for Barbara obtained with generative model approaches and different decoding models. In **B**, approaches with binary (top) and continuous (bottom) latents are separated. EBSC and ES3C are considered as using a shallow, linear decoder. Listed are best performances of several runs of each algorithm. *VLAE uses importance sampling-based log-likelihood estimation. **C** compares decoder outputs for singleton (i.e., one-hot) input vectors. See Sec. S2.4 for details.

EPLL [71], BDGAN [70] and DPDNN [12] all make use of clean training data (typically hundreds of thousands of data points or more). EPLL, KSVD [15], WNNM [23] and BM3D [10] leverage a-priori noise level information (these algorithms use the ground-truth noise level of the test image as input parameter). As noisy data is very frequently occurring, lifting the requirement of clean data has been of considerable recent interest with approaches such as Noise2Noise (N2N; [37]), N2V, and S2S having received considerable attention.

Considering Fig. 1 D, first note that TVAE consistently improves PSNRs of N2V, also when comparing to a variant trained on external data with matched-noise level (N2V† in Fig. 1 D). At high noise level ($\sigma = 50$), PSNRs of TVAE represent state-of-the-art performance in the 'zero-shot' category (Fig. 1 D, top); compared to methods which exploit additional a-priori information (Fig. 1 D, middle and bottom), the denoising performance of TVAE (at high noise level) is improved only by WNNM, BDGAN and DPDNN. At lower noise levels, TVAE still performs competitively in the 'zero-shot' setting, yet highest PSNRs are obtained by other methods (S2S and ES3C). Figure 1 D reveals that TVAE can improve on two competing VAE approaches, namely GSVAE (which uses Gumbel-softmax-based optimization for discrete latents) and VLAE (which uses continuous latents and Gaussian posterior approximations). For more systematic comparison, we applied the VAE approaches using identical decoder architectures and identical patch sizes (details in Sec. S2.4). As striking difference between the approaches, we observed GSVAE to learn a significantly denser encoding compared to TVAE. Furthermore, we observed that the sparse encodings of TVAE resulted in strong performance not only in terms of denoising PSNR but also in terms of lower bounds (see Fig. 2).

Inpainting. Finally, we applied TVAE to 'zero-shot' inpainting tasks. For TVAE, the treatment of missing data is directly available given the probabilistic formulation of the model. Concretely, when evaluating log-joint probabilities of a datapoint, missing values are treated as unknown observables (details in Sec. S2.5).

	House 50%
Papyan et al.	34.58
BPFA	38.02
DIP	39.16
ES3C	**39.59**
TVAE	38.56

	Castle 50%	Castle 80%
MTMKL	n/a	28.94
BPFA	36.45	29.12
ES3C	**38.23**	**29.66**
TVAE	37.33	28.93
PLE	**38.34**	**30.07**
IRCNN	n/a	28.74

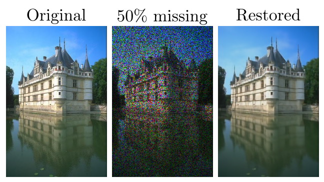


Fig. 3. Inpainting results for House (50% missing pixels) and Castle (50% and 80% missing; top group lists 'zero-shot' approaches). PSNR for Papyan et al. as reported in [64]. Depicted is TVAE's restoration for 50% missing pixels (sparsity $\frac{\sum_h \pi_h}{H} = \frac{10.56}{512}$).

In contrast, amortized approaches need to specify how the deterministic encoder DNNs should treat missing values. Figure 3 evaluates performance of TVAE on two standard inpainting benchmarks with randomly missing pixels. Methods compared to include MTMKL, BPFA [69], ES3C, the method of Papyan et al. [46], DIP [64], PLE [66], and IRCNN [6]. PLE uses the noise level as a-priori information, and IRCNN is trained on external clean images. On House, TVAE improves the performance of Papyan et al. and BPFA; highest PSNRs for this benchmark are obtained by DIP (which, in contrast to TVAE, is not permutation invariant and uses large U-nets) and ES3C (which is based on a SSSC model and EVO-based training). On Castle, PSNRs of TVAE are higher in comparison to SSSC-based BPFA (for 50% missing pixels) and IRCNN.

4 Discussion

We investigated a novel approach built upon Evolutionary Variational Optimization [14] to train VAEs with binary latents. Compared to all previous optimizations suggested for VAEs with discrete latents, the approach followed here differs the most substantially from conventional VAE training. While all other VAEs maintain amortization and reparameterization as key elements, the TVAE approach instead uses a direct and non-amortized optimization. Recent work using elementary generative models such as mixtures and shallow models [17,14] have made considerations of direct VAE optimization possible for intermediately large scales. A conceptual advantage of the here developed approach is its concise formulation (compare Fig. S2) with fewer algorithmic elements, fewer hyperparameters and fewer model parameters (e.g., no parameters of encoder DNNs). Functional advantages of the approach are its avoidance of an amortization gap (e.g., [30,9]), its ability to learn sparse codes, and its generality (it does not use a specific posterior model, and can be applied to other noise models, for instance). However, non-amortized approaches do in general have the disadvantage of a lower computational efficiency: an optimization of variational parameters for each data point is more costly (Tab. S3). Conventional amortized approaches (for discrete or continuous VAEs) are consequently preferable for large-scale data sets and for the optimization of large, intricate DNNs. There are, however, alternatives such as transformers (which can use >150M parameters) or diffusion nets, which both are considered to perform more strongly than VAEs for large-scale settings and density modeling ([7,31] for recent comparisons).

At the same time, direct discrete optimization can be feasible and can be advantageous. For image patch data, for instance, we showed that TVAEs with intermediately large decoder DNNs perform more strongly than Gumbel-softmax VAEs (GSVAE), and TVAEs are also outperforming a recent continuous VAE baseline (VLAE; Figs. 1 and 2). The stronger performance of TVAE is presumably, at least in part, due to the approach not being subject to an amortization gap, due to it avoiding factored variational distributions, and, more generally, due to the emerging sparse codes being well suited for modeling image patch data. In comparison, the additional methods to treat discrete latents in GSVAE seem to result in dense codes with significantly lower performance than TVAE. Compared to GSVAE, the VLAE approach, which uses standard non-sparse (i.e. Gaussian) latents, is more competitive on the benchmarks we considered. The reason is presumably that VLAE's continuous latents are able to better capture component intensities in image patches. This advantage does not outweigh the advantages of sparse codes learned by TVAE, however. If sparse codes and continuous latents are combined, the example of ES3C shows that strong performances can be obtained (Figs. 1 to 3). For the here considered binary latents, however, a linear decoder (compare EBSC) is much inferior to a deep decoder (Figs. 1, 2 and S9), which suggests future work on VAEs with more complex, sparse priors if the goal is to improve 'zero-shot' denoising and inpainting. Dense codes are notably not necessarily disadvantageous for image data. On the contrary, for datasets with many images of single objects like CIFAR, the dense codes of GSVAE and also of VLAE are, in terms of ELBO values, similar or better compared to TVAE (Tab. S4). The suitability of sparse versus dense encoding consequently seems to highly depend on the data, and here we confirm the suitability of sparse codes for image patches. In addition to learning sparse codes, direct optimization can have further advantages compared to conventional training. One such advantage is highlighted by the inpainting task: in contrast to other (continuous or discrete) VAEs, it is not required to additionally specify how missing data shall be treated by an encoder DNN (compare Sec. S2.5)

We conclude that direct discrete optimization can, depending on the data and task, serve as an alternative for training discrete VAEs. In a sense, the approach can be considered more brute-force than conventional amortized training: direct optimization is slower but at scales at which it can be applied, it is more effective. To our knowledge, the approach is also the first training method for discrete VAEs not using gradient optimization of encoder models, and can thus contribute to our understanding of how good representations can be learned by different approaches.

Acknowledgments. We acknowledge funding by the German Research Foundation (DFG) under grant SFB 1330/1&2 (B2), ID 352015383 (JD), and the H4a cluster of excellence EXC 2177/1, ID 390895286 (JL), and by the German Federal Ministry of Education and Research (BMBF) through a Wolfgang Gentner scholarship (awarded to EG, ID 13E18CHA). Furthermore, computing infrastructure support is acknowledged through the HPC Cluster CARL at UOL (DFG, grant INST 184/157-1 FUGG) and the HLRN Alliance, grant ID nim00006.

References

1. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy Layer-Wise Training of Deep Networks. In: NeurIPS (2007)
2. Bengio, Y., Léonard, N., Courville, A.: Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. arXiv:1308.3432 (2013)
3. Berliner, A., Rotman, G., Adi, Y., Reichart, R., Hazan, T.: Learning Discrete Structured Variational Auto-Encoder using Natural Evolution Strategies. In: ICLR (2022)
4. Bingham, E., Chen, J.P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P., Horsfall, P., Goodman, N.D.: Pyro: Deep Universal Probabilistic Programming. JMLR **20**(1), 973–978 (2019)
5. Bowman, S., Vilnis, L., Vinyals, O., Dai, A.M., Jozefowicz, R., Bengio, S.: Generating Sentences from a Continuous Space. In: CoNLL (2016)
6. Chaudhury, S., Roy, H.: Can fully convolutional networks perform well for general image restoration problems? In: IAPR Int. Conf. MVA (2017)
7. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating Long Sequences with Sparse Transformers. arXiv:1904.10509 (2019)
8. Cong, Y., Zhao, M., Bai, K., Carin, L.: GO Gradient for Expectation-Based Objectives. In: ICLR (2019)
9. Cremer, C., Li, X., Duvenaud, D.: Inference Suboptimality in Variational Autoencoders. In: ICML (2018)
10. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image Denoising by Sparse 3D Transform-Domain Collaborative Filtering. IEEE Trans. Image Proc. **16**(8), 2080–2095 (2007)
11. Dimitriev, A., Zhou, M.: CARMS: Categorical-Antithetic-REINFORCE Multi-Sample Gradient Estimator. NeurIPS (2021)
12. Dong, W., Wang, P., Yin, W., Shi, G., Wu, F., Lu, X.: Denoising Prior Driven Deep Neural Network for Image Restoration. TPAMI **41**(10), 2305–2318 (2019)
13. Dong, Z., Mnih, A., Tucker, G.: Coupled Gradient Estimators for Discrete Latent Variables. In: NeurIPS (2021)
14. Drefs, J., Guiraud, E., Lücke, J.: Evolutionary Variational Optimization of Generative Models. JMLR **23**(21), 1–51 (2022)
15. Elad, M., Aharon, M.: Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries. IEEE Trans. Image Proc. **15**, 3736–3745 (2006)
16. Eldar, Y.C., Kutyniok, G.: Compressed Sensing: Theory and Applications. Cambridge University Press (2012)
17. Exarchakis, G., Oubari, O., Lenz, G.: A sampling-based approach for efficient clustering in large datasets. In: CVPR (2022)
18. Fajtl, J., Argyriou, V., Monekosso, D., Remagnino, P.: Latent Bernoulli Autoencoder. In: ICML (2020)
19. Foerster, J., Farquhar, G., Al-Shedivat, M., Rocktäschel, T., Xing, E., Whiteson, S.: DiCE: The Infinitely Differentiable Monte Carlo Estimator. In: ICML (2018)
20. Goodfellow, I.J., Courville, A., Bengio, Y.: Scaling Up Spike-and-Slab Models for Unsupervised Feature Learning. TPAMI **35**(8), 1902–1914 (2013)

21. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. In: NeurIPS (2014)
22. Grathwohl, W., Choi, D., Wu, Y., Roeder, G., Duvenaud, D.: Backpropagation through the Void: Optimizing control variates for black-box gradient estimation. In: ICLR (2018)
23. Gu, S., Zhang, L., Zuo, W., Feng, X.: Weighted Nuclear Norm Minimization with Application to Image Denoising. In: CVPR (2014)
24. Hajewski, J., Oliveira, S.: An Evolutionary Approach to Variational Autoencoders. In: CCWC (2020)
25. van Hateren, J.H., van der Schaaf, A.: Independent component filters of natural images compared with simple cells in primary visual cortex. Proc. Roy. Soc. London. Series B **265**, 359–66 (1998)
26. Hirschberger, F., Forster, D., Lücke, J.: A Variational EM Acceleration for Efficient Clustering at Very Large Scales. TPAMI (2022)
27. Hughes, M.C., Sudderth, E.B.: Fast Learning of Clusters and Topics via Sparse Posteriors. arXiv:1609.07521 (2016)
28. Imamura, R., Itasaka, T., Okuda, M.: Zero-Shot Hyperspectral Image Denoising With Separable Image Prior. In: ICCV Workshops (2019)
29. Jang, E., Gu, S., Poole, B.: Categorical Reparameterization with Gumbel-Softmax. In: ICLR (2017)
30. Kim, Y., Wiseman, S., Miller, A., Sontag, D., Rush, A.: Semi-Amortized Variational Autoencoders. In: ICML (2018)
31. Kingma, D.P., Salimans, T., Poole, B., Ho, J.: Variational diffusion models. In: NeurIPS (2021)
32. Kingma, D.P., Welling, M.: Auto-Encoding Variational Bayes. In: ICLR (2014)
33. Kiran, B., Thomas, D., Parakkal, R.: An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. Journal of Imaging **4**(2), 36 (2018)
34. Kool, W., van Hoof, H., Welling, M.: Estimating Gradients for Discrete Random Variables by Sampling without Replacement. In: ICLR (2020)
35. van Krieken, E., Tomczak, J.M., Teije, A.T.: Storchastic: A Framework for General Stochastic Automatic Differentiation. In: NeurIPS (2021)
36. Krull, A., Buchholz, T.O., Jug, F.: Noise2Void - Learning Denoising from Single Noisy Images. In: CVPR (2019)
37. Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., Aila, T.: Noise2Noise: Learning Image Restoration without Clean Data. In: ICML (2018)
38. Liu, R., Regier, J., Tripuraneni, N., Jordan, M., Mcauliffe, J.: Rao-Blackwellized Stochastic Gradients for Discrete Distributions. In: ICML (2019)
39. Lorberbom, G., Gane, A., Jaakkola, T., Hazan, T.: Direct Optimization through arg max for Discrete Variational Auto-Encoder. In: NeurIPS (2019)
40. Maaløe, L., Sønderby, C.K., Sønderby, S.K., Winther, O.: Auxiliary Deep Generative Models. In: ICML (2016)
41. Maddison, C.J., Mnih, A., Teh, Y.W.: The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In: ICLR (2017)
42. Mairal, J., Elad, M., Sapiro, G.: Sparse Representation for Color Image Restoration. IEEE Trans. Image Proc. **17**(1), 53–69 (2008)
43. Olshausen, B., Field, D.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature **381**, 607–9 (1996)
44. van den Oord, A., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. In: NeurIPS (2017)

45. Paiton, D.M., Frye, C.G., Lundquist, S.Y., Bowen, J.D., Zarcone, R., Olshausen, B.A.: Selectivity and robustness of sparse coding networks. Journal of Vision **20**(12), 10–10 (2020)
46. Papyan, V., Romano, Y., Sulam, J., Elad, M.: Convolutional Dictionary Learning via Local Processing. In: ICCV (2017)
47. Park, Y., Kim, C., Kim, G.: Variational Laplace Autoencoders. In: ICML (2019)
48. Paulus, M.B., Maddison, C.J., Krause, A.: Rao-Blackwellizing the Straight-Through Gumbel-Softmax Gradient Estimator. In: ICLR (2021)
49. Potapczynski, A., Loaiza-Ganem, G., Cunningham, J.P.: Invertible Gaussian Reparameterization: Revisiting the Gumbel-Softmax. In: NeurIPS (2020)
50. Quan, Y., Chen, M., Pang, T., Ji, H.: Self2Self With Dropout: Learning Self-Supervised Denoising From Single Image. In: CVPR (2020)
51. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In: ICML (2014)
52. Roberts, A., Engel, J., Raffel, C., Hawthorne, C., Eck, D.: A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music. In: ICML (2018)
53. Rolfe, J.T.: Discrete Variational Autoencoders. In: ICLR (2017)
54. Schulman, J., Heess, N., Weber, T., Abbeel, P.: Gradient Estimation Using Stochastic Computation Graphs. In: NeurIPS (2015)
55. Sheikh, A.S., Lücke, J.: Select-and-Sample for Spike-and-Slab Sparse Coding. In: NeurIPS (2016)
56. Sheikh, A.S., Shelton, J.A., Lücke, J.: A Truncated EM Approach for Spike-and-Slab Sparse Coding. JMLR **15**, 2653–2687 (2014)
57. Shelton, J., Bornschein, J., Sheikh, A.S., Berkes, P., Lücke, J.: Select and Sample - A Model of Efficient Neural Inference and Learning. NeurIPS (2011)
58. Shelton, J.A., Gasthaus, J., Dai, Z., Lücke, J., Gretton, A.: GP-Select: Accelerating EM Using Adaptive Subspace Preselection. Neur. Comp. **29**(8), 2177–2202 (2017)
59. Shocher, A., Cohen, N., Irani, M.: "Zero-Shot" Super-Resolution using Deep Internal Learning. In: CVPR (2018)
60. Sulam, J., Muthukumar, R., Arora, R.: Adversarial Robustness of Supervised Sparse Coding. NeurIPS (2020)
61. Titsias, M.K., Lázaro-Gredilla, M.: Spike and Slab Variational Inference for Multi-Task and Multiple Kernel Learning. In: NeurIPS (2011)
62. Tomczak, J.M., Welling, M.: VAE with a VampPrior. In: AISTATS (2018)
63. Tonolini, F., Jensen, B.S., Murray-Smith, R.: Variational Sparse Coding. In: UAI (2020)
64. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Deep Image Prior. In: CVPR (2018)
65. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning **8**(3), 229–256 (1992)
66. Yu, G., Sapiro, G., Mallat, S.: Solving Inverse Problems With Piecewise Linear Estimators: From Gaussian Mixture Models to Structured Sparsity. IEEE Trans. Image Proc. **21**(5), 2481–2499 (2012)
67. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. IEEE Trans. Image Proc. **26**(7), 3142–3155 (2017)
68. Zhou, M., Chen, H., Paisley, J., Ren, L., Sapiro, G., Carin, L.: Non-Parametric Bayesian Dictionary Learning for Sparse Image Representations. In: NeurIPS (2009)
69. Zhou, M., Chen, H., Paisley, J., Ren, L., Li, L., Xing, Z., Dunson, D., Sapiro, G., Carin, L.: Nonparametric Bayesian Dictionary Learning for Analysis of Noisy and Incomplete Images. IEEE Trans. Image Proc. **21**(1), 130–144 (2012)

70. Zhu, S., Xu, G., Cheng, Y., Han, X., Wang, Z.: BDGAN: Image Blind Denoising Using Generative Adversarial Networks. In: PRCV (2019)
71. Zoran, D., Weiss, Y.: From Learning Models of Natural Image Patches to Whole Image Restoration. In: ICCV (2011)