DEEP ATTENTIVE VARIATIONAL INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Stochastic Variational Inference is a powerful framework for learning large-scale probabilistic latent variable models. However, typical assumptions on the factorization or independence of the latent variables can substantially restrict its capacity for inference and generative modeling. A major line of active research aims at building more expressive variational models by designing deep hierarchies of interdependent latent variables. Although these models exhibit superior performance and enable richer latent representations, we show that they incur diminishing returns: adding more stochastic layers to an already very deep model yields small predictive improvement while substantially increasing the inference and training time. Moreover, the architecture for this class of models favors local interactions among the latent variables between neighboring layers when designing the conditioning factors of the involved distributions. This is the first work that proposes attention mechanisms to build more expressive variational distributions in deep probabilistic models by explicitly modeling both local and global interactions in the latent space. Specifically, we propose deep attentive variational autoencoder and test it on a variety of established datasets. We show it achieves state-of-the-art log-likelihoods while using fewer latent layers and requiring less training time than existing models. The proposed non-local inference reduces computational footprint by alleviating the need for deep hierarchies.

1 INTRODUCTION

A core line of research in both supervised and unsupervised learning relies on deep probabilistic models. This class of models uses deep neural networks to model distributions that express hypotheses about the way in which the data have been generated. Such architectures are preferred due to their capacity to express complex, non-linear relationships between the random variables of interest while enabling tractable inference and sampling. Latent variable probabilistic models augment the set of the observed variables with auxiliary latent variables. They are characterized by a posterior distribution over the latent variables, one which is generally intractable and typically approximated by closed-form alternatives. They provide an explicit parametric specification of the joint distributions over the expanded random variable space, while the distribution of the observed variables is computed by marginalizing over the latent variables. The Variational Autoencoder (VAE) (Kingma & Welling, 2014) belongs to this model category. The VAE uses neural networks for learning the parametrization of both the inference network (which defines the posterior distribution of the latent variables) and the generative network (which defines the prior distribution of the latent variables and the conditional data distribution given the latent variables). Their parameters are jointly learned via stochastic variational inference (Paisley et al., 2012; Hoffman et al., 2013).

Early VAE architectures (Rezende et al., 2014) make strong assumptions about the posterior distribution—specifically, it is standard to assume that the posterior is approximately factorial. Since then, research has progressed on learning more expressive latent variable models. For example, Rezende & Mohamed (2015); Kingma et al. (2016); Chen et al. (2017) aim at modeling more complex posterior distributions, constructed with autoregressive layers. Theoretical research focuses on deriving tighter bounds (Burda et al., 2016; Li & Turner, 2016; Masrani et al., 2019) or building more informative latent variables by mitigating posterior collapse (Razavi et al., 2019a; Lucas et al., 2019). Sinha & Dieng (2021) improve generalization by enforcing regularization in the latent space for semantics-preserving transformations of the data. Taking a different approach, hierarchical VAEs (Gulrajani et al., 2017; Sønderby et al., 2016; Maaløe et al., 2019; Vahdat & Kautz, 2020; Child,

2020) leverage increasingly deep and interdependent layers of latent variables, similar to how subsequent layers in a discriminative network are believed to learn increasingly abstract representations of the data. These architectures exhibit superior generative and reconstructive capabilities since they allow for modeling of much richer structures in the latent space.

Previous work overlooks the effect of long-range correlations among the latent variables. In this work, we propose to restructure common hierarchical VAE architectures in order to increase the receptive field of the variational distributions. We first provide experimental evidence that conditional dependencies in deep probabilistic hierarchies may be implicitly disregarded by current models. Subsequently, we propose a decomposed, attention-guided scheme that allows a long-range flow of both the latent and the observed information both across different, potentially far apart, stochastic layers and within the same layer and we investigate the importance of each proposed change through extensive ablation studies. Finally, we demonstrate that our model is both computationally more economical and can attain state-of-the-art performance across a diverse set of benchmark datasets.

2 Proposed Method

2.1 Deep Variational Inference

A latent variable generative model defines the joint distribution of a set of observed variables, $\boldsymbol{x} \in \mathbb{R}^D$, and auxiliary latent variables, $\boldsymbol{z}$, coming from a prior distribution $p(\boldsymbol{z})$. To perform inference, the marginal likelihood of the distribution of interest, $p(\boldsymbol{x})$, can be computed by integrating out the latent variables:

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{z})\, d\boldsymbol{z}. \tag{1}$$

Since this integration is generally intractable, a lower bound on the marginal likelihood is maximized instead. This is done by introducing an approximate posterior distribution $q(\boldsymbol{z} \mid \boldsymbol{x})$ and applying Jensen's inequality:

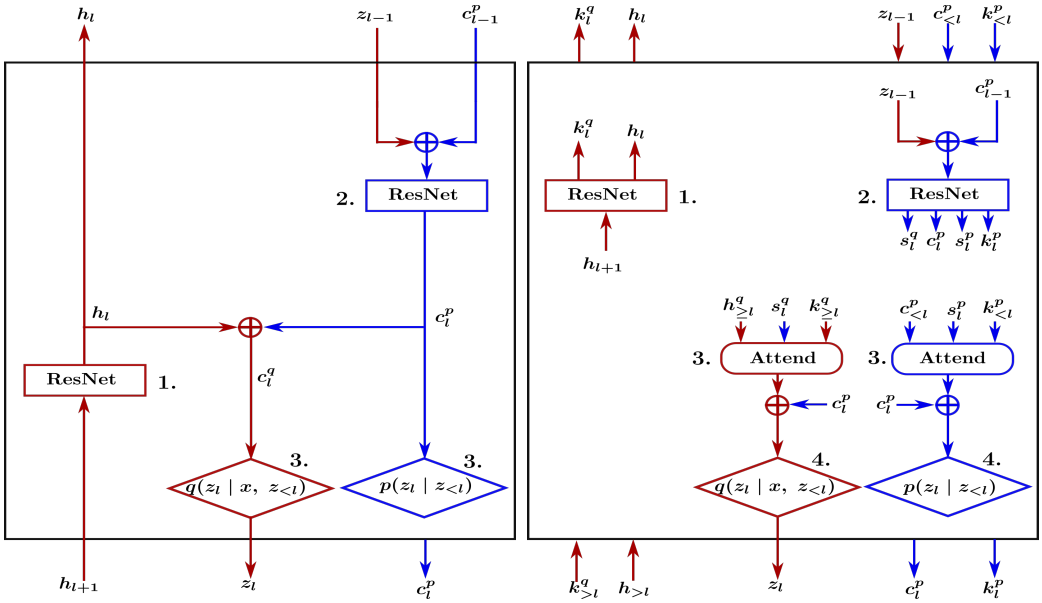
$$\begin{aligned} &\log p(\boldsymbol{x}) = \log \int p(\boldsymbol{x}, \boldsymbol{z})\, d\boldsymbol{z} = \log \int \frac{q(\boldsymbol{z} \mid \boldsymbol{x})}{q(\boldsymbol{z} \mid \boldsymbol{x})} p(\boldsymbol{x}, \boldsymbol{z})\, d\boldsymbol{z} \geq \int q(\boldsymbol{z} \mid \boldsymbol{x}) \log \left[\frac{p(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z})}{q(\boldsymbol{z} \mid \boldsymbol{x})} \right] d\boldsymbol{z} \\ &\implies \log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x} \mid \boldsymbol{z})] - D_{KL}(q(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z})), \end{aligned} \tag{2}$$

where $\boldsymbol{\theta}$, $\boldsymbol{\phi}$ parameterize $p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta})$ and $q(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\phi})$ respectively. For ease of notation, we omit $\boldsymbol{\theta}, \boldsymbol{\phi}$ in the derivations. This objective is called the Evidence Lower BOund (ELBO) and can be optimized efficiently for continuous $\boldsymbol{z}$ via stochastic gradient descent (Kingma & Welling, 2014; Rezende et al., 2014).

For ease of implementation, it is common to assume that both $q(\boldsymbol{z} \mid \boldsymbol{z})$ and $p(\boldsymbol{z})$ are fully factorized Gaussian distributions. However, this assumption may be too limiting in cases of complex underlying distributions. To compensate for this modeling constraint, many works focus on stacking and improving the stability of multiple layers of stochastic latent features which are partitioned in groups such that $\boldsymbol{z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_L\}$, where L is the number of such groups (Rezende et al., 2014; Gulrajani et al., 2017; Kingma et al., 2016; Sønderby et al., 2016; Maaløe et al., 2019; Vahdat & Kautz, 2020; Child, 2020). Our work builds on architectures of bidirectional inference with a deterministic bottom-up pass. The schematic diagram of a stochastic layer in such a deep variational model is depicted in Figure 1. In a bidirectional inference architecture with a deterministic bottom-up pass (left part in Figure 1a), posterior sampling is preceded by a sequence of non-linear transformations, T_l^q, of the evidence, $\boldsymbol{x}$, i.e., $\boldsymbol{h}_l = T_l^q(\boldsymbol{h}_{l+1})$, with $\boldsymbol{h}_{L+1} = \boldsymbol{x}$. The inference $q(\boldsymbol{z} \mid \boldsymbol{x})$ and generative $p(\boldsymbol{z})$ network decompose in an identical topological ordering: $q(\boldsymbol{z} \mid \boldsymbol{x}) = \prod_l q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l})$ and $p(\boldsymbol{z}) = \prod_l p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$. The top-down pass (right part in Figure 1a) generates the posterior samples $\boldsymbol{z}$ that feed the conditional data distribution $p(\boldsymbol{x} \mid \boldsymbol{z})$.

2.2 Motivation

Kingma et al. (2016) proposed a strongly connected directed probabilistic graphical model for the generative and inference network, so that each variable depends on all the previous in the hierarchy:



(a) Locally connected variational layer. (b) Strongly connected variational layer **(Ours)**.

Figure 1: **Computational graph of the l-th variational layer in a hierarchy.** The numbers denote the order of the computations. The inference path forms the posterior $q(\boldsymbol{z}_l \mid \boldsymbol{x}, \boldsymbol{z}_{<l})$ given the condition $\boldsymbol{c}_l^q$. The generative path forms the prior $p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$ given the condition $\boldsymbol{c}_l^p$. $\boldsymbol{c}_l^p$ represents context information that consists of both deterministic features and latent samples with $\boldsymbol{c}_{-1}^p$ being a constant. $\boldsymbol{h}_l$ represents hidden features of the data with $\boldsymbol{h}_{L+1} \equiv \boldsymbol{x}$. Multiple such blocks are stacked. $p(\boldsymbol{x} \mid \boldsymbol{z})$ receives the sample $\boldsymbol{z}_L$ and the context $\boldsymbol{c}_L^p$ of the last layer. The $\oplus$ symbol denotes a module responsible for combining two streams of features. In Figure 1a, the layer is connected only with the adjacent layers in the hierarchy. Latent information of earlier layers $\boldsymbol{z}_{<l-1}$ is carried through $\boldsymbol{c}_{l-1}^p$. In Figure 1b, the layer is connected with all the layers below (above) at the bottom-up pass (top-down pass). The ResNet transformations are extended to produce key $\boldsymbol{k}$ and query $\boldsymbol{s}$ feature maps. The generative network queries the layers of the generative network (inference network) above (below) to identify the most relevant conditioning factors of the prior (posterior) according to attention scores computed by the attention modules (see Figure 2).

$p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$. Similarly, for the inference model: $q(\boldsymbol{z}_l \mid \boldsymbol{x}, \boldsymbol{z}_{<l})$. This is in contrast to other works (Sønderby et al., 2016) that consider statistical dependencies between successive layers only, i.e., $p(\boldsymbol{z}_l \mid \boldsymbol{z}_{l-1})$. Maaløe et al. (2019) also highlight the importance of this modification. The long-range conditional dependencies are implicitly enforced via deterministic features that are mixed with the latent variables and propagated through the hierarchy (see feature $\boldsymbol{c}_l^p$ in Figure 1a).

State-of-the-art models (Vahdat & Kautz, 2020; Child, 2020) leverage this fully-connected factorization and rely on the increased depth to improve performance and deliver results comparable to that of autoregressive models (Salimans et al., 2017). Table 1 reports the absolute and relative decrease in the negative log-likelihood (in bits per dimension) as one increases the number of stochastic layers in an NVAE (Vahdat & Kautz, 2020). We observe that the predictive gains diminish as depth increases. We hypothesize that this may be because the effect of the latent variables of earlier layers diminishes as the context feature $\boldsymbol{c}_l^p$ traverses the hierarchy and is updated with latent information from subsequent layers.

Table 1: $-\log p(\boldsymbol{x})$ **for varying depth L (bits/dim)**.

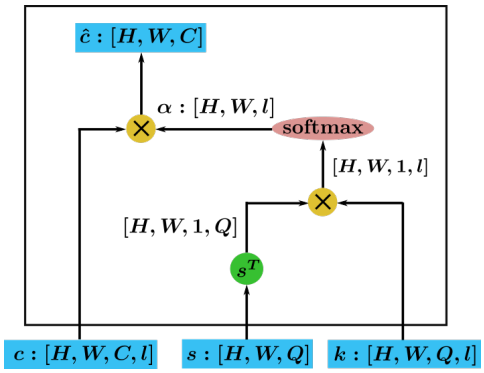
Depth (L)	bits/dim ↓	$\Delta(\cdot)\%$
2	3.50	–
4	3.26	−6.8
8	3.06	−6.1
16	2.96	−3.2
30	2.91	−1.7

In this work, we improve the flexibility of the prior $p(\boldsymbol{z})$ and posterior $q(\boldsymbol{z} \mid \boldsymbol{x})$ by designing more informative conditioning factors of the conditional distributions $p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$ and $q(\boldsymbol{z}_l \mid \boldsymbol{x}, \boldsymbol{z}_{<l})$. We do this by designing a hierarchy of densely connected stochastic layers that learn to attend to the latent and observed information that is most critical for inference. Figure 1b is a graphical illustration of the proposed model. Inference on the model is also summarized in Appendix A.3.

2.3 GENERATIVE MODEL

As shown in Figure 1, the conditioning factor of the prior is a 3D context vector $\boldsymbol{c}_l^p \in \mathbb{R}^{H \times W \times C}$. $\boldsymbol{c}_l^p$ is the output of a mapping T_l^p, typically implemented as a cascade of ResNet cells (He et al., 2016a;b), such that $\boldsymbol{c}_l^p = T_l^p(\boldsymbol{z}_{l-1} \oplus \boldsymbol{c}_{l-1}^p)$. T_l^p corresponds to the blue ResNet module in Figures 1a, 1b. The operator $\oplus$ combines information from two branches in the network (e.g. by summation or concatenation). $\boldsymbol{z}_{l-1}$, $\boldsymbol{c}_{l-1}^p$ are passed in from the previous layer. $\boldsymbol{c}_{l-1}^p$ carries the latent information from previous layers $\boldsymbol{z}_{<l-1}$. However, by construction, very deep VAE architectures favor only local dependencies in the latent space, attenuating long-range conditional dependencies between $\boldsymbol{z}_l$ and $\boldsymbol{z}_{<l-1}$. This means that in practice the network may no longer respect the factorization of the prior $p(\boldsymbol{z}) = \prod_l p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$ leading to diminished performance (see Table 1).

Figure 2: **Attend$(\boldsymbol{c}, \boldsymbol{s}, \boldsymbol{k})$– a depth-wise attention block.** $\boldsymbol{c}$, $\boldsymbol{k}$ are the sequences of l contexts and corresponding keys with C and Q feature maps accordingly, while $\boldsymbol{s}$ is the query vector. The multiplication is applied to the inner matrix dimensions. The normalization of the softmax is applied to the last dimension, treating each pixel independently from the others.



To circumvent this issue, we adopt direct connections between each pair of stochastic layers, as shown in Figure 1b. That is, a variational layer l receives the context of all previous layers $\boldsymbol{c}_{l'}^p$, with $l' \in \{1, \ldots, l-1\}$ along with a key feature $\boldsymbol{k}_{l'}^p \in \mathbb{R}^{H \times W \times Q}$, where each $\boldsymbol{k}_{l'}^p$ can be perceived as a hidden representation of $\boldsymbol{c}_{l'}^p$ of smaller dimension Q, such that $Q \ll C$. Each layer selectively attends to the context instances according to a query feature $\boldsymbol{s}_l^p \in \mathbb{R}^{H \times W \times Q}$. In our architecture, $\boldsymbol{c}_l^p$, $\boldsymbol{k}_l^p$, and $\boldsymbol{s}_l^p$ are jointly learned:

$$[\boldsymbol{c}_l^p,\ \boldsymbol{k}_l^p,\ \boldsymbol{s}_l^p] \leftarrow T_l^p(\boldsymbol{z}_{l-1} \oplus \boldsymbol{c}_{l-1}^p). \tag{3}$$

To mitigate the computational requirements, we treat each pixel independently from the rest. The resulting feature map of a pixel (i, j), with $1 \leq i \leq H$, $1 \leq j \leq W$, $\hat{\boldsymbol{c}}_l^p(i,j) \in \mathbb{R}^C$ is:

$$\hat{\boldsymbol{c}}_l^p(i,j) = \sum_{l' < l} \boldsymbol{\alpha}_{l',l}^p(i,j) \boldsymbol{c}_{l'}^p(i,j), \tag{4}$$

$$\boldsymbol{\alpha}_{l',l}^p(i,j) = \frac{\exp(\boldsymbol{k}_{l'}^p(i,j)^T \boldsymbol{s}_l^p(i,j))}{\sum_{l'<l} \exp(\boldsymbol{k}_{l'}^p(i,j)^T \boldsymbol{s}_l^p(i,j))}. \tag{5}$$

In words, the generative model queries through the feature $\boldsymbol{s}_l^p(i,j) \in \mathbb{R}^Q$ the relevance of the feature $\boldsymbol{c}_{l'}^p(i,j) \in \mathbb{R}^C$, that is represented by $\boldsymbol{k}_{l'}^p(i,j) \in \mathbb{R}^Q$ to build the final context of the prior. $\boldsymbol{\alpha}_{l',l}^p(i,j) \in \mathbb{R}$ is the resulting metric of relevance. Figure 2 demonstrates the procedure described above.

To ease training of architectures consisting of many stochastic layers, we adopt the following variant of the normalization scheme proposed by Chen et al. (2020):

$$\boldsymbol{c}_{<l}^p \leftarrow \boldsymbol{c}_{<l}^p + \mathrm{Gelu}(\mathrm{LayerNorm}(\boldsymbol{c}_{<l}^p)), \tag{6}$$

$$\hat{\boldsymbol{c}}_l^p \leftarrow \hat{\boldsymbol{c}}_l^p + \mathrm{Gelu}(\mathrm{LayerNorm}(\hat{\boldsymbol{c}}_l^p)), \tag{7}$$

where Gelu is the GELU non-linearity (Hendrycks & Gimpel, 2016) and $\mathrm{LayerNorm}$ a layer normalization operation (Ba et al., 2016).

To facilitate training further, we initially let the variational layer rely on local dependencies, captured by $\boldsymbol{c}_l^p$, by wrapping the above operation in a residual connection, such that the final prior condition is given by:

$$\boldsymbol{c}_l^p \leftarrow \boldsymbol{c}_l^p + \gamma_l^p \hat{\boldsymbol{c}}_l^p, \tag{8}$$

where γ_l^p is a learnable scalar parameter initialized to zero. Finally, we assume a Gaussion prior distribution that is given by:

$$p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{c}_l^p), \boldsymbol{\sigma}(\boldsymbol{c}_l^p)). \tag{9}$$

$\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are transformations implemented as convolutions.

2.4 INFERENCE MODEL

In this section, we extend the mechanism presented in section 2.3 to refine the posterior distribution $q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l})$. We adopt a Gaussian residual parametrization of the posterior (Vahdat & Kautz, 2020) (see also Equation 21 in the Appendix). As shown in Figure 1a, the condition $\boldsymbol{c}_l^q$ of q results from combining the observed representation $\boldsymbol{h}_l$ and the stochastic factor provided by the generative network $\boldsymbol{c}_l^p$: $\boldsymbol{c}_l^q = \boldsymbol{h}_l \oplus \boldsymbol{c}_l^p$. Here, we focus on enhancing the representation of the observation $\boldsymbol{x}$.

Towards this end, we allow the inference model of layer l to have access to all subsequent hidden data representations $\boldsymbol{h}_{\geq l}$. The bottom-up, red path of the inference model in Figure 1b of the inference model in Figure 1b is queried by the top-down, blue path of the generative model so that it decides the hidden data representations $\boldsymbol{h}_{\geq l}$ that are most important to the inference according to the collected latent information contained in $\boldsymbol{c}_l^p$. Like the generative model, the bottom-up path is extended in order to emit key features $\boldsymbol{k}_l^q$ of the hidden data features $\boldsymbol{h}_l$ such that:

$$[\boldsymbol{h}_l,\ \boldsymbol{k}_l^q] \leftarrow T_l^q(\boldsymbol{h}_{l+1} \oplus \boldsymbol{k}_{l+1}^q). \tag{10}$$

A slice of $\boldsymbol{c}_l^p$ serves as the query feature $\boldsymbol{s}_l^q$ of the generative model. Equations 4, 5, are symmetrical replacing $\boldsymbol{c}_{l'}^p$ with $\boldsymbol{h}_{l'}$, $\boldsymbol{s}_l^p$ with $\boldsymbol{s}_l^q$, $\boldsymbol{k}_{l'}^p$ with $\boldsymbol{k}_{l'}^q$, $\hat{\boldsymbol{c}}_l^p$ with $\hat{\boldsymbol{c}}_l^q$, $\boldsymbol{\alpha}_{l',l}^p$ with $\boldsymbol{\alpha}_{l',l}^q$ and the ordering in the summation $l' < l$ with $l' \geq l$. Likewise, in Equations 8, 6, 7 the p superscript is replaced with q. The complete description is delegated to Appendix A.1 and Algorithm 3 in Appendix A.3.

The above procedure can also be viewed as statistically driven skip connections that selectively connect each layer with representations closer to the data, hence facilitating the flow of the gradients during training. More importantly, it establishes further communication between the inference and generative model, as is motivated by Sønderby et al. (2016). This interaction conduces to a better coordination between the prior and the posterior. This fact, in turn, mitigates the KL loss in Equation 2, which is prone to explode, and therefore promotes stable optimization. T

2.5 SPATIALLY NON-LOCAL VARIATIONAL LAYERS

In sections 2.3 and 2.4, we allowed the perceptive field of each pixel at layer l to encompass only pixels at the same locations in earlier layers during inference, thereby treating $H \times W$ sequences of length $l-1$ independently. In order to fully capture distant dependencies in the latent space, we still need to coordinate portions of information in the spatial domain. To increase the computational efficiency of our model, this occurs as a second stage operation. Conceptually, this two-step procedure resembles the factorized attention of efficient transformers (Child et al., 2019), albeit deep variational models are inherently amenable to a different, inter-layer and intra-layer, decomposition. This observation decreases the time and memory complexity of the non-local operations for computing the attention scores from $\mathcal{O}(L^3 \times W^2 \times H^2)$ to $\mathcal{O}(L^2 + L \times W^2 \times H^2)$, while still providing inference with global information.

To accomplish this, we interleave spatial self-attention blocks (Wang et al., 2018) in the residual cells of the ResNet modules in Figure 1, that are scaled and added back to the input feature, as shown in Figure 3. Such blocks treat a context feature $\boldsymbol{c} \in \mathbb{R}^{H \times W \times C}$ as a sequence of length $H \times W$ of $C-$dimensional features. Each $\boldsymbol{c}(i) \in \mathbb{R}^C$ is mapped to feature $\boldsymbol{v}(i) \in \mathbb{R}^C$ as a normalized combination of all the features in the sequence such that: $\boldsymbol{v}(i) \propto \sum_j f(\boldsymbol{c}(i), \boldsymbol{c}(j)) g(\boldsymbol{c}(j))$, where $f(\boldsymbol{c}(i), \boldsymbol{c}(j)) \in \mathbb{R}$ reflects the importance of $\boldsymbol{c}(j)$ to $\boldsymbol{c}(i)$ when determining $\boldsymbol{v}(i)$. We use the softmax function as an attention distribution. f is a scaled dot-product of linear embeddings of $\boldsymbol{c}(i)$ and $\boldsymbol{c}(j)$, while $g(\boldsymbol{c}(j)) \in \mathbb{R}^C$ is also implemented as a linear transformation. The exact formulation is similar to that of Wang et al. (2018) and is provided in the Appendix A.2.

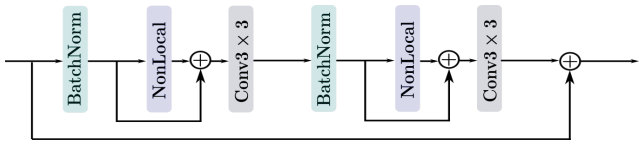


Figure 3: **The spatially non-local ResNet cell.** The non-local operations (Wang et al., 2018) that are interleaved in the convolutions are responsible for capturing inter-pixel (equivalently, intra-layer) long-range interactions in the stochastic features of the hierarchy.

3 EXPERIMENTAL STUDIES

We conduct three series of experiments. In the first set of experiments (Section 3.1.1), we apply the non-local variational path proposed in this paper on VAEs that are trained on two datasets of binary images: the dynamically binarized MNIST and OMNIGLOT. We show that the skip connections introduced by the attention-based mechanisms consistently improve the predictive likelihood compared to state-of-the-art models. Subsequently, in section 3.1.2, we investigate the effectiveness of the proposed techniques on large-scale latent spaces that are used for generating the CIFAR-10 natural images. We verify the significant impact of non-local variational inference that leads to better utilization of the latent space and to shorter hierarchies while outperforming computationally intensive state-of-the-art models. Finally, in section 3.2 we conduct an ablation study and report the benefits of each proposed attention module separately.

All hyperparameters for experiments are available in the Appendix B. We note here that we choose to not use spectral regularization, and we find this does not compromise training stability of our model (in contrast to results reported by Vahdat & Kautz (2020)). The fact that our model is able to achieve better performance without this regularization may indicate that the non-local features help stabilize training.

3.1 MAIN QUANTITATIVE RESULTS

3.1.1 APPLICATION ON BINARY IMAGES

Datasets. We evaluate the models on two benchmark datasets: MNIST (LeCun et al., 1998), a dataset of 28×28 images of handwritten digits, and OMNIGLOT (Lake et al., 2013), an alphabet recognition dataset of 28×28 images. For convenience, we add two zero pixels to each border of the training images. In both cases, the observations are dynamically binarized by being resampled from the normalized real values using a Bernoulli distribution after each epoch, as suggested by Burda et al. (2016), which prevents over-fitting. We use the standard splits of MNIST into 60,000 training and 10,000 test examples, and of OMNIGLOT into 24,345 training and 8,070 test examples.

Set-up. For both datasets, we use a hierarchy of $L = 15$ variational layers. We use a Bernoulli distribution in the image decoder. In the generative model a non-local block is inserted at the end of the module responsible for combining the context with the latent sample provided by the previous layer. In the inference layer, a non-local block is inserted at the end of the module responsible for concatenating the deterministic, data-dependent features of the bottom-up pass and the context containing latent information that is provided by the generative model. Non-local blocks are also inserted at the post-processing cells, at the end of the variational hierarchy and right before the layers that implement the data distribution (see Appendix B for details). MNIST images are downsampled twice before being passed to the hierarchy leading to latent spaces of spatial dimension 8×8 while the OMNIGLOT images are downsampled only once resulting to 16×16 latent spaces.

Results. Table 10 reports the estimated marginal likelihood of our model along with the performance achieved by state-of-the-art models. We observe that in both datasets, non-local variational inference consistently improves performance. More importantly, our model outperforms models that increase the flexibility of the variational posterior by making use of normalizing flows (Kingma et al., 2016) or the data distribution by making use of autoregressive decoders (Chen et al., 2017). The use of autoregressive layers is a model refinement orthogonal to ours and it could potentially further improve the results presented in Table 10. The architecture closest to ours is NVAE (Vahdat & Kautz, 2020), and the performance gap can be solely attributed to the non-local operations described in sections 2.3, 2.4, and 2.5.

3.1.2 APPLICATION ON NATURAL IMAGES

Dataset. CIFAR-10 is a dataset of 32×32 natural images. The raw pixel values are first scaled to $[-1, 1]$.

Set-up. For CIFAR-10, we use a hierarchy of $L = 16$ variational layers. We use a mixture of discretized Logistic distributions (Salimans et al., 2017) for the data distribution. We use the spatially non-local residual cells of Figure 3 in both the generative and inference network of each layer. Non-local blocks are also inserted at the pre-processing blocks, at the beginning of the bottom-up pass

Table 2: **Dynamically binarized MNIST** (Burda et al., 2016) **and OMNIGLOT** (Lake et al., 2013) **performance on the test set.** All models except for IWAE are trained with a single importance sample. IWAE is trained with 50 importance samples. The marginal loglikelihood is estimated with 500 importance samples. Attentive VAE outperforms all state-of-the-art VAEs with or without autoregressive components.

Model	$\log\ p(\boldsymbol{x}) \geq \uparrow$	
	MNIST	OMNIGLOT
Attentive VAE (ours)	$\mathbf{-77.63}$	$\mathbf{-89.50}$
NVAE (Vahdat & Kautz, 2020)	-78.01	-90.18
MAE (Ma et al., 2019)	-77.98	$-$
BIVA (Maaløe et al., 2019)	-78.41	-93.54
PixelVAE++ (Sadeghi et al., 2019)	-78.00	$-$
DVAE++ (Vahdat et al., 2018)	-78.49	$-$
VampPrior (Tomczak & Welling, 2018)	-78.45	-89.76
SA-VAE (Kim et al., 2018)	$-$	-90.05
Lossy VAE (Chen et al., 2017)	-78.53	-89.83
Ladder VAE (Sønderby et al., 2016)	-81.74	-102.11
IAF-VAE (Kingma et al., 2016)	-79.10	$-$
DVAE (Rolfe, 2016)	-81.01	-97.43
Conv DRAW (Gregor et al., 2016)	$-$	-91.00
IWAE (Burda et al., 2016)	-82.90	-103.38

(see Appendix B for details). CIFAR-10 images are downsampled only once resulting to 16×16 latent spaces. We note that it helps optimization if we bound the log of the prior standard deviation such that $\log \sigma_p \geq -1.0$, yielding less confident prior assumptions. We also empirically find that adding Gaussian noise with $\sigma_{noise} = 0.001$ in both the log of the prior scale $\log \sigma_p$ and the posterior scale $\log \sigma_q$ helps network's generalization.

Results. Table 3 shows the performance of the proposed model both among other VAEs and other purely generative models. As we can see, attention-guided variational inference is conducive to building more informative latent spaces, rendering the model the best performing in its class. Moreover, it almost closes the performance gap between VAEs and expensive generative models that rely on fully-autoregressive distributions. Finally, it is possible our results in Table 3 could be improved by considering deeper hierarchies.

Table 3: **CIFAR-10** (Krizhevsky et al., 2009) **performance on the test set.** The marginal loglikelihood is estimated with 100 importance samples. A shallower Attentive VAE outperforms all state-of-the-art VAEs with or without autoregressive components. Attentive VAE performs on par with fully autoregressive generative models. However, it permits fast sampling that requires a single network evaluation per sample as opposed to D, where D the dimension of the data distribution.

Model	VAE	Depth (L)	Autoregressive Decoder	$-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$
Attentive VAE (ours) trained for 400 epochs	✓	16	✗	**2.82**
Attentive VAE (ours) trained for 500 epochs	✓	16	✗	**2.81**
Attentive VAE (ours) trained for 900 epochs	✓	16	✗	**2.79**
Very Deep VAE (Child, 2020)	✓	45	✗	2.87
NVAE (Vahdat & Kautz, 2020)	✓	30	✗	2.91
BIVA (Maaløe et al., 2019)	✓	15	✗	3.08
IAF-VAE (Kingma et al., 2016)	✓	12	✗	3.11
δ-VAE (Razavi et al., 2019a)	✓		✓	2.83
PixelVAE++ (Sadeghi et al., 2019)	✓		✓	2.90
Lossy VAE (Chen et al., 2017)	✓		✓	2.95
MAE (Ma et al., 2019)	✓		✓	2.95
PixelCNN++ (Salimans et al., 2017)	✗		✓	2.92
PixelSNAIL (Chen et al., 2018)	✗		✓	2.85
Image Transformer (Parmar et al., 2018)	✗		✓	2.90
Sparse Transformer (Child et al., 2019)	✗		✓	**2.80**

In Table 4, we compare the computing demands for training our model with the one of the other two most competitive VAEs presented in Table 3. As we see, our 16-layer VAE requires almost $3/5$ of the GPU hours needed for training a 30-layer NVAE, while still outperforming the more efficient VAE of Child (2020). In this work, we implement full rank matrices for the self-attention scores of the non-local residual cells of Figure 3.

Efficient attention approximations such as those proposed by Child et al. (2019) and Choromanski et al. (2020) could further reduce the computational requirements of our models and are left as future work.

Table 4: **Comparison of the computational requirements for training deep state-of-the-art VAE models.** All models are trained on 32GB V100 GPUs. The additional cost for computing the attention scores is compensated by the smaller number of stochastic layers in the hierarchy without sacrificing the generative capacity of the model, see Table 3.

Model	batch size / GPU	# GPUs	Training Time	Total GPU hours
Attentive VAE (ours), 400 epochs	32	4	68 hours	272
Attentive VAE (ours), 500 epochs	32	4	84 hours	336
Attentive VAE (ours), 900 epochs	32	4	152 hours	608
NVAE	32	8	55 hours	440
Very Deep VAE	32	2	6 days	288

3.2 Ablation Study

To verify the effectiveness on the performance of the different attention schemes proposed in section 2, we conduct comprehensive ablation studies on architectures of a variable number of stochastic layers on the CIFAR-10 dataset. We incrementally add different combinations of the proposed non-local components of sections 2.3, 2.4, 2.5 in the NVAE architecture and we evaluate their impact on the estimated marginal likelihood reported in bits per dimension. Our results are shown in Table 5. We make the following observations:

- Regardless of the number of stochastic layers in the hierarchy, we observe improvement over the baseline model (Case 1 vs Case 8 and Case 9 vs Case 16 in Table 5).
- We notice that the 8-layer architecture that employs an attention mechanism in both the generative network and the inference counterpart of the model (Case 8 in Table 5) almost reaches the performance level of the baseline 16-layer architecture (Case 9 in Table 5). This fact indicates that our model results in a better utilization of the latent space. This is orthogonal to other works which aim at improvement by stabilizing deeper architectures instead.
- The effect of the spatially non-local stochastic layers is larger in the shallower architecture (Case 1 vs Case 2 and Case 9 vs Case 10 in Table 5). This may be because some loss of the latent information is compensated by additional layers. However, the total improvement yielded by the non-local components is larger in the deeper architecture, highlighting the importance of the inter-layer connectivity when constructing the prior and the posterior of each layer.

Table 5: **A comparison of the effect of the non-local features in a deep variational model on the CIFAR-10 dataset.** The NVAE (Vahdat & Kautz, 2020) is used as a baseline case (no non-local features).

Depth (L)	Case	Non-local layers	Non-local generative	Non-local inference	# Parameters ($\times 10^6$)	$-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$
8	1.	✗	✗	✗	39.473	3.062
	2.	✓	✗	✗	44.585	2.998
	3.	✗	✓	✗	44.716	3.019
	4.	✓	✓	✗	50.471	2.98
	5.	✗	✗	✓	46.3	3.034
	6.	✓	✗	✓	52.151	2.985
	7.	✗	✓	✓	51.925	2.986
	8.	✓	✓	✓	58.451	2.963
16	9.	✗	✗	✗	79.206	2.939
	10.	✓	✗	✗	89.602	2.912
	11.	✗	✓	✗	90.916	2.918
	12.	✓	✓	✗	102.744	2.872
	13.	✗	✗	✓	92.983	2.92
	14.	✓	✗	✓	104.882	2.905
	15.	✗	✓	✓	105.541	2.861
	16.	✓	✓	✓	118.966	2.823

4 RELATED WORK AND DISCUSSION

Our work builds on recent deep variational autoencoders (Vahdat & Kautz, 2020; Child, 2020), which first perform a fully deterministic bottom-up pass followed by a top-down pass involving both stochastic and deterministic features, as first proposed by Kingma et al. (2016). This hybrid top-down pass enables fully connected factorizations of the joint prior and posterior distributions, according to which the latent variables of each layer are conditioned on the latent variables of all the layers above in the hierarchy. However, modeling such conditional dependencies relies on local architectures which assume connectivity only between successive layers. We improve on prior work by introducing attention-guided mechanisms to actively discover long-range statistical dependencies in the latent space. We build on top of NVAE (Vahdat & Kautz, 2020) to demonstrate our method, but our suggestions are generic and could be applied to deeper and thinner hierarchies such as that of Child (2020), or the BIVA hierarchy which extends the deterministic bottom-up pass to latent variables (Maaløe et al., 2019). These configurations could further improve the results reported in Tables 10, 3. Moreover, the proposed model is amenable to other tools for improved variational inference such as the use of an autoregressive posterior (or prior) distribution (Kingma et al., 2016), a second-stage ancenstral sampling by an auxiliary VAE that learns the aggregated posterior (Dai & Wipf, 2019), or regularization techniques that promote congruence between latent representations of the original and transformed inputs (Sinha & Dieng, 2021).

Vector Quantized Variational Autoencoders (VQ-VAEs), first proposed by Razavi et al. (2019b), learn discrete latent representations. However, the training objective of these models does not formally constitute a lower bound on the marginal log-likelihood. Although VQ-VAEs have exhibited high-quality generative capability on high-resolution images, they base their performance on fully autoregressive PixelCNN priors (van den Oord et al., 2016) which render them too slow to sample from in case of large latent spaces. Most related to ours is the work of Esser et al. (2021). The authors use a fully autoregressive transformer in the latent space. However, they manage to handle low-dimensional latent spaces, leading to log-likelihoods infererior to that of Razavi et al. (2019b). Scaling this model to larger latent spaces, by leveraging the decomposed, attention-guided latent hierarchy proposed in this work could be an interesting future research direction.

Several works on generative modeling leverage self-attention mechanisms (Parmar et al., 2018; Chen et al., 2018; 2020; Child et al., 2019). However, these models are autoregressive, yielding conditional per-pixel distributions, and apply attention in the pixel space. Therefore, as already discussed, they are slow to sample from. Our work has log-likelihood performance comparable to that of these works, while using a non-autoregressive data distribution. The work of Tulsiani & Gupta (2021) uses a transformer-based encoder to learn pixel distributions conditioned on a small set of observed pixel values at arbitrary locations. Diffusion models (Ho et al., 2020; Kingma et al., 2021) can be viewed as VAEs and achieve impressive generative quality. However, similarly to the autoregressive models, sampling from them requires multiple network evaluations. Morever, their performance when trained with data augmentation (Kingma et al., 2021) can be similar to that of deep regularized VAEs (Sinha & Dieng, 2021). Finally, the work by Zhang et al. (2019) boosts GANs by incorporating non-local blocks (Wang et al., 2018) in the generator and the discriminator.

5 CONCLUSION

We have presented deep attentive VAE, the first attention-driven framework for variational inference in deep probabilistic models. We argue that the expressivity of current deep probabilistic models can be improved by selectively prioritizing statistical dependencies between latent variables that are potentially distant from each other. We introduced a scalable factorized attention scheme that first discovers inter-layer correlations, succeeded by intra-layer attention operations. Moreover, the generative model is allowed to query its posterior counterpart based on the available latent information to identify the data-dependent representations most informative for inference. We extensively evaluated the proposed architecture on multiple public datasets and showed that our model outperforms all existing deep VAEs, while requiring significantly less training time.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders. In *4th International Conference on Learning Representations, ICLR*, 2016.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pp. 1691–1703. PMLR, 2020.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational Lossy Autoencoder. In *5th International Conference on Learning Representations, ICLR*, 2017.

Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. Pixelsnail: An improved autoregressive generative model. In *International Conference on Machine Learning*, pp. 864–872. PMLR, 2018.

Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Bin Dai and David Wipf. Diagnosing and enhancing VAE models. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=B1e0X3C9tQ`.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12873–12883, 2021.

Karol Gregor, Frederic Besse, Danilo Jimenez Rezende, Ivo Danihelka, and Daan Wierstra. Towards conceptual compression. *Advances In Neural Information Processing Systems*, 29:3549–3557, 2016.

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taïga, Francesco Visin, David Vázquez, and Aaron C. Courville. PixelVAE: A Latent Variable Model for Natural Images. In *5th International Conference on Learning Representations, ICLR*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, June 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016b.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(5), 2013.

Yoon Kim, Sam Wiseman, Andrew Miller, David Sontag, and Alexander Rush. Semi-amortized variational autoencoders. In *International Conference on Machine Learning*, pp. 2678–2687. PMLR, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR*, 2014.

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 29*, 2016.

Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. One-shot learning by inverting a compositional causal process. Neural Information Processing Systems Foundation, Inc., 2013.

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

Yingzhen Li and Richard E Turner. R\'enyi divergence variational inference. *arXiv preprint arXiv:1602.02311*, 2016.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

James Lucas, George Tucker, Roger B Grosse, and Mohammad Norouzi. Don't blame the elbo! a linear vae perspective on posterior collapse. *Advances in Neural Information Processing Systems*, 32:9408–9418, 2019.

Xuezhe Ma, Chunting Zhou, and Eduard Hovy. Mae: Mutual posterior-divergence regularization for variational autoencoders. *arXiv preprint arXiv:1901.01498*, 2019.

Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. In *Advances in Neural Information Processing Systems 32*, 2019.

Vaden Masrani, Tuan Anh Le, and Frank Wood. The thermodynamic variational objective. *arXiv preprint arXiv:1907.00031*, 2019.

John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pp. 4055–4064. PMLR, 2018.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. Preventing posterior collapse with delta-vaes. *arXiv preprint arXiv:1901.03416*, 2019a.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In *Advances in neural information processing systems*, pp. 14866–14876, 2019b.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning , ICML*, 2014.

Jason Tyler Rolfe. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016.

Hossein Sadeghi, Evgeny Andriyash, Walter Vinci, Lorenzo Buffoni, and Mohammad H Amin. Pixelvae++: Improved pixelvae with discrete prior. *arXiv preprint arXiv:1908.09948*, 2019.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, 29:901–909, 2016.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Samarth Sinha and Adji B Dieng. Consistency regularization for variational auto-encoders. *arXiv preprint arXiv:2105.14859*, 2021.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. *Advances in neural information processing systems*, 29:3738–3746, 2016.

Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pp. 1214–1223. PMLR, 2018.

Shubham Tulsiani and Abhinav Gupta. Pixeltransformer: Sample conditioned signal generation. *arXiv preprint arXiv:2103.15813*, 2021.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, 2020.

Arash Vahdat, William Macready, Zhengbing Bian, Amir Khoshaman, and Evgeny Andriyash. Dvae++: Discrete variational autoencoders with overlapping transformations. In *International Conference on Machine Learning*, pp. 5035–5044. PMLR, 2018.

Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *Proceedings of the 33nd International Conference on Machine Learning, ICML*, 2016.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7794–7803, 2018.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pp. 7354–7363. PMLR, 2019.

SUPPLEMENTARY MATERIAL FOR THE PAPER: DEEP ATTENTIVE VARIATIONAL INFERENCE

Anonymous authors
Paper under double-blind review

A MODEL ARCHITECTURE

A.1 INFERENCE NETWORK

In this section, we provide the formal description of the attention-enhanced inference network presented in section 2.4. Equivalently, we will present the procedure for the construction of the conditioning factor of the posterior: $q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l})$.

The inference network is first traversed from the bottom (stochastic layer closer to the data $\boldsymbol{x}$) to the top, to compute successive hidden data representations $\boldsymbol{h}_l \in R^{H\times W\times C}$ and their keys: $\boldsymbol{k}_l^q \in R^{H\times W\times Q}$, with $Q \ll C$:

$$[\boldsymbol{h}_l,\ \boldsymbol{k}_l^q] \leftarrow T_l^q(\boldsymbol{h}_{l+1} \oplus \boldsymbol{k}_{l+1}^q), \quad \text{for} \quad l = L, L-1, \ldots, 1. \tag{11}$$

L is the number of stochastic layers. We assume $\boldsymbol{h}_{L+1} \equiv \boldsymbol{x}$. T_l^q is a non-linear transformation, typically implemented as a cascade of ResNet cells (He et al., 2016a;b).

Afterwards, samples from the posterior are drawn in top-down order in the topological ordering of the generative model:

$$\begin{aligned} &\boldsymbol{z}_1 \sim q(\boldsymbol{z}_1 \mid \boldsymbol{x}), \\ &\boldsymbol{z}_l \sim q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l}), \quad \text{for} \quad l = 2, \ldots, L. \end{aligned} \tag{12}$$

The inference network receives the prior context $\boldsymbol{c}_l^p \in \mathbb{R}^{H\times W\times C}$ and a query feature $\boldsymbol{s}_l^q \in \mathbb{R}^{H\times W\times Q}$ from the generative model, see Equation 3. $\boldsymbol{c}_l^p$ and $\boldsymbol{s}_l^q$ contain latent information of previous layers $\boldsymbol{z}_{<l}$. The observed data-dependent feature $\boldsymbol{h}_l$ and the latent feature $\boldsymbol{c}_l^p$ are combined via an operator $\oplus$ to give the context of the posterior $\boldsymbol{c}_l^q \in \mathbb{R}^{H\times W\times C}$ along with its key $\boldsymbol{m}_l^q \in \mathbb{R}^{H\times W\times Q}$:

$$[\boldsymbol{c}_l^q,\ \boldsymbol{m}_l^q] \leftarrow \boldsymbol{h}_l \oplus \boldsymbol{c}_l^p. \tag{13}$$

As already pointed out, $\boldsymbol{m}_l^q$ contains stochastic, latent information, while $\boldsymbol{k}_l^q$ of Equation 11 contains only deterministic, observed information.

We allow variational layer l to have access to all subsequent data representations $\boldsymbol{h}_{\geq l}$. During inference, the variational layer decides parts of the deterministic features $\boldsymbol{h}_{\geq l}$ and the stochastic feature $\boldsymbol{c}_l^q$ to pay attention to. For pixel (i, j), with $1 \leq i \leq H$, $1 \leq j \leq W$, the generative model queries according to $\boldsymbol{s}_l^q(i, j) \in \mathbb{R}^Q$ and computes an attention distribution, as follows:

$$\boldsymbol{\alpha}_{l',l}^q(i,j) = \frac{\exp(\boldsymbol{k}_{l'}^q(i,j)^T \boldsymbol{s}_l^q(i,j))}{\mathcal{Z}^q(i,j)}, \tag{14}$$

$$\boldsymbol{\alpha}_{p,l}^q(i,j) = \frac{\exp(\boldsymbol{m}_l^q(i,j)^T \boldsymbol{s}_l^q(i,j))}{\mathcal{Z}^q(i,j)}, \tag{15}$$

$$\mathcal{Z}^q(i,j) = \exp(\boldsymbol{m}_{l'}^q(i,j)^T \boldsymbol{s}_l^q(i,j)) + \sum_{l' \geq l} \exp(\boldsymbol{k}_{l'}^q(i,j)^T \boldsymbol{s}_l^q(i,j)). \tag{16}$$

$\boldsymbol{\alpha}_{p,l}^q(i,j) \in \mathbb{R}$ reflects the importance of the stochastic context $\boldsymbol{c}_l^q(i,j) \in \mathbb{R}^C$ to pixel (i, j). Similarly, $\boldsymbol{\alpha}_{l',l}^q(i,j) \in \mathbb{R}$ reflects the importance of the deterministic feature $\boldsymbol{h}_{l'}(i,j) \in \mathbb{R}^C$ of layer l' to pixel (i, j). $\mathcal{Z}^q(i,j) \in \mathbb{R}$ is a normalization constant.

The posterior context for pixel (i, j), $\hat{\boldsymbol{c}}_l^q(i, j) \in \mathbb{R}^C$, is the weighted sum:

$$\hat{\boldsymbol{c}}_l^q(i, j) = \boldsymbol{\alpha}_{p,l}^q(i, j)\boldsymbol{c}_l^q(i, j) + \sum_{l' \geq l} \boldsymbol{\alpha}_{l',l}^q(i, j)\boldsymbol{h}_{l'}(i, j). \tag{17}$$

To scale the deep variational model with ease, we use the following normalization:

$$\hat{\boldsymbol{c}}_l^q \leftarrow \hat{\boldsymbol{c}}_l^q + \mathrm{Gelu}(\mathrm{LayerNorm}(\hat{\boldsymbol{c}}_l^q)), \tag{18}$$

where Gelu is the GELU non-linearity (Hendrycks & Gimpel, 2016) and $\mathrm{LayerNorm}$ a layer normalization operation (Ba et al., 2016).

At the beginning of the training, we let the inference model rely on local dependencies, as captured by $\boldsymbol{c}_l^q$, by introducing the residual connection:

$$\boldsymbol{c}_l^q \leftarrow \boldsymbol{c}_l^q + \gamma_l^q \hat{\boldsymbol{c}}_l^q, \tag{19}$$

where γ_l^q is a learnable scalar parameter initialized to zero.

We adopt the Gaussian residual parametrization between the prior and the posterior proposed by Vahdat & Kautz (2020).

Recall that the prior is given by:

$$p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{c}_l^p), \boldsymbol{\sigma}(\boldsymbol{c}_l^p)). \tag{20}$$

The posterior is then given by:

$$q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{c}_l^p) + \Delta\boldsymbol{\mu}(\boldsymbol{c}_l^q), \boldsymbol{\sigma}(\boldsymbol{c}_l^p)\Delta\boldsymbol{\sigma}(\boldsymbol{c}_l^q)), \tag{21}$$

where the summation and multiplication are pointwise, and $\boldsymbol{c}_l^q$ is defined in Equation 19. This process is described in detail in Algorithm 3.

A.2 Spatially Non-Local Variational Layers

In this section, we provide the formal description of the non-local modules illustrated in Figure 3.

Let $\boldsymbol{c} \in \mathbb{R}^{H \times W \times C}$ a 3D context feature. We treat $\boldsymbol{c}$ as a sequence of N, with $N = H \times W$, $C-$dimensional features, such that $\boldsymbol{c} \in \mathbb{R}^{N \times C}$. The non-local block maps $\boldsymbol{c}$ to a feature $\boldsymbol{v} \in \mathbb{R}^{N \times C}$ of the same dimension by taking into account the response of all the locations in the sequence.

The context feature $\boldsymbol{c}$ is first mapped to two feature spaces of a lower dimension Q, with $Q << C$, through a key $\boldsymbol{\alpha}$ and a query $\boldsymbol{\beta}$ linear transformation:

$$\boldsymbol{\alpha}(\boldsymbol{c}) = \boldsymbol{c}\boldsymbol{W}_\alpha, \tag{22}$$

$$\boldsymbol{\beta}(\boldsymbol{c}) = \boldsymbol{c}\boldsymbol{W}_\beta, \tag{23}$$

where the linear transformations $\boldsymbol{W}_{\boldsymbol{\alpha}} \in \mathbb{R}^{C \times Q}$, $\boldsymbol{W}_{\boldsymbol{\beta}} \in \mathbb{R}^{C \times Q}$ are implemented as 1×1 convolutions, and $\boldsymbol{\alpha}(\boldsymbol{c}) \in \mathbb{R}^{N \times Q}$, $\boldsymbol{\beta}(\boldsymbol{c}) \in \mathbb{R}^{N \times Q}$.

The energy matrix $\boldsymbol{E}(\boldsymbol{c}) \in \mathbb{R}^{N \times N}$ is computed as:

$$\boldsymbol{E}(\boldsymbol{c}) = \boldsymbol{\beta}(\boldsymbol{c})\boldsymbol{\alpha}^T(\boldsymbol{c}). \tag{24}$$

Each element $\boldsymbol{E}_{i,j}(\boldsymbol{c}) = \boldsymbol{\beta}_i(\boldsymbol{c}) \times \boldsymbol{\alpha}_j^T(\boldsymbol{c})$, where $\boldsymbol{\beta}_i$ is the $i-$th row of $\boldsymbol{\beta}$, reflects the relevance of pixel j to pixel i. The attention distribution is computed by a scaled softmax row normalization, such that the corresponding attention score $\boldsymbol{A}_{i,j}(\boldsymbol{c})$ is :

$$\boldsymbol{A}_{i,j}(\boldsymbol{c}) = \frac{\exp(\omega\boldsymbol{\beta}_i(\boldsymbol{c})\boldsymbol{\alpha}_j^T(\boldsymbol{c}))}{\mathcal{Z}(i)}, \tag{25}$$

$$\mathcal{Z}(i) = \sum_{j'=1}^{N} \exp(\omega\boldsymbol{\beta}_i(\boldsymbol{c})\boldsymbol{\alpha}_{j'}^T(\boldsymbol{c})). \tag{26}$$

$\omega \in \mathbb{R}$ is learnable scalar parameter initialized to one. $\boldsymbol{c}$ is also linearly projected to an embedding space via a transformation $\boldsymbol{\eta}$:

$$\boldsymbol{\eta}(\boldsymbol{c}) = \boldsymbol{c}\boldsymbol{W}_{\boldsymbol{\eta}}, \tag{27}$$

where $\boldsymbol{W}_{\boldsymbol{\eta}} \in \mathbb{R}^{C \times C}$ and $\boldsymbol{\eta}(\boldsymbol{c}) \in \mathbb{R}^{N \times C}$. $\boldsymbol{W}_{\boldsymbol{\eta}}$ is implemented as a 1×1 convolution.

The global output $\boldsymbol{v} \in \mathbb{R}^{N \times C}$ is:

$$\boldsymbol{v} = \boldsymbol{A}(\boldsymbol{c})\boldsymbol{\eta}(\boldsymbol{c}). \tag{28}$$

$\boldsymbol{A}(\boldsymbol{c}) \in \mathbb{R}^{N \times N}$ is the matrix of the attention scores defined in Equation 26. Finally, $\boldsymbol{v}$ represents a scaled, residual mapping of $\boldsymbol{c}$ such that the final output is:

$$\boldsymbol{c} \leftarrow \boldsymbol{c} + \gamma \boldsymbol{v}, \tag{29}$$

where $\gamma \in \mathbb{R}$ is a learnable scalar parameter initialized to zero.

A.3 Algorithms

Algorithms 1, 2, and 3 summarize the inference procedure of our model. T_l^p and T_l^q refer to a block of ResNet cells (Figure 3). F^d, F_l^p, and F_l^q refer to a single convolution computing the parameters of the data, prior and posterior distribution respectively. $\oplus$ is a merging operator unless one of its operands is zero, in which case it corresponds to an identity mapping.

Algorithm 1 Infer

`This algorithm computes a lower bound on the marginal data log-likelihood (Equation 2).`

Inputs
  $\boldsymbol{x}$: The data.
$\boldsymbol{h},\ \boldsymbol{k}^q \leftarrow \text{Bottom} - \text{Up Pass}\ (\boldsymbol{x})$ (Algo. 2).
$\boldsymbol{c}^p,\ \boldsymbol{z},\ p(\boldsymbol{z}),\ q(\boldsymbol{z} \mid \boldsymbol{x}) \leftarrow \text{Top} - \text{Down Pass}\ \big(\boldsymbol{h},\ \boldsymbol{k}^q\big)$ (Algo. 3).
$\boldsymbol{c}_{L+1}^p \leftarrow T_{L+1}^p(\boldsymbol{c}_L^p \oplus \boldsymbol{z}_L)$.
$\boldsymbol{\theta}(\boldsymbol{z}) \leftarrow F^d(\boldsymbol{c}_{L+1}^p)$.
$p(\boldsymbol{x} \mid \boldsymbol{z}) \triangleq p(\boldsymbol{x}; \boldsymbol{\theta}(\boldsymbol{z}))$.
ELBO $\approx \log p(\boldsymbol{x} \mid \boldsymbol{z}) - D_{KL}(q(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z}))$.
Return
  ELBO: The evidence lower bound.

Algorithm 2 Bottom-Up Pass

`This algorithm computes a sequence of deterministic, hidden features and keys of the data.`

Inputs
  $\boldsymbol{x}$: The data.
Set $\boldsymbol{h}_{L+1} \equiv \boldsymbol{x}$ and $\boldsymbol{k}_{L+1}^q \equiv 0$.
for $l = L, L-1, \ldots, 1$ **do**
  $[\boldsymbol{h}_l,\ \boldsymbol{k}_l^q] \leftarrow T_l^q(\boldsymbol{h}_{l+1} \oplus \boldsymbol{k}_{l+1}^q)$.
end for
Return
  $\boldsymbol{h} \triangleq \{\boldsymbol{h}_l\}_{l=1}^L$: The hidden data features.
  $\boldsymbol{k}^q \triangleq \{\boldsymbol{k}_l^q\}_{l=1}^L$: The data keys.

Algorithm 1 computes the optimization objective of Equation 2. It consists of three main steps. During the bottom-up pass described in Algorithm 2 (see also left part of Figures 1a, 1b), the data $\boldsymbol{x}$ are passed through a sequence of ResNet transformations T_l^q, that will be used by the stochastic layers of the VAE hierarchy when forming the posterior distribution, in a reverse order. Note that in Algorithm 2, $\boldsymbol{k}_{L+1}^q$ is set to zero since layer L has access only to its own $\boldsymbol{h}_L$ and it is not connected with another layer from the bottom. Algorithms 1 and 2 are very similar to that of typical deep VAEs. The only difference is that the ResNet transformations are also producing the data keys $\boldsymbol{k}^q$ along with the data features $\boldsymbol{h}$. The second, top-down phase of Algorithm 1 infers samples of the latent variables $\boldsymbol{z}$ along with the prior and posterior distributions. It is described in Algorithm 3 and

Algorithm 3 Top-Down Pass

This algorithm infers a sequence of stochastic, latent representations of the data.

Inputs
  $\boldsymbol{h} \triangleq \{\boldsymbol{h}_l\}_{l=1}^L$: The hidden data features.
  $\boldsymbol{k}^q \triangleq \{\boldsymbol{k}_l^q\}_{l=1}^L$: The data keys.
Set $\boldsymbol{z}_0 \equiv 0$, $\boldsymbol{c}_0^p \equiv 0$, $\boldsymbol{s}_1^p \equiv 0$, $\boldsymbol{s}_L^q \equiv 0$, and $\boldsymbol{k}_L^p \equiv 0$.
for $l = 1, 2, \ldots, L$ **do**
  $[\boldsymbol{s}_l^q,\ \boldsymbol{m}_l^p, \boldsymbol{s}_l^p,\ \boldsymbol{k}_l^p] \leftarrow T_l^p(\boldsymbol{z}_{l-1} \oplus \boldsymbol{c}_{l-1}^p)$.
  $\boldsymbol{c}_l^p \leftarrow \boldsymbol{m}_l^p \oplus \boldsymbol{s}_l^q$.

  Form the prior distribution.
  if $l > 1$ **then**
    $\boldsymbol{c}_l^p \leftarrow \boldsymbol{c}_l^p + \mathrm{Gelu}(\mathrm{LayerNorm}(\boldsymbol{c}_l^p))$.
    $\boldsymbol{c}_{l-1}^p \leftarrow \mathrm{LayerNorm}\big(\boldsymbol{c}_{l-1}^p\big)$.
    $\hat{\boldsymbol{c}}_l^p \leftarrow \mathrm{Attend}\big(\{\boldsymbol{c}_{l'}^p\}_{l'=1}^{l-1},\ \boldsymbol{s}_l^p,\ \{\boldsymbol{k}_{l'}^p\}_{l'=1}^{l-1}\big)$.
    $\hat{\boldsymbol{c}}_l^p \leftarrow \hat{\boldsymbol{c}}_l^p + \mathrm{Gelu}(\mathrm{LayerNorm}(\hat{\boldsymbol{c}}_l^p))$.
    $\hat{\boldsymbol{c}}_l^p \leftarrow \boldsymbol{c}_l^p + \gamma_l^p \hat{\boldsymbol{c}}_l^p$.
  else
    $\hat{\boldsymbol{c}}_l^p \leftarrow \boldsymbol{c}_l^p$.
  end if
  $[\boldsymbol{\mu}_l^p,\ \boldsymbol{\sigma}_l^p] \leftarrow F_l^p(\hat{\boldsymbol{c}}_l^p)$.
  $p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l}) = \mathcal{N}(\boldsymbol{\mu}_l^p, \boldsymbol{\sigma}_l^p)$.

  Form and sample from the posterior distribution.
  if $l < L$ **then**
    $[\boldsymbol{c}_l^q,\ \boldsymbol{m}_l^q] \leftarrow \boldsymbol{h}_l \oplus \hat{\boldsymbol{c}}_l^p$.
    $\boldsymbol{c}_l^q \leftarrow \boldsymbol{c}_l^q + \mathrm{Gelu}(\mathrm{LayerNorm}(\boldsymbol{c}_l^q))$.
    $\hat{\boldsymbol{c}}_l^q \leftarrow \mathrm{Attend}\big(\{\boldsymbol{h}\}_{l'=l}^L \cup \{\boldsymbol{c}_l^q\},\ \boldsymbol{s}_l^q,\ \{\boldsymbol{k}_{l'}^q\}_{l'=l}^L \cup \{\boldsymbol{m}_l^q\}\big)$.
    $\hat{\boldsymbol{c}}_l^q \leftarrow \hat{\boldsymbol{c}}_l^q + \mathrm{Gelu}(\mathrm{LayerNorm}(\hat{\boldsymbol{c}}_l^q))$.
    $\hat{\boldsymbol{c}}_l^q \leftarrow \boldsymbol{c}_l^q + \gamma_l^q \hat{\boldsymbol{c}}_l^q$.
  else
    $\hat{\boldsymbol{c}}_l^q \leftarrow \boldsymbol{h}_l \oplus \hat{\boldsymbol{c}}_l^p$.
  end if
  $[\Delta\boldsymbol{\mu}_l^q,\ \Delta\boldsymbol{\sigma}_l^q] \leftarrow F_l^q(\hat{\boldsymbol{c}}_l^q)$.
  $q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l}) = \mathcal{N}(\boldsymbol{\mu}_l^p + \Delta\boldsymbol{\mu}_l^q, \boldsymbol{\sigma}_l^p \Delta\boldsymbol{\sigma}_l^q)$.
  $\boldsymbol{z}_l \sim q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l})$.
end for
Return
  $\boldsymbol{c}^p \triangleq \{\boldsymbol{c}_l^p\}_{l=1}^L$: The latent context vectors.
  $\boldsymbol{z} \triangleq \{\boldsymbol{z}_l\}_{l=1}^L$: The inferred latent variables.
  $p(\boldsymbol{z}) = \prod_l p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$: The prior distribution of the latent variables.
  $q(\boldsymbol{z} \mid \boldsymbol{x}) = \prod_l q(\boldsymbol{z}_l \mid \boldsymbol{x},\ \boldsymbol{z}_{<l})$: The approximate posterior distribution of the latent variables.

explained in detail below. In Algorithm 1, $\boldsymbol{c}_L^p$ encapsulates latent information of $\boldsymbol{z}_{<L}$. Along with $\boldsymbol{z}_L$, they feed the ResNet block T_{L+1}^p succeeded by the convolution F^d to compute the conditional data likelihood $p(\boldsymbol{x} \mid \boldsymbol{z})$ that is parameterized by $\boldsymbol{\theta}$.

Algorithm 3 describes the top-down pass that is also visualized in the right part of Figures 1a, 1b. The layers are traversed from the top ($l = 0$) to the bottom-up ($l = L$), sampling latent variables $\boldsymbol{z}_l$ from the posterior distribution and computing context vectors $\boldsymbol{c}_l^p$ that contain processed latent information of previous layers. The prior distribution is also computed in order to obtain the prior regularization term $D_{KL}(q(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z}))$ of the objective function in Equation 2.

The first layer does not receive latent information or context from a previous layer, therefore it sets $\boldsymbol{z}_0$ and $\boldsymbol{c}_0^p$ to zero. Similarly, since there is no other layer in the generative network to connect with, it does not produce a query $\boldsymbol{s}_1^p$ to attend with. The last layer does not pass any prior information to a subsequent layer, therefore it does not compute a key $\boldsymbol{k}_L^p$. Similarly, the last layer can use only its own data features $\boldsymbol{h}_L$. Therefore, the generative portion of the layer does not compute a query for the inference portion setting $\boldsymbol{s}_L^q \equiv 0$.

Each layer l, unless in a marginal case described above, computes four types of features $\boldsymbol{k}_l^p$, $\boldsymbol{s}_l^p$, $\boldsymbol{s}_l^q$ and $\boldsymbol{m}_l^p$. Key features $\boldsymbol{k}_l^p$ are used at the depth-wise Attend step of next layers $> \boldsymbol{l}$. $\boldsymbol{s}_l^p$ is a query to the generative network of previous layers $< l$. $\boldsymbol{s}_l^q$ is a query to the inference network of current and next layers $\geq l$. Finally, $\boldsymbol{m}_l^p$ are stochastic features that, along with $\boldsymbol{s}_l^q$, will be used to form the condition of the prior distribution (Line 6). We note here that $\boldsymbol{s}_l^q$ is used both to form the condition of the prior $\hat{\boldsymbol{c}}_l^p$ and to query the inference network when forming the condition of the posterior $\hat{\boldsymbol{c}}_l^q$. A layer in a deep VAE that is not attention-guided does not produce $\boldsymbol{k}_l^p, \boldsymbol{s}_l^q, \boldsymbol{s}_l^p$ and the computation falls into the "else" cases of Algorithm 3 (Lines 15, 26).

In Line 11, the generative network queries previous layers (via $\boldsymbol{s}_l^p$) to identify latent context features $\boldsymbol{c}_{l'}^p$ represented by keys $\boldsymbol{k}_{l'}^p$ that are most critical when forming its prior beliefs. The resulting feature $\hat{\boldsymbol{c}}_l^p$, after being normalized in Line 12, rectifies the initial context $\boldsymbol{c}_l^p$ by the residual connection of Line 13. The initial prior context $\boldsymbol{c}_l^p$ is being used by next layers while the attention-guided context $\hat{\boldsymbol{c}}_l^p$ is the conditioning factor of the prior distribution formed in Lines 17, 18.

A similar procedure takes place to form the posterior distribution. $\boldsymbol{c}_l^q$ is the initial posterior context that merges deterministic information $\boldsymbol{h}$ and stochastic, latent information $\hat{\boldsymbol{c}}_l^p$, as seen in Lines 20, 26. $\boldsymbol{c}_l^q$ is updated with an attention-driven feature computed in Line 22 by the residual connection of Line 24. In Line 29, the residual parametrization of the posterior, as proposed by (Vahdat & Kautz, 2020) and also described in Equation 21, is derived.

B IMPLEMENTATION DETAILS

The experiments of section 3.1.1 are run on NVIDIA GeForce RTX 2080 Ti while the experiments of sections 3.1.2 and 3.2 are run on 32GB V100-SXM2. For all the experiments, we use the AdaMax (Kingma & Ba, 2014) optimizer with the default parameters for training. The learning rate is decayed according to the cosine schedule with warm-up during the first 5 epochs. When we train for 900 epochs, we use a cosine decay schedule with restarts with $T_0 = 300$ and $T_{mult} = 2$ follwoing the notation of Loshchilov & Hutter (2016). We apply weight normalization (Salimans & Kingma, 2016) in all our models. We use the Swish (Ramachandran et al., 2017) activation function in the ResNet cells of the architecture.

Table 6 summarizes the main training and architectural hyperparameters for the experiments presented in section 3. However, it should be noted that due to the heavy computing requirements, we do not exhaustively optimize the hyperparameters. Therefore, the results reported could potentially be further improved.

Table 6: **Summary of key hyperparamaters for experiments.** The first set pertains to training hyperparameters. The second set refers to statistical hyperparameters specific to variational models. The last set includes architectural hyperparameters.

Hyperparameter	OMNIGLOT	MNIST	CIFAR-10
# epochs	400	400	600
batch size per GPU	16	50	40
initial learning rate	0.01	0.01	0.01
BN momentum	0.9	0.9	0.9
BN epsilon	$1e-5$	$1e-5$	$1e-5$
# GPUs	2	2	4
l_2-regularization λ	$1.15e-4$	$1.15e-4$	$1.15e-4$
KL annealing	$0.1 \rightarrow 1$	$0.1 \rightarrow 1$	$0.1 \rightarrow 1$
# KL annealing epochs	16	16	16
σ_{noise}	✗	✗	0.001
$\log \sigma_p \geq$	-5.0	-1.0	-1.0
number of layers (L)	15	15	16
spatial dims of latent space $\boldsymbol{z}$	16×16	8×8	16×16
# channels in $\boldsymbol{z}$	20	20	20
# pre-proc blocks	1	2	1
# ResNet cells in pre-proc blocks	3	3	2
# initial channels in pre-proc blocks	32	32	128
# channels in encoder	64	128	256
# ResNet cells in encoder	2	2	2
# channels in decoder	64	128	256
# ResNet cells in decoder	2	2	2
# post-proc blocks	1	2	1
# ResNet cells in pre-proc blocks	3	3	2
# initial channels in post-proc blocks	64	128	256
key dim in top-down non-local path	8	8	20
layer norm in top-down non-local path	✓	✓	✓
key dim in bottom-up non-local path	8	8	20
layer norm in bottom-up non-local path	✓	✓	✓
key dim in spatially non-local blocks	8	16	32

KL Annealing. We apply KL annealing (Sønderby et al., 2016) for the first 16 training epochs starting from 0.1 to 1. This technique introduces a scheduled regularization coefficient β for the KL-divergence. Formally, the regularized ELBO objective becomes:

$$\mathbb{E}_q[\log p(\boldsymbol{x} \mid \boldsymbol{z})] - \beta D_{KL}(q(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z})), \quad \beta_0 \leq \beta \leq 1, \ \beta_0 < 1, \tag{30}$$

where β linearly increases from β_0 to 1 for a number of epochs at the beginning of the training.

Variational Distributions. We use Gaussian prior and posterior distributions connected via a residual parametrization, see Equation 21. To ease training stability, we bound the log-scale $\log \boldsymbol{\sigma}$ of the prior as reported in Table 6. In case of CIFAR-10, we also apply additive zero mean, Gaussian noise to the prior scale. The mean $\boldsymbol{\mu}$ and the log-scale $\log \boldsymbol{\sigma}$ are jointly learned by using a shared convolution. In particular, a 1×1 convolution with an ELU activation is used for the prior parameters. For the posterior parameters of all but the first layer, two 3×3 convolutions with an ELU non-linearity are used. For the posterior of the first layer, a single 1×1 convoltion with no activation is used.

Data Distributions. For MNIST and OMNIGLOT, we assume a logit-based parametrization of a Bernoulli distribution. For CIFAR-10, we use a mixture of discretized Logistic distributions (Salimans et al., 2017) with five mixture components. The parameters are jointly learned by a single 3×3 convolution with an ELU activation.

Pre-Processing and Post-Processing Blocks. The data $\boldsymbol{x}$, before being passed to the stochastic hierarchy of L layers, are first pre-processed by a cascade of ResNet cells. Each block is responsible for downsampling the spatial dimension by a factor of 2. For MNIST, the image is downsampled twice by two blocks, leading to latent variables with spatial dimension 8×8. For OMNIGLOT and CIFAR-10, one preprocessing block is being used leading to latent spaces of dimension 16×16. No further downsampling is performed by the encoders in the main hierarchy. Each block consists of a series of ResNet cells as shown in Figure 3. The last cell of the block is responsible for downsampling while doubling the number of the output channels. For CIFAR-10, we interleave non-local blocks as shown in Figure 3. Symmetrically, the context of the last layer L is first being post-processed by a cascade of post-processing blocks before being passed to the data distribution of the decoder. Each such block is responsible for upsampling the context by a factor of 2, in order to mach the initial spatial dimension of the image. The upsampling operation is performed by the first residual cell while halving the number of the output channels. For MNIST and OMNIGLOT, we assume the spatially non-local residual cells of Figure 3.

C Ablation studies

In this section, we perform ablation experiments to investigate the effect of different architectural choices on the model performance.

For the experiments in Tables 7, 8 and 10, the models were trained on CIFAR-10 for 500 epochs. We used shallower attentive VAE models of 8 layers for these experiments. We considered model instances of a varying size, with 64 and 128 channels at the beginning of the bottom-up network and at the end of the top-down network. As already noted in section B, the number of channels is halved when a residual cell upsamples the spatial domain and vice versa. In all cases, model instances with more channels in the ResNet modules in encoder and decoder perform significantly better. We can also see that weight normalization (Salimans & Kingma, 2016), Table 7, and batch normalization layers interleaved with the convolutions of both the bottom-up and the top-down model, Table 8, improves performance. These results are consistent with the observations made by Vahdat & Kautz (2020). Moreover, their effect is greater on the larger models. In Table 9, we report the performance of different activation functions. The Swish (Ramachandran et al., 2017) and GELU (Hendrycks & Gimpel, 2016) activations yield the best performance. This is in accordance with similar choices in current deep VAE architectures (Vahdat & Kautz, 2020; Child, 2020). The rest of the hyperparameters are identical to that of section B.

Table 7: **Weight Normalization.** Predictive performance with and without weight normalization for two model sizes. $-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$ is reported.

# initial channels	w/ Weight Norm.	w/o Weight Norm.
64	3.07	3.09
128	**2.94**	2.99

Table 8: **Batch Normalization.** Predictive performance with and without batch normalization for two model sizes. $-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$ is reported.

# initial channels	w/ Batch Norm.	w/o Batch Norm.
64	3.07	3.09
128	**2.94**	2.97

Table 9: **Activation Function.** Predictive performance for various activation functions for two model sizes. $-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$ is reported.

# initial channels	Activation			
	ReLU	ELU	Swish	GELU
64	3.12	3.09	3.07	3.07
128	2.99	2.99	**2.94**	**2.94**

In Table 10, we take a closer look at the formulation of the normalized depth-wise attention proposed in section 2.3. The differences in the performance were more noticeable for deep VAE models of 16 layers. We train all model instances for 400 epochs. We omit the application of the normalization scheme presented in Equations 6 and 7 in the generative and/or inference network. We confirm that usage of depth-wise normalization facilitates training and helps scale the depth of the proposed architecture.

Table 10: **Depth-wise Attention Normalization.** Predictive performance for normalized and unormalized depth-wise attention in decoder and encoder in a deep, 16-layer, VAE. $-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$ is reported.

Normalized Depth-Wise Attention		
Generative Network	Inference Network	$-\log p(\boldsymbol{x}) \leq$ (bits/dim) $\downarrow$
✓	✓	**2.82**
✗	✓	2.89
✓	✗	2.86
✗	✗	2.90

D EMPIRICAL EVIDENCE OF BETTER LATENT VARIABLE UTILIZATION BY ATTENTIVE VAE

In this section, we probe the capacity of attentive VAE to utilize the latent information inserted in the model. We first examine the Kullback–Leibler divergence $D_{KL}(q \parallel p)$ observed on the test data across the hierarchy during training. Unless it explodes, a high $D_{KL}(q \parallel p)$ loss term is often desirable since it leads to a smaller negative ELBO (NELBO) loss in Equation 2. This is because higher $D_{KL}(q \parallel p)$ indicates more informative latent factors $\boldsymbol{z}$, and in turn, higher conditional likelihood $p(\boldsymbol{x} \mid \boldsymbol{z})$. For example, Kingma et al. (2016) use a modified objective function that lower bounds $D_{KL}(q \parallel p)$ by some free bits. Similarly, Vahdat & Kautz (2020) use a residual parametrization between the prior and the posterior to keep more latent variables active.

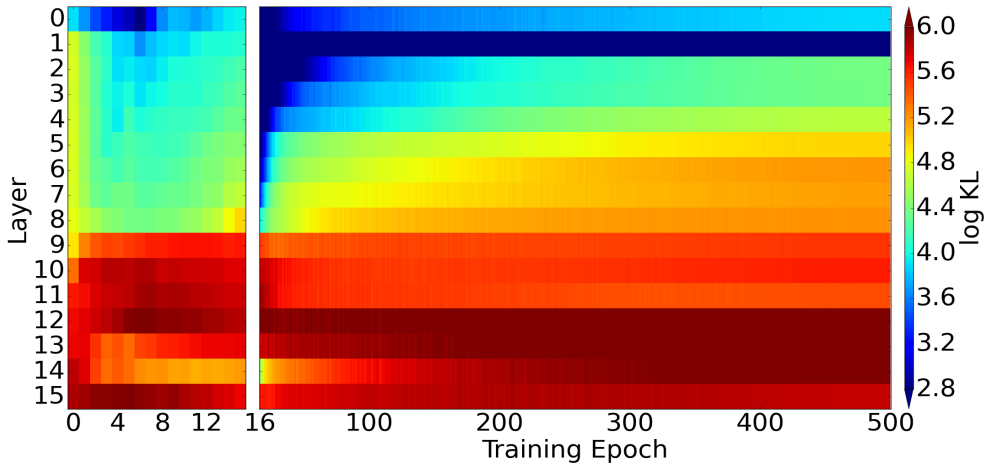


Figure 4: **Latent space activation map for NVAE.** The $\log D_{KL}(q \parallel p)$ for each latent layer $\boldsymbol{z}_l$ as a function of the training epoch on CIFAR-10 for the baseline NVAE.

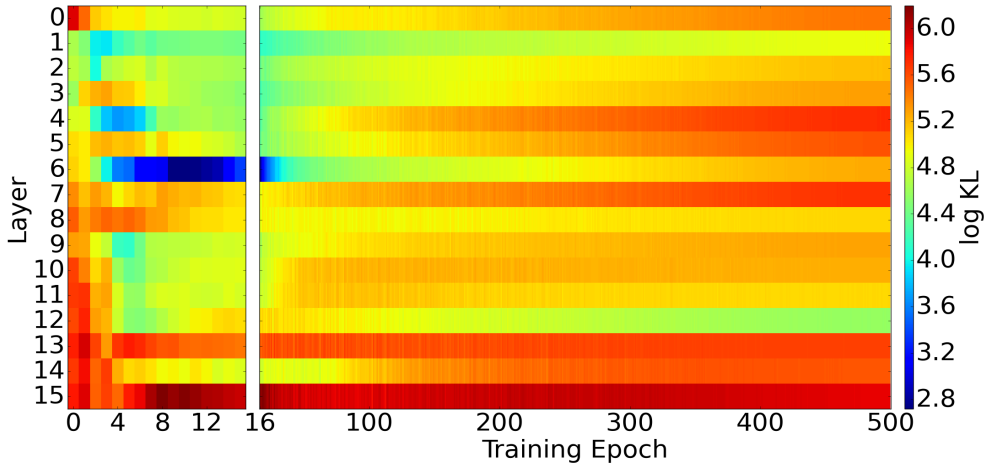


Figure 5: **Latent space activation map for Attentive VAE (ours).** The $\log D_{KL}(q \parallel p)$ for each latent layer $\boldsymbol{z}_l$ as a function of the training epoch on CIFAR-10 for the attention boosted VAE.

Table 11: $-\log p(\boldsymbol{x})$ **for varying depth** L **(bits/dim) for attentive VAE (ours).**

Depth (L)	bits/dim $\downarrow$	$\Delta(\cdot)\%$
2	3.31	–
4	3.12	-5.7
8	2.96	-5.1
16	2.81	-5.1

In this work, we tackle posterior collapse by imposing attention-driven connections between the layers. The proposed depth-wise attention naturally realizes long-range couplings between the layers at a pixel level. Therefore, it selectively preserves the most informative latent information which may lie far beyond in the hierarchy. In Figure 4, we show $\log D_{KL}(q \parallel p)$ per layer $\boldsymbol{z}_l$ for the baseline NVAE during the first 16 KL annealing epochs (left part) and for the rest of the training (right part). Similarly, in Figure 5, we show $\log D_{KL}(q \parallel p)$ for the proposed model.

Given the above plots, we make the following observations:

- Although in both models, $\log D_{KL}(q \parallel p)$ remains large (always positive), our model presents a more balanced activation map in the latent space. In contrast, in Figure 4, we see that the first two layers are significantly much less active compared to the layers closer to $p(\boldsymbol{x} \mid \boldsymbol{z})$. This phenomenon is exacerbated in deeper hierarchies.
- In Figure 4, $\log D_{KL}(q \parallel p)$ increases with depth. In contrast, in Figure 5 of our model, the first layer converges to a much higher $D_{KL}(q \parallel p)$ compared to layer 12 although it is much farther from $p(\boldsymbol{x} \mid \boldsymbol{z})$.
- In Figure 5, we notice that as the learning progresses, the attention scores can help activate some latent units. For example, layer 6 is relatively dormant at the beginning of the training, while it converges at a highly informative posterior distribution at the end.

Finally, in Table 11, we show the bits per dimension as the depth varies for our model. All models in Table 11 were trained for 500 epochs. As we can see the predictive improvement gained when doubling the size of the model from 8 to 16 layers is on par with the improvement accomplished when transitioning from a model of 2 layers to a model of 4 layers. In contrast, deep VAEs without attention experience half the amount of relative improvement, as seen in Table 1. We also note that the first two smallest models have converged during the last 100 training epochs which is not the case for the last two model instances. Therefore, the gap in the relative decrease for the models of 8 and 16 layers could be further closed by longer training.

E VISUALIZATION OF ATTENTION PATTERNS

In order to get further insights into the attention-driven inference, we visualize the attention maps proposed in this work. In Figure 6, we show the layer connectivity in the generative network as explained in section 2.3 for some test images that belong to different classes. For the same images, in Figure 7 we illustrate the depth-wise attention scores in the inference network we describe in section 2.4, and explain in greater detail in section A.1. For both cases, we chose to visualize the layer which has the largest receptive field, i.e, the last layer for $L = 16$ which is right before the conditional likelihood $p(\boldsymbol{x} \mid \boldsymbol{z}_L)$ in the decoder, and the first layer which has access to all hidden data-dependent feature maps of the bottom-up pass in the encoder.

This analysis validates the hypothesis made in this work and suggests the following conclusions:

- For all pixels, the layer relies greatly on the immediately above layer when forming the prior distribution, as shown in Figure 6.
- Different latent pixels of the same image can discover related latent clues that lie in different layers.
- Similarly, the connectivity patterns in the generative network vary significantly across images.
- Although the context $\boldsymbol{c}_{14}^{p}$ already carries latent information from all previous layers, the decoder does not depend solely on that. Instead, it chooses to activate direct connections with layers far beyond in the hierarchy. This reinforces our hypothesis that the effect of latent variables in early layers may be obfuscated by more recent latent information leading in practice to non fully factorized prior distributions: $p(\boldsymbol{z}) = \prod_l p(\boldsymbol{z}_l \mid \boldsymbol{z}_{<l})$.
- The inference attention scores in Figure 7 are more balanced across layers compared to the generative attention scores in Figure 6.
- Mitigation of posterior collapse as described in D is accomplished in a dual fashion. First, all layers in the bottom-up pass of the inference network can directly interact with the initial data feature maps of the last layers in the hierarchy, as shown in Figure 7. Second, early latent variables of the top layers condition the prior distribution of the latent variables in layers with closer affinity to the conditional data distribution in the decoder, as shown in Figure 6.

In Figures 8, 9, 10, we plot the attention scores among pixels that lie in the same layer as discussed in section 2.5 and A.2. The attention scores pertain to the non-local block of the last convolution of the last residual cell, as depicted in Figure 3, of the ResNet module in the generative portion of each layer.

We derive the following observations:

- The connectivity patterns in all layers are highly structured. This implies that the model could be benefited by sparse attention factorizations such as those proposed by Child et al. (2019), that rely on a similar assumption.
- The attention schemes vary widely across layers. For example, the activation map is more scattered in the first layer of Figure 8 compared to that of the last layer of Figure 10 which is sparser and gathered in periodic spatial locations. This may align with the functionality of the latent information as discussed in current research (Child, 2020), according to which top layers capture global, high-level features while lower layers are responsible for capturing finer details.
- Different images exhibit different spatial attention maps. The differences are more pronounced in the middle layer, as depicted in Figure 9.

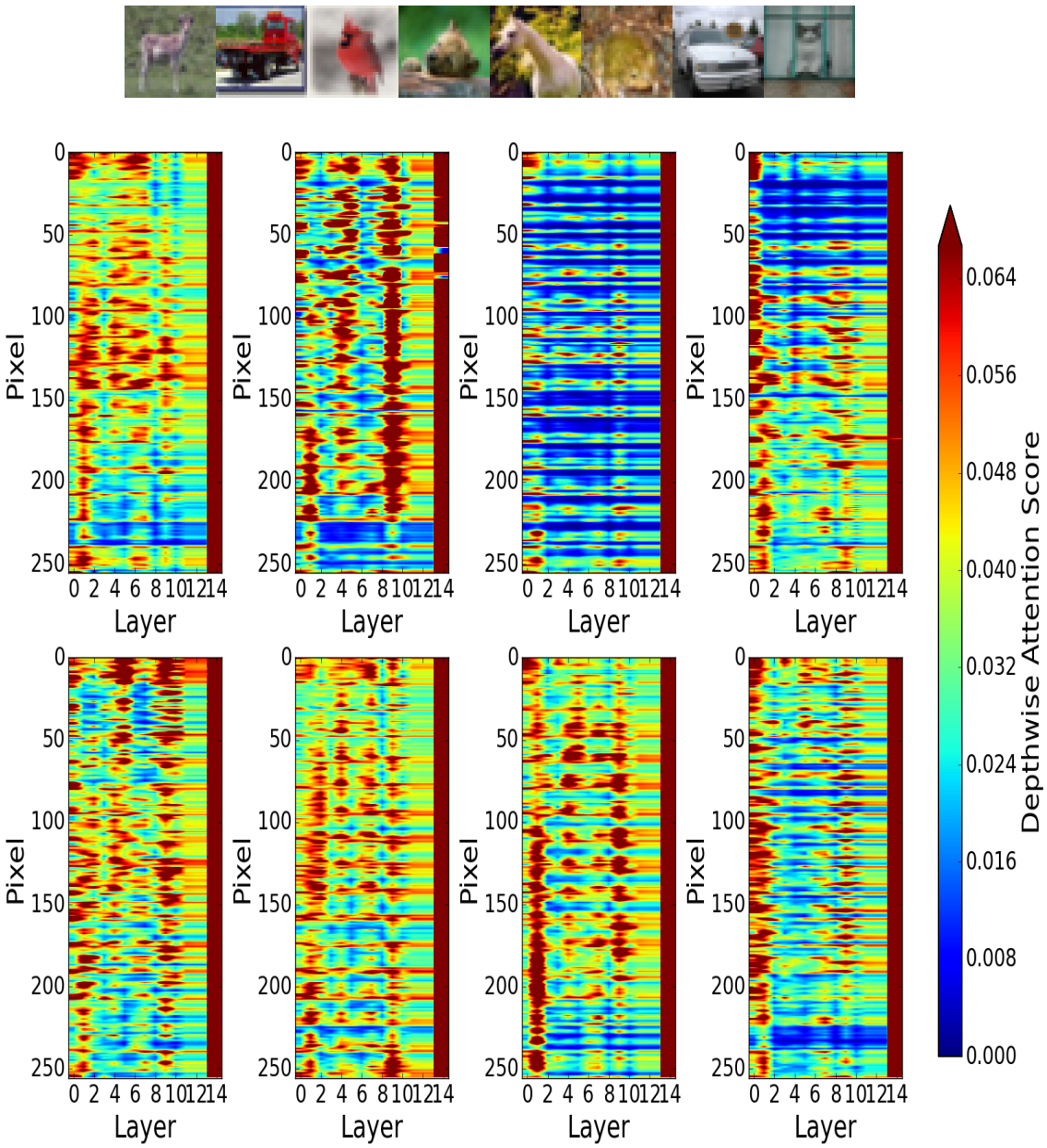


Figure 6: **Depth-wise generative attention scores**. Connectivity map at pixel granularity between the last (16th) layer and the preceding layers in the hierarchy in the generative network using the proposed depth-wise attention.

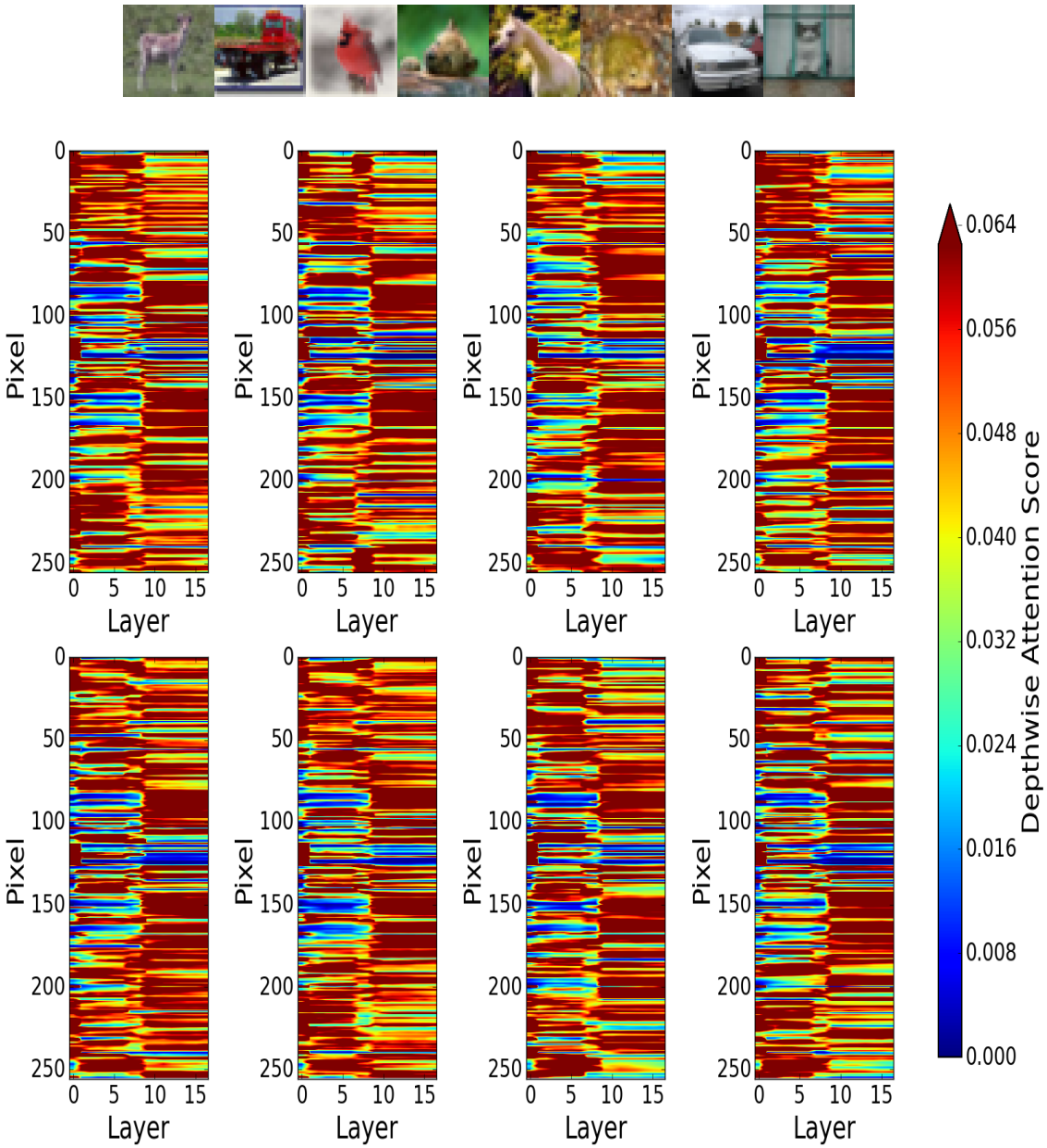


Figure 7: **Depth-wise inference attention scores**. Connectivity map at pixel granularity between the first layer and the subsequent layers in the inference network as dictated by the generative network using the proposed depth-wise attention.

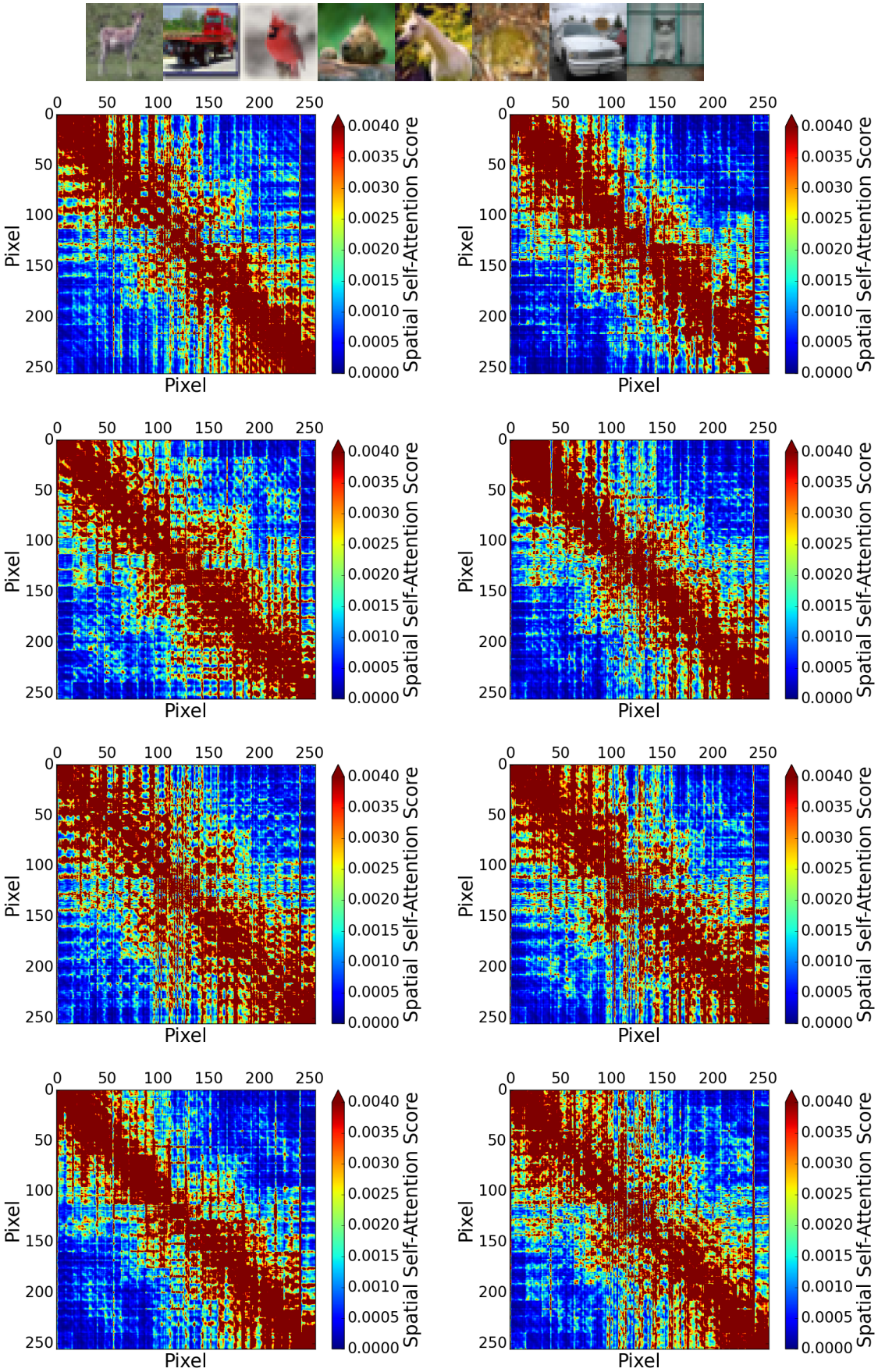


Figure 8: **Spatial attention maps.** Pixel connectivity at the final latent feature map of the first layer in the generative network.

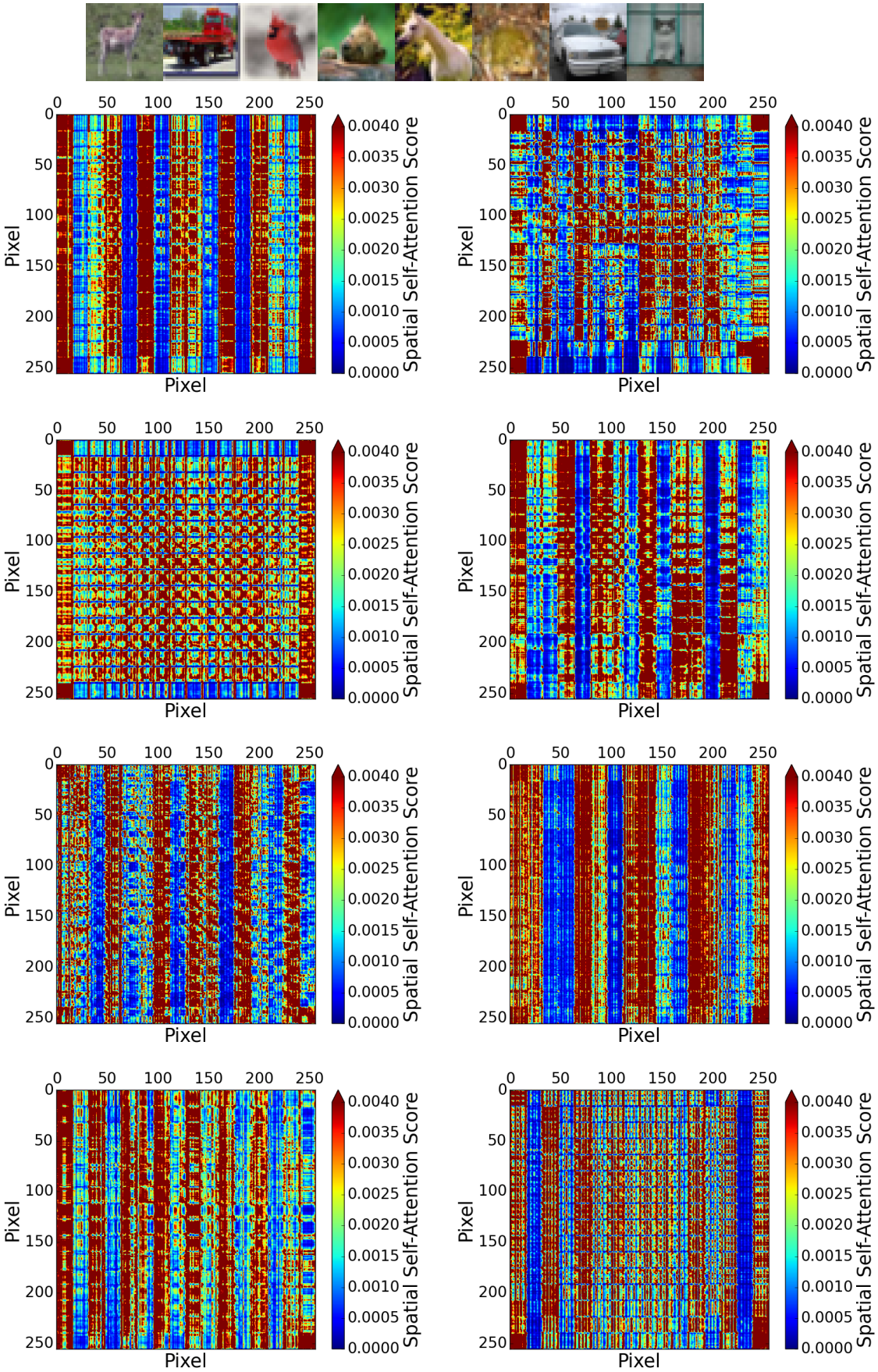


Figure 9: **Spatial attention maps.** Pixel connectivity at the final latent feature map of the middle (8th) layer in the generative network.

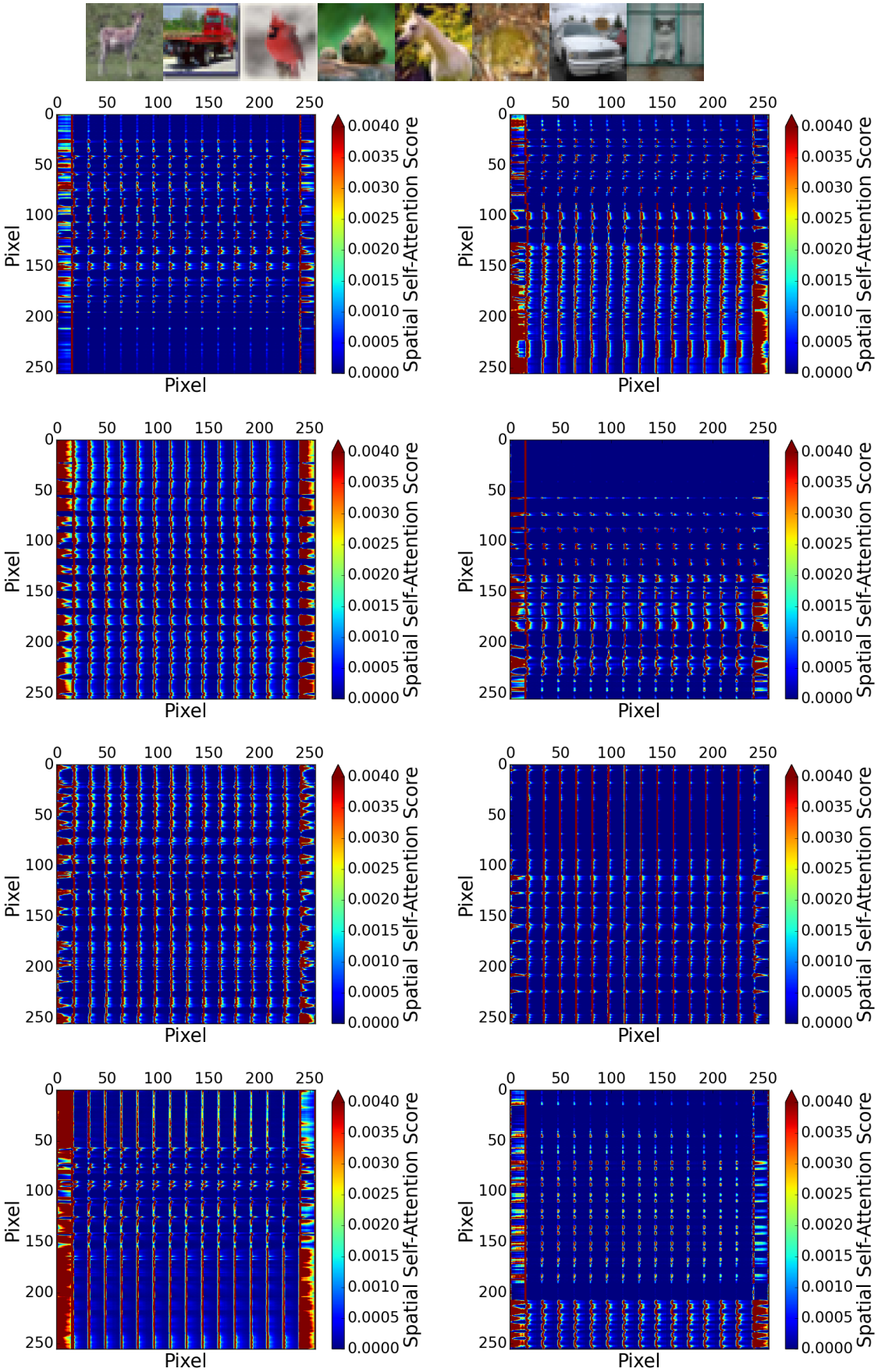
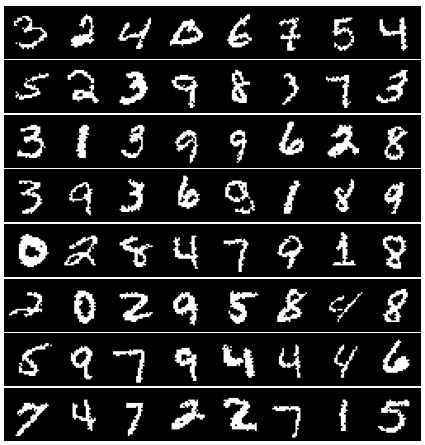


Figure 10: **Spatial attention maps.** Pixel connectivity at the final latent feature map of the last (16th) layer in the generative network.

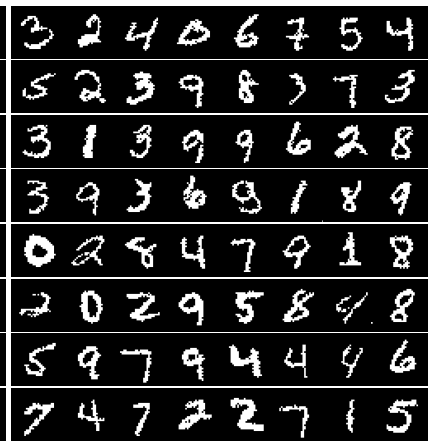
F RECONSTRUCTIONS AND FANTASY SAMPLES GENERATED FROM OUR MODEL

In this section, we qualitatively assess the Attentive VAE presented in the paper. Figures 11b and 12b illustrate reconstructed MNIST and CIFAR-10 test images from the inferred latent variables.

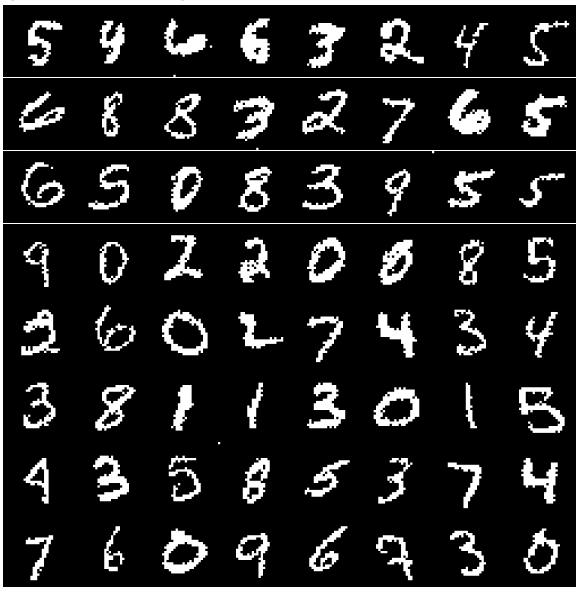
Figures 11c and 12c show novel MNIST and CIFAR-10 images sampled by the generative model.



(a) Original MNIST images.

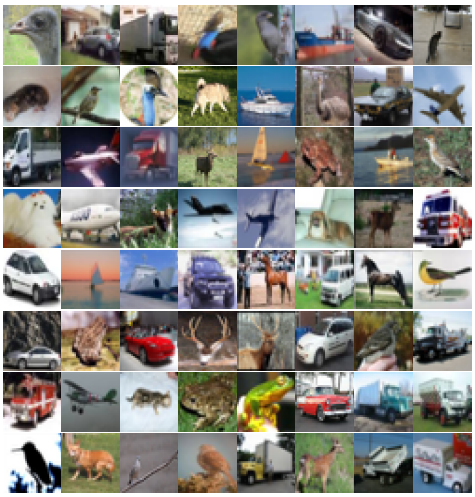


(b) Reconstructed MNIST images.

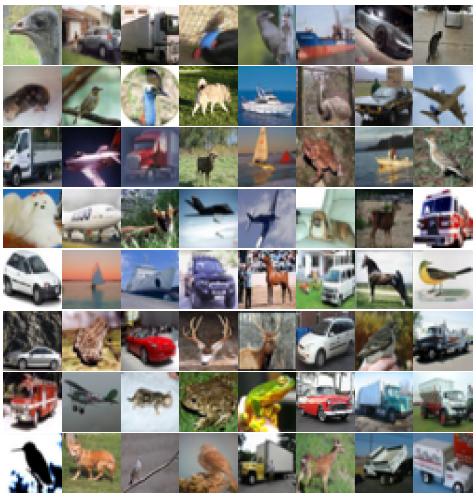


(c) Uncurated generated MNIST images.

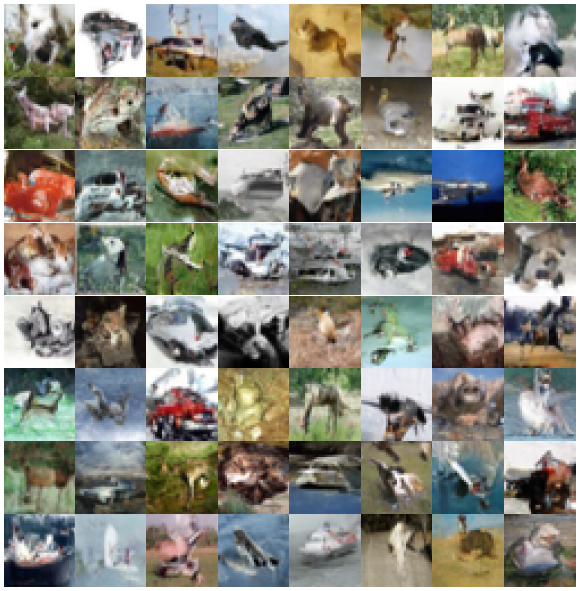
Figure 11: Qualitative evaluation on MNIST images.



(a) Original CIFAR-10 images.



(b) Reconstructed CIFAR-10 images.



(c) Uncurated generated CIFAR-10 images.

Figure 12: Qualitative evaluation on CIFAR-10 images.