Poisson Variational Autoencoder

Hadi Vafaii[1]
vafaii@berkeley.edu

Dekel Galor[1]
galor@berkeley.edu

Jacob L. Yates[1]
yates@berkeley.edu

[1]UC Berkeley

Abstract

Variational autoencoders (VAEs) employ Bayesian inference to interpret sensory inputs, mirroring processes that occur in primate vision across both ventral [1] and dorsal [2] pathways. Despite their success, traditional VAEs rely on continuous latent variables, which deviates sharply from the discrete nature of biological neurons. Here, we developed the Poisson VAE ($\mathcal{P}$-VAE), a novel architecture that combines principles of predictive coding with a VAE that encodes inputs into discrete spike counts. Combining Poisson-distributed latent variables with predictive coding introduces a metabolic cost term in the model loss function, suggesting a relationship with sparse coding which we verify empirically. Additionally, we analyze the geometry of learned representations, contrasting the $\mathcal{P}$-VAE to alternative VAE models. We find that the $\mathcal{P}$-VAE encodes its inputs in relatively higher dimensions, facilitating linear separability of categories in a downstream classification task with a much better ($5\times$) sample efficiency. Our work provides an interpretable computational framework to study brain-like sensory processing and paves the way for a deeper understanding of perception as an inferential process.
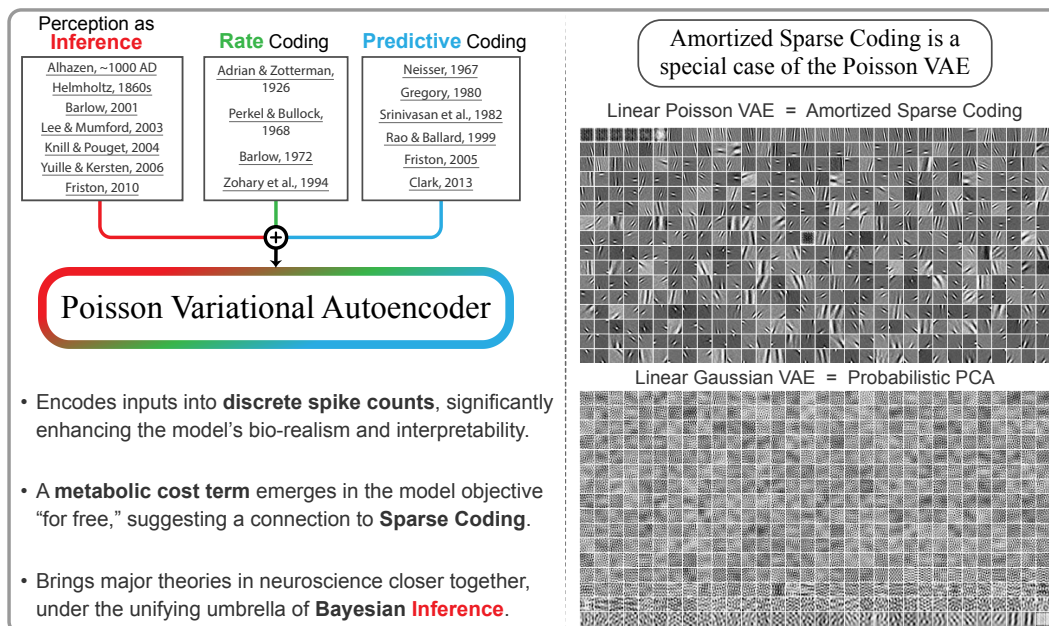
Figure 1: Graphical abstract. Introducing the Poisson Variational Autoencoder ($\mathcal{P}$-VAE), which draws on key concepts in neuroscience. When trained on natural image patches, $\mathcal{P}$-VAE with a linear decoder develops Gabor-like feature selectivity, reminiscent of Sparse Coding [3]. In sharp contrast, the standard Gaussian VAE learns the principal components [4]. Our code, data, and model checkpoints are available at this repository: https://github.com/hadivafaii/PoissonVAE

1 Introduction

The study of artificial neural networks (ANN) and neuroscience has always been closely linked, driving advancements in both fields [5–10]. Despite the close proximity of the two fields, most ANN models deviate substantially from biological brains [11, 12]. A major challenge is designing models that not only perform well computationally but also exhibit "brain-like" structure and function. This is seen both as a goal for improving ANNs [13–15], and better understanding biological brains [8, 9, 16–19], which has recently been referred to as the *neuroconnectionist* research programme [20].

Drawing from neuroscience, a major guiding idea is that perception is a process of inference [21, 22], where the brain constructs a representation of the external world by inferring the causes of sensory inputs [23–26]. This concept is mirrored in "generative AI" where models learn the generative process underlying their inputs [27–29]. However, in this vein, there is a tension between small well-understood models that are directly inspired by cortex, such as sparse coding [3] and predictive coding [30], and deep generative models that perform well [31–34].

The variational autoencoder (VAE; [35, 36]) model family is a promising candidate for neuroconnectionist goals for multiple reasons. First, VAEs learn probabilistic generative models of their inputs and are grounded in Bayesian probability theory, providing a solid theoretical foundation that directly incorporates the concept of perceptual inference [10, 22]. Second, the VAE model family, specifically hierarchical VAEs, is broad with other generative models, such as diffusion models, understood as special cases of hierarchical VAEs [37–39]. Finally, VAEs learn representations that are similar to cortex [1, 2, 40], exhibit cortex-like topographic organization [41, 42], and make perceptual errors that mimic those of humans [43], indicating a significant degree of neural, organizational, and psychophysical alignment with the brain.

However, standard VAEs diverge from brains in the way they encode information. Biological neurons fire all-or-none action potentials [44], and are thought to represent information via firing rate [45–49]. These firing rates must be positive and generate discrete "spike" counts, which exhibit conditionally Poisson-like statistics in small counting windows [49–51]. In contrast, VAEs are typically parameterized with real-valued, continuous, Gaussian distributions [52].

Contributions. In this work, we address this discrepancy by introducing the Poisson Variational Autoencoder ($\mathcal{P}$-VAE), a novel architecture that combines perceptual inference with two other inspirations from neuroscience (Fig. 1). First, that information is encoded in the rates of discrete spike counts, which are approximately Poisson-distributed on short time intervals. And second, that feedforward connections encode deviations from expectations contained in feedback connections (Fig. 2a; [30, 53]). We introduce a reparameterization trick for Poisson samples (Algorithm 1), and derive the evidence lower bound (ELBO) objective for the $\mathcal{P}$-VAE (eq. (3)). Overall, we believe $\mathcal{P}$-VAE introduces a promising new model at the intersection of computational neuroscience and machine learning that offers several appealing features over existing VAE architectures:

- The $\mathcal{P}$-VAE loss derivation (eq. (3)) naturally results in a metabolic cost term that penalizes high firing rates, such that $\mathcal{P}$-VAE with a linear decoder implements amortized sparse coding (Fig. 2b). We validate this prediction empirically.
- $\mathcal{P}$-VAE largely avoids the prevalent posterior collapse issue, maintaining many more active latents compared to alternative VAE models (Table 1), especially the continuous ones.
- $\mathcal{P}$-VAE encodes its inputs in relatively higher dimensions, facilitating linear separability of categories in a downstream classification task with a much better ($5\times$) sample efficiency.

We evaluate these results on two natural image datasets and MNIST. The $\mathcal{P}$-VAE paves the way for the future development of interpretable hierarchical models that perform "brain-like" inference.

2 Background & Related work

Perception as inference: connections to neuroscience and machine learning. A centuries-old idea [21, 22], "perception as inference" argues that coherent perception of the world results from the unconscious inference over the causes of the senses. In other words, the brain learns a generative model of the sensory inputs. This has led to fruitful theoretical work in neuroscience [23, 54–56] and machine learning [57, 58], including VAEs [52]. See Marino [10] for a review.

Efficient, predictive, and sparse coding. Another longstanding idea in neuroscience is that brains are adapted to the statistics of the environment. Efficient coding states that brains represent as much information about the environment as possible while minimizing neural resource use [59, 60].

Predictive coding [30, 61, 62] postulates that the brain generates a statistical prediction of its inputs, with feedforward networks carrying only the prediction errors or unexplained information [63]. More recently, ANNs based on predictive coding have been shown to capture a wide range of phenomena in biological neurons across the visual system [64, 65]. More broadly, prediction in time has emerged as an objective that lends itself to brain-like representations [66, 67].

Sparse coding (SC) is directly inspired by efficient coding, aiming to explain inputs as sparsely as possible [47, 68]. SC was the first unsupervised model to learn representations closely resembling the receptive fields of V1 neurons [3] and predicts an array of empirical features of neural activity [69–79]. SC is formalized with a generative model where neural activations z are sampled from a sparsity-inducing prior, $z \sim p(z)$, and the input image x is reconstructed as a linear combination of basis vectors Φ, plus additive Gaussian noise, $\hat{x} = \Phi z + \varepsilon$. The SC loss is as follows:

$$\mathcal{L}_{\text{SparseCoding}}(x; \Phi, z) = \|x - \Phi z\|_2^2 + \beta \|z\|_1. \tag{1}$$

Commonly used algorithms for sparse coding include the locally competitive algorithm (LCA; [80]), which is a biologically plausible algorithm to optimize eq. (1), and iterative shrinkage-thresholding algorithm (ISTA; [81, 82]), which has shown robust performance in learning sparse codes given a fixed dictionary Φ.

VAE objective. VAEs define a probabilistic generative model $p(x, z)$, where x denotes the observed data and z are some latent variables. The generative process samples z from a prior distribution $p(z)$ and then generates the observed data x from the conditional distribution $p_\theta(x|z)$, also known as the "decoder". The "encoder", $q_\phi(z|x)$, performs approximate inference on the inputs. Model parameters are learned by maximizing the evidence lower bound (ELBO) objective, which is derived from variational inference (see appendix B for the full set of derivations). The ELBO is given by:

$$\log p(x) \geq \mathbb{E}_{q_\phi(z|x)}\Big[\log p_\theta(x|z)\Big] - \mathcal{D}_{\text{KL}}\Big(q_\phi(z|x) \,\|\, p(z)\Big) = \mathcal{L}_{\text{VAE}}(x; \theta, \phi). \tag{2}$$

The first term captures the reconstruction performance of the decoder, and the second term, the "KL term," captures the divergence of the approximate posterior from the prior.

The specific form of these distributions is up to the practitioner. In standard VAEs, factorized Gaussians are typically used: $q = \mathcal{N}(z; \mu(x), \sigma^2(x))$ and $p = \mathcal{N}(z; 0, 1)$. The likelihood, $p_\theta(x|z)$, is also typically modeled as a Gaussian conditioned on a parameterized neural network $\text{dec}_\theta(z)$.

Amortized inference in VAEs. A major contribution of VAEs is the idea of amortizing inference over the latents z with a black box ANN [83, 84]. "Amortized" inference borrows a term from finance to capture the idea of spreading out costs—here, the cost of performing inference over multiple samples. In amortized inference, a neural network learns (during training) how to map a data sample to a distribution over latent variables given the sample. The cost is paid during training, but the trained model can then be used to perform inference on future samples efficiently. It has been argued that the brain performs amortized inference for computational efficiency [85].

VAEs connection to biology. VAEs have been shown to contain individual latents that resemble neurons, capturing a wide range of the phenomena observed in visual cortical areas [40] and human perceptual judgments [43]. Like many other ANN models [86, 87], VAEs have been found to learn representations that are predictive of single-neuron activity in both the ventral [1] and dorsal [2] streams. However, unlike most ANNs, the mapping from certain VAEs to neural activity is incredibly sparse, even one-to-one in some cases [1, 2].

Discrete VAEs. VAEs with discrete latent spaces, such as VQ-VAE [88] and Categorical VAE [89], are designed to capture complex data structures by mapping inputs to a finite set of latent variables. Unlike traditional VAEs that use continuous latent spaces, these models leverage discrete representations to enhance interpretability and can yield high performance with lower capacity [90].

Algorithm 1 Reparameterized sampling (rsample) for Poisson distribution.

Input:
$\boldsymbol{\lambda} \in \mathbb{R}_{>0}^{B \times K}$ # rate parameter; B, batch size; K, latent dimensionality
n_exp # number of exponential samples to generate
temperature   # controls the sharpness of the thresholding

1: **procedure** RSAMPLE($\boldsymbol{\lambda}, n\_\text{exp}, \text{temperature}$)
2:     Exp $\leftarrow$ Exponential($\boldsymbol{\lambda}$)   ▷ create exponential distribution
3:     $\Delta t \leftarrow$ Exp.rsample$((n\_\text{exp},))$ ▷ sample inter-event times, $\Delta t : [n\_\text{exp} \times B \times K]$
4:     times $\leftarrow$ **cumsum**($\Delta t, \text{dim}=0$)   ▷ compute arrival times, same shape as Δt
5:     indicator $\leftarrow$ sigmoid $\left(\frac{1-\text{times}}{\text{temperature}} \right)$ ▷ soft indicator for events within unit time
6:     $\boldsymbol{z} \leftarrow$ sum(indicator, dim=0)   ▷ event counts, or number of spikes, $\boldsymbol{z} : [B \times K]$
7:     **return** $\boldsymbol{z}$
8: **end procedure**

VAEs connection to sparse coding.   Previous work has attempted to connect sparse coding and VAEs directly [91–93], with each approaching the problem differently. Geadah et al. [91] introduced sparsity-inducing priors (such as Laplace or Cauchy) and a linear decoder with an overcomplete latent space. Tonolini et al. [92] introduced a spike and slab prior into a modified ELBO, and Xiao et al. [93] added a sparse coding layer learned by ISTA to the latent space of a VQ-VAE. Notably, none of the three ended up minimizing the sparse coding loss. Two of the three maintain the linear generative model with an overcomplete latent space, but the ELBO in both requires an additional approximation step for the KL term [91, 92].

3   Introducing the Poisson Variational Autoencoder ($\mathcal{P}$-VAE)

Our main contribution is integrating Poisson-distributed latents into VAEs, where both the approximate posterior and the prior are parameterized as Poisson distributions. Critically, the latents $\boldsymbol{z}$ are no longer continuous variables, but rather they are discrete spike counts. To perform inference over discrete latents, we introduce a Poisson reparameterization trick. We then derive the KL term and obtain the full $\mathcal{P}$-VAE objective.

Poisson reparameterization trick. For a homogeneous Poisson process [94–96], given a window size $\Delta t = 1$, and rate λ, we can generate Poisson distributed counts by drawing randomly distributed wait-times from an exponential distribution with mean $1/\lambda$ and counting all events where the cumulative time is less than 1. Because the exponential distribution is trivially reparameterized [35], and PyTorch contains an implementation [97], we need only to approximate the hard threshold for comparing cumulative wait times with the window size. We accomplish this by replacing the indicator function with a sigmoid as in refs. [89, 98].
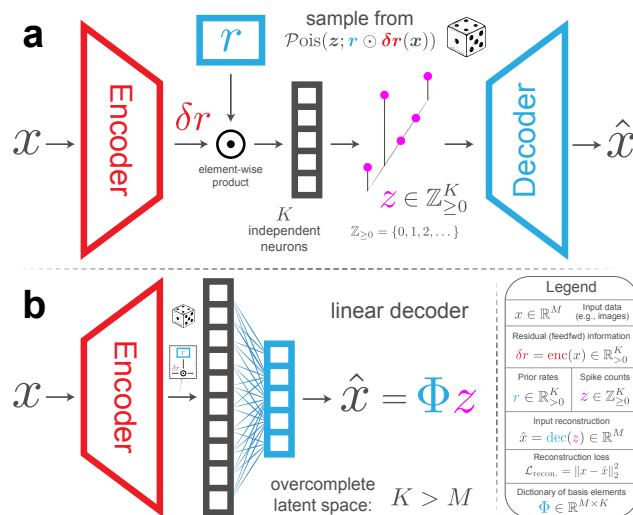

Figure 2: **(a)** Model architecture. Colored shapes indicate learnable model parameters, including the prior firing rates, $\boldsymbol{r}$. We color code the model's inference and generative components using red and blue, respectively. The $\mathcal{P}$-VAE encodes its inputs in discrete spike counts, $\boldsymbol{z}$, significantly enhancing its biological realism. **(b)** "Amortized Sparse Coding" is a special case within the $\mathcal{P}$-VAE model family: it's a $\mathcal{P}$-VAE with a linear decoder and an overcomplete latent space.

Algorithm 1 demonstrates the steps: Given a matrix of rates $\boldsymbol{\lambda}$, sample n_exp wait times $t_1, t_2, ...t_{n\_exp}$ for each element of $\boldsymbol{\lambda}$ by sampling from an exponential distribution with mean $1/\boldsymbol{\lambda}$. We then calculate the cumulative event times $S(n\_exp) = \sum_{j=1}^{n\_exp} t_j$, pass them through a sigmoid $\sigma(\frac{1-S}{\text{temperature}})$, and sum over samples to get event counts, $\boldsymbol{z}$. The temperature controls the sharpness of the thresholding. We adaptively scale the number of samples, n_exp, by keeping track of the maximum rate in each batch, $\lambda_{\max}$, and then use the inverse cumulative density function (cdf) for Poisson to find the number of samples, n_exp, such that $\text{cdf}(n\_exp; \lambda_{\max}) = 0.99999$.

At non-zero temperatures, our parameterization algorithm provides a continuous relaxation of the Poisson distribution. Figure 3 shows histograms of samples drawn using Algorithm 1 for rate $\lambda = 1$ and temperatures $T = 1.0, 0.1, 0.01$, and 0. The latter case ($T = 0$, true Poisson) is equivalent to `torch.poisson()`.
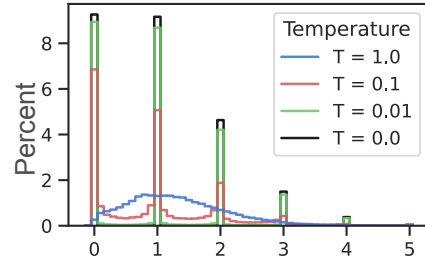

Figure 3: Relaxed Poisson distribution. Samples are drawn using Algorithm 1 for $\lambda = 1$. At non-zero temperatures, samples are non-integer, but approach the true Poisson distribution as $T \to 0$.

$\mathcal{P}$-VAE architecture and residual parameterization. The architecture of $\mathcal{P}$-VAE captures the interactions between feedforward and feedback connections that are present in all visual cortical areas [99, 100]. Feedforward areas carry sensory information and feedback connections are thought to carry modulatory signals such as attention [53] or prediction [30], which interact multiplicatively with feedforward inputs [53, 101].

$\mathcal{P}$-VAE embodies this idea by having the posterior rates depend on the prior, such that $\boldsymbol{r}_{\text{prior}} = \boldsymbol{r}$ and $\boldsymbol{r}_{\text{post.}} = \boldsymbol{r} \odot \boldsymbol{\delta r}(\boldsymbol{x})$, where $\odot$ is the Hadamard (element-wise) product. The prior rates, $\boldsymbol{r} \in \mathbb{R}^K$, are learnable parameters that capture expectations about the statistics of the input. The encoder outputs, $\boldsymbol{\delta r}(\boldsymbol{x}) \in \mathbb{R}^K$, capture *deviations* from the prior. Thus, $\mathcal{P}$-VAE models the interaction between prior expectations, and deviations from them, in a multiplicative and symmetric way. This results in a posterior, $q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r} \odot \boldsymbol{\delta r}(\boldsymbol{x}))$, and prior, $p(\boldsymbol{z}) = \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r})$, where $\boldsymbol{z}$ is the spike count variable and $\mathcal{P}\text{ois}(z; \lambda) = \lambda^z e^{-\lambda}/z!$ is the Poisson distribution. Notably, this multiplicative relationship is maximally general, as any pair of positive variables, $\boldsymbol{r}_{\text{prior}}$, and $\boldsymbol{r}_{\text{post.}}$ can be expressed as a base variable, $\boldsymbol{r} := \boldsymbol{r}_{\text{prior}}$, multiplied by their relative ratio, $\boldsymbol{\delta r} := \boldsymbol{r}_{\text{post.}}/\boldsymbol{r}$. See Fig. 2a.

$\mathcal{P}$-VAE loss function. For a comprehensive derivation of the $\mathcal{P}$-VAE objective, see appendix B. Here, we report the final result:

$$\mathcal{L}_{\text{PVAE}} = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r} \odot \boldsymbol{\delta r})} \left[\|\boldsymbol{x} - \text{dec}(\boldsymbol{z})\|_2^2 \right] + \sum_{i=1}^{K} r_i f(\delta r_i), \quad (3)$$

where $\text{dec}(\cdot)$ is the decoder neural network, and $f(y) := 1 - y + y \log y$ (see supplementary Fig. 6).

$\mathcal{P}$-VAE relationship to sparse coding. The KL term in eq. (3) penalizes firing rates. Both $\boldsymbol{r}$ and $\boldsymbol{\delta r}$ are positive by definition, and $f(y) \geq 0$, strongly resembling the sparsity penalty in Olshausen and Field [3]. To make this connection more explicit, we make two additional assumptions (Fig. 2b):

1. The decoder is a linear generative model: $\hat{\boldsymbol{x}} = \boldsymbol{\Phi}\boldsymbol{z}$, with $\boldsymbol{x} \in \mathbb{R}^M$ and $\boldsymbol{\Phi} \in \mathbb{R}^{M \times K}$.
2. The latent space is overcomplete: $K > M$.

Because both $\mathbb{E}_{\boldsymbol{z} \sim \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{\lambda})}[z_i]$ and $\mathbb{E}_{\boldsymbol{z} \sim \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{\lambda})}[z_i z_j]$ have closed-form solutions (eq. (22)), the reconstruction term in eq. (3) can be computed analytically for a linear decoder, resulting in:

$$\mathcal{L}_{\text{SC-PVAE}}(\boldsymbol{x}; \boldsymbol{\delta r}, \boldsymbol{r}, \boldsymbol{\Phi}) = \|\boldsymbol{x} - \boldsymbol{\Phi}\boldsymbol{\lambda}\|_2^2 + \boldsymbol{\lambda}^T \text{diag}(\boldsymbol{\Phi}^T \boldsymbol{\Phi}) + \beta \sum_{i=1}^{K} r_i f(\delta r_i). \quad (4)$$

where $\boldsymbol{\lambda} = \boldsymbol{r} \odot \boldsymbol{\delta r}(\boldsymbol{x})$ are the posterior firing rates, $f(y)$ is defined as above, and β is a hyperparameter that scales the contribution of the KL term [102], and changes the sparsity penalty for the $\mathcal{P}$-VAE.

Table 1: Models considered in this paper.

Discrete		Continuous	
Poisson VAE ($\mathcal{P}$-VAE)	Categorical VAE ($\mathcal{C}$-VAE; [89, 98])	Gaussian VAE ($\mathcal{G}$-VAE; [35, 36])	Laplace VAE ($\mathcal{L}$-VAE; [40, 91])

The relationship between the linear $\mathcal{P}$-VAE loss (eq. (4)) and the sparse coding loss (eq. (1)) can now be seen. Both contain a term that minimizes the squared error of the reconstruction and a term (two terms for $\mathcal{P}$-VAE) that penalizes non-zero firing rates. Unlike prior work that directly implemented amortized sparse coding [91, 92], here the activity penalty naturally emerges from the derivations, and the only additional assumption was an overcomplete linear generative model. The inference is accomplished using a parameterized feed-forward neural network, $\delta r(x)$, thus, it is amortized [83]. We call this specific case of $\mathcal{P}$-VAE "Amortized Sparse Coding" (Fig. 2b).

Note that a closed-form derivation of the reconstruction term is possible for any VAE with a linear decoder and a generating distribution that has a mean and variance (see eq. (21)).

This closed-form expression of the loss given a linear decoder is useful because we can see how different parameters contribute to the loss. Furthermore, we can compute gradients of the whole loss exactly, and use this to evaluate our Poisson reparameterization.

4 Experiments

To evaluate the $\mathcal{P}$-VAE, we perform three sets of experiments. First, we utilize the theoretical results for a linear decoder (eqs. (4) and (21)) to test the effectiveness of our reparameterization algorithm. We compare to alternative VAE models with established reparameterization tricks (e.g., Gaussian).

Second, to confirm $\mathcal{P}$-VAE with a linear decoder not only resembles amortized sparse coding but practically performs like sparse coding, we compare to standard and well-established sparse coding algorithms such as the locally competitive algorithm (LCA; [80]) and the widely-used iterative shrinkage-thresholding algorithm (ISTA; [81, 82]) to see if $\mathcal{P}$-VAE reproduces their results.

Third, we test the $\mathcal{P}$-VAE in a generic representation learning context and evaluate the geometry of learned representations for downstream tasks. For these experiments, both the encoder and decoder's architecture is a ResNet (see appendix C for full architecture and training details).

Architecture notation. We experimented with both convolutional and linear architectures. We highlight the encoder and decoder networks using red and blue, respectively. We use the ⟨enc|dec⟩ convention to clearly specify which architecture type was used. For example, ⟨conv|lin⟩ represents a model with a convolutional encoder and a linear decoder. Using this notation, we note that ⟨lin|lin⟩ and ⟨conv|lin⟩ architectures were used for the first and second sets of experiments, while ⟨conv|conv⟩ architectures were employed for the third.

Alternative models. We compare $\mathcal{P}$-VAE to both discrete and continuous VAEs (Table 1). Other than the traditional Gaussian, we compare to Laplace-distributed VAEs because previous work found the Laplace distribution supported robust sparse representations [40, 91]. Additionally, we compare to Categorical VAEs, trained using the Gumbel-Softmax trick [89, 98]. We use PyTorch's implementation which is based on Maddison et al. [98].

Finally, we test models where Gaussian latents are passed through an activation function before passing to the decoder. We call these models $\mathcal{G}$-VAE$_{+\text{act}}$, where $\text{act} \in \{\text{relu}, \exp\}$, capturing other families of distributions (truncated Gaussian and log-normal). We include these to test the hypothesis that positive constraints (and not discrete latents) are the key contribution of Poisson [103].

Datasets. For sparse coding results, we use 101 natural images from the van Hateren dataset [104]. We tile the images to extract 16×16 patches and apply whitening and contrast normalization, as is typically done in sparse coding literature [3, 105]. To test the generalizability of our sparse coding results, we repeat these steps on CIFAR10 [106], a dataset we call CIFAR$_{16\times16}$. For the general representation learning results, we use MNIST. See appendix C for additional details.

Table 2: Reparameterized gradient estimators perform comparably to exact ones across datasets and encoder architectures (linear vs. convolutional). Exact gradients are only computable for linear decoders (see eqs. (21), (23) and (24)). Values represent percent drop in validation loss (lower is better), shown as mean±99% confidence interval calculated from $n = 5$ random initializations. The best-performing case was selected as the single best random seed for models of the same architecture and dataset across gradient methods (1 out of: 15 for $\mathcal{P}$-VAE, 10 for $\mathcal{G}$-VAE). See supplementary Fig. 7 for a visualization of the same data presented in this table. For actual loss values, see supplementary Table 5. EX: exact; MC: Monte Carlo; ST: straight-through [107].

Model		van Hateren		CIFAR$_{16\times16}$		MNIST	
-------	---	$\langle$lin$\vert$lin$\rangle$	$\langle$conv$\vert$lin$\rangle$	$\langle$lin$\vert$lin$\rangle$	$\langle$conv$\vert$lin$\rangle$	$\langle$lin$\vert$lin$\rangle$	$\langle$conv$\vert$lin$\rangle$
	EX	$0.6_{\pm.5}$	$0.1_{\pm.1}$	$0.0_{\pm.1}$	$0.0_{\pm.0}$	$0.1_{\pm.1}$	$0.5_{\pm.6}$
$\mathcal{P}$-VAE	MC	$0.0_{\pm.1}$	$0.7_{\pm.1}$	$0.2_{\pm.0}$	$0.5_{\pm.1}$	$0.7_{\pm.4}$	$0.9_{\pm.5}$
	ST	$7.3_{\pm.1}$	$10.5_{\pm.1}$	$9.1_{\pm.1}$	$12.5_{\pm.1}$	$8.1_{\pm.3}$	$11.8_{\pm.2}$
$\mathcal{G}$-VAE	EX	$0.1_{\pm.1}$	$0.0_{\pm.0}$	$0.0_{\pm.1}$	$0.0_{\pm.0}$	$0.1_{\pm.2}$	$0.1_{\pm.2}$
	MC	$0.1_{\pm.1}$	$0.0_{\pm.0}$	$0.1_{\pm.1}$	$0.0_{\pm.0}$	$0.4_{\pm.1}$	$0.3_{\pm.1}$

Statistical tests. In the VAE literature, it is known that random seeds can have a large effect compared to architecture or regularization [108]. Therefore, we train each configuration using 5 different random initializations. We report 99% confidence intervals throughout, and perform paired t-tests, reporting significance for $p < 0.01$ (FDR corrected using the Benjamini-Hochberg method).

Evaluating the Poisson reparameterization algorithm. $\mathcal{P}$-VAE with a linear decoder has a closed form solution (eq. (4)), which lets us evaluate how well our reparameterized gradients perform compared to the exact ones. We compare our results to the gold-standard Gaussian (Table 2), as well as Categorical and Laplace VAEs (supplementary Table 5). In Table 2, we report the percent performance drop relative to the best fit, enabling meaningful comparisons across architectures and datasets. Monte Carlo sampling with Poisson reparameterization closely matches exact inference just like established methods for Gaussian and Laplace. In contrast, the straight-through (ST; [107]) estimator performs poorly (Table 2; see also supplementary Fig. 7).

Annealing the temperature. The temperature parameter (T) is a crucial hyperparameter in our Poisson reparameterization trick (Algorithm 1). To assess its impact, we followed standard practice [89] and annealed T during the first half of training, starting from a large value ($T_{\text{start}} = 1$) and gradually decreasing it to a small value ($T_{\text{final}} = 0.05$ in the main paper). Figure 9 shows the performance on the van Hateren dataset as a function of various T_{final}, two architectures ($\langle$lin$\vert$lin$\rangle$ and $\langle$conv$\vert$lin$\rangle$), as well as two annealing schedules (linear vs. exponential; see inset). We find that final temperatures $T_{\text{final}} \leq 0.1$ and either annealing strategy work well.

During training, we maintain $T > 0$, which results in continuous (floating) latent variables, z. At test time, we set $T = 0$ to produce genuine integer Poisson samples. Crucially, all reported results use $T = 0$ at test time. We also explored a "hard-forward" scheme during the latter half of training, where T remains nonzero only in the backward pass. This *surrogate gradients* approach provides integer latents in the forward pass but, somewhat unexpectedly, underperformed our "relaxed Poisson" method (Fig. 9). These findings suggest that surrogate gradient methods might benefit from relaxing the hard-forward strategy during training. We believe this observation will be of particular interest to the spiking neural network community, which often relies on surrogate gradients for training.

The $\mathcal{P}$-VAE learns basis vectors similar to those from sparse coding. A major result from sparse coding is that it learns basis vectors (dictionaries) that resemble the "Gabor-like" receptive fields of cortical neurons [3, 109, 110]. Inspecting the dictionaries learned by different models demonstrates this is not trivial (Fig. 4). As expected from theoretical results [4], $\mathcal{G}$-VAE (top left) learn probabilistic PCA, but with many noisy elements. As demonstrated previously [40, 91], $\mathcal{L}$-VAE (lower left) learn Gabor-like elements. However, there are a large number of noisy basis vectors. It is of note that previous work did not show complete dictionaries for their results with Laplace latents [40, 91]. In contrast, $\mathcal{P}$-VAE (top middle) learns Gabor-like filters that cover space, orientation, and spatial
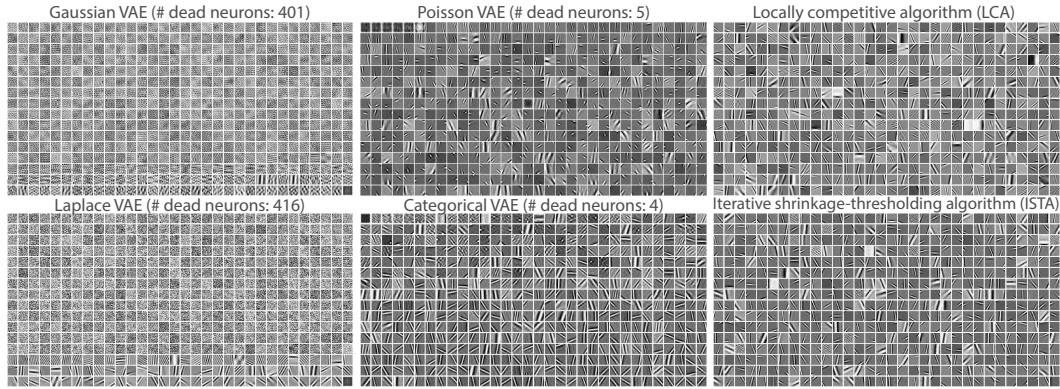
Figure 4: Learned basis elements for various $\langle \texttt{lin}|\texttt{lin}\rangle$ VAEs (first two columns) and standard sparse coding models (last column). There are a total of $K = 512$ elements, each made of $16 \times 16 = 256$ pixels (i.e., $\mathbf{\Phi} \in \mathbb{R}^{256 \times 512}$). Features are ordered from top-left to bottom-right, in ascending order of their associated KL divergence ($\mathcal{P}$-VAE, $\mathcal{G}$-VAE, $\mathcal{L}$-VAE), or the magnitude of posterior logits ($\mathcal{C}$-VAE). The sparse coding results (LCA and ISTA) are ordered randomly.

frequency. The quality is comparable to sparse coding dictionaries learned with LCA/ISTA (top/lower right panels). $\mathcal{C}$-VAE also learns Gabors, although there are significantly more noisy basis elements.

The $\mathcal{P}$-VAE avoids posterior collapse.   A striking feature of Fig. 4 is the sheer number of noisy basis vectors for both continuous VAEs ($\mathcal{G}$-VAE, $\mathcal{L}$-VAE). We suspected this reflected dead neurons with vanishing KL, which is indicative of a collapsed latent dimension that's no longer encoding information. To quantify this, we binned the distribution of KL values and thresholded the resulting distribution at discontinuous points (see supplemental Fig. 10). Table 3 shows the results of this analysis for all VAEs with valid KL terms. Across all datasets, both continuous VAEs suffered from large numbers of dead neurons, whereas $\mathcal{P}$-VAE largely avoided this problem. On both natural image datasets, $\mathcal{P}$-VAE had $\sim 2\%$ dead neurons compared to $\sim 80\%$ for $\mathcal{G}$-VAE and $\mathcal{L}$-VAE. Having a more expressive encoder slightly increases this percentage, but a dramatic difference between $\mathcal{P}$-VAE and continuous VAEs ($\mathcal{G}$-VAE, $\mathcal{L}$-VAE) persists.

The $\mathcal{P}$-VAE learns sparse representations.   To quantify whether $\mathcal{P}$-VAE learns sparse representations, we compared our VAE models to sparse coding trained with LCA and ISTA and quantified the lifetime sparsity [69]. The lifetime sparsity of the j-th latent is:

$$ s_j = \left(1 - \frac{1}{N}\right)^{-1} \left(1 - \frac{1}{N} \frac{\left(\sum_i z_{ij}\right)^2}{\sum_i z_{ij}^2}\right), \tag{5} $$

where N is the number of images, and z_{ij} is sampled from the posterior for the i-th image. Intuitively, $s_j = 1$ whenever neuron j responds to a single stimulus out of the entire set (highly selective). In contrast, $s_j = 0$ whenever the neuron responds equally well to all stimuli indiscriminately.

Fig. 5a shows the reconstruction performance (MSE) compared to lifetime sparsity (s, eq. (5)) for all VAEs. Empty and solid circles represent $\langle \texttt{conv}|\texttt{lin}\rangle$ and $\langle \texttt{lin}|\texttt{lin}\rangle$ architectures, respectively. The $\mathcal{G}$-VAE finds good reconstructions (MSE = 71.49) but with low sparsity ($s = 0.37$). Because the $\mathcal{P}$-VAE KL term explicitly penalizes rate (eq. (3)), we explored different β values for $\mathcal{P}$-VAE with both $\langle \texttt{lin}|\texttt{lin}\rangle$ and $\langle \texttt{conv}|\texttt{lin}\rangle$ architectures (Fig. 5a, blue curves). This maps out rate-distortion curves, enabling us to compare the sparsity levels at which $\mathcal{P}$-VAE matches $\mathcal{G}$-VAE performance.

With a simpler (linear) encoder, $\langle \texttt{lin}|\texttt{lin}\rangle$ $\mathcal{P}$-VAE matches $\langle \texttt{conv}|\texttt{lin}\rangle$ $\mathcal{G}$-VAE performance while achieving $1.7\times$ greater sparsity at $\beta = 0.6$. A $\langle \texttt{conv}|\texttt{lin}\rangle$ $\mathcal{P}$-VAE further increases this gap to $2.4\times$ greater sparsity. Adding a relu activation to $\mathcal{G}$-VAE also increases sparsity ($s = 0.69$). By comparing $\langle \texttt{lin}|\texttt{lin}\rangle$ and $\langle \texttt{conv}|\texttt{lin}\rangle$ $\mathcal{P}$-VAE models, we observe that enhancing encoder complexity for the same $\beta = 1$ (gray arrows) preserves MSE performance while achieving greater sparsity. This highlights how amortization quality can significantly influence rate-distortion curves [33, 111–113].

Table 3: Proportion of active neurons. All models considered in this table had a latent dimensionality of $K = 512$, with either $\langle \texttt{lin}|\texttt{lin}\rangle$ or $\langle \texttt{conv}|\texttt{lin}\rangle$ architectures. See also supplementary Fig. 10.

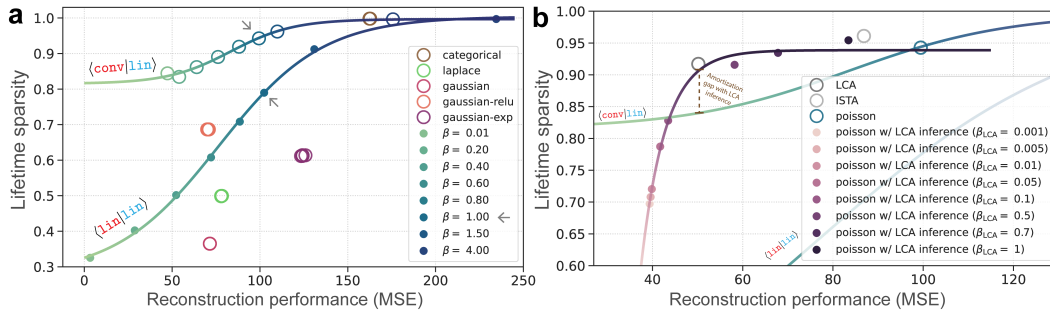| Model	van Hateren		CIFAR$_{16\times16}$		MNIST	
	linear	conv	linear	conv	linear	conv
$\mathcal{P}$-VAE	$\mathbf{0.984}_{\pm.011}$	$\mathbf{0.819}_{\pm.041}$	$\mathbf{0.999}_{\pm.002}$	$\mathbf{0.928}_{\pm.045}$	$\mathbf{0.537}_{\pm.008}$	$\mathbf{0.426}_{\pm.011}$
$\mathcal{L}$-VAE	$0.188_{\pm.000}$	$0.222_{\pm.003}$	$0.193_{\pm.003}$	$0.230_{\pm.000}$	$0.027_{\pm.000}$	$0.034_{\pm.002}$
$\mathcal{G}$-VAE	$0.218_{\pm.003}$	$0.246_{\pm.000}$	$0.105_{\pm.008}$	$0.246_{\pm.000}$	$0.027_{\pm.000}$	$0.031_{\pm.000}$

Figure 5: Reconstruction performance vs. sparsity of representations. **(a)** Results for the VAE model family. The curves are sigmoid fit to $\langle \texttt{lin}|\texttt{lin}\rangle$ and $\langle \texttt{conv}|\texttt{lin}\rangle$ $\mathcal{P}$-VAE results across varying β values (β from eq. (4)). Empty circles correspond to $\langle \texttt{conv}|\texttt{lin}\rangle$ architectures. **(b)** Amortization gap for $\mathcal{P}$-VAE (blue open circle) compared to sparse coding (LCA/ISTA). Solid points show results from applying the LCA inference algorithm to $\mathcal{P}$-VAE basis vectors at different sparsity levels (β_{LCA} from eq. (1)). The purple curve is a sigmoid fit, and curves from part (a) are also included for comparison.

Does $\mathcal{P}$-VAE match the performance of traditional sparse coding trained with LCA or ISTA? Figure 5b compares $\mathcal{P}$-VAE to sparse coding models that were trained using a wide range of hyperparameters, and the best models were selected for each class (appendix C). $\mathcal{P}$-VAE achieves a similar sparsity to LCA and ISTA ($s = 0.94, 0.91$, and 0.96, respectively), but the best LCA model drastically outperforms $\mathcal{P}$-VAE on MSE for similar levels of sparsity. This suggests our convolutional encoder is struggling to close the amortization gap. To test this hypothesis, we performed LCA inference on basis elements learned by $\mathcal{P}$-VAE (Fig. 5b curve/solid points). We explored a range of hyperparameters to determine whether the MSE improved for similar sparsity levels. Indeed, LCA inference using $\mathcal{P}$-VAE dictionary was able to nearly match the performance of sparse coding LCA for similar levels of sparsity. This confirms our hypothesis that a large amortization gap remains for the specific encoder architectures we tested, highlighting the need for improved inference algorithms/architectures [112].

The $\mathcal{P}$-VAE is more sample efficient in downstream tasks.   To assess downstream performance, we trained $\langle \texttt{conv}|\texttt{conv}\rangle$ VAE models with a $K = 10$ latent dimension on MNIST (see supplementary Fig. 12 for generated samples and reconstructions from these models). We then extracted representations from the trained encoders and evaluated their ability to classify MNIST digits. We define representations as mean vectors $\boldsymbol{\mu}$ for continuous VAEs ($\mathcal{G}$-VAE, $\mathcal{L}$-VAE) following conventions in the VAE literature [108], and use $\log \boldsymbol{\delta r}$ for $\mathcal{P}$-VAE, and logits for $\mathcal{C}$-VAE.

We split the MNIST validation set into two 5,000 sample sets, used as train/test sets for this task. We train K-nearest neighbors (KNN) classifiers with a varying number of limited supervised samples ($N = 200, 1000, 5000$) drawn without replacement from the first set (train), to measure classification accuracy on the withheld set (test). KNN is nonparametric, and its performance is directly influenced by the geometry of representations by explicitly capturing the distance between encoded samples [114]. We find that using only $N = 200$ samples, $\mathcal{P}$-VAE achieves $\sim 82\%$ accuracy in held out data; whereas, $\mathcal{G}$-VAE achieves the same level of accuracy at $N = 1000$ samples (Table 4). By this measure, $\mathcal{P}$-VAE is $5\times$ more sample efficient. But from Alleman et al. [115], we know that the choice of activation function changes the geometry of learned representations. Therefore, we also tested $\mathcal{G}$-VAE models with an activation function (relu and $\exp$) applied to latents after sampling from the

Table 4: Geometry of representations ($K = 10$ only; see Table 6 for the full set of results).

Latent dim.	Model	KNN classification (N, # labeled samples)			Shattering dim.
		$N = 200$	$N = 1,000$	$N = 5,000$	
$K = 10$	$\mathcal{P}$-VAE	$\mathbf{0.815}_{\pm.002}$	$\mathbf{0.919}_{\pm.001}$	$\mathbf{0.946}_{\pm.017}$	$\mathbf{0.797}_{\pm.009}$
	$\mathcal{C}$-VAE	$0.705_{\pm.002}$	$0.800_{\pm.002}$	$0.853_{\pm.040}$	$\mathbf{0.795}_{\pm.006}$
	$\mathcal{L}$-VAE	$0.757_{\pm.003}$	$0.869_{\pm.002}$	$\mathbf{0.924}_{\pm.028}$	$0.751_{\pm.008}$
	$\mathcal{G}$-VAE	$0.673_{\pm.003}$	$0.813_{\pm.002}$	$0.891_{\pm.033}$	$0.758_{\pm.007}$
	$\mathcal{G}$-VAE $_{+\mathrm{relu}}$	$0.694_{\pm.003}$	$0.817_{\pm.003}$	$0.877_{\pm.045}$	$0.762_{\pm.007}$
	$\mathcal{G}$-VAE $_{+\mathrm{exp}}$	$0.642_{\pm.003}$	$0.784_{\pm.002}$	$0.863_{\pm.032}$	$0.737_{\pm.008}$

posterior. This biological constraint improved $\mathcal{G}$-VAE, but it still underperformed $\mathcal{P}$-VAE (Table 4). We also found this result held for higher dimensional latent spaces (supplementary Table 6).

In supplementary analyses (Fig. 11), we evaluated the representations using logistic regression trained on the full dataset. For larger latent dimensionalities ($K = 50, 100$), $\mathcal{P}$-VAE outperformed all other VAEs, but at lower dimensionalities ($K = 10$), it underperforms both $\mathcal{G}$-VAE and $\mathcal{L}$-VAE.

The $\mathcal{P}$-VAE learns representations with higher dimensional geometry. The preceding results are indicative of substantial differences in the geometry of the representations learned by $\mathcal{P}$-VAE compared to other VAE families (Table 4). To test this more explicitly, we calculated the "shattering dimensionality" of the latent space [116–118]. Shattering dim measures the average accuracy over all possible pairwise classification tasks. This is called "shattering" because if the model shatters data points around into a high dimensional space, they will become more linearly separable. For MNIST with 10 classes, there are $\binom{10}{5} = 252$ possible classifications. We trained logistic regression on the entire training set to classify each of the 252 arbitrary splits and measured the average performance on the entire validation set. The far right column of Table 4 shows the measured shattering dims. For $K = 10$, the shattering dim was significantly higher for discrete VAEs ($\mathcal{P}$-VAE, $\mathcal{C}$-VAE). For higher dimensional latent spaces $\mathcal{P}$-VAE strongly outperformed alternative models (Table 6).

5 Conclusions

In this paper, we introduced the $\mathcal{P}$-VAE, a generative model that encodes inputs into discrete spike counts and unifies established theoretical concepts in neuroscience with modern machine learning. We introduced a Poisson reparameterization algorithm and derived the ELBO for Poisson-distributed latent variables. The $\mathcal{P}$-VAE objective results in a KL term that penalizes firing rates, like sparse coding. We showed that $\mathcal{P}$-VAE with a linear decoder reduces to amortized sparse coding. We evaluated the representations on downstream classification tasks and found that $\mathcal{P}$-VAE encodes its inputs in a higher dimensional space, enabling good linear separability between classes.

Limitations. $\mathcal{P}$-VAE samples Poisson latents. Although this is inspired by the statistics of spike counts in the brain over short time intervals [50], there are deviations from Poisson throughout the cortex over longer time windows [51]. We discuss this point in appendix A. A second limitation is the amortization gap between our current implementation of $\mathcal{P}$-VAE and traditional sparse coding. This could likely be closed with more expressive encoders [119] or through iterative inference [113, 120], but it is an open area of research [112].

Neuroscience implications and future directions. Like biological neurons, the $\mathcal{P}$-VAE generates spikes. This non-negative, discrete representational form closely parallels neuronal spiking activity. Therefore, the $\mathcal{P}$-VAE can be more directly compared to neuronal circuits than unconstrained, continuous VAEs. This analogy facilitates in silico perturbation experiments (e.g., "stimulating" or "silencing" $\mathcal{P}$-VAE neurons) to mirror in vivo causal manipulations. It also allows applying methods like *Most Exciting Inputs* (MEI; [121]), which assume non-negative activations. Future work could explore hierarchical $\mathcal{P}$-VAEs, finding a sweet spot between interpretability and performance. Overall, the biologically inspired representational form of $\mathcal{P}$-VAE brings computational modeling closer to experimental neuroscience and opens new avenues for advancing NeuroAI research [13, 20].

6   Code and data

Our code, data, and model checkpoints are available here: https://github.com/hadivafaii/PoissonVAE.

7   Acknowledgments

References

[1]   Irina Higgins et al. "Unsupervised deep learning identifies semantic disentanglement in single inferotemporal face patch neurons". In: *Nature Communications* 12.1 (2021), p. 6456. DOI: 10.1038/s41467-021-26751-5.

[2]   Hadi Vafaii et al. "Hierarchical VAEs provide a normative account of motion processing in the primate brain". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=1wOkHN9JK8.

[3]   Bruno A Olshausen and David J Field. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". In: *Nature* 381.6583 (1996), pp. 607–609. DOI: 10.1038/381607a0.

[4]   Michael E. Tipping and Christopher M. Bishop. "Probabilistic Principal Component Analysis". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 61.3 (Jan. 1999), pp. 611–622. ISSN: 1369-7412. DOI: 10.1111/1467-9868.00196.

[5]   Warren S McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5 (1943), pp. 115–133. DOI: 10.1007/BF02478259.

[6]   Patricia S Churchland and Terrence J Sejnowski. "Perspectives on cognitive neuroscience". In: *Science* 242.4879 (1988), pp. 741–745. DOI: 10.1126/science.3055294.

[7]   Michael SC Thomas and James L McClelland. "Connectionist models of cognition". In: *The Cambridge handbook of computational psychology* (2008), pp. 23–58. URL: http://www7.bbk.ac.uk/psychology/dnl/wp-content/uploads/2023/10/Thomas-McClelland-proof.pdf.

[8]   Nikolaus Kriegeskorte. "Deep neural networks: a new framework for modeling biological vision and brain information processing". In: *Annual Review of Vision Science* 1 (2015), pp. 417–446. DOI: 10.1101/029876.

[9]   Grace W. Lindsay. "Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future". In: *Journal of Cognitive Neuroscience* 33.10 (Sept. 2021), pp. 2017–2031. ISSN: 0898-929X. DOI: 10.1162/jocn_a_01544.

[10]   Joseph Marino. "Predictive coding, variational autoencoders, and biological connections". In: *Neural Computation* 34.1 (2022), pp. 1–44. DOI: 10.1162/neco_a_01458.

[11]   Jeffrey S Bowers et al. "Deep problems with neural network models of human vision". In: *Behavioral and Brain Sciences* 46 (2023), e385. DOI: 10.1017/S0140525X22002813.

[12]   Felix A. Wichmann and Robert Geirhos. "Are Deep Neural Networks Adequate Behavioral Models of Human Visual Perception?" In: *Annual Review of Vision Science* 9.Volume 9, 2023 (2023), pp. 501–524. ISSN: 2374-4650. DOI: 10.1146/annurev-vision-120522-031739.

[13]   Anthony Zador et al. "Catalyzing next-generation Artificial Intelligence through NeuroAI". In: *Nature Communications* 14.1 (2023), p. 1597. DOI: 10.1038/s41467-023-37180-x.

[14]   Fabian H Sinz et al. "Engineering a less artificial intelligence". In: *Neuron* 103.6 (2019), pp. 967–979. DOI: 10.1016/j.neuron.2019.08.034.

[15] Demis Hassabis et al. "Neuroscience-inspired artificial intelligence". In: *Neuron* 95.2 (2017), pp. 245–258. DOI: 10.1016/j.neuron.2017.06.011.

[16] Nancy Kanwisher et al. "Using artificial neural networks to ask 'why' questions of minds and brains". In: *Trends in Neurosciences* (2023). DOI: 10.1016/j.tins.2022.12.008.

[17] Blake Richards et al. "The application of artificial intelligence to biology and neuroscience". In: *Cell* 185.15 (2022), pp. 2640–2643. DOI: 10.1016/j.cell.2022.06.047.

[18] Blake A Richards et al. "A deep learning framework for neuroscience". In: *Nature Neuroscience* 22.11 (2019), pp. 1761–1770. DOI: 10.1038/s41593-019-0520-2.

[19] David GT Barrett et al. "Analyzing biological and artificial neural networks: challenges with opportunities for synergy?" In: *Current Opinion in Neurobiology* 55 (2019), pp. 55–64. DOI: 10.1016/j.conb.2019.01.007.

[20] Adrien Doerig et al. "The neuroconnectionist research programme". In: *Nature Reviews Neuroscience* (2023), pp. 1–20. DOI: 10.1038/s41583-023-00705-w.

[21] Ibn al-Haytham. *Book of optics (Kitab Al-Manazir)*. 1011–1021 AD.

[22] Hermann Von Helmholtz. *Handbuch der physiologischen Optik*. Vol. 9. Voss, 1867.

[23] Tai Sing Lee and David Mumford. "Hierarchical Bayesian inference in the visual cortex". In: *JOSA A* 20.7 (2003), pp. 1434–1448. DOI: 10.1364/JOSAA.20.001434.

[24] Bruno A. Olshausen. "Perception as an Inference Problem". In: *The Cognitive Neurosciences (5th edition)* (2014). Ed. by Michael Gazzaniga and George R. Mangun. DOI: 10.7551/mitpress/9504.003.0037. URL: http://rctn.org/bruno/papers/perception-as-inference.pdf.

[25] Edwin Garrigues Boring. "Perception of objects". In: *American Journal of Physics* (1946). DOI: 10.1119/1.1990807.

[26] Karl Friston. "The free-energy principle: a unified brain theory?" In: *Nature Reviews Neuroscience* 11.2 (2010), pp. 127–138. DOI: 10.1038/nrn2787.

[27] Sam Bond-Taylor et al. "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models". In: *IEEE transactions on pattern analysis and machine intelligence* 44.11 (2021), pp. 7327–7347. DOI: 10.1109/TPAMI.2021.3116668.

[28] Stanley H. Chan. *Tutorial on Diffusion Models for Imaging and Vision*. 2024. arXiv: 2403.18103 [cs.LG].

[29] Wayne Xin Zhao et al. *A Survey of Large Language Models*. 2023. arXiv: 2303.18223 [cs.CL].

[30] Rajesh PN Rao and Dana H Ballard. "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects". In: *Nature Neuroscience* 2.1 (1999), pp. 79–87. DOI: 10.1038/4580.

[31] Robin Rombach et al. "High-Resolution Image Synthesis With Latent Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10684–10695. URL: https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

[32] Tero Karras et al. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019. URL: https://openaccess.thecvf.com/content_CVPR_2019/html/Karras_A_Style-Based_Generator_Architecture_for_Generative_Adversarial_Networks_CVPR_2019_paper.html.

[33] Arash Vahdat and Jan Kautz. "NVAE: A Deep Hierarchical Variational Autoencoder". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 19667–19679. URL: https://papers.nips.cc/paper_files/paper/2020/hash/e3b21256183cf7c2c7a66be163579d37-Abstract.html.

[34] Rewon Child. "Very Deep {VAE}s Generalize Autoregressive Models and Can Outperform Them on Images". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=RLRXCV6DbEJ.

[35] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: (2014). arXiv: 1312.6114v11 [stat.ML].

[36] Danilo Jimenez Rezende et al. "Stochastic backpropagation and approximate inference in deep generative models". In: *International Conference on Machine Learning*. PMLR. 2014, pp. 1278–1286. URL: https://proceedings.mlr.press/v32/rezende14.html.

[37] Diederik P Kingma and Ruiqi Gao. "Understanding Diffusion Objectives as the ELBO with Simple Data Augmentation". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=NnMEadcdyD.

[38] Karsten Kreis et al. *NeurIPS 2023 Tutorial on Latent Diffusion Models*. https://neurips2023-ldm-tutorial.github.io/. 2023.

[39] Diederik Kingma et al. "Variational Diffusion Models". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 21696–21707. URL: https://proceedings.neurips.cc/paper/2021/hash/b578f2a52a0229873fefc2a4b06377fa-Abstract.html.

[40] Ferenc Csikor et al. "Top-down perceptual inference shaping the activity of early visual cortex". In: *bioRxiv* (2023). DOI: 10.1101/2023.11.29.569262.

[41] T. Anderson Keller et al. "Modeling Category-Selective Cortical Regions with Topographic Variational Autoencoders". In: *SVRHM 2021 Workshop @ NeurIPS*. 2021. URL: https://openreview.net/forum?id=yGRq_lW54bI.

[42] T. Anderson Keller and Max Welling. "Topographic VAEs learn Equivariant Capsules". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 28585–28597. URL: https://proceedings.neurips.cc/paper/2021/hash/f03704cb51f02f80b09bffba15751691-Abstract.html.

[43] Katherine R Storrs et al. "Unsupervised learning predicts human perception and misperception of gloss". In: *Nature Human Behaviour* 5.10 (2021), pp. 1402–1417. DOI: 10.1038/s41562-021-01097-6.

[44] Edgar Adrian. *The activity of the nerve fibres*. https://www.nobelprize.org/prizes/medicine/1932/adrian/lecture/. 1932.

[45] Edgar Douglas Adrian and Yngve Zotterman. "The impulses produced by sensory nerve-endings: Part II. The response of a Single End-Organ". In: *The Journal of Physiology* (1926), pp. 151–71. DOI: 10.1113/jphysiol.1926.sp002281.

[46] Donald H Perkel and Theodore H Bullock. "Neural coding". In: *Neurosciences Research Program Bulletin* (1968). URL: https://ntrs.nasa.gov/citations/19690022317.

[47] Horace B Barlow. "Single units and sensation: a neuron doctrine for perceptual psychology?" In: *Perception* 1.4 (1972), pp. 371–394. DOI: 10.1068/p010371.

[48] Ehud Zohary et al. "Correlated neuronal discharge rate and its implications for psychophysical performance". In: *Nature* 370.6485 (1994), pp. 140–143. DOI: 10.1038/370140a0.

[49] Fred Rieke et al. *Spikes: exploring the neural code*. MIT press, 1999. URL: https://mitpress.mit.edu/9780262181747/spikes/.

[50] Malvin C Teich. "Fractal character of the auditory neural spike train". In: *IEEE Transactions on Biomedical Engineering* 36.1 (1989), pp. 150–160. DOI: 10.1109/10.16460.

[51] Robbe LT Goris et al. "Partitioning neuronal variability". In: *Nature Neuroscience* 17.6 (2014), pp. 858–865. DOI: 10.1038/nn.3711.

[52] Diederik P Kingma and Max Welling. "An introduction to variational autoencoders". In: *Foundations and Trends® in Machine Learning* 12.4 (2019), pp. 307–392. DOI: 10.1561/2200000056.

[53] Charles D Gilbert and Wu Li. "Top-down influences on visual processing". In: *Nature Reviews Neuroscience* 14.5 (2013), pp. 350–363. DOI: 10.1038/nrn3476.

[54] David C Knill and Alexandre Pouget. "The Bayesian brain: the role of uncertainty in neural coding and computation". In: *Trends in Neurosciences* 27.12 (2004), pp. 712–719. DOI: 10.1016/j.tins.2004.10.007.

[55] Horace Barlow. "Redundancy reduction revisited". In: *Network: computation in neural systems* 12.3 (2001), p. 241. DOI: 10.1080/net.12.3.241.253.

[56] Karl Friston. "The free-energy principle: a rough guide to the brain?" In: *Trends in Cognitive Sciences* 13.7 (2009), pp. 293–301. DOI: 10.1016/j.tics.2009.04.005.

[57] Peter Dayan et al. "The Helmholtz machine". In: *Neural Computation* 7.5 (1995), pp. 889–904. DOI: 10.1162/neco.1995.7.5.889.

[58] William Lotter et al. "Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=B1ewdt9xe.

[59] Fred Attneave. "Some informational aspects of visual perception." In: *Psychological review* 61.3 (1954), p. 183. DOI: 10.1037/h0054663.

[60] Horace B. Barlow. "Possible principles underlying the transformation of sensory messages". In: *Sensory communication* 1.01 (1961), pp. 217–233. URL: https://www.cnbc.cmu.edu/~tai/microns_papers/Barlow-SensoryCommunication-1961.pdf.

[61] Mandyam Veerambudi Srinivasan et al. "Predictive coding: a fresh view of inhibition in the retina". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 216.1205 (1982), pp. 427–459. DOI: 10.1098/rspb.1982.0085.

[62] Karl Friston. "A theory of cortical responses". In: *Philosophical transactions of the Royal Society B: Biological Sciences* 360.1456 (2005), pp. 815–836. DOI: 10.1098/rstb.2005.1622.

[63] Andy Clark. "Whatever next? Predictive brains, situated agents, and the future of cognitive science". In: *Behavioral and brain sciences* 36.3 (2013), pp. 181–204. DOI: 10.1017/S0140525X12000477.

[64] William Lotter et al. "A neural network trained for prediction mimics diverse features of biological neurons and perception". In: *Nature machine intelligence* 2.4 (2020), pp. 210–219. DOI: 10.1038/s42256-020-0170-9.

[65] Beren Millidge et al. "Predictive coding networks for temporal prediction". In: *PLOS Computational Biology* 20.4 (2024), e1011183. DOI: 10.1371/journal.pcbi.1011183.

[66] Yosef Singer et al. "Hierarchical temporal prediction captures motion processing along the visual pathway". In: *Elife* 12 (2023), e52599. DOI: 10.7554/eLife.52599.

[67] Pierre-Étienne H Fiquet and Eero P Simoncelli. "A polar prediction model for learning to represent visual transformations". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=hyPUZX03Ks.

[68] Bruno A Olshausen and David J Field. "Sparse coding of sensory inputs". In: *Current opinion in neurobiology* 14.4 (2004), pp. 481–487. DOI: 10.1016/j.conb.2004.07.007.

[69] William E Vinje and Jack L Gallant. "Sparse coding and decorrelation in primary visual cortex during natural vision". In: *Science* 287.5456 (2000), pp. 1273–1276. DOI: 10.1126/science.287.5456.1273.

[70] Alison L Barth and James FA Poulet. "Experimental evidence for sparse firing in the neocortex". In: *Trends in neurosciences* 35.6 (2012), pp. 345–355. DOI: 10.1016/j.tins.2012.03.008.

[71] R Quian Quiroga et al. "Sparse but not 'grandmother-cell' coding in the medial temporal lobe". In: *Trends in cognitive sciences* 12.3 (2008), pp. 87–91. DOI: 10.1016/j.tics.2007.12.003.

[72] Tomáš Hromádka et al. "Sparse representation of sounds in the unanesthetized auditory cortex". In: *PLoS biology* 6.1 (2008), e16. DOI: 10.1371/journal.pbio.0060016.

[73] Cindy Poo and Jeffry S Isaacson. "Odor representations in olfactory cortex: "sparse" coding, global inhibition, and oscillations". In: *Neuron* 62.6 (2009), pp. 850–861. DOI: 10.1016/j.neuron.2009.05.022.

[74] Jason Wolfe et al. "Sparse and powerful cortical spikes". In: *Current opinion in neurobiology* 20.3 (2010), pp. 306–312. DOI: 10.1016/j.conb.2010.03.006.

[75] Ben DB Willmore et al. "Sparse coding in striate and extrastriate visual cortex". In: *Journal of neurophysiology* 105.6 (2011), pp. 2907–2919. DOI: 10.1152/jn.00594.2010.

[76] Bilal Haider et al. "Synaptic and network mechanisms of sparse and reliable visual cortical activity during nonclassical receptive field stimulation". In: *Neuron* 65.1 (2010), pp. 107–121. DOI: 10.1016/j.neuron.2009.12.005.

[77] Sylvain Crochet et al. "Synaptic mechanisms underlying sparse coding of active touch". In: *Neuron* 69.6 (2011), pp. 1160–1175. DOI: 10.1016/j.neuron.2011.02.022.

[78] Carl CH Petersen. "Sensorimotor processing in the rodent barrel cortex". In: *Nature Reviews Neuroscience* 20.9 (2019), pp. 533–546. DOI: 10.1038/s41583-019-0200-y.

[79] Emmanouil Froudarakis et al. "Population code in mouse V1 facilitates readout of natural scenes through increased sparseness". In: *Nature neuroscience* 17.6 (2014), pp. 851–857. DOI: 10.1038/nn.3707.

[80] Christopher J Rozell et al. "Sparse coding via thresholding and local competition in neural circuits". In: *Neural Computation* 20.10 (2008), pp. 2526–2563. DOI: 10.1162/neco.2008.03-07-486.

[81] I. Daubechies et al. "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint". In: *Communications on Pure and Applied Mathematics* 57.11 (2004), pp. 1413–1457. DOI: 10.1002/cpa.20042.

[82] Amir Beck and Marc Teboulle. "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems". In: *SIAM Journal on Imaging Sciences* 2.1 (2009), pp. 183–202. DOI: 10.1137/080716542.

[83] Ankush Ganguly et al. "Amortized Variational Inference: A Systematic Review". In: *Journal of Artificial Intelligence Research* 78 (2023), pp. 167–215. DOI: 10.1613/jair.1.14258.

[84] Brandon Amos. "Tutorial on Amortized Optimization". In: *Foundations and Trends® in Machine Learning* 16.5 (2023), pp. 592–732. ISSN: 1935-8237. DOI: 10.1561/2200000102.

[85] Samuel Gershman and Noah Goodman. "Amortized inference in probabilistic reasoning". In: *Proceedings of the annual meeting of the cognitive science society*. Vol. 36. 36. 2014. URL: https://escholarship.org/uc/item/34j1h7k5.

[86] Colin Conwell et al. "What can 1.8 billion regressions tell us about the pressures shaping high-level visual representation in brains and machines?" In: *bioRxiv* (2023). DOI: 10.1101/2022.03.28.485868.

[87] Eric Elmoznino and Michael F Bonner. "High-performing neural network models of visual cortex benefit from high latent dimensionality". In: *bioRxiv* (2022), pp. 2022–07. DOI: 10.1101/2022.07.13.499969.

[88] Aaron van den Oord et al. "Neural Discrete Representation Learning". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://papers.nips.cc/paper_files/paper/2017/hash/7a98af17e63a0ac09ce2e96d03992fbc-Abstract.html.

[89] Eric Jang et al. "Categorical Reparameterization with Gumbel-Softmax". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=rkE3y85ee.

[90] Hiromichi Kamata et al. "Fully spiking variational autoencoder". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 6. 2022, pp. 7059–7067. DOI: 10.1609/aaai.v36i6.20665.

[91] Victor Geadah et al. "Sparse-Coding Variational Autoencoders". In: *Neural Computation* 36.12 (Nov. 2024), pp. 2571–2601. ISSN: 0899-7667. DOI: 10.1162/neco_a_01715.

[92] Francesco Tonolini et al. "Variational Sparse Coding". In: *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*. Ed. by Ryan P. Adams and Vibhav Gogate. Vol. 115. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 690–700. URL: https://proceedings.mlr.press/v115/tonolini20a.html.

[93] Pan Xiao et al. "SC-VAE: Sparse Coding-based Variational Autoencoder with Learned ISTA". In: (2024). arXiv: 2303.16666 [cs.CV].

[94] David Roxbee Cox and Valerie Isham. *Point processes*. Vol. 12. CRC Press, 1980.

[95] Oleksandr Shchur et al. "Fast and Flexible Temporal Point Processes with Triangular Maps". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 73–84. URL: https://proceedings.neurips.cc/paper_files/paper/2020/hash/00ac8ed3b4327bdd4ebbebcb2ba10a00-Abstract.html.

[96] Oleksandr Shchur. "Modeling Continuous-time Event Data with Neural Temporal Point Processes". PhD thesis. Technische Universität München, 2022. URL: https://mediatum.ub.tum.de/doc/1662914/2wdsxe2av36cxz519qo7xolo3.dissertation.pdf.

[97] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://papers.nips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[98] Chris J. Maddison et al. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: *International Conference on Learning Representations*. 2017. URL: `https://openreview.net/forum?id=S1jE5L5gl`.

[99] Daniel J Felleman and David C Van Essen. "Distributed hierarchical processing in the primate cerebral cortex". In: *Cerebral Cortex* 1.1 (1991), pp. 1–47. DOI: `10.1093/CERCOR/1.1.1`.

[100] Nikola T Markov et al. "Anatomy of hierarchy: feedforward and feedback pathways in macaque visual cortex". In: *Journal of Comparative Neurology* 522.1 (2014), pp. 225–259. DOI: `10.1002/cne.23458`.

[101] Anita A Disney. "Neuromodulatory control of early visual processing in macaque". In: *Annual Review of Vision Science* 7 (2021), pp. 181–199. DOI: `10.1146/annurev-vision-100119-125739`.

[102] Irina Higgins et al. "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *International Conference on Learning Representations*. 2017. URL: `https://openreview.net/forum?id=Sy2fzU9gl`.

[103] James C. R. Whittington et al. "Disentanglement with Biological Constraints: A Theory of Functional Cell Types". In: *The Eleventh International Conference on Learning Representations*. 2023. URL: `https://openreview.net/forum?id=9Z_GfhZnGH`.

[104] J Hans Van Hateren and Arjen van der Schaaf. "Independent component filters of natural images compared with simple cells in primary visual cortex". In: *Proceedings of the Royal Society of London. Series B: Biological Sciences* 265.1394 (1998), pp. 359–366. DOI: `10.1098/rspb.1998.0303`.

[105] Victor Boutin et al. "Sparse deep predictive coding captures contour integration capabilities of the early visual system". In: *PLoS computational biology* 17.1 (2021), e1008629. DOI: `10.1371/journal.pcbi.1008629`.

[106] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images". In: (2009). URL: `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[107] Yoshua Bengio et al. "Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation". In: (2013). arXiv: `1308.3432 [cs.LG]`.

[108] Francesco Locatello et al. "Challenging common assumptions in the unsupervised learning of disentangled representations". In: *international conference on machine learning*. PMLR. 2019, pp. 4114–4124. URL: `https://proceedings.mlr.press/v97/locatello19a.html`.

[109] David H. Hubel and Torsten N. Wiesel. "Receptive fields of single neurones in the cat's striate cortex". In: *The Journal of Physiology* 148 (1959). DOI: `10.1113/jphysiol.1959.sp006308`.

[110] David H. Hubel and Torsten N. Wiesel. "Receptive fields and functional architecture of monkey striate cortex". In: *The Journal of Physiology* 195 (1968). DOI: `10.1113/jphysiol.1968.sp008455`.

[111] Alexander Alemi et al. "Fixing a Broken ELBO". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 159–168. URL: `https://proceedings.mlr.press/v80/alemi18a.html`.

[112] Chris Cremer et al. "Inference suboptimality in variational autoencoders". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1078–1086. URL: `https://proceedings.mlr.press/v80/cremer18a.html`.

[113] Joe Marino et al. "Iterative Amortized Inference". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 3403–3412. URL: `https://proceedings.mlr.press/v80/marino18a.html`.

[114] Kilian Q. Weinberger and Lawrence K. Saul. "Distance Metric Learning for Large Margin Nearest Neighbor Classification". In: *Journal of Machine Learning Research* 10.9 (2009), pp. 207–244. URL: `http://jmlr.org/papers/v10/weinberger09a.html`.

[115] Matteo Alleman et al. "Task structure and nonlinearity jointly determine learned representational geometry". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: `https://openreview.net/forum?id=k9t8dQ30kU`.

[116] Mattia Rigotti et al. "The importance of mixed selectivity in complex cognitive tasks". In: *Nature* 497.7451 (2013), pp. 585–590. DOI: 10.1038/nature12160.

[117] Silvia Bernardi et al. "The geometry of abstraction in the hippocampus and prefrontal cortex". In: *Cell* 183.4 (2020), pp. 954–967. DOI: 10.1016/j.cell.2020.09.031.

[118] Matthew T Kaufman et al. "The implications of categorical and category-free mixed selectivity on representational geometries". In: *Current opinion in neurobiology* 77 (2022), p. 102644. DOI: 10.1016/j.conb.2022.102644.

[119] Karol Gregor and Yann LeCun. "Learning fast approximations of sparse coding". In: *Proceedings of the 27th international conference on international conference on machine learning*. 2010, pp. 399–406. URL: https://dl.acm.org/doi/abs/10.5555/3104322.3104374.

[120] Yoon Kim et al. "Semi-Amortized Variational Autoencoders". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 2678–2687. URL: https://proceedings.mlr.press/v80/kim18e.html.

[121] Edgar Y Walker et al. "Inception loops discover what excites neurons most using deep predictive models". In: *Nature neuroscience* 22.12 (2019), pp. 2060–2065. DOI: 10.1038/s41593-019-0517-x.

[122] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

[123] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[124] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830. DOI: 10.5555/1953048.2078195.

[125] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134.

[126] John D Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in science & engineering* 9.03 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[127] Michael L Waskom. "Seaborn: statistical data visualization". In: *Journal of Open Source Software* 6.60 (2021), p. 3021. DOI: 10.21105/joss.03021.

[128] David J Tolhurst et al. "The statistical reliability of signals in single neurons in cat and monkey visual cortex". In: *Vision research* 23.8 (1983), pp. 775–785. DOI: 10.1016/0042-6989(83)90200-6.

[129] AF Dean. "The variability of discharge of simple cells in the cat striate cortex". In: *Experimental Brain Research* 44.4 (1981), pp. 437–440. DOI: 10.1007/BF00238837.

[130] Michael N Shadlen and William T Newsome. "The variable discharge of cortical neurons: implications for connectivity, computation, and information coding". In: *Journal of neuroscience* 18.10 (1998), pp. 3870–3896. DOI: 10.1523/JNEUROSCI.18-10-03870.1998.

[131] Bruno B Averbeck et al. "Neural correlations, population coding and computation". In: *Nature Reviews Neuroscience* 7.5 (2006), pp. 358–366. DOI: 10.1038/nrn1888.

[132] Peter Dayan and Laurence F Abbott. "Theoretical Neuroscience". In: (2001). URL: https://mitpress.mit.edu/9780262041997/theoretical-neuroscience/.

[133] Wilson Truccolo et al. "A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects". In: *Journal of neurophysiology* 93.2 (2005), pp. 1074–1089. DOI: 10.1152/jn.00697.2004.

[134] Zachary F Mainen and Terrence J Sejnowski. "Reliability of spike timing in neocortical neurons". In: *Science* 268.5216 (1995), pp. 1503–1506. DOI: 10.1126/science.7770778.

[135] Michael R DeWeese et al. "Binary spiking in auditory cortex". In: *Journal of Neuroscience* 23.21 (2003), pp. 7940–7949. DOI: 10.1523/JNEUROSCI.23-21-07940.2003.

[136] William H Calvin and CHARLES F Stevens. "Synaptic noise and other sources of randomness in motoneuron interspike intervals". In: *Journal of neurophysiology* 31.4 (1968), pp. 574–587. DOI: 10.1152/jn.1968.31.4.574.

[137] Christina Allen and Charles F Stevens. "An evaluation of causes for unreliability of synaptic transmission". In: *Proceedings of the National Academy of Sciences* 91.22 (1994), pp. 10380–10383. DOI: 10.1073/pnas.91.22.10380.

[138] Matteo Carandini. "Amplification of trial-to-trial response variability by neurons in visual cortex". In: *PLoS biology* 2.9 (2004), e264. DOI: 10.1371/journal.pbio.0020264.

[139] Alison I Weber and Jonathan W Pillow. "Capturing the dynamical repertoire of single neurons with generalized linear models". In: *Neural Computation* 29.12 (2017), pp. 3260–3289. DOI: 10.1162/neco_a_01021.

[140] Daniel A Butts et al. "Nonlinear computations shaping temporal processing of precortical vision". In: *Journal of Neurophysiology* 116.3 (2016), pp. 1344–1357. DOI: 10.1152/jn.00878.2015.

[141] Jagruti J Pattadkal et al. "Synchrony dynamics underlie irregular neocortical spiking". In: *bioRxiv* (2024), pp. 2024–10. DOI: 10.1101/2024.10.15.618398.

[142] Casper Kaae Sønderby et al. "Ladder Variational Autoencoders". In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc., 2016. URL: https://papers.nips.cc/paper_files/paper/2016/hash/6ae07dcb33ec3b7c814df797cbda0f87-Abstract.html.

[143] Cina Aghamohammadi et al. "A doubly stochastic renewal framework for partitioning spiking variability". In: *bioRxiv* (2024), pp. 2024–02.

[144] David M Blei et al. "Variational inference: A review for statisticians". In: *Journal of the American statistical Association* 112.518 (2017), pp. 859–877. DOI: 10.1080/01621459.2017.1285773.

[145] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[146] Prajit Ramachandran et al. "Searching for Activation Functions". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=SkBYYyZRZ.

[147] Stefan Elfwing et al. "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning". In: *Neural Networks* 107 (2018), pp. 3–11. DOI: 10.1016/j.neunet.2017.12.012.

[148] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: (2014). arXiv: 1412.6980 [cs.LG].

[149] Ilya Loshchilov and Frank Hutter. "SGDR: Stochastic Gradient Descent with Warm Restarts". In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=Skq89Scxx.

[150] Samuel R. Bowman et al. "Generating Sentences from a Continuous Space". In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 10–21. DOI: 10.18653/v1/K16-1002.

[151] Hao Fu et al. "Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 240–250. DOI: 10.18653/v1/N19-1021.

[152] Michael Teti. *LCA-PyTorch*. [Computer Software] https://doi.org/10.11578/dc.20230728.4. June 2023. DOI: 10.11578/dc.20230728.4. URL: https://doi.org/10.11578/dc.20230728.4.

[153] Shakir Mohamed et al. "Monte Carlo Gradient Estimation in Machine Learning". In: *Journal of Machine Learning Research* 21.132 (2020), pp. 1–62. URL: http://jmlr.org/papers/v21/19-346.html.

A Are real neurons truly Poisson?

In this section, we discuss empirical and theoretical observations from neuroscience that motivated our Poisson assumption.

"Poisson-like" noise in neuroscience has a long history. It begins with observations that neurons do not fire the same sequence of spikes to repeated presentations of the same input, and that the variance is proportional to the mean [128, 129], and was followed by the observation that for short counting windows, that proportionality is one [49, 50, 130–132]. Larger windows and higher visual areas are notably super-Poisson, but that can be attributed to a modulation of the rate of an inhomogeneous Poisson process [51].

In other words, neurons are conditionally Poisson, not marginally Poisson [133].

Spike-generation, it is argued, is not noisy [134–136], but synaptic noise [137], or noise on the membrane potential, can create a Poisson-like distribution of spikes [138]. An important caveat is that the well-known example of precision in spike generation by Mainen and Sejnowski [134] is effectively captured by a Poisson-process Generalized Linear Model (GLM; Weber and Pillow [139]). However, this precision relies on a Bernoulli approximation to a Poisson process, allowing only 0 or 1 spikes. There is a widely-held misconception that precise timing cannot be produced by spike-rate models, but inhomogeneous rate models can produce precise spiking patterns at high time resolution [140]. In contrast, recent work has shown that correlations in excitatory inputs drive Poisson-like variability, explaining the widespread observation of Poisson-like noise in real neurons [141].

In summary, neurons are not literally Poisson, but it is a good choice. To set up the ELBO, one has to choose an approximate posterior and prior. Because spike counts are integer and cannot be negative, Poisson is a more natural choice than Gaussian without knowing anything about neural firing statistics. Here, we found that the Poisson assumption led to a model with interesting theoretical and empirical properties, where sparse coding emerged from the ELBO with Poisson.

Extending the $\mathcal{P}$-VAE to hierarchical architectures [2, 33, 34, 142] will make the latents conditionally Poisson, but not marginally Poisson (as they are modulated by top-down rates). Further extensions could implement doubly-stochastic spike generation [51, 143].

B Full derivations

In this section, we provide a self-contained and pedagogical introduction to VAEs, derive the $\mathcal{P}$-VAE loss function, and highlight how combining Poisson-distributed latents with predictive coding leads to the emergence of a metabolic cost term in the $\mathcal{P}$-VAE loss. For the case of a linear decoder, the reconstruction loss assumes a closed-form solution. This means we can compute the gradients analytically, which we can then use to evaluate the Poisson reparameterization trick.

B.1 Deriving the evidence lower bound (ELBO) loss

For completeness, let's first go over the basics. This section will provide a quick refresher on variational inference and how to derive the VAE loss from scratch. Assume the data $\boldsymbol{x} \in \mathbb{R}^M$ and K-dimensional latent variables $\boldsymbol{z}$ are jointly distributed as $p(\boldsymbol{x}, \boldsymbol{z})$, with the data generated through the following process:

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{z}) \, d\boldsymbol{z} = \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}) \, d\boldsymbol{z}, \qquad (6)$$

In Bayesian posterior inference, the goal is to identify which latents $\boldsymbol{z}$ are likely given data $\boldsymbol{x}$. In other words, we want to approximate $P(\boldsymbol{z}|\boldsymbol{x})$, the optimal but (typically) intractable posterior distribution.

B.1.1 Variational inference and VAE loss function

To achieve approximate Bayesian inference, a common approach is to define a family of variational densities $\mathcal{Q}$ and find a member $q(\boldsymbol{z}|\boldsymbol{x}) \in \mathcal{Q}$ such that it sufficiently approximates the optimal posterior [144]. We call $q(\boldsymbol{z}|\boldsymbol{x})$ the *approximate posterior*. The general aim of variational inference (VI) can be summarized as follows:

$$\texttt{VI}: \quad \text{find a } q(\boldsymbol{z}|\boldsymbol{x}) \in \mathcal{Q} \text{ such that } q(\boldsymbol{z}|\boldsymbol{x}) \text{ is a good approximation of } p(\boldsymbol{z}|\boldsymbol{x}). \qquad (7)$$

The goodness of our approximate posterior, or its closeness to the true posterior, is measured using the Kullback-Leibler (KL) divergence:

$$q^* = \operatorname*{argmin}_{q \in \mathcal{Q}} \mathcal{D}_{\mathrm{KL}}\Big(q(\boldsymbol{z}|\boldsymbol{x}) \,\big\|\, p(\boldsymbol{z}|\boldsymbol{x}) \Big). \tag{8}$$

We cannot directly optimize eq. (8), because $p(\boldsymbol{z}|\boldsymbol{x})$ is often intractable. Instead, we rearrange some terms and arrive at the following loss function:

$$\mathcal{L}_{\mathrm{NELBO}}(q) = - \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x})}\Big[\log p(\boldsymbol{x}|\boldsymbol{z}) \Big] + \mathcal{D}_{\mathrm{KL}}\Big(q(\boldsymbol{z}|\boldsymbol{x}) \,\big\|\, p(\boldsymbol{z}) \Big). \tag{9}$$

NELBO stands for negative ELBO, also known as "variational free energy." Notably, finding a $q \in \mathcal{Q}$ that minimizes $\mathcal{L}_{\mathrm{NELBO}}(q)$ in eq. (9) is equivalent to finding the optimal q^* in eq. (8).

The first term in eq. (9), often called the reconstruction term, captures the likelihood of the observed data $\boldsymbol{x}$, given latents $\boldsymbol{z}$, under the approximate posterior. For all our VAE models, we assume a Gaussian conditional likelihood with a fixed variance, as is typically done in the literature. This approximates the reconstruction term as the mean squared error between input data and their reconstructed version. The second term, known as the KL term, is more interesting. This term can assume very different forms depending on the distribution used.

B.2   The KL term

In this section, we will derive closed-form expressions for the KL term for different choices of the distributions $q(\boldsymbol{z}|\boldsymbol{x})$ and $p(\boldsymbol{z})$. Specifically, we will focus on Gaussian and Poisson parameterizations.

Predictive coding assumption.   We will draw inspiration from predictive coding and assume that the bottom-up inference pathway only encodes the residual information relative to the top-down, or predicted information. We will apply this idea to both Gaussian and Poisson cases, and find that only in the Poisson case, the outcome becomes interpretable and resembles sparse coding objective.

B.2.1   KL term derivation: Gaussian

Let $q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_q(\boldsymbol{x}), \boldsymbol{\sigma}_q(\boldsymbol{x}))$ and $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_p, \boldsymbol{\sigma}_p)$, where the mean and variance are either outputs of the encoder network or parameters of the decoder network.

Now, let us implement the predictive coding assumption, where the encoder only keeps track of residual information that is not already contained in the prior information. Mathematically, this idea can be formalized as follows:

$$\begin{aligned} \boldsymbol{\mu}_p &\to \boldsymbol{\mu}, & \boldsymbol{\mu}_q &\to \boldsymbol{\mu} + \boldsymbol{\delta\mu} \\ \boldsymbol{\sigma}_p &\to \boldsymbol{\sigma}, & \boldsymbol{\sigma}_q &\to \boldsymbol{\sigma} \cdot \boldsymbol{\delta\sigma} \end{aligned} \tag{10}$$

With these modifications, the Gaussians KL term becomes:

$$\mathcal{D}_{\mathrm{KL}}\left(q \,\|\, p \right) = \frac{1}{2}\Big(\frac{\boldsymbol{\delta\mu}^2}{\boldsymbol{\sigma}^2} + \boldsymbol{\delta\sigma}^2 - \log \boldsymbol{\delta\sigma}^2 - \boldsymbol{1} \Big). \tag{11}$$

In standard Gaussian VAEs, the prior has no learnable parameter. Instead, we have $\boldsymbol{\mu} \to \boldsymbol{0}$ and $\boldsymbol{\sigma} \to \boldsymbol{1}$. Therefore, the final form of the KL term for a standard Gaussian VAE is:

$$\mathcal{D}_{\mathrm{KL}}\left(q \,\|\, \mathcal{N}(\boldsymbol{0}, \boldsymbol{1}) \right) = \frac{1}{2}\Big(\boldsymbol{\delta\mu}^2 + \boldsymbol{\delta\sigma}^2 - \log \boldsymbol{\delta\sigma}^2 - \boldsymbol{1} \Big). \tag{12}$$

We observe that the KL term vanishes when $\boldsymbol{\delta\mu} \to \boldsymbol{0}$ and $\boldsymbol{\delta\sigma} \to \boldsymbol{1}$. This happens whenever no new information is propagated through the encoder, a phenomenon known as posterior collapse.

Other than this trivial observation, eq. (12) does not really lend itself to interpretation. In contrast, will show below that a Poisson parameterization of VAEs leads to a much more interpretable outcome for the KL term.

B.2.2 KL term derivation: Poisson

Now suppose $q(z|\boldsymbol{x}) = \mathcal{P}\text{ois}(z; r\delta r(\boldsymbol{x}))$, and $p(z) = \mathcal{P}\text{ois}(z; r)$, where z is literally the spike count of a single latent dimension—or shall we say, neuron?

In the Poisson case, the KL term becomes more interpretable, as we will show below. Recall that the Poisson distribution for a single variable z, given rate $\lambda \in \mathbb{R}_{>0}$, is given by:

$$\mathcal{P}\text{ois}(z; \lambda) = \frac{\lambda^z e^{-\lambda}}{z!}. \tag{13}$$

Plug this expressions into the KL divergence definition to get:

$$
\begin{aligned}
\mathcal{D}_{\text{KL}}\left(q \,\|\, p\right) &= \mathbb{E}_{z\sim q}\left[\log \frac{q}{p}\right] \\
&= \mathbb{E}_{z\sim q}\left[\log \frac{(r\delta r)^z e^{-r\delta r}/z!}{r^z e^{-r}/z!}\right] \\
&= \mathbb{E}_{z\sim q}\left[\log \left(\left(\frac{r\delta r}{r}\right)^z e^{-r\delta r + r}\right)\right] \\
&= \mathbb{E}_{z\sim q}\left[\log \delta r^z + \log e^{-r\delta r + r}\right] \\
&= \mathbb{E}_{z\sim q}\left[z\log \delta r - r\delta r + r\right] \\
&= \mathbb{E}_{z\sim q}\left[z\right] \log \delta r - r\delta r + r \\
&= r\delta r \log \delta r - r\delta r + r \\
&= r\left(1 - \delta r + \delta r \log \delta r\right) \\
&= r f(\delta r),
\end{aligned}
\tag{14}
$$

where we have define $f(y) := 1 - y + y\log y$.

To examine the behavior of the Poisson KL term, we assume $\delta r = 1 + \epsilon$, where $\epsilon \ll 1$, then Taylor expand f. Calculating the first and second derivatives of $f(y) = 1 - y + y\log y$ gives $f'(y) = \log y$ and $f''(y) = 1/y$. Thus:

$$
\begin{aligned}
f(1 + \epsilon) &= f(1) + \epsilon f'(1) + \frac{\epsilon^2}{2!}f''(1) + \mathcal{O}(\epsilon^3) \\
&= 0 + 0 + \frac{\epsilon^2}{2!} + \mathcal{O}(\epsilon^3) \\
&\approx \frac{1}{2}\epsilon^2
\end{aligned}
\tag{15}
$$

Plug this back into eq. (14) to get:

$$
\begin{aligned}
\mathcal{D}_{\text{KL}}\left(q \,\|\, p\right) &= r f(\delta r) \\
&= r f(1 + \epsilon) \\
&\approx \frac{1}{2}r\epsilon^2.
\end{aligned}
\tag{16}
$$

For small deviations ϵ, the KL term simplifies to the product of the prior firing rate, r, and ϵ^2. See Fig. 6 for a visualization of the full function, $f(\delta r) = 1 - \delta r + \delta r \log \delta r$, along with its quadratic approximation near $\delta r = 1$.

In general, there are two ways to minimize the KL term: dead prior neurons ($r \to 0$), or posterior collapse ($\delta r \to 1$).
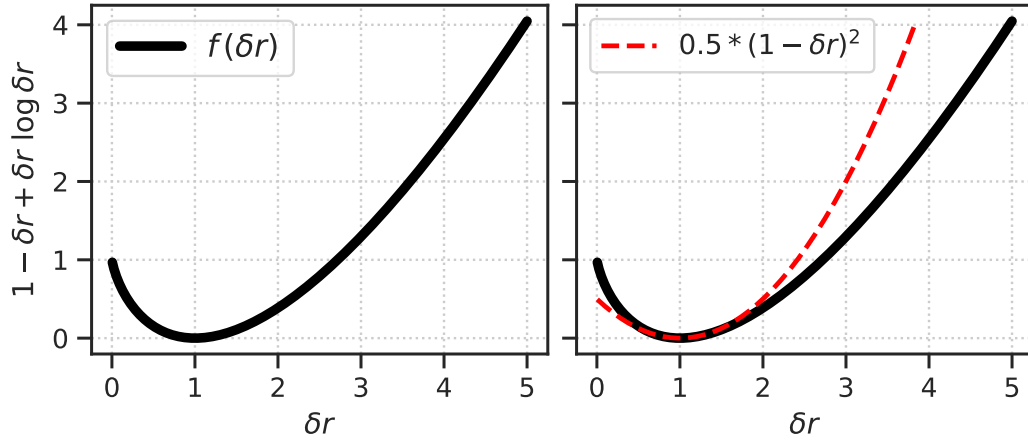
Figure 6: Left, residual term $f(\delta r)$ from eq. (14). Right, quadratic approximation of f from eq. (15).

Together with the reconstruction loss, the NELBO for a 1-dimensional $\mathcal{P}$-VAE reads:

$$\mathcal{L}_{\text{PVAE}}\left(r, \delta r\right) = \mathcal{L}_{\text{recon.}}\left(r, \delta r\right) + r\left(1 - \delta r + \delta r \log \delta r\right). \tag{17}$$

Finally, it is easy to show that for K-dimensional latent space, eq. (14) generalizes to:

$$\mathcal{D}_{\text{KL}}\Big(\mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r} \odot \boldsymbol{\delta r}(\boldsymbol{x})) \,\big\|\, \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r})\Big) = \boldsymbol{r} \cdot f(\boldsymbol{\delta r}), \tag{18}$$

where $\odot$ and $\cdot$ denote the Hadamard (element-wise) and vector products, respectively.

B.3   Connection to sparse coding

Equation (17) mirrors sparse coding due to the presence of the firing rate in the objective function. Furthermore, it follows the principle of predictive coding by design. Thus, our Poisson formulation of VAEs effectively unifies these two major themes in theoretical neuroscience. Let's explore this curious connection to sparse coding more closely below.

B.4   Statistically independent neurons

Suppose our $\mathcal{P}$-VAE has K statistically independent neurons, and $\boldsymbol{z} \in \mathbb{Z}_{\geq 0}^{K}$ is the spike count variable, where $\mathbb{Z}_{\geq 0} = \{0, 1, 2, \dots\}$ is the set of non-negative integers. Let us use bold font $\boldsymbol{r}$ and $\boldsymbol{\delta r}$ to refer to the firing rate vectors of the representation and error units, respectively. Recall that we allowed these variables to interact in a multiplicative way to construct the posterior rates, $\lambda_i(\boldsymbol{x}) = r_i \delta r_i(\boldsymbol{x})$. More explicitly, we have:

$$
\begin{aligned}
q(\boldsymbol{z}|\boldsymbol{x}) &= \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r} \odot \boldsymbol{\delta r}) = \prod_{i=1}^{K} \mathcal{P}\text{ois}(z_i; r_i \delta r_i) = \prod_{i=1}^{K} \frac{\lambda_i^{z_i} e^{-\lambda_i}}{z_i!}, \\
p(\boldsymbol{z}) &= \mathcal{P}\text{ois}(\boldsymbol{z}; \boldsymbol{r}) = \prod_{i=1}^{K} \mathcal{P}\text{ois}(z_i; r_i) = \prod_{i=1}^{K} \frac{r_i^{z_i} e^{-r_i}}{z_i!}.
\end{aligned}
\tag{19}
$$

Note that, unlike a standard Gaussian VAE, the prior in $\mathcal{P}$-VAE is parameterized using $\boldsymbol{r}$, which is learned from data along with the other parameters. Similar to standard Gaussian VAEs, $\boldsymbol{\delta r}(\boldsymbol{x})$ is parameterized as a neural network.

B.5 Linear decoder

Following the sparse coding literature [3], we will now assume our decoder generates the input image $\boldsymbol{x} \in \mathbb{R}^M$ as a linear sum of K basis elements, $\boldsymbol{\Phi} \in \mathbb{R}^{M \times K}$. Additionally, we choose a diagonal Gaussian distribution with fixed variance as our conditional likelihood, resulting in a mean squared error between the input $\boldsymbol{x}$, and its reconstruction $\boldsymbol{\Phi z}$.

Given these assumptions, the reconstruction loss for a VAE with approximate posterior q can be expressed as follows:

$$\mathcal{L}_{\text{recon.}}(\boldsymbol{x}; q) = \mathbb{E}_{\boldsymbol{z} \sim q(Z|X=\boldsymbol{x})}\left[\|\boldsymbol{x} - \boldsymbol{\Phi z}\|_2^2\right]. \tag{20}$$

For a linear decoder, the reconstruction term $\|\boldsymbol{x} - \boldsymbol{\Phi z}\|_2^2$ contains only the first and second moments of $\boldsymbol{z}$. Consequently, the expectation in eq. (20) can be analytically computed. This results in a close-form expression for the reconstruction loss, and consequently, its gradients as well.

In general, whenever the VAE decoder is linear, the following result holds:

$$\boxed{\mathcal{L}_{\text{recon.}}(\boldsymbol{x}; q, \boldsymbol{\Phi}) = \|\boldsymbol{x} - \boldsymbol{\Phi}\,\mathbb{E}_q[Z]\|_2^2 + \text{Var}_q[Z]^T \text{diag}(\boldsymbol{\Phi}^T\boldsymbol{\Phi}).} \tag{21}$$

Note that a linear decoder is the only assumption we needed to obtain this closed-form solution. There are no restrictions on the form of the encoder: it can be linear, or as complicated as we want. We only have to compute the mean and variance of the posterior.

Specifically, for the Poisson case, we only need to know the following expectation values:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{z} \sim \mathcal{P}\text{ois}(\boldsymbol{z};\boldsymbol{\lambda})}\left[z_i\right] &= \lambda_i, \\
\mathbb{E}_{\boldsymbol{z} \sim \mathcal{P}\text{ois}(\boldsymbol{z};\boldsymbol{\lambda})}\left[z_i z_j\right] &= \lambda_i \lambda_j + \delta_{ij}\lambda_i.
\end{aligned} \tag{22}$$

Here are the reconstruction losses for both Poisson and Gaussian VAEs with linear decoders, put side-by-side for comparison:

$$\boxed{\begin{aligned}
\text{Poisson:} &\qquad \mathcal{L}_{\text{recon.}}(\boldsymbol{x}; \boldsymbol{\lambda}, \boldsymbol{\Phi}) = \|\boldsymbol{x} - \boldsymbol{\Phi\lambda}\|_2^2 + \boldsymbol{\lambda}^T \text{diag}(\boldsymbol{\Phi}^T\boldsymbol{\Phi}), \\
\text{Gaussian:} &\qquad \mathcal{L}_{\text{recon.}}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\Phi}) = \|\boldsymbol{x} - \boldsymbol{\Phi\mu}\|_2^2 + (\boldsymbol{\sigma}^2)^T \text{diag}(\boldsymbol{\Phi}^T\boldsymbol{\Phi}).
\end{aligned}} \tag{23}$$

Given these assumptions, the NELBO (eq. (9)) for $\mathcal{P}$-VAE with a linear decoder becomes:

$$\boxed{\mathcal{L}_{\text{SC-PVAE}}(\boldsymbol{x}; \boldsymbol{\delta r}, r, \boldsymbol{\Phi}) = \|\boldsymbol{x} - \boldsymbol{\Phi\lambda}\|_2^2 + \boldsymbol{\lambda}^T \text{diag}(\boldsymbol{\Phi}^T\boldsymbol{\Phi}) + \beta \sum_{i=1}^{K} r_i f(\delta r_i).} \tag{24}$$

Recall that we have $f(y) = 1 - y + y \log y$ (see Fig. 6). We introduced the β term here to control the trade-off between the reconstruction and the KL term [102]. Additionally, we dropped the explicit dependence of $\boldsymbol{\delta r}(\boldsymbol{x})$ on the input image $\boldsymbol{x}$ to enhance readability.

B.6 Linear encoder

We can further simplify the $\mathcal{P}$-VAE architecture by making the encoder also linear. Let $\boldsymbol{W} \in \mathbb{R}^{K \times M}$ denote the encoder's weight matrix, and assume an exponential link function mapping the input to residual firing rates, i.e., $\boldsymbol{\delta r} = \exp(\boldsymbol{Wx})$.

Starting from eq. (24), substituting $\log \boldsymbol{\delta r} = \boldsymbol{Wx}$, and rearranging terms yields the following loss function for the $\langle \texttt{lin}|\texttt{lin}\rangle$ $\mathcal{P}$-VAE:

$$\mathcal{L}_{\text{Lin-PVAE}} = \boldsymbol{\lambda}^T \boldsymbol{\Phi}^T \boldsymbol{\Phi\lambda} + \boldsymbol{\lambda}^T \text{diag}(\boldsymbol{\Phi}^T\boldsymbol{\Phi} - \beta\boldsymbol{I}) + \boldsymbol{\lambda}^T(\beta\boldsymbol{W} - 2\boldsymbol{\Phi}^T)\boldsymbol{x} + \beta \sum_{i=1}^{K} r_i + \boldsymbol{x}^T\boldsymbol{x}. \tag{25}$$

C Architecture, training, and hyperparameter details

C.1 Datasets: additional details

We consider three datasets in this paper. We tile up the van Hateren dataset of natural images [104] and CIFAR10 into 16×16 patches and apply whitening and contrast normalization using the code made available by Boutin et al. [105]. This operation results in the following total number of samples:

- **van Hateren**: #train = 107,520, #validation = 28,224,
- **CIFAR$_{16 \times 16}$**: #train = 200,000, #validation = 40,000.

We use the MNIST dataset primarily for the downstream classification task. After the training is done, we use the following train/validation split to evaluate the models:

- **K-nearest neighbor classification** (tables 4 and 6): For this task, we only make use of the validation set for both training and testing of the classifier. We divide up the $N = 10,000$ validation samples into two disjoint sets of $N = 5,000$ samples each. We then draw random samples (without replacement) from the first half and use them for training the KNN classifier. We then test the performance on the other half.
- **Shattering dimensionality** (tables 4 and 6, last column): We use the entire MNIST training set ($N = 60,000$ samples) to train logistic regression classifiers on extracted representations. We then test the results using the entire validation set ($N = 10,000$ samples).

C.2 Architecture details

For sparse coding results, we focused on models with linear decoders. For the fully linear models (Figs. 4 and 10) both the encoder and decoder were linear layers, without bias.

For the convolutional components, we use residual layers without batch norm. For van Hateren and CIFAR$_{16 \times 16}$ datasets, the encoders had 5 layers ($2 \times$ conv each). The decoders had 8 convolutional layers ($1 \times$ conv each). For the MNIST dataset, the encoders had 7 layers ($2 \times$ conv each). The decoders had 10 convolutional layers ($1 \times$ conv each). For all convolutional encoders, the output from ResNet was followed by a learned pooling layer. The pooled output was then fed into a feed-forward layer inspired by Transformers [145], which includes a layer norm as the final operation, the output of which was fed into a linear layer that projects features into posterior distribution parameters. For all convolutional decoders, nearest neighbor upsampling was performed to scale up the spatial dimension of reconstructions, as suggested by Child [34].

We experimented with both leaky_relu and swish activation functions [146, 147], and found that swish consistently outperformed leaky_relu in all our experiments across datasets and VAE models.

Please see our code for the full architecture details.

C.3 Training details

We used a variety of learning rates and batch sizes, depending on the dataset and architecture. For $\langle \texttt{lin}|\texttt{lin} \rangle$ and $\langle \texttt{conv}|\texttt{lin} \rangle$ models, we used $lr = 0.005$, and for $\langle \texttt{conv}|\texttt{conv} \rangle$ models we used $lr = 0.002$. All models were trained using the AdaMax optimizer [148] with a cosine learning rate schedule [149]. Please see our code for the full details of training hyperparameters. Overall, we trained 195 VAE models, $n = 5$ seeds each, resulting in a total of $195 \times 5 = 975$ VAEs. For sparse coding models, we ran ISTA [81, 82] and LCA [80] with 270 hyperparameter combinations each. Training all models took roughly a week on 8 RTX 6000 Ada GPUs.

Temperature annealing for discrete VAEs. We also annealed the temperature from a large value to a smaller value during the same first half of training for $\mathcal{P}$-VAE and $\mathcal{C}$-VAE. We found that the specific functional form of temperature annealing (e.g., linear, exponential, etc.) did not matter as much as the final temperature (Fig. 9). For both $\mathcal{P}$-VAE and $\mathcal{C}$-VAE, we start from $T_{\text{start}} = 1.0$ and anneal down to $T_{\text{stop}} = 0.05$ for $\mathcal{P}$-VAE, and $T_{\text{stop}} = 0.1$ for $\mathcal{C}$-VAE. We found that the $\mathcal{C}$-VAE performance was not very sensitive to the choice of T_{stop}, corroborating previous reports [89, 98].

The $\mathcal{P}$-VAE was relatively more sensitive to the value of T_{stop}, and we found marginal improvements when reducing it from 0.1 to 0.05. See Fig. 9 for comprehensive experiments exploring the effect of the final temperature, as well as a "hard-forward" training method where we set $T = 0$ in the forward pass (ensuring integer samples) and use a non-zero T only during the backward pass (surrogate gradients). We find that our "relaxed Poisson" approach (Fig. 3) consistently outperforms the hard-forward approach.

KL annealing for VAEs. For all VAE models, we annealed the KL term during the first half of the training, which is known to be an effective trick in training VAEs [2, 33, 142, 150, 151].

C.3.1 Training: sparse coding models

To fit LCA and ISTA models, we explored a combination of 6 β schedules (same β as in eq. (1)), 3 numbers of iteration (for inference), 3 learning rates, and 5 different seeds (for dictionary initialization). The code for LCA was obtained from the public python library "lca-pytorch" ([152]), and the code for ISTA was obtained from public "sparsecoding" repository of the Redwood Center for Theoretical Neuroscience (with added clipping of coefficients to be nonnegative, following the thresholding step).

We explored learning rates of 1×10^{-1}, 1×10^{-2}, and 1×10^{-3}. We trained all models for 100 epochs. We scheduled the β parameters linearly, starting from β_{start}, and stepped it up every five epochs by β_{step}, until it reached β_{end}. We explored the following β schedules (expressed as β_{start}:β_{end}:β_{step}):

$$0.05{:}0.7{:}0.1, \quad 0.01{:}0.1{:}0.01, \quad 0.1{:}1.0{:}0.1, \quad 0.05{:}0.7{:}0.05, \quad 0.05{:}0.5{:}0.05, \quad 0.1{:}0.1{:}0$$

We also explored the inference iteration limits of 100, 500, and 900 iterations. We selected the best fits to include in the main results shown in Figs. 4 and 5.

D   Supplementary results

In this section, we include additional results that further support those reported in the main paper, including:

Table 5 contains the negative ELBO values for all VAE models with a linear decoder. This table reveals a comparable performance between using Monte Carlo samples to estimate gradients, versus optimizing the exact loss (see eqs. (4), (21), (23) and (24)), highlighting the effectiveness of our Poisson reparameterization algorithm.

Figure 7 uses the same data from the main paper Table 2 to visualize the effects.

Figure 8 shows the dependence of loss on latent dimensionality. We find that increasing the number of latent dimensions consistently improves ELBO for $\langle$`conv`|`lin`$\rangle$ architectures, but $\langle$`lin`|`lin`$\rangle$ models either overfit (for van Hateren) or fail to improve (for $\text{CIFAR}_{16 \times 16}$) once K becomes large.

Figure 9 demonstrates the robustness of our Poisson reparameterization trick (Algorithm 1) to variations in the temperature parameter. Importantly, we also explore a "hard-forward" training approach, where we fix $T = 0$ during the forward pass but allow $T > 0$ in the backward pass. This is also known as *surrogate gradients*. We find that, somewhat surprisingly, this hard-forward method performs significantly worse than our "relaxed Poisson" approach (Fig. 3).

Figure 10 shows how the distribution of KL values (or the norm of decoder weights in the case of linear decoders) can be used to determine dead neurons that don't contribute to the encoding of information.

Table 6 contains the full set of downstream classification results. Related to Table 4.

Figure 11 shows the performance of a simple linear classifier (logistic regression) trained on unsupervised representations learned by various $\langle$`conv`|`conv`$\rangle$ VAEs. We find that increasing the latent dimension (K) generally improves the performance of $\mathcal{P}$-VAE, but at lower dimensions, other methods like $\mathcal{L}$-VAE and $\mathcal{G}$-VAE can outperform it.

Figure 12 shows MNIST samples generated from the latent space of different $\langle$`conv`|`conv`$\rangle$ VAE models, as well as their reconstruction performance.

Table 5: The reparameterized gradient estimators work as well as exact ones, across datasets and encoder architectures (linear vs. conv). Note that exact gradients are only computable for linear decoders (see eqs. (21), (23) and (24)). The values are negative ELBO (lower is better), shown as mean$\pm$99% confidence interval calculated from $n = 5$ different random initializations. For MNIST, our use of Gaussian conditional likelihoods means the numerical performance values are not directly comparable to studies that use binarized MNIST with a cross-entropy decoder. EX, exact, MC, Monte-Carlo, ST, straight-through [107]. See also Table 2 and supplementary Fig. 7.

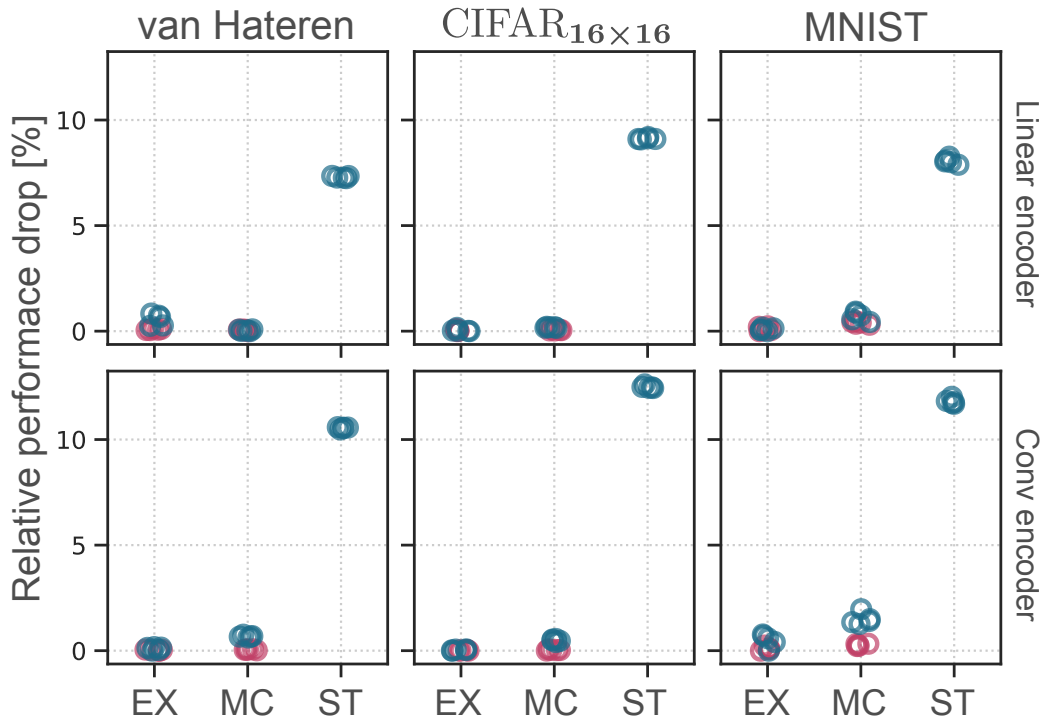| Model		van Hateren		CIFAR$_{16\times16}$		MNIST	
		$\langle$lin|lin$\rangle$	$\langle$conv|lin$\rangle$	$\langle$lin|lin$\rangle$	$\langle$conv|lin$\rangle$	$\langle$lin|lin$\rangle$	$\langle$conv|lin$\rangle$
	EX	$168.0_{\pm.8}$	$162.4_{\pm.2}$	$167.1_{\pm.2}$	$162.1_{\pm.1}$	$41.5_{\pm.1}$	$39.7_{\pm.2}$
$\mathcal{P}$-VAE	MC	$167.2_{\pm.1}$	$163.4_{\pm.1}$	$167.3_{\pm.1}$	$162.9_{\pm.2}$	$41.7_{\pm.2}$	$40.1_{\pm.2}$
	ST	$179.3_{\pm.1}$	$179.4_{\pm.1}$	$182.3_{\pm.1}$	$182.3_{\pm.2}$	$44.8_{\pm.1}$	$44.2_{\pm.1}$
$\mathcal{G}$-VAE	EX	$160.3_{\pm.1}$	$154.4_{\pm.1}$	$165.9_{\pm.1}$	$149.2_{\pm.0}$	$40.6_{\pm.1}$	$40.0_{\pm.1}$
	MC	$160.3_{\pm.1}$	$154.4_{\pm.1}$	$165.9_{\pm.1}$	$149.2_{\pm.1}$	$40.7_{\pm.1}$	$40.1_{\pm.0}$
	EX	$174.9_{\pm.1}$	$186.3_{\pm.8}$	$177.1_{\pm.1}$	$180.6_{\pm.5}$	$56.1_{\pm.7}$	$59.1_{\pm.0}$
$\mathcal{C}$-VAE	MC	$170.5_{\pm.1}$	$171.9_{\pm.2}$	$174.7_{\pm.1}$	$176.5_{\pm.1}$	$39.7_{\pm.2}$	$59.1_{\pm.0}$
	ST	$174.2_{\pm.2}$	$181.1_{\pm.3}$	$180.2_{\pm.0}$	$185.6_{\pm.2}$	$49.3_{\pm.1}$	$63.8_{\pm3.4}$
$\mathcal{L}$-VAE	EX	$167.3_{\pm.0}$	$159.0_{\pm.2}$	$170.1_{\pm.1}$	$154.3_{\pm.1}$	$42.1_{\pm.1}$	$41.0_{\pm.0}$
	MC	$167.3_{\pm.0}$	$159.2_{\pm.2}$	$170.1_{\pm.1}$	$154.5_{\pm.1}$	$42.1_{\pm.0}$	$41.0_{\pm.0}$

Figure 7: Performance drop relative to the best fit. Blue circles indicate $\mathcal{P}$-VAE results, red circles indicate $\mathcal{G}$-VAE results, and each set of $n = 5$ circles corresponds to five random initializations. Using Monte Carlo samples [153] and our Poisson reparameterization trick (Algorithm 1) to estimate gradients performs comparably to using exact gradients (see eqs. (21), (23) and (24)). Table 2 provides a tabular summary of these results. EX, exact, MC, Monte-Carlo, ST, straight-through [107].
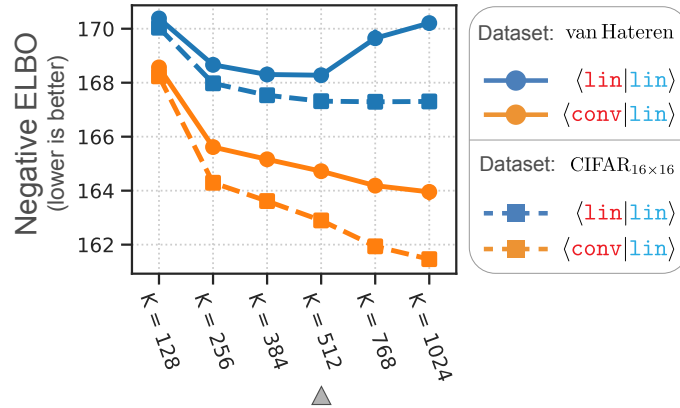
Figure 8: The effect of latent dimensionality on model performance across datasets and encoder architectures. For all convolutional encoder cases, ELBO improves as a function of latent dimensionality. However, for linear encoders, we see that the van Hateren dataset starts to overfit for $K > 512$, and it stagnates for the $\text{CIFAR}_{16\times16}$ dataset. In conclusion, more expressive encoders can find nonlinear features, represented using additional latent dimensions, but simple linear encoders struggle to utilize additional dimensions. The gray triangle indicates the setting used in the main results.
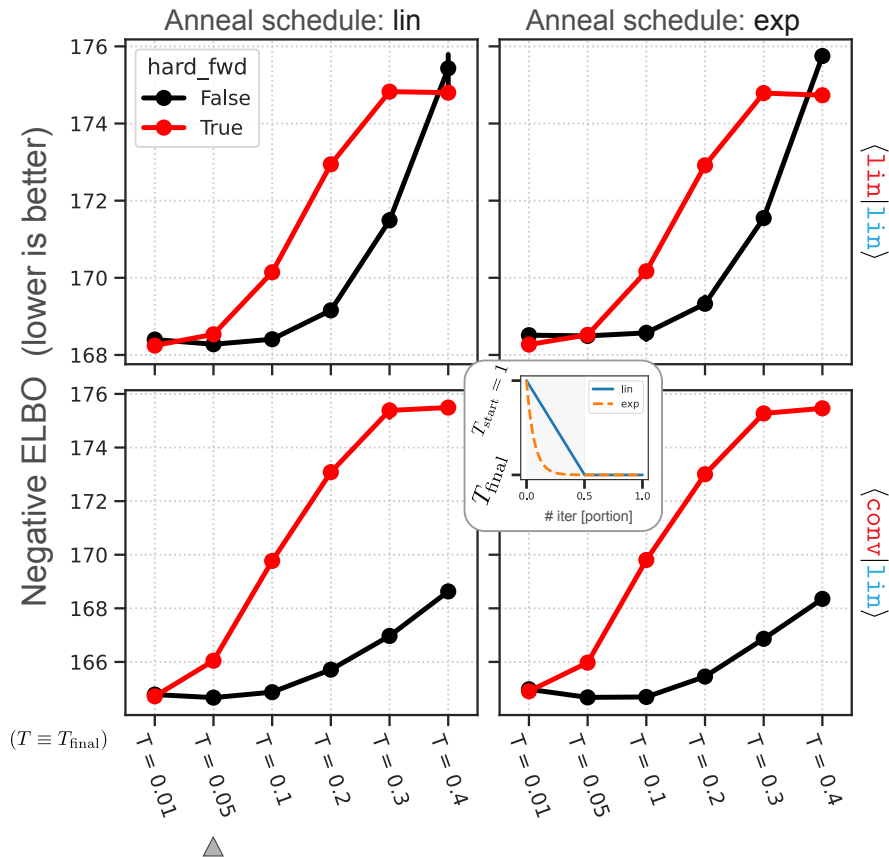

Figure 9: Performance as a function of the final temperature (T_{final}), annealing schedule (linear vs. exponential; inset), and the "hard-forward" approach. The hard-forward approach uses exact integer samples ($T = 0$) in the forward pass and applies nonzero temperatures only in the backward pass (i.e., "surrogate gradients"). Although all results are evaluated at $T = 0$ during testing, the hard-forward approach still underperforms our "relaxed Poisson" method (Fig. 3), which employs continuous (floating) samples during training due to a non-zero T (Algorithm 1). The gray triangle indicates the setting used in the main results: $T_{\text{final}} = 0.05$ with a linear annealing schedule.
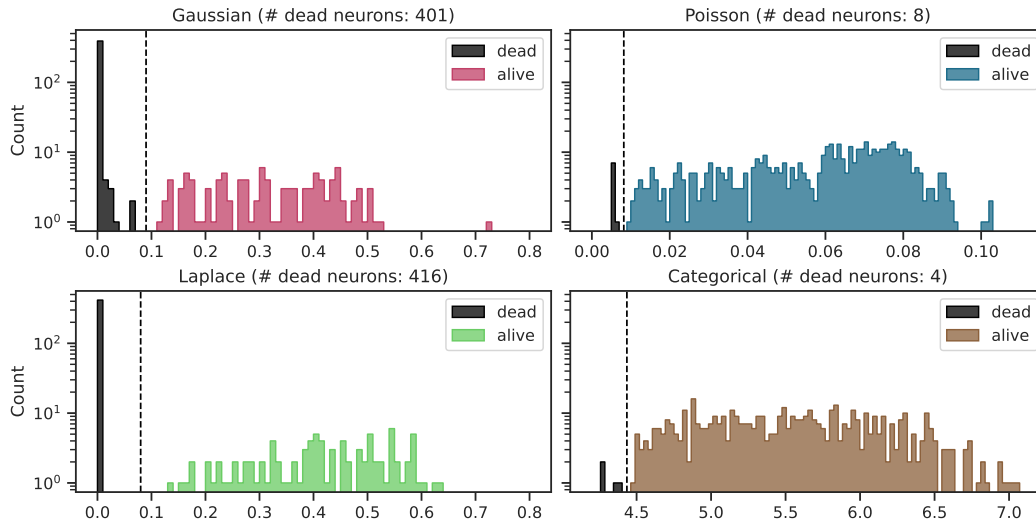
Figure 10: Identifying dead neurons using a histogram-based method. We bin the KL values and determine the gap between small values and larger ones. We identify neurons with KL values lower than the identified threshold (black dashed lines) and pronounce them dead. The figure shows the distribution of KL values over all neurons ($K = 512$) for $\mathcal{P}$-VAE, $\mathcal{G}$-VAE, and $\mathcal{L}$-VAE. The KL term is a single number for the $\mathcal{C}$-VAE because its latent space consists of a single one-hot categorical distribution with $K = 512$ categories. Therefore, for the $\mathcal{C}$-VAE, we use the distribution of decoder weight norms instead. These are the same models shown in Fig. 4, where both encoder and decoder are linear. Table 3 uses this method to quantify the proportion of active neurons for VAEs across different datasets and the choice of encoder architectures.

Table 6: Geometry of representations. Full set of results. Related to Table 4.

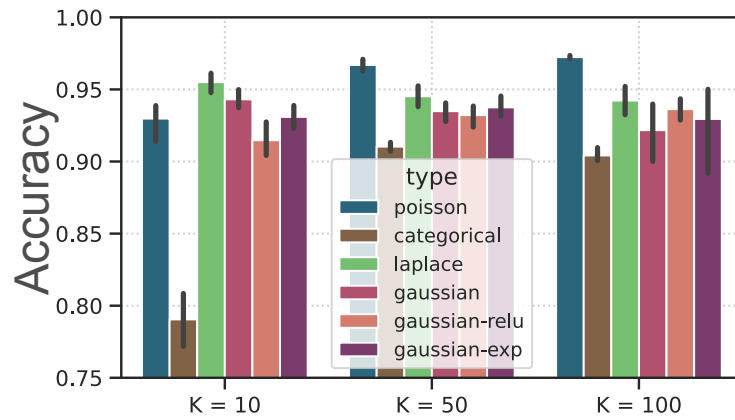| Latent dim.	Model	KNN classification (N, # labeled samples)			Shattering dim.
		$N = 200$	$N = 1{,}000$	$N = 5{,}000$	
$K = 10$	$\mathcal{P}$-VAE	$\mathbf{0.815}_{\pm.002}$	$\mathbf{0.919}_{\pm.001}$	$\mathbf{0.946}_{\pm.017}$	$\mathbf{0.797}_{\pm.009}$
	$\mathcal{C}$-VAE	$0.705_{\pm.002}$	$0.800_{\pm.002}$	$0.853_{\pm.040}$	$\mathbf{0.795}_{\pm.006}$
	$\mathcal{L}$-VAE	$0.757_{\pm.003}$	$0.869_{\pm.002}$	$\mathbf{0.924}_{\pm.028}$	$0.751_{\pm.008}$
	$\mathcal{G}$-VAE	$0.673_{\pm.003}$	$0.813_{\pm.002}$	$0.891_{\pm.033}$	$0.758_{\pm.007}$
	$\mathcal{G}$-VAE $_{+\mathrm{relu}}$	$0.694_{\pm.003}$	$0.817_{\pm.003}$	$0.877_{\pm.045}$	$0.762_{\pm.007}$
	$\mathcal{G}$-VAE $_{+\mathrm{exp}}$	$0.642_{\pm.003}$	$0.784_{\pm.002}$	$0.863_{\pm.032}$	$0.737_{\pm.008}$
$K = 50$	$\mathcal{P}$-VAE	$\mathbf{0.825}_{\pm.002}$	$\mathbf{0.927}_{\pm.001}$	$\mathbf{0.957}_{\pm.005}$	$\mathbf{0.935}_{\pm.003}$
	$\mathcal{C}$-VAE	$0.770_{\pm.002}$	$0.880_{\pm.001}$	$0.920_{\pm.009}$	$0.899_{\pm.004}$
	$\mathcal{L}$-VAE	$0.710_{\pm.003}$	$0.836_{\pm.003}$	$0.902_{\pm.038}$	$0.770_{\pm.007}$
	$\mathcal{G}$-VAE	$0.604_{\pm.003}$	$0.746_{\pm.002}$	$0.837_{\pm.022}$	$0.743_{\pm.007}$
	$\mathcal{G}$-VAE $_{+\mathrm{relu}}$	$0.710_{\pm.002}$	$0.844_{\pm.002}$	$0.904_{\pm.026}$	$0.786_{\pm.006}$
	$\mathcal{G}$-VAE $_{+\mathrm{exp}}$	$0.694_{\pm.003}$	$0.836_{\pm.002}$	$0.906_{\pm.027}$	$0.762_{\pm.007}$
$K = 100$	$\mathcal{P}$-VAE	$\mathbf{0.807}_{\pm.002}$	$\mathbf{0.925}_{\pm.001}$	$\mathbf{0.958}_{\pm.013}$	$\mathbf{0.949}_{\pm.002}$
	$\mathcal{C}$-VAE	$0.753_{\pm.002}$	$0.876_{\pm.001}$	$0.925_{\pm.005}$	$0.884_{\pm.004}$
	$\mathcal{L}$-VAE	$0.701_{\pm.004}$	$0.830_{\pm.003}$	$\mathbf{0.896}_{\pm.046}$	$0.767_{\pm.007}$
	$\mathcal{G}$-VAE	$0.636_{\pm.003}$	$0.789_{\pm.002}$	$0.875_{\pm.024}$	$0.763_{\pm.007}$
	$\mathcal{G}$-VAE $_{+\mathrm{relu}}$	$0.757_{\pm.002}$	$0.881_{\pm.001}$	$\mathbf{0.933}_{\pm.019}$	$0.818_{\pm.006}$
	$\mathcal{G}$-VAE $_{+\mathrm{exp}}$	$0.695_{\pm.003}$	$0.846_{\pm.002}$	$0.918_{\pm.024}$	$0.793_{\pm.006}$

Figure 11: Downstream classification performance using a simple linear classifier. After unsupervised training of $\langle \texttt{conv}|\texttt{conv} \rangle$ VAEs on MNIST, we extracted latent representations and applied logistic regression. For $K = 100$, $\mathcal{P}$-VAE achieves the highest accuracy, while for $K = 10$, both $\mathcal{L}$-VAE and $\mathcal{G}$-VAE outperform it.
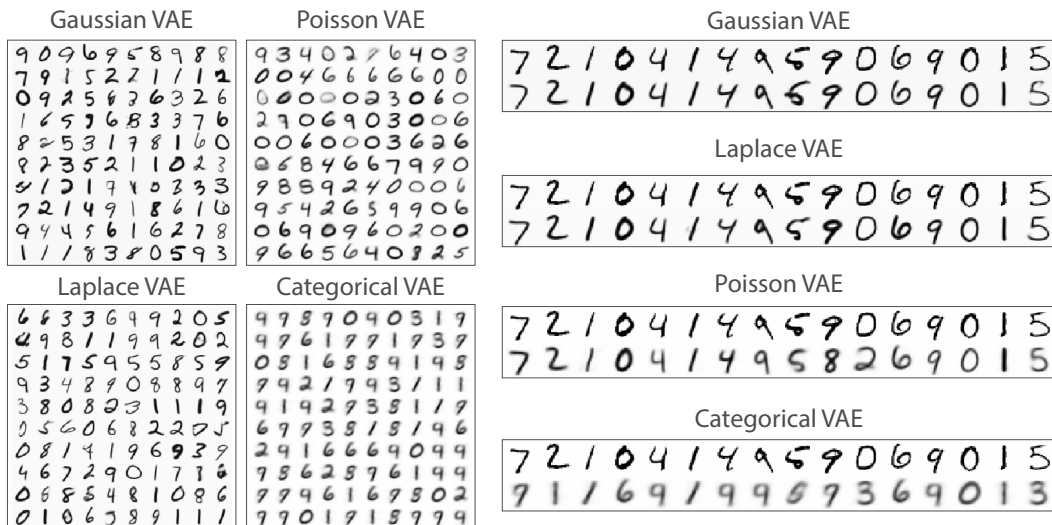

Figure 12: Generated samples (left) and reconstruction performance (right). These results shown here are from models with a $\langle \texttt{conv}|\texttt{conv} \rangle$ architectures and latent dimensionality of $K = 10$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: we provide comprehensive theoretical and empirical evidence to support our claims of (1) introducing the $\mathcal{P}$-VAE and its reparameterization trick; (2) $\mathcal{P}$-VAE containing amortized sparse coding as a special case; (3) $\mathcal{P}$-VAE largely avoiding posterior collapse; and (4) $\mathcal{P}$-VAE facilitating linear separability of categories at better sample efficiency, in sections 3 and 4, and supplemental appendices B to D.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of (1) Poisson possibly not being a perfect description of cortical activity, and (2) amortization gap, are shown explicitly and thoroughly discussed in sections 4 and 5. Specifically, we have a dedicated paragraph for limitations in section 5. We evaluated our claims using multiple well-known datasets such as the van Hateren natural images [104], CIFAR10, and MNIST, on tasks such as reconstruction, sparse coding, and classification.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We provide the full derivation of the $\mathcal{P}$-VAE loss function, which is self-contained in the paper (section 3) and supplement (appendix B).

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We disclose all details relating to the algorithm, including the optimization objective (eq. (3)), architecture and training details (appendix C), and pseudo-code for Poisson reparameterized sampling (Algorithm 1). In addition, we intend to release all code and data needed for replicating our work.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Our code, data, and model checkpoints are available from the following GitHub repository: https://github.com/hadivafaii/PoissonVAE.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All details about how the data was used for training and testing, as well as which hyperparameters were used, are available at appendix C. In addition, the provided code replicates our results and therefore contains all details of implementation.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: As stated in section 4, the paper reports confidence intervals and t-test significance tests, using false discovery rate (FDR) correction for multiple comparisons. The exact implementation details are included in the provided code for reproducibility.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide details about our compute resources (GPUs), and duration of training in section 4.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: Our paper follows the code of ethics, including preserving anonymity (such as in releasing code anonymously).

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: Our paper is considered foundational research, and does not target practical tasks that can be deployed outside of the research field. Thus we do not anticipate negative social impacts from this work.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our paper utilizes publicly domain datasets (not scraped), and poses no safety risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We properly cite papers that introduce algorithms (such as LCA), datasets (such as MNIST, CIFAR10, van Hateren), and code (such as LCA and ISTA).

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: New assets introduced in the paper consist of our codebase which includes notebooks to replicate our experiments and analyses, and contains documentation.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.