
NEGATIVE BINOMIAL VARIATIONAL AUTOENCODERS FOR OVERDISPERSED LATENT MODELING

Yixuan Zhang

School of Statistics and Data Science
Southeast University

Wenxin Zhang

School of Computer Science and Technology
University of the Chinese Academy of Sciences

Hua Jiang

School of Computer Science
University of Sydney

Quyu Kong

Alibaba Cloud

Feng Zhou

Center for Applied Statistics and School of Statistics
Renmin University of China

August 8, 2025

ABSTRACT

Biological neurons communicate through spike trains—discrete, irregular bursts of activity that exhibit variability far beyond the modeling capacity of conventional variational autoencoders (VAEs). Recent work, such as the Poisson-VAE, makes a biologically inspired move by modeling spike counts using the Poisson distribution. However, they impose a rigid constraint: equal mean and variance, which fails to reflect the true stochastic nature of neural activity. In this work, we challenge this constraint and introduce NegBio-VAE, a principled extension of the VAE framework that models spike counts using the negative binomial distribution. This shift grants explicit control over dispersion, unlocking a broader and more accurate family of neural representations. We further develop two ELBO optimization schemes and two differentiable reparameterization strategies tailored to the negative binomial setting. By introducing one additional dispersion parameter, NegBio-VAE generalizes the Poisson latent model to a negative binomial formulation. Empirical results demonstrate this minor yet impactful change leads to significant gains in reconstruction fidelity, highlighting the importance of explicitly modeling overdispersion in spike-like activations.

1 Introduction

Although artificial neural networks (ANNs) have historically been inspired by the brain, they differ fundamentally from biological neural systems in their structure, neural computation, and learning mechanisms [40, 1]. One of the most striking distinctions lies in how information is transmitted. In the brain, neurons communicate via sequences of action potentials, also known as spike trains [30]. A growing body of research emphasizes the critical role of these spike trains in neural encoding [26, 2, 13]. Therefore, bridging the gap between artificial and biological systems is an important research direction for advancing ANNs towards brain-like architectures. This not only improves performance but also deepens our understanding of the mechanisms underlying biological brain function [43, 12].

One promising direction in developing brain-like architectures is the variational autoencoder (VAE) [19], a powerful generative model grounded in Bayesian inference that learns structured latent representations of data. VAEs are considered brain-like because their approach to learning latent variables closely resembles how the brain encodes information from the external world [27, 41, 38]. While VAEs have shown success in various domains, they typically employ continuous latent variables, which contrasts with the discrete nature of biological neurons that fire all-or-none action potentials. This discrepancy between the continuous latent variables in VAEs and the discrete spike counts observed in biological neurons has been addressed by recent works [14, 44, 42].

The main improvement presented in this paper is based on the Poisson VAE (\mathcal{P} -VAE) [42], which encodes inputs as discrete spike counts drawn from a Poisson distribution. The Poisson distribution, while offering a biologically plausible model for spike counts, it inherently assumes that the mean and variance of the spike counts are equal. However,



Figure 1: Reconstructions from different VAEs (256 dims) on MNIST. Despite using the same latent dimensionality, NegBio-VAE recovers sharper digit structures (e.g., the gap in **0** and the loop in **4**) that others miss. This advantage stems from its ability to model overdispersed latent spike counts, providing greater variance flexibility and enabling sharper distinctions in high-variability regions.

empirical studies in neuroscience have consistently shown that spike counts often exhibit overdispersion, a condition where the variance significantly exceeds the mean [39, 28, 37]. This overdispersion is a defining characteristic of neural activity, suggesting that Poisson models may underestimate the inherent variability of spike trains, especially in complex or noisy neural environments.

To address this limitation and align the brain-like VAE with both biological and perceptual variability, we adopt the negative binomial (NB) distribution [36], a two-parameter generalization of the Poisson distribution. This introduces an additional dispersion parameter, allowing us flexibly model discrete latent spike counts with excess variance. By incorporating the NB distribution into the VAE framework, we propose NegBio-VAE, a biologically inspired generative model that maintains the interpretability of spike-based representations while more accurately capturing the statistical properties of neural activity. Empirically, we demonstrate that the added flexibility of the NB latent space not only improves biological plausibility but also enhances the model’s capacity to capture fine-grained details in image reconstruction Fig. 1. By decoupling the mean and variance of spike-like activations, NegBio-VAE better accommodates overdispersed latent signals, which are poorly modeled by conventional VAEs. As a result, the learned latent codes are more expressive, leading to clearer reconstructions and stronger downstream performance even with very low-dimensional latents.

Our main contributions can be summarized as follows: (1) NegBio-VAE introduces an additional parameter to explicitly model the overdispersion of latent spike counts. This decouples the mean and variance of latent spike counts, allowing the model to better reflect the variability of neural responses while preserving biological plausibility. (2) We develop two KL estimation strategies: Monte Carlo for flexibility and dispersion sharing for stability. Additionally, we introduce two reparameterizations (Gumbel-Softmax and continuous-time) to support gradient-based learning with discrete latent variables. (3) NegBio-VAE achieves sharper reconstructions, more expressive and active latent spaces, and superior downstream performance in low dimensions. It also remains robust across architectures and inference settings, highlighting its broad applicability.

2 Related Works

Brain-like ANNs, emerging at the intersection of neuroscience and machine learning, aim to mirror the brain’s functionality and structure. Related works can be categorized into two types: spiking neural networks (SNNs) and brain-like generative models. SNNs [12, 5, 48, 9, 24], like biological neurons, use discrete spikes for communication instead of continuous activations as in traditional ANNs. A notable model is the leaky integrate-and-fire (LIF) model, which simulates the temporal dynamics of spike generation. The second category includes generative models that learn data representations similar to how brain processes sensory information. Key works in this area include brain-like VAEs [15, 45, 42], GANs [20, 35, 10], and diffusion models [25, 4, 16]. Our work extends the \mathcal{P} -VAE [42] by incorporating a NB distribution to better capture overdispersion in latent spike counts, providing a more accurate representation of neural activity.

Discrete VAEs are typically categorized into two types: discrete representations and discrete data. In VAEs with discrete representations, discrete latent variables model the underlying structure of the data, capturing discrete, non-continuous patterns. Most current works on discrete-representation VAEs use categorical distributions for the latent variables [44, 11, 14, 8]. Other works employ Bernoulli [15, 34] or Poisson distributions [42, 46]. These methods have achieved significant success in speech synthesis and image generation. The second category focuses on VAEs for discrete data, such as text, categorical, or count data. These models reconstruct non-continuous data, making them suitable for tasks like natural language processing and structured prediction. Recent works include [47, 32], with [47] employing the NB distribution to model count data while keeping the latent variables continuous. In contrast, our work is a discrete-representation VAE that uses the NB distribution to model count-type latent variables.

3 Preliminaries

In this section, we provide background knowledge on VAEs and \mathcal{P} -VAE. We first review the standard VAE framework to establish the foundation for understanding latent variable modeling. Then, we discuss how \mathcal{P} -VAE adapts this framework to represent neural spike counts using the Poisson distribution.

3.1 Variational Autoencoder

VAE [18] is a probabilistic generative model that defines a joint distribution $p(\mathbf{x}, \mathbf{z})$ over observed data \mathbf{x} and latent variables \mathbf{z} . It generates data \mathbf{x} by first sampling $\mathbf{z} \sim p(\mathbf{z})$ (the prior) and then decoding it using $p_\theta(\mathbf{x} | \mathbf{z})$. To infer the latent variables, VAEs introduce an approximate posterior $q_\phi(\mathbf{z} | \mathbf{x})$, which estimates the true posterior distribution and is known as the encoder. The model parameters are learned by maximizing the Evidence Lower Bound (ELBO), a tractable surrogate for the marginal log-likelihood $\log p(\mathbf{x})$, derived through variational inference. The ELBO objective encourages accurate reconstruction of the input data while regularizing the latent space to learn meaningful representations, and is usually given as:

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

The first term enforces faithful reconstruction of the data by encouraging the decoder $p_\theta(\mathbf{x} | \mathbf{z})$ to accurately reproduce the input. The second term penalizes deviations from the approximate posterior $q_\phi(\mathbf{z} | \mathbf{x})$ from the prior $p(\mathbf{z})$, typically enforcing structure or smoothness in the latent space. To enable gradient-based optimization, VAEs employ the reparameterization trick [18], introducing a differentiable sampling step between the encoder and decoder. In standard implementations, both the prior $p(\mathbf{z})$ and posterior $q_\phi(\mathbf{z} | \mathbf{x})$ are modeled as isotropic Gaussians, which simplifies computation but limits expressiveness in certain settings.

3.2 Poisson VAE

To better mimic biological neuron activity, the \mathcal{P} -VAE [42] was proposed to model spike counts as discrete latent variables. Specifically, it uses the Poisson distribution to represent the spike counts of K neurons, with the latent variable $\mathbf{z} \in \mathbb{Z}_0^{+K}$. The prior and variational posterior are defined as:

$$\text{Prior: } p(\mathbf{z}) = \text{Poi}(\mathbf{z}; \mathbf{r}) \quad \text{Posterior: } q(\mathbf{z} | \mathbf{x}) = \text{Poi}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x})),$$

where both the prior Poisson and the posterior Poisson are factorized, i.e., $\text{Poi}(\mathbf{z}) = \prod_{i=1}^K \text{Poi}(z_i)$. Here, $\mathbf{r} \in \mathbb{R}^{+K}$ denotes the prior firing rates, and $\mathbf{r} \odot \delta_r(\mathbf{x})$ gives the posterior firing rates, with \odot denoting element-wise multiplication. The encoder output $\delta_r(\mathbf{x}) \in \mathbb{R}^{+K}$ modulates the ratio of posterior to prior firing rates based on the input. In contrast to standard VAEs, where latent variables are continuous and typically drawn from a Gaussian, the \mathcal{P} -VAE models \mathbf{z} as a vector of discrete counts, which better resembles neural firing behavior. The objective of the \mathcal{P} -VAE is given by:

$$\mathcal{L}_{\mathcal{P}\text{-VAE}} = \mathbb{E}_{\text{Poi}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}))} [\log p_\theta(\mathbf{x} | \mathbf{z})] + \sum_{i=1}^K r_i g(\delta_{r_i}), \quad (1)$$

where $g(a) = 1 - a + a \log a$ corresponds to the KL divergence between the posterior and prior Poisson distributions.

4 Methodology

A key limitation of the Poisson distribution is its restrictive assumption that the mean and variance of spike counts are equal. This assumption fails to capture the overdispersion frequently observed in neural data. To address this, we propose the NegBio-VAE, which applies a more flexible NB distribution. As a two-parameter generalization of the Poisson, the NB distribution introduces a dispersion parameter that allows the variance to exceed the mean. This makes it more suitable for modeling overdispersed spike counts. The NB distribution has been widely applied in various fields, such as spiking neuron models [31], RNA sequence analysis [7], and language modeling [49].

We begin by defining the prior and posterior distributions over the discrete latent variables $\mathbf{z} \in \mathbb{Z}_0^{+K}$ as $p(\mathbf{z}) = \text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p})$ and $q(\mathbf{z} | \mathbf{x}) = \text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x}))$, respectively. Similar to \mathcal{P} -VAE, both the prior and posterior NB distribution are factorized, i.e., $\text{NB}(\mathbf{z}) = \prod_{i=1}^K \text{NB}(z_i)$, $\delta_r(\mathbf{x})$ and $\delta_p(\mathbf{x})$ are outputs of the encoder, which captures the ratio of the posterior parameters to the prior parameters. With this setup, the ELBO of NegBio-VAE becomes:

$$\mathcal{L} = \mathbb{E}_{\text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x}))} [\log p_\theta(\mathbf{x} | \mathbf{z})] - \mathcal{D}_{\text{KL}}[\text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x})) || \text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p})]. \quad (2)$$

While this formulation enables greater flexibility, it also introduces two key technical challenges during the training of NegBio-VAE: (1) The second term in Eq. (2) requires calculating the KL divergence between two NB distributions; (2) The first term in Eq. (2) requires reparameterized sampling from the NB distribution. We address each of these issues in the following sections.

4.1 KL Divergence between NB Distributions

In both vanilla VAE and \mathcal{P} -VAE, the KL term is tractable due to closed-form solutions for Gaussian and Poisson distributions. However, no such form exists for the KL divergence between two NB distributions, which poses the first challenge for training NegBio-VAE. To address this, we propose two strategies: a **Monte Carlo** method for direct approximation, and a **dispersion sharing** technique that simplifies the KL divergence by partially tying posterior parameters to the prior.

(1) **Monte Carlo.** Optimizing the ELBO for NegBio-VAE presents a significant challenge due to the absence of an analytical solution for the KL divergence between two NB distributions. This prevents direct computation of the KL term in the objective function. However, we can overcome this issue by resorting to Monte Carlo estimation. Using the identity $\mathcal{D}_{\text{KL}}[q(\mathbf{z})||p(\mathbf{z})] = \mathbb{E}_{q(\mathbf{z})} [\log q(\mathbf{z}) - \log p(\mathbf{z})]$, we can approximate the KL divergence by Monte Carlo sampling from the variational posterior and computing the sample average of the log-density difference between the posterior and the prior.

Substituting this expression into the ELBO in Eq. (2) we obtain the following objective:

$$\mathcal{L} = \mathbb{E}_{\text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x}))} [\log p_\theta(\mathbf{x} | \mathbf{z}) - \log \text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x})) + \log \text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p})].$$

Clearly, as long as we can implement reparameterized sampling from the NB distribution, we can use the above objective function to train NegBio-VAE.

(2) **Dispersion Sharing.** Although the KL divergence between two general NB distributions, $\text{NB}(z; r_1, p_1)$ and $\text{NB}(z; r_2, p_2)$, does not have an analytical solution, a tractable analytical form exists when the dispersion parameters are shared, i.e., $r_1 = r_2 = r$.

Based on this observation, we propose an alternative strategy for computing the KL term in the NegBio-VAE by constraining the prior $\text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p})$ and the posterior $\text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x}))$ to share the same dispersion parameter, i.e., setting $\delta_r(\mathbf{x})$ to be $\mathbf{1}$. Then, the KL term in Eq. (2) admits a closed-form solution:

$$\mathcal{D}_{\text{KL}}[\text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p} \odot \delta_p(\mathbf{x})) || \text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p})] = \sum_{i=1}^K r_i g(p_i, \delta_{p_i}), \quad (3)$$

where $g(a, b)$ is defined as: $g(a, b) = \log b + \frac{1-ab}{ab} \log \left(\frac{1-ab}{1-a} \right)$, with $a \in (0, 1)$ and $b > 0$. The complete derivation can be found in Appendix A. Then, the final NegBio-VAE objective becomes:

$$\mathcal{L} = \mathbb{E}_{\text{NB}(\mathbf{z}; \mathbf{r}, \mathbf{p} \odot \delta_p(\mathbf{x}))} [\log p_\theta(\mathbf{x} | \mathbf{z})] + \sum_{i=1}^K r_i g(p_i, \delta_{p_i}). \quad (4)$$

Importantly, *sharing the same dispersion parameter between the prior and posterior does not imply that they have identical means or variances*. For the NB distribution, the mean is given by $r(1-p)/p$ and the variance by $r(1-p)/p^2$. Thus, even when r is the same for both, different p still allows the posterior to capture different distributional properties from the prior.

Both methods have their advantages and limitations. The Monte Carlo approach avoids assumptions about the variational posterior but introduces higher gradient variance, potentially leading to unstable training. In contrast, the dispersion-sharing method enables analytical computation of the KL divergence and reduces gradient variance by assuming a shared dispersion parameter. Notably, this constraint does not compromise the model’s ability to capture overdispersion, offering a practical trade-off between expressiveness and training stability. In our experiments, we thoroughly compare the performance of both strategies, the results are in Section 5.2 and Appendix D.3.

4.2 Reparameterized Sampling for NB Distribution

The second challenge in training NegBio-VAE lies in the sampling process. The expectation term in the ELBO requires reparameterized sampling from the NB distribution to allow efficient gradient-based optimization. Reparameterizing discrete distributions is more challenging compared to continuous ones, but it can be achieved through suitable relaxation techniques. In this section, we describe how to apply reparameterization to the NB distribution by leveraging a key property: the NB distribution can be represented as a continuous mixture of Poisson distributions, where the mixing weight being a Gamma distribution:

$$\text{NB}(z; r, p) = \int_0^\infty \text{Poi}(z|\lambda) \text{Gamma}(\lambda; r, \frac{p}{1-p}) d\lambda. \quad (5)$$

This implies that a sample from $\text{NB}(z; r, p)$ can be obtained by first sampling $\lambda \sim \text{Gamma}(r, \frac{p}{1-p})$, followed by sampling $z \sim \text{Poi}(\lambda)$.

The first step, sampling from the Gamma distribution, is straightforward to reparameterize. In practice, PyTorch’s `Gamma.rsample()` function supports gradient propagation, as it uses the Marsaglia-Tsang algorithm in its underlying implementation and ensures differentiability through implicit gradient computation. The second step, sampling from the Poisson distribution, is more challenging, as it lacks a standard reparameterizable form. To address this, we adopt approximate relaxation techniques such as **Gumbel-Softmax Relaxation**[14] and **Continuous-Time Simulation**[42]. Both methods rely on a temperature parameter to transform “hard” discrete counts into “soft” discrete counts, thereby enabling differentiability. For implementation details, please see Appendix B.1.

(1) Gumbel-Softmax Relaxation. To enable differentiable sampling from a Poisson distribution, we adopt a relaxation-based strategy that treats the Poisson as a categorical distribution over a truncated support $\{0, 1, \dots, Z_{\max}\}$. By applying the Gumbel-Softmax trick [14], we construct a soft approximation of the discrete counts:

$$\tilde{z} = \sum_{z=0}^{Z_{\max}} z \cdot \text{softmax} \left(\frac{\log \text{Poi}(z) + \epsilon_z}{\tau} \right),$$

where $\epsilon_z \sim \text{Gumbel}(0, 1)$ is an i.i.d. Gumbel noise and $\tau > 0$ is a temperature parameter controlling the degree of relaxation. As $\tau \rightarrow 0$, the soft sample \tilde{z} converges to the Poisson distribution.

(2) Continuous-Time Simulation. This method leverages the connection between the Poisson distribution and homogeneous Poisson processes. Specifically, it models a Poisson-distributed count as the number of events occurring within the interval $[0, 1]$, where inter-arrival times follow an exponential distribution with rate λ . The soft count is computed by simulating the inter-arrival times and accumulating a temperature-smoothed approximation of the total event count:

$$\tilde{z} = \sum_{n=1}^M \sigma \left(\frac{1 - S_n}{\tau} \right), \quad \text{where } S_n = \sum_{i=1}^n s_i, \quad 1 \leq n \leq M, \quad \{s_i\}_{i=1}^M \sim \text{Exponential}(\lambda),$$

where $\sigma(\cdot)$ is the sigmoid function, $\tau > 0$ is a temperature parameter, and $\tau \rightarrow 0$ converges to the Poisson distribution. This approach allows for differentiable Poisson sampling through reparameterizable exponential sampling.

Both Gumbel-Softmax and continuous-time are used for the Poisson step in the NB reparameterization sampling. Theoretically, both approaches are valid. Empirically, under the same temperature, we find that the continuous-time relaxation tends to produce smoother spike counts, while Gumbel-Softmax encourages more discrete ones. A detailed comparison of the two methods is provided in Section 5.2.

4.3 Implementation

The encoder $\text{NB}(\mathbf{z}; \mathbf{r} \odot \delta_r(\mathbf{x}), \mathbf{p} \odot \delta_p(\mathbf{x}))$ is implemented as a neural network that takes input \mathbf{x} and outputs $\delta_p(\mathbf{x})$, and optionally $\delta_r(\mathbf{x})$. The decoder $p_\theta(\mathbf{x} | \mathbf{z})$ is modeled as a Gaussian: $p_\theta(\mathbf{x} | \mathbf{z}) = \mathcal{N}(\mathbf{x}; f_\theta(\mathbf{z}), \sigma^2 \mathbf{I})$, where σ^2 is a hyperparameter. This results in a reconstruction term $\log p_\theta(\mathbf{x} | \mathbf{z}) = -\frac{1}{2\sigma^2} \|\mathbf{x} - f_\theta(\mathbf{z})\|_2^2 + \text{const}$, which is equivalent to applying a coefficient $\beta = 2\sigma^2$ to the KL term, balancing the trade-off between reconstruction accuracy and adherence to the prior. In experiments, we investigate the effect of β (or equivalently, σ^2) on model performance.

5 Experiments

In this section, we compare NegBio-VAE with well-known VAE models on standard benchmark datasets. These experiments are designed to assess the effectiveness of our model in terms of reconstruction quality, latent representation, and biological plausibility.

5.1 Experimental Setup

Datasets. We evaluate our method across a variety of widely-used benchmark datasets: **MNIST** [23, 6], **CIFAR_{16×16}** [21], **Omniglot** [22], and **SVHN** [29]. MNIST consists of grayscale handwritten digits with a simple, low-dimensional structure. CIFAR_{16×16} is a downsampled version of CIFAR-10, containing colored natural images with diverse textures and object structures. Omniglot includes handwritten characters from over 50 alphabets, providing a challenging setting with high visual diversity. SVHN comprises real-world images of house numbers with significant background noise and intra-class variation. Pre-processing details are provided in Appendix B.2.

Metrics. We evaluate NegBio-VAE using both reconstruction metrics and latent space diagnostics. Reconstruction quality is assessed via Mean Squared Error (**MSE**), Fréchet Inception Distance (**FID**), and Inception Score (**IS**), measuring pixel-wise error, distributional similarity, and sample fidelity/diversity, respectively. To assess latent space properties, we compute the overdispersion index (**ODI**, also known as the Fano Factor in neuroscience), where is defined as $\text{Var}(z_i)/(\mathbb{E}[z_i] + \epsilon)$, where values greater than 1 indicate deviations from the Poisson distribution and justify the use of the NB distribution. We also report the dead neuron ratio (**DNR**), defined as the fraction of latent units with near-zero KL divergence. Metric definitions are detailed in Appendix D.

Baselines. We compare NegBio-VAE to representative VAE models that use both continuous and discrete latent variables. Continuous baselines include the Gaussian VAE (\mathcal{G} -VAE) [19] and a Laplace variant (\mathcal{L} -VAE), while discrete baselines include the Categorical VAE (\mathcal{C} -VAE) [14] and the Poisson VAE (\mathcal{P} -VAE) [42]. We also evaluate four variants of our model: NegBio-VAE_{DS-G}, NegBio-VAE_{DS-C}, NegBio-VAE_{MC-G}, and NegBio-VAE_{MC-C}. Here, **DS** uses analytical KL with dispersion sharing, **MC** uses Monte Carlo KL, **G** and **C** denote Gumbel-Softmax relaxation and continuous-time simulation reparameterization, respectively. These variants allow us to investigate the effects of KL computation and reparameterized sampling within our model.

Setup. We conduct two types of experiments to evaluate model performance. The first focuses on reconstruction quality and latent space properties and the results are reported in Section 5.2. The second assesses the utility of the learned representations through downstream tasks, using latent codes extracted from the encoder, as detailed in Section 5.3. Unless otherwise specified, all VAEs use convolutional encoders and decoders, with the latent dimensionality fixed at 256. Experiments are implemented in PyTorch (Python 3.9.12) and run on two NVIDIA A40 GPUs (40GB), one NVIDIA A100 GPU (44GB), and a 2.60GHz Xeon(R) Gold 6240 CPU.

5.2 Performance Comparison

All metrics are averaged over 5 runs. As shown in Table 1, NegBio-VAE variants consistently achieve competitive or superior reconstruction performance, especially in MSE and visual fidelity, even when not always yielding the best FID or IS scores. This highlights their strong ability to recover fine-grained details under overdispersed conditions. *Our FID scores are notably lower than those reported in prior work, primarily because we adopt a lighter feature dimension (64 instead of the standard 2048) and evaluate only on 5,000 images, which together lead to underestimated FID values.*

It is worth noting that the ODI is computed over all latent variables z across multiple batches and training epochs. As a result, the aggregated latent variables will exhibit overdispersion in \mathcal{P} -VAE and somewhat underdispersion in NegBio-VAE_{DS-G} across the batches over training epochs. To validate this, we recompute ODI on MNIST dataset using a fixed input x (one batch) and measure $z \sim \text{Poisson}(\lambda(x))$ without aggregating across batches and epochs. *In this setting as shown in Table 2, the ODI is consistently ≈ 1 , confirming that Poisson VAE adheres to $\text{Var}(z) = \mathbb{E}[z]$ as expected. In contrast, NegBio-VAE maintains $\text{ODI} > 1$ even when measured per-batch, showing it explicitly captures the input-driven overdispersion.*

While the ODI of different NegBio-VAE variants appear similar when measured on single batches, their aggregated ODI over the full training process diverges notably. This divergence arises from how different KL estimation and reparameterization strategies interact with dispersion dynamics throughout training. Specifically, the dispersion-sharing variant with continuous-time relaxation (DS-C) shows the highest aggregated ODI, while DS with Gumbel-Softmax (DS-G) yields the lowest—sometimes even below 1, indicating underdispersion. This stems from two key factors: (1) Dispersion sharing ties prior and posterior variance, which, over training, may lead to inflated mean counts and increased variability in DS-C; and (2) Gumbel-Softmax often results in more discrete and peaked latent activations, suppressing variance across batches, while continuous relaxations produce smoother samples that, under weak KL regularization, allow greater variability. These results highlight that even subtle design choices can lead to significant cumulative effects on latent dispersion during training.

5.3 Latent Representation Analysis

To evaluate learned latent representations, we follow the setup in Vafaii et al. [42], using MNIST with a fixed latent dimensionality of 10 and convolutional encoder-decoders for all models. We randomly split the test set into two sets of 5,000 samples each and train logistic regression classifiers using $N = 200, 1000, 5000$ labeled samples from one set. We then report accuracy on the other set (Table 3). For NegBio-VAE, we use the variant of DS-C. Higher-dimensional results are provided in Appendix D.1.1. We also assess latent space structure via empirical shattering dimensionality [3, 17, 33, 42], defined as the average binary classification accuracy over disjoint class partitions using linear classifiers. As shown in Table 3, NegBio-VAE consistently outperforms baselines, especially when using a small number of labels ($N = 200$, achieving a 1.4% gain over the next best model). Moreover, it achieves the highest

Table 1: Reconstruction quality evaluation and latent space diagnostics of various VAE models on MNIST, CIFAR_{16×16}, Omniglot and SVHN datasets. Lower values indicate better performance for MSE and FID, while higher values are better for IS. We highlight the best results for these metrics in bold. For ODI and DNR, which serve as diagnostic indicators rather than direct performance objectives, we do not mark the best values, as higher or lower is not necessarily better in all contexts.

Dataset	Method	Quantitative metrics			Latent space diagnostics	
		MSE(↓)	IS(↑)	FID(↓)	ODI	DNR
MNIST	\mathcal{G} -VAE	0.006±0.000	2.303±0.039	0.009±0.000	0.087±0.000	0.000±0.000
	\mathcal{L} -VAE	0.006±0.000	2.364±0.030	0.010±0.003	0.113±0.000	0.000±0.000
	\mathcal{P} -VAE	0.013±0.000	2.350±0.040	0.031±0.000	1.468±0.000	0.004±0.000
	\mathcal{C} -VAE	0.023±0.000	2.406±0.048	0.085±0.001	0.102±0.000	0.715±0.000
	NegBio-VAE _{DS-G}	0.009±0.000	2.277±0.097	0.017±0.000	0.739±0.000	0.000±0.000
	NegBio-VAE _{DS-C}	0.004 ±0.000	2.278±0.049	0.005±0.000	145.141±0.000	0.000±0.000
	NegBio-VAE _{MC-G}	0.004 ±0.000	2.267±0.048	0.002 ±0.000	1.411±0.000	0.000±0.000
	NegBio-VAE _{MC-C}	0.012±0.000	2.421 ±0.052	0.027±0.000	18.247±0.000	0.000±0.000
CIFAR _{16×16}	\mathcal{G} -VAE	0.008±0.000	1.862±0.041	0.059±0.000	0.085±0.000	0.000±0.000
	\mathcal{L} -VAE	0.008±0.000	1.889±0.026	0.053±0.000	0.110±0.000	0.000±0.000
	\mathcal{P} -VAE	0.024±0.000	1.844±0.174	0.062±0.000	1.126±0.000	0.203±0.000
	\mathcal{C} -VAE	0.023±0.000	1.600±0.019	0.233±0.000	0.069±0.000	0.008±0.000
	NegBio-VAE _{DS-G}	0.013±0.000	1.900±0.035	0.031±0.000	0.734±0.000	0.098±0.000
	NegBio-VAE _{DS-C}	0.007 ±0.000	1.883±0.040	0.021±0.000	143.899±0.000	0.000±0.000
	NegBio-VAE _{MC-G}	0.007 ±0.000	1.965 ±0.052	0.007 ±0.000	1.436±0.000	0.000±0.000
	NegBio-VAE _{MC-C}	0.016±0.000	1.771±0.044	0.106±0.001	8.613±0.001	0.000±0.000
Omniglot	\mathcal{G} -VAE	0.024±0.000	1.830±0.170	0.065±0.000	1.130±0.000	0.207±0.000
	\mathcal{L} -VAE	0.024±0.000	1.292±0.183	0.057±0.000	1.124±0.000	0.203±0.000
	\mathcal{P} -VAE	0.028±0.000	2.003 ±0.146	0.082±0.000	1.743±0.000	0.008±0.000
	\mathcal{C} -VAE	0.042±0.000	2.061±0.071	0.277±0.000	0.075±0.000	0.008±0.000
	NegBio-VAE _{DS-G}	0.022±0.000	1.810±0.147	0.838±0.000	0.876±0.000	0.086±0.000
	NegBio-VAE _{DS-C}	0.015±0.000	1.762±0.164	0.004 ±0.000	163.946±0.003	0.000±0.000
	NegBio-VAE _{MC-G}	0.010 ±0.000	1.865±0.206	0.006±0.000	1.457±0.000	0.000±0.000
	NegBio-VAE _{MC-C}	0.022±0.000	1.919±0.133	0.050±0.000	9.401±0.019	0.000±0.000
SVHN	\mathcal{G} -VAE	0.004 ±0.000	2.148±0.060	0.035±0.021	0.085±0.000	0.000±0.000
	\mathcal{L} -VAE	0.004 ±0.000	2.159±0.044	0.032 ±0.010	0.110±0.000	0.000±0.000
	\mathcal{P} -VAE	0.008±0.000	2.159±0.050	0.052±0.020	1.607±0.005	0.004±0.000
	\mathcal{C} -VAE	0.017±0.000	2.097±0.051	0.322±0.069	0.080±0.003	0.009±0.002
	NegBio-VAE _{DS-G}	0.007±0.000	2.159±0.066	0.040±0.017	0.829±0.012	0.060±0.055
	NegBio-VAE _{DS-C}	0.004 ±0.000	2.294 ±0.040	0.083±0.062	130.937±1.568	0.000±0.000
	NegBio-VAE _{MC-G}	0.010±0.000	2.264±0.036	0.038±0.006	1.477±0.007	0.000±0.000
	NegBio-VAE _{MC-C}	0.008±0.000	2.140±0.052	0.118±0.042	9.804±0.193	0.000±0.000

Table 2: ODI on a single MNIST batch. To remove the effect of aggregation across batches and epochs, we compute the ODI using a fixed input batch. P-VAE yields an ODI ≈ 1 , consistent with the Poisson assumption. In contrast, all NegBio-VAE variants maintain ODI > 1 , confirming their ability to capture overdispersion at the per-batch level.

Method	NB-VAE _{MC-G}	NB-VAE _{MC-C}	NB-VAE _{DS-G}	NB-VAE _{DS-C}	P-VAE
ODI	2.8044	2.8434	2.8745	2.8586	1.0343

shattering score. t-SNE visualizations (Fig. 2) further show that NegBio-VAE produces compact, well-separated clusters, unlike the entangled (\mathcal{G}), dispersed (\mathcal{P}), or noisy (\mathcal{C}) latent spaces of other models. These results underscore the advantages of modeling overdispersed spike counts for learning structured and semantically meaningful representations.

Table 3: Evaluation of latent representations on MNIST. Higher accuracy and shattering dimensionality indicate more structured and generalizable latents.

Latent Dim.	Model	N=200	N = 1000	N = 5000	Shattering Dim.
10	\mathcal{G} -VAE	0.803 ± 0.009	0.847 ± 0.004	0.865 ± 0.002	0.755 ± 0.004
	\mathcal{L} -VAE	0.766 ± 0.019	0.817 ± 0.006	0.830 ± 0.003	0.724 ± 0.004
	\mathcal{P} -VAE	0.773 ± 0.015	0.824 ± 0.005	0.841 ± 0.003	0.770 ± 0.005
	\mathcal{C} -VAE	0.715 ± 0.012	0.787 ± 0.004	0.799 ± 0.003	0.747 ± 0.003
	NegBio-VAE	0.817 ± 0.018	0.896 ± 0.005	0.908 ± 0.004	0.795 ± 0.005

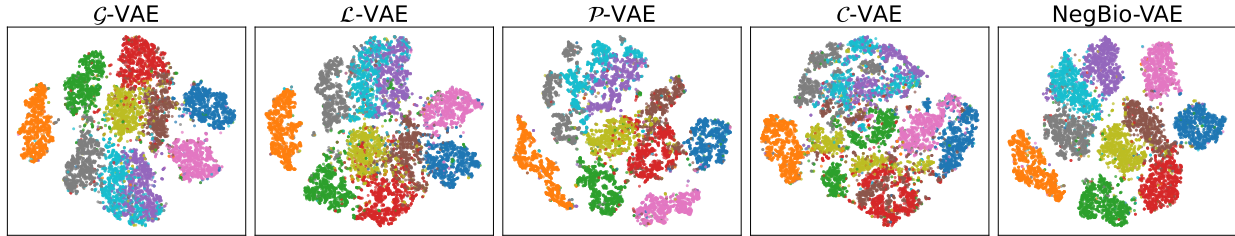


Figure 2: t-SNE of latent representations from different VAEs on MNIST with latent dimensionality fixed at 10. We visualize the 2D embeddings of latent representations learned by four VAE baselines and compare them with NegBio-VAE. Each point is colored by the corresponding digit label. A more compact and cluster-aligned structure suggests better class separation in the latent space.

5.4 Ablation Study

Encoder-Decoder Architectures. To explore the impact of encoder architecture on model performance, we conduct an ablation study using the MNIST dataset, with the decoder fixed as a linear network. As shown in Table 4, both the MLP and convolutional encoders outperform the simple linear encoder in terms of MSE and FID. Notably, the MLP achieves the lowest MSE and the convolutional encoder achieves competitive scores across all metrics. These results suggest that non-linear encoding functions can significantly enhance representational capacity. A comprehensive comparison of different models is presented in Appendix D.2.

Effect of β Scaling. We examine how varying β affects the performance of NegBio-VAE variants. As shown in Fig. 3a, decreasing β generally leads to improved reconstruction quality, particularly for DS+C and MC+C. These variants also show stable IS values across β , indicating robustness in generative quality. In contrast, higher β values tend to increase ODI, especially for DS+C, suggesting stronger overdispersion. However, this increased expressiveness comes at the cost of reduced reconstruction performance and increased DNR in some configurations (DS+G).

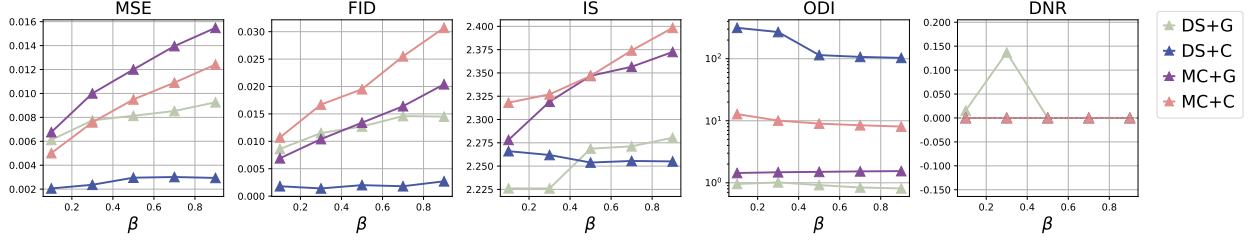
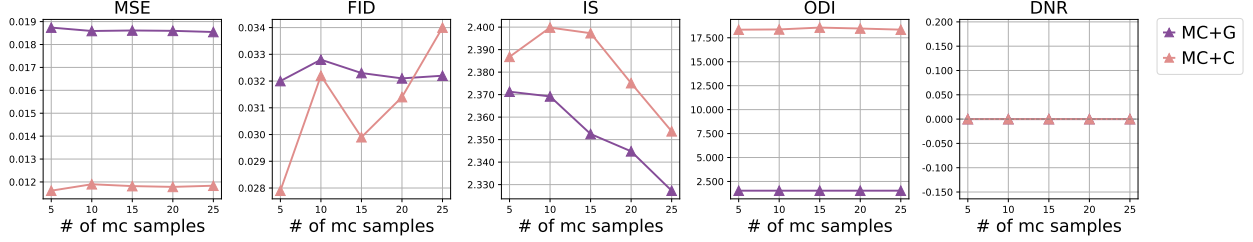
Table 4: Ablation study of encoder architectures on MNIST with decoder fixed as a linear network.

Encoder	MSE	IS	FID
Linear	0.015 ± 0.006	2.132 ± 0.046	0.206 ± 0.018
MLP	0.006 ± 0.000	2.170 ± 0.047	0.132 ± 0.007
Conv	0.007 ± 0.000	2.144 ± 0.054	0.135 ± 0.001

Effect of Number of MC Samples. We further evaluate how the number of MC samples affects model performance in Fig. 3b. Interestingly, the two MC-based variants respond differently to increased sample size. MC+C consistently shows improvements across all metrics, while MC+G remains a relatively stable performance. This is likely because continuous-time reparameterization is more sensitive to noisy KL gradients, and benefits substantially from improved estimation. In contrast, Gumbel-softmax inherently smooths the latent space, making it less affected by MC samples.

6 Conclusions

In this work, we presented NegBio-VAE, a biologically inspired generative model that uses the NB distribution to capture the overdispersed nature of neural spike counts. By introducing a dispersion parameter, the model extends beyond Poisson assumptions with minimal modification. Despite its simplicity, NegBio-VAE yields substantial improvements in reconstruction quality across various benchmark datasets. This extension decouples the mean and variance of spike counts, allowing the model to better capture neural variability while maintaining biological plausibility. Experimental

(a) Effect of varying β on generation quality and overdispersion control.

(b) Impact of Monte Carlo sample size on reconstruction quality and latent space metrics.

Figure 3: Ablation study of NegBio-VAE. a explores the effect of KL weighting via β on various metrics; b shows how the number of Monte Carlo samples affects model performance.

results show that NegBio-VAE outperforms existing VAE baselines in reconstruction fidelity, sample quality, and the biological plausibility of latent representations. In future work, we can incorporate temporal dynamics of spike trains and explore hierarchical latent structures to further align artificial models with the complexity of biological neural systems.

7 Limitation

While NegBio-VAE introduces greater flexibility in modeling overdispersed spike-like activations, it also introduces new design choices such as KL estimation strategy and reparameterization method, which can significantly affect stability and representational quality. Although we provide initial empirical guidance, a deeper theoretical understanding of these trade-offs remains underexplored. In future work, we plan to investigate adaptive mechanisms for selecting reparameterization strategies based on data characteristics, and explore extensions to hierarchical or structured spike-based latent spaces to further improve model expressiveness and biological alignment.

References

- [1] Michael A Arbib. *The handbook of brain theory and neural networks*. MIT press, 2003.
- [2] Wyeth Bair and Christof Koch. Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey. *Neural computation*, 8(6):1185–1202, 1996.
- [3] Silvia Bernardi, Marcus K. Benna, Mattia Rigotti, Jérôme Munuera, Stefano Fusi, and C. Daniel Salzman. The geometry of abstraction in the hippocampus and prefrontal cortex. *Cell*, 183(4):954–967.e21, Nov 2020. doi: 10.1016/j.cell.2020.09.031.
- [4] Jiahang Cao, Ziqing Wang, Hanzhong Guo, Hao Cheng, Qiang Zhang, and Renjing Xu. Spiking denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 4912–4921, 2024.
- [5] Xiang Cheng, Yunzhe Hao, Jiaming Xu, and Bo Xu. Lissn: Improving spiking neural networks with lateral interactions for robust object recognition. In *IJCAI*, pages 1519–1525. Yokohama, 2020.
- [6] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

- [7] Yanming Di, Daniel W Schafer, Jason S Cumbie, and Jeff H Chang. The nbp negative binomial model for assessing differential gene expression from rna-seq. *Statistical applications in genetics and molecular biology*, 10(1), 2011.
- [8] Emilien Dupont. Learning disentangled joint continuous and discrete representations. *Advances in neural information processing systems*, 31, 2018.
- [9] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2661–2671, 2021.
- [10] Linghao Feng, Dongcheng Zhao, and Yi Zeng. Spiking generative adversarial network with attention scoring decoding. *Neural Networks*, 178:106423, 2024.
- [11] Vincent Fortuin, Matthias Hüser, Francesco Locatello, Heiko Strathmann, and Gunnar Rätsch. Som-vae: Interpretable discrete representation learning on time series. In *International Conference on Learning Representations*, 2019.
- [12] Samanwoy Ghosh-Dastidar and Hojjat Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.
- [13] Tim Gollisch and Markus Meister. Rapid neural coding in the retina with relative spike latencies. *science*, 319(5866):1108–1111, 2008.
- [14] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.
- [15] Hiromichi Kamata, Yusuke Mukuta, and Tatsuya Harada. Fully spiking variational autoencoder. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 7059–7067, 2022.
- [16] Jaivardhan Kapoor, Auguste Schulz, Julius Vetter, Felix Pei, Richard Gao, and Jakob H Macke. Latent diffusion for neural spiking data. *Advances in Neural Information Processing Systems*, 37:118119–118154, 2024.
- [17] Matthew T. Kaufman, Marcus K. Benna, Mattia Rigotti, Fabio Stefanini, Stefano Fusi, and Anne K. Churchland. The implications of categorical and category-free mixed selectivity on representational geometries. *Current Opinion in Neurobiology*, 77:102644, Dec 2022. doi: 10.1016/j.conb.2022.102644.
- [18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.
- [19] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.
- [20] Vineet Kotariya and Udayan Ganguly. Spiking-gan: A spiking generative adversarial network using time-to-first-spike coding. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2022.
- [21] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.
- [22] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.
- [23] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, 2, 2010.
- [24] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *Advances in neural information processing systems*, 34:23426–23439, 2021.
- [25] Mingxuan Liu, Jie Gan, Rui Wen, Tao Li, Yongli Chen, and Hong Chen. Spiking-diffusion: Vector quantized discrete diffusion model with spiking neural networks. In *2024 5th International Conference on Computer, Big Data and Artificial Intelligence (ICCBD+ AI)*, pages 627–631. IEEE, 2024.
- [26] Zachary F Mainen and Terrence J Sejnowski. Reliability of spike timing in neocortical neurons. *Science*, 268(5216):1503–1506, 1995.
- [27] Joseph Marino. Predictive coding, variational autoencoders, and biological connections. *Neural Computation*, 34(1):1–44, 2022.
- [28] Dina Moshitch and Israel Nelken. Using tweedie distributions for fitting spike count data. *Journal of neuroscience methods*, 225:13–28, 2014.
- [29] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

- [30] Donald H Perkel, George L Gerstein, and George P Moore. Neuronal spike trains and stochastic point processes: Ii. simultaneous spike trains. *Biophysical journal*, 7(4):419–440, 1967.
- [31] Jonathan Pillow and James Scott. Fully bayesian inference for neural models with negative-binomial spiking. *Advances in neural information processing systems*, 25, 2012.
- [32] Daniil Polykovskiy and Dmitry Vetrov. Deterministic decoding for discrete data in variational autoencoders. In *International conference on artificial intelligence and statistics*, pages 3046–3056. PMLR, 2020.
- [33] Mattia Rigotti, Omri Barak, Melissa R. Warden, Xiao-Jing Wang, Nathaniel D. Daw, Earl K. Miller, and Stefano Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, 497:585–590, 05 2013. doi: <https://doi.org/10.1038/nature12160>.
- [34] Jason Tyler Rolfe. Discrete variational autoencoders. In *International Conference on Learning Representations*, 2017.
- [35] Bleema Rosenfeld, Osvaldo Simeone, and Bipin Rajendran. Spiking generative adversarial networks with a neural network discriminator: Local training, bayesian models, and continual meta-learning. *IEEE Transactions on Computers*, 71(11):2778–2791, 2022.
- [36] GJS Ross and DA Preece. The negative binomial distribution. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 34(3):323–335, 1985.
- [37] Ian H Stevenson. Flexible models for spike count data with both over-and under-dispersion. *Journal of computational neuroscience*, 41:29–43, 2016.
- [38] Katherine R Storrs, Barton L Anderson, and Roland W Fleming. Unsupervised learning predicts human perception and misperception of gloss. *Nature Human Behaviour*, 5(10):1402–1417, 2021.
- [39] Wahiba Taouali, Giacomo Benvenuti, Pascal Wallisch, Frédéric Chavane, and Laurent U Perrinet. Testing the odds of inherent vs. observed overdispersion in neural spike counts. *Journal of neurophysiology*, 115(1):434–444, 2016.
- [40] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks. *Neural networks*, 111:47–63, 2019.
- [41] Hadi Vafaii, Jacob Yates, and Daniel Butts. Hierarchical vaes provide a normative account of motion processing in the primate brain. *Advances in Neural Information Processing Systems*, 36:46152–46190, 2023.
- [42] Hadi Vafaii, Dekel Galor, and Jacob Yates. Poisson variational autoencoder. *Advances in Neural Information Processing Systems*, 37:44871–44906, 2024.
- [43] Gido M Van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):4069, 2020.
- [44] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.
- [45] Srishti Yadav, Anshul Pundhir, Tanish Goyal, Balasubramanian Raman, and Sanjeev Kumar. Differentially private spiking variational autoencoder. In *International Conference on Pattern Recognition*, pages 96–112. Springer, 2025.
- [46] Qiugang Zhan, Ran Tao, Xiurui Xie, Guisong Liu, Malu Zhang, Huajin Tang, and Yang Yang. Esvae: An efficient spiking variational autoencoder with reparameterizable poisson spiking sampling. *arXiv preprint arXiv:2310.14839*, 2023.
- [47] He Zhao, Piyush Rai, Lan Du, Wray Buntine, Dinh Phung, and Mingyuan Zhou. Variational autoencoders for sparse and overdispersed discrete data. In *International conference on artificial intelligence and statistics*, pages 1684–1694. PMLR, 2020.
- [48] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11062–11070, 2021.
- [49] Mingyuan Zhou and Lawrence Carin. Negative binomial process count and mixture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):307–320, 2013.

A Derivation of KL Term

We derive an analytical expression for the KL divergence between two Negative Binomial distributions under the assumption that the encoder does not modify the dispersion parameter (i.e., $\delta_r = 1$). the univariate Negative Binomial distribution, given dispersion r and success probability p , is defined as:

$$\text{NB}(z; r, p) = \binom{z+r-1}{z} (1-p)^z p^r.$$

Substituting this into the KL divergence for a single z yields:

$$\begin{aligned} \mathcal{D}_{\text{KL}}(q||p) &= \mathbb{E}_{z \sim q} \left[\log \frac{q}{p} \right] \\ &= \mathbb{E}_{z \sim q} \left[\log \frac{\binom{z+r\delta_r-1}{z} (1-p\delta_p)^z (p\delta_p)^{r\delta_r}}{\binom{z+r-1}{z} (1-p)^z p^r} \right] \\ &= \mathbb{E}_{z \sim q} \left[\log \left[\binom{z+r\delta_r-1}{z} \right] - \log \left[\binom{z+r-1}{z} \right] + (r\delta_r) \log[p\delta_p] - r \log p + z \log \left(\frac{1-p\delta_p}{1-p} \right) \right] \\ &= \mathbb{E}_{z \sim q} \left[r \log \frac{p\delta_p}{p} + z \log \left(\frac{1-p\delta_p}{1-p} \right) \right] \quad \text{By taking } \delta_r = 1 \\ &= r \log \frac{p\delta_p}{p} + \mathbb{E}_{z \sim q} \left[z \log \left(\frac{1-p\delta_p}{1-p} \right) \right] \\ &= r \log \delta_p + \log \left(\frac{1-p\delta_p}{1-p} \right) \mathbb{E}_{z \sim q} [z] \\ &= r \log \delta_p + r \frac{1-p\delta_p}{p\delta_p} \log \left(\frac{1-p\delta_p}{1-p} \right) \quad \text{for } z \sim \text{NB}(r\delta_r, p\delta_p), \mathbb{E}_{z \sim q} [z] = \frac{r(1-p\delta_p)}{p\delta_p} \\ &= r \left[\log \delta_p + \frac{1-p\delta_p}{p\delta_p} \log \left(\frac{1-p\delta_p}{1-p} \right) \right] \\ &= rg(p, \delta_p), \end{aligned}$$

The first two terms of the full KL expression (Line 2) involve binomial coefficients. Taking the logarithm of these terms and computing the expectation with respect to the posterior makes the KL divergence intractable. As a result, Monte Carlo sampling or variational approximation techniques are typically required, which often introduce high variance in the gradient estimates or rely on additional approximating assumptions and can lead to unstable or biased training. To make the expression tractable, we introduce a simplifying assumption: $\delta_r = 1$, i.e., the encoder does not adjust the prior parameter r , and thus the posterior and prior share the same dispersion parameter. This assumption is reasonable because the NB distribution is parameterized by both r and p . Therefore, even when r is fixed, we can still adjust the distribution (i.e., its mean and variance) by varying p . This leads to a closed-form approximation:

$$\mathcal{D}_{\text{KL}}(q||p) = r \left[\log \delta_p + \frac{1-p\delta_p}{p\delta_p} \log \left(\frac{1-p\delta_p}{1-p} \right) \right],$$

which we denote as $rg(p, \delta_p)$, where:

$$g(a, b) := \log b + \frac{1-ab}{ab} \log \left[\frac{1-ab}{1-a} \right], \quad a \in (0, 1), \quad b > 0.$$

This expression is simple, interpretable and has useful boundary properties. When $\delta_p = 1$ (i.e., the encoder does not shift p), $g(a, 1) = 0$, and the KL divergence vanishes. As $ab \rightarrow 0$ (i.e., posterior sparsity increases), the KL grows rapidly, penalizing excessive deviation from the prior. This behavior mirrors that of \mathcal{P} -VAE, which strongly discourages low-rate posterior collapse. While \mathcal{P} -VAE already provides an elegant analysis of sparsity through its KL structure, we do not emphasize this aspect in the main text. However, our formulation shares the same desirable sparsity behavior: when δ_p approaches 0, the KL diverges, discouraging extreme posterior sparsification. Moreover, our formulation retains an analytical form even for overdispersed distributions, enabling tractable training without Monte Carlo estimation.

Similar to the \mathcal{P} -VAE [42], which analyzes the behavior of its KL divergence near the prior via a Taylor expansion of the function $f(\delta_r) = 1 - \delta_r + \delta_r \log \delta_r$, we perform a similar analysis for the closed-form KL term in NegBio-VAE.

To better understand the behavior of the closed-form KL divergence near the prior, we expand $g(a, b)$ at $b = 1 + \epsilon$, with $\epsilon \ll 1$:

$$g(a, 1 + \epsilon) \approx \frac{a}{2(1-a)} \epsilon^2 + \mathcal{O}(\epsilon^3). \quad (6)$$

Thus, when $\delta_p = 1 + \epsilon$, the KL becomes:

$$\mathcal{D}_{\text{KL}} \approx r \cdot \frac{a}{2(1-a)} \epsilon^2.$$

This reveals that, like in \mathcal{P} -VAE, the KL divergence grows quadratically near the prior, encouraging smooth and stable optimization. However, our formulation provides a tunable growth rate via the parameter $a = p$, allowing more flexible control over sparsity regularization. Unlike Poisson VAEs, which assume equal mean and variance, our Negative Binomial model accommodates overdispersion and remains analytically tractable—enabling stable training without Monte Carlo approximation. These properties make our approach better suited for modeling realistic, variable spike-based neural activity.

B Implementation Details

We include all the implementation details in this section, including the sampling techniques and detailed experimental settings.

B.1 Sampling techniques

We adopt two sampling techniques for our model, while we have introduced the main idea in the main text, for completeness, we include the details here: (1) **Gumbel-Softmax Relaxation** This method approximates discrete Poisson sampling using continuous relaxation.

1. Limit the maximum count value to Z_{\max} .
2. Compute the log-probability for $z = 0, 1, \dots, Z_{\max}$,
$$\log \text{Poi}(z) = z \log \lambda - \lambda - \log \Gamma(z + 1),$$
3. For each z , generate noise $\epsilon_z \sim \text{Gumbel}(0, 1)$.
4. Apply the Gumbel-Softmax trick with temperature τ ,

$$\tilde{z} = \sum_{z=0}^{Z_{\max}} z \cdot \text{softmax} \left(\frac{\log \text{Poi}(z) + \epsilon_z}{\tau} \right),$$

where $\tau \rightarrow 0$ recovers discrete sampling.

The proof of this reparameterization can be found in Jang et al. [14], and will not be repeated here.

(2) **Continuous-Time Simulation** This method models Poisson processes with intensity λ on $[0, 1]$ using exponentially distributed inter-arrival times.

1. Sample inter-arrival times from an exponential distribution:

$$\{s_i\}_{i=1}^M \sim \text{Exponential}(\lambda),$$

where M is a sufficiently large integer, the exponential distribution is easily reparameterized and PyTorch contains an implementation.

2. Accumulate inter-arrival times:

$$S_n = \sum_{i=1}^n s_i, \quad 1 \leq n \leq M.$$

3. Soft count of events:

$$\tilde{z} = \sum_{n=1}^M \sigma \left(\frac{1 - S_n}{\tau} \right),$$

where $\tau \rightarrow 0$ recovers discrete sampling.

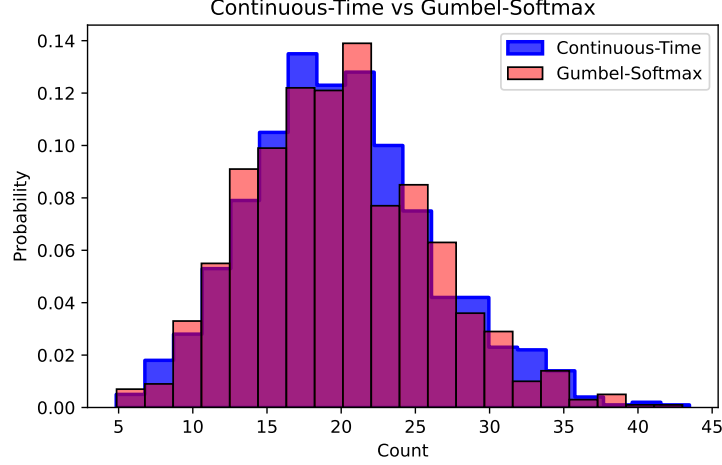


Figure 4: Empirical distributions of Negative Binomial samples generated using Continuous-Time Simulation and Gumbel-Softmax Relaxation. Both methods successfully approximate count-valued outputs consistent with NB sampling behavior, validating their use as differentiable reparameterization strategies. Each method generates 1000 samples using parameters $r = 20$, $p = 0.5$, and temperature $\tau = 0.1$.

This reparameterization exploits the relationship between the Poisson distribution and the Poisson process. We can generate Poisson counts from $\text{Poi}(\lambda)$ by counting events on a homogeneous Poisson process with intensity λ over the interval $[0, 1]$.

To verify that both proposed reparameterization methods can successfully generate valid count samples from the NB distribution, we generate 1000 samples from each method using $r = 20$, $p = 0.5$, and temperature $\tau = 0.1$, and the empirical distribution is shown in Fig. 4. Both methods produce plausible count distributions with unimodal structure and similar mean values, confirming that each method can successfully approximate NB samples in a differentiable manner. This validates their use as practical reparameterization techniques for NegBio-VAE.

B.2 Experimental Implementation

In this section, we provide additional implementation details that complement the experiments described in the main text.

B.2.1 Datasets

We implement all data loading pipelines using PyTorch Lightning’s `LightningDataModule` interface, ensuring consistent structure across datasets. For each dataset, we apply preprocessing transformations tailored to its modality and input requirements:

- **MNIST:** Grayscale images are normalized to $[0, 1]$ using `transforms.ToTensor()` without additional augmentation. Images are optionally flattened if required by the encoder structure.
- **CIFAR_{16×16}:** We use CIFAR-10 as a base dataset and uniformly downsample all images to 16×16 resolution. Color images are normalized to $[-1, 1]$ using mean and standard deviation $(0.5, 0.5, 0.5)$ and are optionally augmented via horizontal flipping.
- **Omniglot:** Following standard protocols, we resize all images to 28×28 , invert foreground-background polarity via `ImageOps.invert`, and convert them to grayscale. Only background characters are used for training, while evaluation is performed on the non-background split.
- **SVHN:** We use both the training and test splits. Images are converted to tensors and normalized to $[-1, 1]$ across all three RGB channels.

All preprocessing logic is encapsulated in a shared function `get_transform()`, which dynamically composes transformations based on dataset type, grayscale conversion, data flattening, and augmentation flags.

B.2.2 Encoder and Decoder Architectures

Our model supports interchangeable encoder and decoder architectures to accommodate various data modalities and representation structures. Following the same setting in [42], we include linear, convolutional, and MLP-based designs.

Encoders:

- **Linear Encoder:** A single fully connected layer that maps flattened inputs to the latent space. Optionally applies weight normalization.
- **MLP Encoder:** A two-layer perceptron with an intermediate residual dense layer followed by a linear projection. This encoder supports flexible nonlinearity and is used when richer transformations are required from vectorized inputs.
- **Convolutional Encoder:** A two-stage convolutional network with ReLU activations, followed by flattening and a fully connected projection. It adapts the input channel size (1 for grayscale datasets, 3 for RGB), and auto-computes the flattening shape based on the dataset resolution. LayerNorm is optionally applied to the final latent layer.

Decoders:

- **Linear Decoder:** Mirrors the linear encoder with a fully connected layer projecting latent vectors to pixel space. Output is passed through either a Sigmoid or Tanh nonlinearity depending on the expected pixel scale.
- **MLP Decoder:** A three-layer feedforward network with residual blocks and configurable nonlinearity (e.g., Swish), designed for richer reconstructions from compact latent codes. The final layer uses Sigmoid or Tanh.
- **Convolutional Decoder:** Used in image-based settings, this decoder first expands latent vectors through a fully connected layer into a low-resolution feature map, then applies transposed convolutions to upscale to the desired image size. The initial size is determined by dataset type (e.g., 7×7 for MNIST, 4×4 for CIFAR_{16×16}).

B.2.3 Shattering Dimensionality

To quantitatively evaluate the geometry of the learned latent space, we compute the shattering dimensionality following prior work [3, 17, 33, 42]. Specifically, we measure how well the latent space supports linear separation across all balanced binary label partitions. Concretely, given a label set such as digits 0–9, we enumerate all disjoint splits into two non-overlapping and balanced class groups (e.g., 0,1,2,3,4 vs. 5,6,7,8,9), where each partition defines a binary classification task. For each split, we relabel samples in one group as class 0 and those in the other group as class 1, producing binary-labeled data. For a dataset with 10 classes (e.g., MNIST), we generate all possible balanced, disjoint 5-vs-5 class splits. This results in a total of 252 unique binary classification tasks, corresponding to all combinations of 5 classes out of 10 without regard to class order or labeling symmetry. A linear classifier (e.g., logistic regression) is then trained on latent representations from a subset of the training data. The classification accuracy is computed on the validation set, and the final shattering dimensionality is taken as the average accuracy across all 252 tasks. The implementation uses the `itertools.combinations` function to enumerate all unique 5-class subsets, and constructs their complementary partitions to define the 5-vs-5 classification groups.

B.2.4 FID Computation Details

To quantitatively evaluate the visual fidelity and distributional similarity of generated images, we adopt the *Fréchet Inception Distance* (FID) as a standard evaluation metric. FID measures the distance between the real and generated image distributions in the feature space of a pretrained Inception network. Specifically, it assumes both distributions are Gaussian, and computes the Fréchet distance between them as:

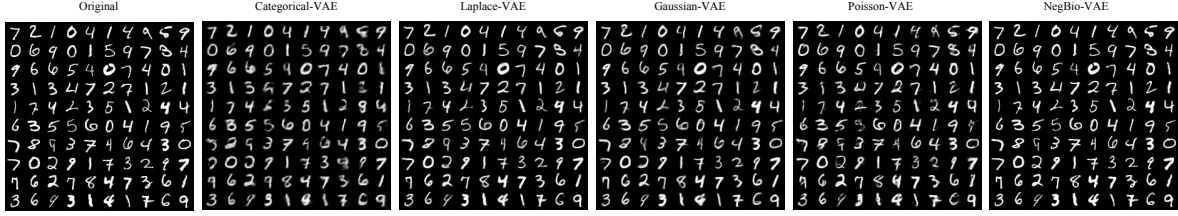
$$\text{FID} = \|\mu_{\text{real}} - \mu_{\text{gen}}\|^2 + \text{Tr} \left(\Sigma_{\text{real}} + \Sigma_{\text{gen}} - 2(\Sigma_{\text{real}}\Sigma_{\text{gen}})^{1/2} \right),$$

where μ_{real} , Σ_{real} and μ_{gen} , Σ_{gen} denote the mean and covariance of the real and generated feature activations, respectively. A lower FID score indicates that the generated samples are more similar to the real data in terms of both image quality and diversity. In our experiments, we extract features from the `pool3` layer of a pretrained Inception-V3 model. For low-resolution datasets such as MNIST, we use early convolutional features with reduced dimensionality (e.g., `feature=64`) to ensure compatibility and stability of the metric.

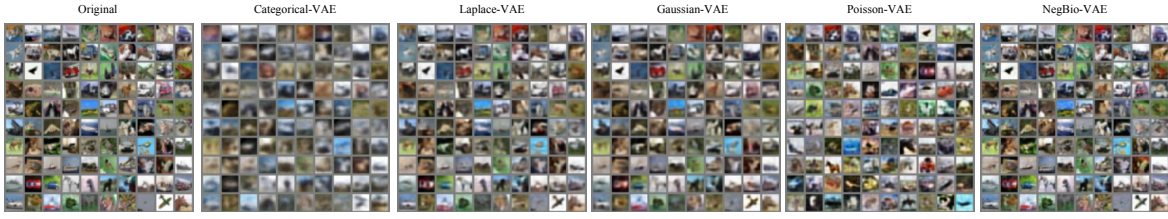
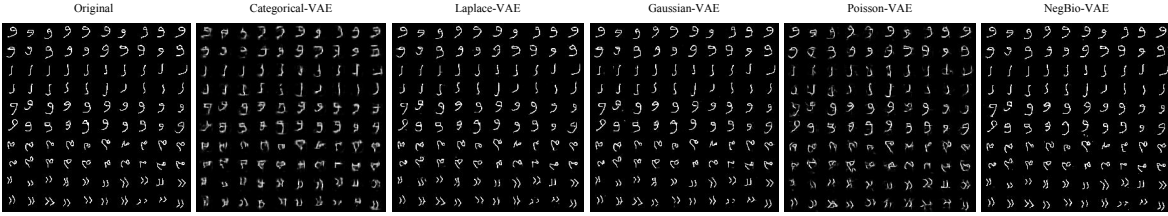
B.2.5 IS Computation Details

To compute the Inception Score (IS), we follow the standard protocol using a pretrained Inception v3 network. Given a batch of generated images with shape $[N, 3, H, W]$ and pixel values normalized to the range $[0, 1]$, we first resize each image to 299×299 using bilinear interpolation to match the input requirements of the Inception v3 model. The resized images are then passed through the pretrained network (without input transformation or fine-tuning) to obtain the class probability distribution via the softmax output. We use a batch size of 32 during evaluation and compute softmax probabilities on CPU for compatibility and memory efficiency.

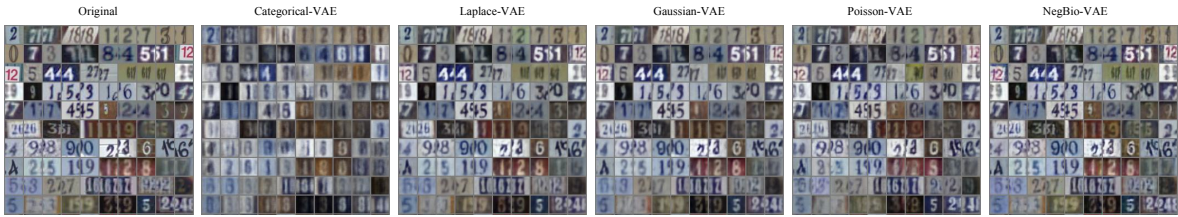
C Reconstruction Image Comparisons



(a) MNIST

(b) CIFAR_{16×16}

(c) Omniglot



(d) SVHN

Figure 5: Reconstruction comparison across datasets using a shared architecture and 256-dimensional latent space. NegBio-VAE consistently reconstructs clearer and more accurate images across MNIST, SVHN, CIFAR_{16×16}, and Omniglot.

All reconstruction results were obtained using a fixed latent dimensionality of 256 and a shared convolutional encoder-decoder architecture. From Fig. 5, we can see across all evaluated datasets, NegBio-VAE consistently outperforms other methods in reconstructing clearer, more faithful images. On MNIST, NegBio-VAE is capable of preserving fine-grained details such as the gap around the digit “0”, which other models fail to retain. Moreover, it correctly reconstructs the

Table 5: Evaluation of latent representations on MNIST using logistic regression with varying supervision levels ($N = 200, 1000, 5000$), along with a shattering dimensionality that reflects the average linear separability across class partitions. Higher accuracy and shattering dimensionality indicate more structured and generalizable latent codes.

Latent Dim.	Model	N=200	N = 1000	N = 5000	Shattering Dim.
10	\mathcal{G} -VAE	0.803 ± 0.009	0.847 ± 0.004	0.865 ± 0.002	0.755 ± 0.004
	\mathcal{L} -VAE	0.766 ± 0.019	0.817 ± 0.006	0.830 ± 0.003	0.724 ± 0.004
	\mathcal{P} -VAE	0.773 ± 0.015	0.824 ± 0.005	0.841 ± 0.003	0.770 ± 0.005
	\mathcal{C} -VAE	0.715 ± 0.012	0.787 ± 0.004	0.799 ± 0.003	0.747 ± 0.003
	NegBio-VAE	0.817 ± 0.018	0.896 ± 0.005	0.908 ± 0.004	0.795 ± 0.005
128	\mathcal{G} -VAE	0.787 ± 0.012	0.915 ± 0.002	0.962 ± 0.001	0.887 ± 0.006
	\mathcal{L} -VAE	0.712 ± 0.017	0.867 ± 0.004	0.940 ± 0.003	0.853 ± 0.008
	\mathcal{P} -VAE	0.720 ± 0.010	0.859 ± 0.002	0.934 ± 0.003	0.848 ± 0.008
	\mathcal{C} -VAE	0.787 ± 0.013	0.900 ± 0.003	0.947 ± 0.004	0.886 ± 0.005
	NegBio-VAE	0.787 ± 0.006	0.900 ± 0.002	0.950 ± 0.002	0.887 ± 0.008
256	\mathcal{G} -VAE	0.799 ± 0.011	0.872 ± 0.008	0.908 ± 0.004	0.798 ± 0.006
	\mathcal{L} -VAE	0.806 ± 0.010	0.878 ± 0.007	0.914 ± 0.003	0.803 ± 0.006
	\mathcal{P} -VAE	0.833 ± 0.016	0.913 ± 0.006	0.951 ± 0.002	0.853 ± 0.009
	\mathcal{C} -VAE	0.792 ± 0.011	0.903 ± 0.003	0.951 ± 0.002	0.891 ± 0.008
	NegBio-VAE	0.850 ± 0.013	0.924 ± 0.004	0.952 ± 0.004	0.894 ± 0.008

connected stroke in the digit “4”, whereas some models render it more like a “9”, and this demonstrate strong discrete structural modeling of Negbio-VAE. For the low-resolution, visually complex CIFAR_{16×16} dataset, NegBio-VAE shows strong robustness by reconstructing plausible object shapes and textures despite significant downsampling. Similarly, on Omniglot, it accurately captures character structure and stroke variations across diverse alphabets, indicating strong generalization to fine-grained symbolic representations. On SVHN, it produces sharper contours and cleaner backgrounds, reducing noise and artifacts more effectively than competing models. These results highlight the model’s ability to leverage overdispersed latent assumptions to recover richer visual detail, especially under discrete and noisy generative conditions.

D Additional Experiments

This section presents additional empirical results, including the complete main comparison, latent space analysis (downstream classification performance and dead neuron rate in high-dimensional settings), and further evaluations on VAE architecture variants.

D.1 Latent Analysis

In the main text, we presented latent analysis of downstream classification performance using a fixed latent dimensionality of 10. In this section, we provide additional results, including downstream classification performance at higher dimensionalities and dead neuron rate analysis in high-dimensional settings.

D.1.1 Downstream Classification Performance

We evaluate the quality of learned latent representations using logistic regression under varying supervision levels ($N = 200, 1000, 5000$). In addition, we assess the linear separability of the latent space via a shattering score, defined as the average classification accuracy across all $\binom{10}{5} = 252$ binary digit classification tasks. For each digit pair (c_0, c_1) , we relabel the data into binary categories and train a logistic regression classifier on the corresponding latent representations. Higher shattering dimensionality indicates stronger global separability and more structured latent spaces. To ensure fair comparison across models, we follow standard representation choices: posterior mean μ for continuous VAEs (Gaussian, Laplace), encoder output logits for \mathcal{C} -VAE, \mathcal{P} -VAE and NegBio-VAE.

Based on the results in Table 5, NegBio-VAE consistently yields highly structured and discriminative latent representations across all supervision levels and latent dimensionalities. In the low-dimensional setting (dimensionality at 10), NegBio-VAE outperforms all baselines by a clear margin in both logistic regression accuracy and shattering dimensionality, indicating that it can capture meaningful and separable features even under tight capacity constraints.

Table 6: Ablation study of encoder-decoder architectures on the MNIST dataset. We evaluate each model using MSE, FID, IS, ODI, DNR, and ELBO under three encoder configurations: linear (lin), convolutional (conv), and MLP-based (mlp), with the decoder architecture fixed to linear. All models use a latent dimensionality of 256. The table reports the relative changes in each metric compared to the convolutional baseline. The smallest relative changes, indicating greater robustness to architecture variation, are highlighted in bold. A ‘-’ denotes no change from the baseline.

Model	Encoder	MSE	IS	FID	ODI	DNR	ELBO
\mathcal{G} -VAE	conv	+0.0177	(-)0.2798	+0.7353	+0.1992	-	+22.0584
	linear	+0.0240	(-)0.3841	+0.7653	+0.0846	-	+25.0926
	mlp	+0.0176	(-)0.2946	+0.7399	+0.2115	-	+21.9732
\mathcal{L} -VAE	conv	+0.0156	(-)0.2725	+0.7188	+0.2334	-	+29.2157
	linear	+0.0238	(-)0.4157	+0.7473	+0.0628	-	+24.9707
	mlp	+0.0156	(-)0.2931	+0.7130	+0.2473	-	+20.0985
\mathcal{P} -VAE	conv	(-)0.0089	(-)0.1591	(-)0.0209	+2.1372	(-)0.0039	+4.1294
	linear	(-)0.0084	(-)0.1532	(-)0.0120	+1.9801	(-)0.0039	+4.4827
	mlp	(-)0.0085	(-)0.1426	(-)0.0184	+2.5861	(-)0.0039	+4.3650
\mathcal{C} -VAE	conv	+0.0395	(-)1.3892	+1.0160	+0.0908	+0.0037	+49.8702
	linear	+0.0399	(-)1.3954	+1.0165	(-)0.0340	+0.0039	+50.3558
	mlp	+0.0402	(-)1.3982	+1.0101	+0.3628	+0.0156	+50.8826
NegBio-VAE	conv	(-)0.0026	(-)0.1301	+0.1292	+0.5424	(-)0.0469	+5.8500
	linear	+0.0063	(-)0.1182	+0.1881	+0.1130	(-)0.0469	+12.6249
	mlp	(-)0.0029	(-)0.0807	+0.1136	+0.3828	(-)0.0469	+5.5485

This highlights the advantage of modeling overdispersed latent activations, which provide richer expressiveness without requiring high dimensionality. As the latent dimension increases, the performance gap between NegBio-VAE and other models narrows, but our method remains competitive and achieves the best or near-best results across all metrics. Notably, the shattering dimensionality of NegBio-VAE remains the highest or tied for highest in all latent dimensions, confirming that its latent space supports robust linear separability across diverse class partitions. These findings demonstrate that the flexibility of the Negative Binomial prior not only aids reconstruction but also translates into more generalizable and informative latent codes. We hypothesize that the slight performance drop of NegBio-VAE at latent dimension 128 is due to the intermediate capacity not fully aligning with the model’s expressive flexibility. While the Negative Binomial prior enables rich variance modeling, this flexibility can introduce instability or under-regularization at medium dimensionalities. In contrast, at low (10) or high (256) latent sizes, the model benefits from either tighter capacity constraints or sufficient regularization to produce robust and linearly separable codes.

D.1.2 Dead Neuron Rate Analysis on High Dimension

To further evaluate the expressiveness of different latent spaces, we measure the DNR at a high latent dimensionality of 512. A lower DNR indicates a more active and effective latent space, with fewer collapsed units. Among all models, the Gaussian VAE exhibits an extremely high DNR of 0.998, suggesting that nearly all latent dimensions are inactive. The Poisson VAE (0.002) and Categorical VAE (0.0117) show moderate sparsity, while both the Laplace VAE and our NegBio-VAE achieve a DNR of 0, indicating full utilization of the latent space. This demonstrates that NegBio-VAE can maintain activation across all latent units even at high dimensions, highlighting its robustness in avoiding neuron collapse and preserving representational capacity.

D.2 Additional Results on VAE Architecture Variants

Table 6 presents an ablation study on different encoder-decoder architectures using the MNIST dataset, with the latent dimensionality fixed at 256 across all models (for NegBio-VAE, we use NegBio-VAE_{DS-G}). Each model is evaluated under three encoder settings: convolutional (conv), linear (lin), and MLP-based (mlp), while the decoder architecture is fixed to linear. For each metric, the convolutional encoder and convolutional decoder configuration serve as the baseline. Relative differences (denoted by “+” or “-”) are reported with respect to this baseline. We report mean values and omit standard deviations for brevity. From the results, we observe that convolutional architectures consistently outperform

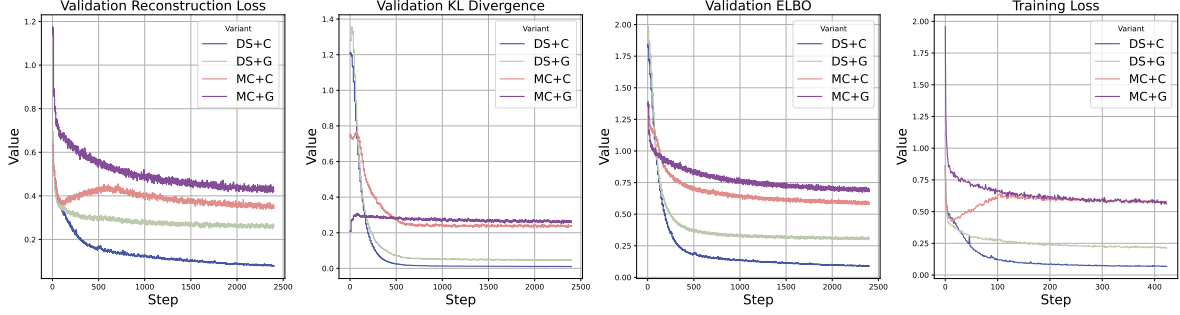


Figure 6: Comparison of four NegBio-VAE variants (DS+C, DS+G, MC+C, MC+G) across validation reconstruction loss, validation KL divergence, validation ELBO and training loss.

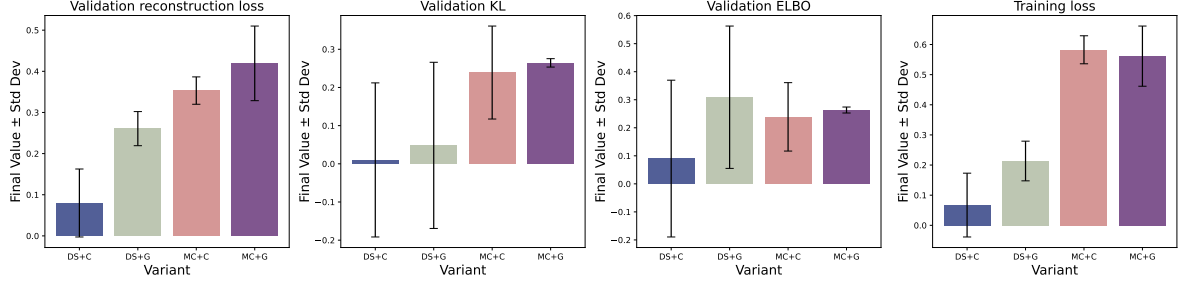


Figure 7: Comparison of convergence and oscillation statistics across four NegBio-VAE variants. Each bar shows the final loss value with standard deviation over training steps.

linear and MLP alternatives in most models, particularly for Gaussian and Laplace VAEs, where the convolutional encoder yields lower MSE and FID scores. Interestingly, the performance of Categorical VAE degrades significantly across all encoder types, indicating limited flexibility on MNIST. In contrast, NegBio-VAE not only achieves the best absolute performance (lowest MSE and competitive FID/IS scores in the convolutional setting), but also demonstrates the smallest relative differences across architectures. This indicates that NegBio-VAE is more robust to architectural variation. Moreover, it maintains high ODI and low DNR, suggesting more active and expressive latent dimensions. These findings underscore the advantage of modeling overdispersed latent distributions for capturing fine image details and achieving architecture-invariant performance.

D.3 Loss Dynamics Across NegBio-VAE Variants

Fig. 6 presents a comparison of the training dynamics for four NegBio-VAE variants across different loss terms (validation reconstruction loss, validation KL, validation ELBO and train loss). We observe notable differences in both the convergence rate and the smoothness of the trajectories, which reflect the influence of the KL estimation method and the reparametrization strategy. Overall, models with MC KL estimation (MC+G and MC+C) exhibit higher variance in the loss curves, with visible oscillations due to the stochastic nature of the Monte Carlo method (which introduces noise into the gradient updates as we have discussed in Section 4.1). In contrast, DS-based variants (DS+C and DS+G), which leverage closed-form KL computation via dispersion sharing, show smoother and more stable curves, suggesting better optimization stability. Comparing reparameterization strategies, models using continuous-time simulation (C) (DS+C and MC+C) tend to achieve lower reconstruction loss and faster ELBO convergence than their Gumbel-softmax (G). This suggests that the continuous-time approach offers a more expressive and stable mechanism for modeling spike-like latent representations. In particular, DS+C demonstrates the most stable and efficient convergence across all loss types, with consistently smooth trajectories and lower final values. We also compute convergence and oscillation statistics across the four variants, including the final value, mean, standard deviation, and range. These results are visualized in Fig. 7. From the plots, we observe that DS+C exhibits the lowest oscillation and the lowest final value in validation reconstruction loss, indicating the most stable training behavior. In contrast, MC+G shows the highest degree of oscillation—with the largest standard deviation—particularly in KL divergence and ELBO, reflecting its instability during training.