Zero-Variance Gradients for Variational Autoencoders

Zilei Shao[1], **Anji Liu**[2], **Guy Van den Broeck**[1]

[1]University of California, Los Angeles
[2]School of Computing, National University of Singapore
{zileishao, guyvdb}@cs.ucla.edu, anjiliu@nus.edu.sg

Abstract

Training deep generative models like Variational Autoencoders (VAEs) is often hindered by the need to backpropagate gradients through the stochastic sampling of their latent variables, a process that inherently introduces estimation variance, which can slow convergence and degrade performance. In this paper, we propose a new perspective that sidesteps this problem, which we call **Silent Gradients**. Instead of improving stochastic estimators, we leverage specific decoder architectures to analytically compute the expected ELBO, yielding a gradient with zero variance. We first provide a theoretical foundation for this method and demonstrate its superiority over existing estimators in a controlled setting with a linear decoder. To generalize our approach for practical use with complex, expressive decoders, we introduce a novel training dynamic that uses the exact, zero-variance gradient to guide the early stages of encoder training before annealing to a standard stochastic estimator. Our experiments show that this technique consistently improves the performance of established baselines, including reparameterization, Gumbel-Softmax, and REINFORCE, across multiple datasets. This work opens a new direction for training generative models by combining the stability of analytical computation with the expressiveness of deep, nonlinear architecture.

1 Introduction

Training of neural networks with stochastic components, such as the sampling of latent variables in generative models, often suffers from high variance in gradient estimates. This variance can impede the optimization process, leading to slower convergence and suboptimal model performance. In Variational Autoencoders (VAEs) (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014), for instance, gradients must be propagated through a stochastic sampling layer. This has led to the development of several estimation techniques. For continuous latent spaces, the reparameterization trick (Kingma and Welling 2014) is commonly used. For discrete spaces, common approaches include the REINFORCE algorithm (Williams 1992) and the Gumbel-Softmax trick (Maddison, Mnih, and Teh 2017; Jang, Gu, and Poole 2017). However, all of these sample-based techniques introduce estimation variance, and in this paper we show that this variance hinders the optimization even in a simple, controlled setting.

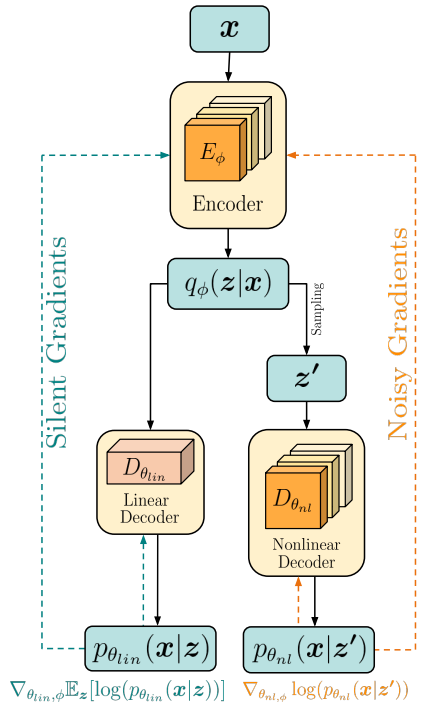

Figure 1: Illustration of the use of Silent Gradients in training VAEs. The encoder (E_ϕ) takes input $\boldsymbol{x}$ and infers a latent distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. These parameters are fed directly to the linear decoder (D_{lin}), which computes the analytical reconstruction log-likelihood, yielding a noise-free (Silent) gradient (dashed teal arrow) used to train the encoder. In parallel, samples $\boldsymbol{z}'$ are drawn from the latent distribution and fed to the nonlinear decoder (D_{nl}), which produces a standard, sample-based loss, resulting in a noisy gradient (dashed orange arrow). The solid black arrows represent the forward pass, while the dashed teal and orange arrows indicate the flow of gradients. During training, we can choose to train the encoder solely with the Silent Gradients or combine it with the noisy gradient using an annealing schedule. At inference time, only the trained encoder E_ϕ and nonlinear decoder D_{nl} are used.

In this paper, we propose a fundamentally different perspective on gradient estimation for VAEs. Given the estimation variance introduced by these stochastic gradient estimators, we argue for a different paradigm. Instead of developing more sophisticated techniques to *estimate* the gradient of an expectation, we explore the possibility of first efficiently[1] computing the expectation itself in closed form, and then differentiating the resulting analytical expression. This path, when available, yields a gradient that is computed exactly and therefore has *zero variance* by definition, in terms of the latent variables.

The feasibility of this approach hinges on the decoder architecture. While it is well-known that for a linear function, the expectation of its output can be computed exactly by linearity of expectation, this does not trivially extend to the full reconstruction log-likelihood. We first show that for a Gaussian likelihood with a fixed variance, the expected loss can still be computed in closed form, as a function of the latent distribution rather than the sampled latent variables. We then empirically demonstrate that using this analytic gradient leads to superior performance and faster convergence compared to standard stochastic estimators in this setting.

Furthermore, we extend this technique to a more expressive setting where the output variance is also a learnable function of the same latent variables, again providing a zero-variance gradient. This analytic gradient component can then boost the performance of existing standard stochastic gradient estimators.

Finally, to generalize our method for more complex and practical settings, we introduce a novel training dynamic, depicted in Figure 1, that combines our analytic gradient component with standard, expressive nonlinear decoders. By using Silent Gradients to guide the initial training of the encoder before annealing to a conventional estimator, our technique serves as a powerful variance reduction tool that consistently improves the performance of established methods.

Our experimental results on the MNIST (Deng 2012), ImageNet (Deng et al. 2009), and CIFAR-10 (Krizhevsky and Hinton 2009) datasets demonstrate a significant and consistent improvement in model performance. This shows that architectural choices that provide exact gradients are a powerful and general strategy for improving the training dynamics of models with stochastic layers.

2 Background

Variational Autoencoders (VAEs) (Kingma and Welling 2014) are a class of generative models for learning the probability distribution $p(\boldsymbol{x})$ that underlies a dataset. VAEs introduce a set of latent variables $\boldsymbol{z}$ that are assumed to generate the observed data $\boldsymbol{x}$. The model consists of two components: a prior distribution over the latent space $p(\boldsymbol{z})$ and a conditional likelihood distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, defined by the decoder D_θ. The marginal likelihood of the model given the data then is $p_\theta(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[p_\theta(\boldsymbol{x}|\boldsymbol{z})]$.

Since direct maximization of this likelihood is generally intractable, VAEs introduce a variational approximation $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to the true posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, called the encoder E_ϕ, parameterized by ϕ. Instead of maximizing the log-likelihood directly, one maximizes the Evidence Lower Bound (ELBO) (Jordan et al. 1999):

$$\mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})). \quad (1)$$

The first term is the expected reconstruction log-likelihood, which encourages the decoder to reconstruct the input $\boldsymbol{x}$ from its latent representation $\boldsymbol{z}$. The second term is the Kullback-Leibler (KL) divergence, which forces the approximate posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to be close to the prior $p(\boldsymbol{z})$. Maximizing the ELBO provides a framework to jointly train the encoder and decoder parameters, ϕ and θ, respectively.

3 Exact ELBO with Linear Decoder

While the KL divergence term is often analytically tractable by employing the mean-field assumption, in which the approximate posterior factorizes across latent dimensions: $q(\boldsymbol{z}|\boldsymbol{x}) := \prod_i q(z_i|\boldsymbol{x})$, the reconstruction term, $\mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})]$, remains intractable to compute. This difficulty comes from the complexity of the decoder function $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, which is typically a deep neural network. Consequently, estimating the gradient of this reconstruction term with respect to the encoder parameters ϕ poses a challenge since the gradient operator ∇_ϕ cannot be passed inside an expectation that depends on those same parameters.

To resolve this problem, various techniques have been introduced (Williams 1992; Jang, Gu, and Poole 2017; Maddison, Mnih, and Teh 2017; Kingma and Welling 2014). However, we show that even widely-used estimators like the reparameterization trick can be far from optimal. Specifically, we compute the average gradient variance of the ELBO for single samples on the MNIST dataset (Deng 2012) at three different training stages (epochs 10, 200, and 500). This allows us to directly compare the gradient noise introduced by each estimator and observe how it evolves during optimization.

As shown in Section 2, the variance of the gradients for standard estimators is substantial. Even for the reparameterization trick, which is considered a low-variance method, the gradient noise is significant and can hinder optimization. This naturally raises the question: how much performance is lost to this estimation variance, and what could be gained if we were able to compute the exact ELBO and its gradients? This motivates our exploration of an analytical approach.

3.1 Analytic gradient

As established, the intractability of the reconstruction term comes from the complexity of the decoder $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, which is typically a deep neural network. This motivates an investigation into specific architectural choices where this analytical bottleneck can be resolved. We show that for a specific model structure, the expectation in the reconstruction term can be computed exactly (i.e., the first term in Eq. 1), bypassing the need for stochastic estimation entirely. Let the data $\boldsymbol{x}$ and latent variable $\boldsymbol{z}$ be column vectors of dimensions k and d, respectively. We consider a generative likelihood $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ that is a Gaussian distribution with a mean produced by a linear decoder and a fixed, scalar variance σ^2:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; \mathbf{W}_\mu \boldsymbol{z}, \sigma^2 I). \quad (2)$$

[1] In time linear in the number of latent dimensions.

	Method	Epoch		
		10	200	500
Continuous	Silent Gradients	0	0	0
	Reparameterization	1.08×10^5	1.78×10^4	1.37×10^4
Discrete	Silent Gradients	0	0	0
	Gumbel-Softmax	1.10×10^5	2.08×10^5	1.68×10^5
	REINFORCE	2.30×10^8	6.30×10^7	4.00×10^7

Table 1: Comparison of the total gradient variance with respect to the encoder parameters. Continuous and Discrete refer to the type of latent space. The variance is measured by repeatedly sampling gradients for a fixed input batch at different training epochs. The results show that existing gradient estimators have substantial variance. In contrast, our method (Silent Gradients) has a true variance of zero as its gradient is computed analytically.

where the decoder is parameterized by the weight matrix $\mathbf{W}_\mu \in \mathbb{R}^{k \times d}$. For simplicity, we will denote the expectation $\mathbb{E}_{\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}$ as $\mathbb{E}$ where the context is clear.

Although this linear setup may seem restrictive, we will show in later sections that this technique forms the basis of a general method applicable to any VAEs. For now, we proceed by substituting this linear decoder structure into the ELBO reconstruction term:

$$\mathbb{E}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] = -\frac{1}{2\sigma^2}\mathbb{E}[||\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z}||_2^2] - \frac{1}{2}\log(2\pi\sigma^2). \tag{3}$$

To compute this term exactly, all we need is to find an analytical form for the expectation $\mathbb{E}[||\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z}||_2^2]$. We begin by expanding the squared L2 norm:

$$\begin{aligned}\mathbb{E}[||\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z}||_2^2] &= \mathbb{E}[(\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z})^T(\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z})], \\ &= \mathbb{E}[||\boldsymbol{x}||_2^2 - 2\boldsymbol{x}^T\mathbf{W}_\mu \boldsymbol{z} + ||\mathbf{W}_\mu \boldsymbol{z}||_2^2].\end{aligned}$$

By linearity of expectation, and because x and W_μ are constants with respect to the expectation over $\boldsymbol{z}$, we simplify:

$$\mathbb{E}[||\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z}||_2^2] = ||\boldsymbol{x}||_2^2 - 2\boldsymbol{x}^T\mathbf{W}_\mu\mathbb{E}[\boldsymbol{z}] + \mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2].$$

The challenge lies in resolving the third term $\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2]$ since a naive computation of this expectation would take time $\mathcal{O}(k^2 d)$, which is quadratic w.r.t. the number of $\mathbf{X}$ variables. However, we can reduce the complexity by exploiting the fact that different variables in $\boldsymbol{z}$ are mutually independent due to the mean-field assumption.

We begin by expanding the quadratic term into a double summation $\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2] = \sum_i \sum_j \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \mathbb{E}[z_i z_j]$, where $\mathbf{w}_{\mu,i}$ is the ith column of $\mathbf{W}_\mu$. Using the identity $\mathbb{E}[z_i z_j] = \mathbb{E}[z_i]\mathbb{E}[z_j] + \text{Cov}(z_i, z_j)$, we split this summation into two parts:

$$\sum_{i,j} \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \mathbb{E}[z_i]\mathbb{E}[z_j] + \sum_{i,j} \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \,\text{Cov}(z_i, z_j).$$

The first term, containing the expectations, can be factored into the square of a sum: $\left(\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]\right)^2$, which can be computed in $\mathcal{O}(kd)$ time. The second covariance term can be simplified due to the independence of the latent variables. The covariance is zero for all z_i, z_j $(i \neq j)$ pairs, zeroing out all cross-terms. This collapses the double summation into a single sum over the variances $\text{Var}(z_i)$. The final analytical expression is the sum of these two simplified components:[2]

$$\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2] = ||\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]||_2^2 + \sum_i ||\mathbf{w}_{\mu,i}||_2^2 \text{Var}(z_i). \tag{4}$$

Therefore, the expected squared error becomes:

$$\begin{aligned}\mathbb{E}[||\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z}||_2^2] = ||\boldsymbol{x}||_2^2 - 2\boldsymbol{x}^T &\left(\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]\right) \\ &+ \left|\left|\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]\right|\right|_2^2 + \sum_i ||\mathbf{w}_{\mu,i}||_2^2 \text{Var}(z_i).\end{aligned}$$

Finally, we substitute this closed-form expression back into the expected reconstruction log-likelihood $\mathbb{E}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})]$:

$$-\frac{1}{2\sigma^2}\left[||\boldsymbol{x} - \mathbf{W}_\mu \boldsymbol{z}||^2 + \sum_i ||\mathbf{w}_{\mu,i}||_2^2 \text{Var}(z_i)\right] - \frac{1}{2}\log(2\pi\sigma^2).$$

This final equation is fully analytical. The expectation over the latent variable $\boldsymbol{z}$ has been entirely eliminated, and the reconstruction loss now depends only on the mean $(\mathbb{E}[z_i])$ and variance $\text{Var}(z_i)$ of the latent distribution. This allows for direct and analytic gradient computation with respect to the encoder parameters.

3.2 Do Analytic Gradients help?

As we show that the ELBO (and its gradients) can be computed exactly in an idealized setting with a linear decoder, we now investigate the practical impact of this zero-variance gradient. We conduct a controlled experiment on the MNIST dataset (Deng 2012) to quantify the performance gains from eliminating gradient estimation noise. This experiment uses the same simple VAE with a linear decoder and fixed output variance ($\sigma^2 = 0.01$) (cf. Eq. (2)) as our our gradient variance analysis in Section 2, a setting designed to isolate the estimator's impact rather than to achieve state-of-the-art results. To ensure a fair comparison, we perform a separate hyperparameter search for each method.

[2]The detailed step-by-step derivation is in Appendix A.1.

	Method	MNIST	
		BPD (↓)	MSE (↓)
Continuous	Silent Gradients	**6.718**	**3.011**
	Reparameterization	6.722	3.059
Discrete	Silent Gradients	**6.900**	**6.103**
	Gumbel-Softmax	6.990	7.670
	REINFORCE	7.208	9.289

Table 2: Performance comparison on MNIST using a linear decoder with a latent dimension of 200 under fixed variance $\sigma^2 = 0.01$. The best BPD and MSE values for the continuous and discrete latent space are in bold, respectively. Silent Gradients consistently outperforms stochastic gradient estimators in terms of BPD and MSE.

By epoch 500, all models have converged, and we report all final metrics at this point. All further details regarding the model architecture and hyperparameters can be found in Appendix B. We benchmark our method, Silent Gradients, against the reparameterization trick in the continuous latent space and against the Gumbel-Softmax estimator and the REINFORCE algorithm in the discrete space. Model performance is evaluated using Bits Per Dimension (BPD) and Mean Squared Error (MSE).

The results, summarized in Section 3.2, demonstrate the consistent advantages of our method. In the discrete latent space setting, our method achieves a substantially lower BPD than the corresponding baseline estimators. In the continuous case, while the final BPD scores are comparable, our method demonstrates significantly faster convergence; Silent Gradients reaches a BPD of 6.73 in just 45 epochs, a milestone that the standard reparameterization trick requires 90 epochs to achieve.

Besides BPD scores, the low MSE of our method confirms the high-fidelity image reconstruction, indicating that the reported BPD is not limited by poor reconstruction quality but is instead constrained by the fixed-variance assumption. The sharp reconstructions in Appendix C visually corroborate this conclusion.

4 More Expressive Decoders

We have shown that Silent Gradients offers a significant performance boost when the analytic gradient of the ELBO can be tractably computed. However, it is still unclear how to apply our method to VAEs with more general decoders. We address this in two steps: first, we demonstrate how to generalize the linear Gaussian decoder setting to make the variance a learnable parameter. Next, we show that this tractable linear component can be integrated with any existing VAE to guide encoder learning.

4.1 Linear Decoders with Learnable Variance

A key limitation of the fixed-variance Gaussian decoder introduced in the previous section is its inability to dynamically adjust confidence across different variables, resulting in significant performance degradation. This motivates generalizing our approach to allow variance to be a learnable, data-dependent function of the latent variable $\boldsymbol{z}$. That is, given latents $\boldsymbol{z}$, we predict both the mean $\mu(\boldsymbol{z})$ and the variance $\sigma^2(\boldsymbol{z})$ of the Gaussian distribution.

Under this parameterization, the first term of the expected reconstruction log-likelihood (i.e., Eq. (3)) is generalized to $\mathbb{E}[\frac{(\boldsymbol{x}-\mu(\boldsymbol{z}))^2}{2\sigma^2(\boldsymbol{z})}]$. Computing the expectation of a reciprocal is #P-hard for simple function classes including multilinear polynomials (Vergari et al. 2021). Furthermore, to ensure the expression is well-defined, $\sigma^2(\mathbf{z})$ must be strictly positive. While this can be enforced through techniques such as lower-bounding the variance by clipping, these approaches introduce discontinuities that complicate the analytical computation of the expectation and may hinder stable optimization. In addition, the second term in the reconstruction log-likelihood, $\frac{1}{2}\log(2\pi\sigma^2(\boldsymbol{z}))$, that involves the expectation of a logarithm, is also computationally hard (Vergari et al. 2021).

To sidestep these challenges, we propose to represent the scale of the Gaussian distribution by the reciprocal of the standard deviation.[3] Formally, this quantity is called precision and is defined as $\alpha(\boldsymbol{z}) = 1/\sigma(\boldsymbol{z})$.

Following Section 3, we define both the mean $\mu(\boldsymbol{z})$ and the precision as linear functions of the latent variable $\boldsymbol{z}$:

$$\mu(\boldsymbol{z}) = \mathbf{W}_\mu \boldsymbol{z}, \quad \alpha(\boldsymbol{z}) = \mathbf{W}_\alpha \boldsymbol{z},$$

which gives the model flexibility to assign pixel-wise uncertainty. Following eq. (2), we define the generative likelihood as a Gaussian distribution where the mean is given by $\mu(\boldsymbol{z})$ and the variance is the element-wise inverse square of $\alpha(\boldsymbol{z})$:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}\left(\boldsymbol{x}; \mu(\boldsymbol{z}), \text{diag}\left(\frac{1}{\alpha(\boldsymbol{z})^2}\right)\right). \tag{5}$$

By the definition of the precision, the expected reconstruction log-likelihood from the ELBO becomes:

$$\begin{aligned}\mathbb{E}[\log p_\theta(x|z)] = -\frac{1}{2}\mathbb{E}\left[||(\boldsymbol{x} - \mu(\boldsymbol{z})) \odot \alpha(\boldsymbol{z})||_2^2\right] \\ + \mathbb{E}[\log(\alpha(\boldsymbol{z}))] - \frac{1}{2}\log(2\pi).\end{aligned} \tag{6}$$

The exact computation of both the first term and the second term is non-trivial. The first term involves an expectation of products of correlated functions of $\boldsymbol{z}$. The second term is hard since $\mathbb{E}[\log(\alpha(\boldsymbol{z}))] \neq \log(\mathbb{E}[\alpha(\boldsymbol{z})])$.

We expand the first expectation term as follows:

$$\begin{aligned}&\mathbb{E}\left[||(\boldsymbol{x} - \mu(\boldsymbol{z}))^T \alpha(\boldsymbol{z})||_2^2\right] = \\ &\quad \mathbf{1}^T\big(||\boldsymbol{x}||_2^2\,\mathbb{E}\left[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2\right] - 2\boldsymbol{x} \odot \mathbb{E}\left[\mathbf{W}_\mu \boldsymbol{z}||\mathbf{W}_\alpha \boldsymbol{z}||_2^2\right] \\ &\quad + \mathbb{E}\left[||\mathbf{W}_\mu \boldsymbol{z}||_2^2||\mathbf{W}_\alpha \boldsymbol{z}||_2^2\right]\big).\end{aligned}$$

The first term in this expansion, $\mathbb{E}\left[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2\right]$, is identical in form to the quadratic term derived in the fixed-variance setting (i.e. Equation (4)). As we showed previously, it can

[3] This parametrization aligns with the classical notion of precision, as originally defined by Gauss (1877); today the term precision is also used to denote the reciprocal of the variance.

be computed analytically, depending on only the mean and variance of latent distribution.

For the two remaining terms, $\mathbb{E}[\mathbf{W}_\mu \boldsymbol{z} ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2]$, and $\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2 ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2]$, we begin the derivation by separating the terms into their expected values and covariances:

$$\mathbb{E}\left[\mathbf{W}_\mu \boldsymbol{z} ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2\right] = \mathbb{E}[\mathbf{W}_\mu \boldsymbol{z}]\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2] + \text{Cov}(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2).$$

$$\mathbb{E}\left[||\mathbf{W}_\mu \boldsymbol{z}||_2^2 ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2\right] = \mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2]\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2] + \text{Cov}(||\mathbf{W}_\mu z||_2^2, ||\mathbf{W}_\alpha z||_2^2).$$

The challenge, therefore, lies in deriving the two covariance terms. Following the principles for computing the covariance of products of random variables by Bohrnstedt and Goldberger (1969), these terms can be decomposed into functions of central moments of the individual latent variables z_i. This makes the tractability of the entire expression dependent on whether these underlying central moments can be computed in closed form, as shown below.

Proposition 1. *(Tractable Central Moments)*
Let $\boldsymbol{z} \in \mathbb{R}^d$ be a random vector with independent components z_i. The first four central moments of each component, $\mathbb{E}[\tilde{z}_i] := \mathbb{E}[(z_i - \mathbb{E}[z_i])^k]$ for $k \in \{1, 2, 3, 4\}$, can be computed in closed form of the parameters of its distribution if z_i follows:

1. A Gaussian distribution, $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$.
2. A Bernoulli distribution, $z_i \sim Bern(p_i)$.

Derivation Sketch. The proof follows from the definitions of the moment-generating functions for each distribution. For a Gaussian variable, the central moments can be derived to be simple functions of its variance σ_i^2 (Winkelbauer 2014). For a Bernoulli variable with probability p_i, the raw moments $\mathbb{E}[z_i^k]$ are trivial to compute, and the central moment of order k is given by $(1 - p_i)(-p_i)^k + p_i(1 - p_i)^k$, resulting in polynomials of p_i. The full derivations are provided in the Appendix A.

With tractability of the individual central moments established, we can now show how this allows for the analytical computation of the full covariance terms.

Theorem 1. *(Analytic Covariance of Linear Projections)*
Let $\mathbf{W}_\mu \boldsymbol{z}$ and $\mathbf{W}_\alpha \boldsymbol{z}$ be two linear projections of a random vector $\boldsymbol{z}$ whose components z_i are independent. The covariance terms $Cov(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||^2)$ and $Cov(||\mathbf{W}_\mu \boldsymbol{z}||_2^2, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)$ can be expressed as a linear combination of the first four central moments of the components z_i. The coefficients of this linear combination are polynomials in the entries of the matrices $\mathbf{W}_\mu$ and $\mathbf{W}_\alpha$.

Proof Sketch. The proof relies on the formula for the covariance of products of random variables. The full derivation is in Appendix A.

1. For the term $\text{Cov}(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)$: We decompose this covariance term into a function of third-order expectations, $\mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_k]$, where $\tilde{\boldsymbol{z}} = \boldsymbol{z} - \mathbb{E}[\boldsymbol{z}]$. Due to the independence of the latent variables z_i, these expectations are non-zero only when all indices are identical ($i = j = k$). This simplifies the expression to a function of the second and third central moments of z_i. The resulting expression is a linear combination of the second and third central moments of z_i. Its coefficients are third-degree polynomials of the weight matrices $\mathbf{W}_\mu$ and $\mathbf{W}_\alpha$.

2. For the term $\text{Cov}(||\mathbf{W}_\mu \boldsymbol{z}||_2^2, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)$: Similarly, this term can be decomposed into functions of fourth-order expectations, $\mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_k \tilde{z}_l]$. Under the independence assumption, these complex expectations simplify into a linear combination of the second, the third, and the fourth central moments. The coefficients are fourth-degree polynomials of the weight matrices.

While the covariance terms can be computed analytically, the full log-likelihood function as in Equation (6) still contains the intractable logarithmic term $\mathbb{E}[\log(\mathbf{W}_\alpha \boldsymbol{z})]$, To ensure the argument of the logarithm is non-negative and to maintain a tractable, zero-variance objective, we approximate the term $\mathbb{E}[\log(||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)]$ using a second order Taylor Expansion around the mean of $||\mathbf{W}_\alpha \boldsymbol{z}||_2^2$ (Teh, Newman, and Welling 2006):

$$\mathbb{E}[\log(||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)] \approx \log(\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)] - \frac{\text{Var}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2]}{2(\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z})^2]||_2^2}.$$

By combining the exact computations for the covariance terms with this approximation, the expected reconstruction log-likelihood can be expressed in an analytical solution:

$$\begin{aligned}
&\mathbb{E}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] \\
&= \frac{1}{2}\left[||\boldsymbol{x}||_2^2 \mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2] - 2\boldsymbol{x}^T(\mathbf{W}_\mu \mathbb{E}[\boldsymbol{z}]\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||^2])\right. \\
&\quad + \text{Cov}(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2) + \mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2]\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2] \\
&\quad \left. + \text{Cov}(||\mathbf{W}_\mu \boldsymbol{z}||_2^2, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)\right] + \frac{1}{2}\log(2\pi) \\
&\quad \left(\frac{1}{2}\log(\mathbb{E}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2]) - \frac{\text{Var}[||\mathbf{W}_\alpha \boldsymbol{z}||_2^2]}{4(\mathbb{E}[||\mathbf{W}_\alpha z||_2^2])^2}\right). \qquad (7)
\end{aligned}$$

This expression now relies only on the tractable moments of the latent distribution and the decoder weights.

4.2 Silent Gradients with General VAEs

While the preceding section demonstrates that analytical gradients are tractable for linear decoders with learnable variance, the expressive power of a purely linear model is limited. To handle more complex data distribution, we now introduce a training strategy that integrates the benefits of our tractable Silent Gradients with general, powerful nonlinear decoders.

Our approach uses a dual-decoder architecture consisting of a shared encoder, a linear decoder for computing the exact ELBO component and computing the exact Silent Gradients, and a parallel, more expressive nonlinear decoder for generating the final reconstructions. A visualization of this pipeline is presented in Figure 1.

The training follows a two-stage process. In the initial stage, the encoder and both decoders are trained, but the encoder parameters are updated only using the analytic gradients from the linear decoder. After a set number of epochs, we freeze the encoder's weights. In the second stage, only

	Method	MNIST		ImageNet		CIFAR-10	
		Without SG	With SG	Without SG	With SG	Without SG	With SG
Continuous	None	–	2.41	–	5.98	–	5.82
	Reparameterization	1.91	**1.80**	5.79	**5.69**	5.70	**5.53**
Discrete	None	–	2.77	–	6.45	–	6.72
	Gumbel-Softmax	2.48	**2.37**	6.31	**6.20**	6.22	**6.19**
	REINFORCE	2.96	**2.94**	6.87	**6.77**	6.74	**6.67**

Table 3: Performance comparison (BPD) (↓) for models with learnable variance across different datasets and methods. The BPD score for the combined method is in bold when it is higher than its corresponding baseline. In both cases, the optimal performance is achieved by combining a standard estimator with our Silent Gradients technique. The results show that combining standard estimators with our Silent Gradients (SG) technique consistently improves performance. Additionally, our method used as a standalone estimator is competitive with and often superior to established baselines like REINFORCE.

Algorithm 1: Training Dynamics for Integrating Silent Gradients

Require: Encoder E_ϕ, Linear Decoder for $\alpha(\boldsymbol{z})$ $D_{lin,\alpha}$, Linear Decoder for $\mu(\boldsymbol{z})$ $D_{lin,\mu}$, Nonlinear Decoder D_{nl}

Require: Training data $\mathcal{D}$, cut-off epoch N_{cutoff}, annealing rate λ

for n_{epoch} = 1 to N_{max} **do**
  if n_{epoch} == N_{cutoff} **then**
    Freeze parameters ϕ of the encoder E_ϕ
  end if
  for batch $\boldsymbol{x}$ in $\mathcal{D}$ **do**
    $\boldsymbol{z}$, stats $\leftarrow E_\phi(\boldsymbol{x})$
    $\mathcal{L}_{lin} \leftarrow -D_{lin}(\text{stats}, \boldsymbol{x})$ ▷ Analytical ELBO component Equation (7)
    $\mathcal{L}_{nl} \leftarrow -\log p_{nl}(\boldsymbol{x}|\boldsymbol{z})$ ▷ Sampled reconstruction loss
    $w_{lin} \leftarrow \max(0, 1 - n_{epoch} \cdot \lambda)$
    $w_{nl} \leftarrow 1 - w_{lin}$
    $\mathcal{L}_{recon} \leftarrow w_{lin} \cdot \mathcal{L}_{lin} + w_{nl} \cdot \mathcal{L}_{nl}$
    $\mathcal{L}_{total} \leftarrow \mathcal{L}_{recon} + D_{KL}$
    Take gradient step on $\mathcal{L}_{total}$ for all unfrozen parameters
  end for
end for

the nonlinear decoder continues to train, fine-tuning its parameters on the now fixed, well-structured latent space provided by the encoder.

The Silent Gradients framework can be extended to boost the performance of existing gradient estimators. Instead of having the encoder rely solely on the linear decoders' analytical gradient, we introduce a gradient annealing schedule. In this combined approach, the gradient signal sent to the encoder E_ϕ is a weighted average:

$$\nabla_{\phi,\text{total}} = w_{\text{lin}} \nabla_{\phi,\text{Silent}} + w_{\text{nl}} \nabla_{\phi,\text{Noisy}}, \tag{8}$$

where $w_{\text{nl}} = 1 - w_{\text{lin}}$, $\nabla_{\phi,\text{Silent}}$ is the analytical gradient from the linear decoder, and $\nabla_{\phi,\text{Noisy}}$ is the noisy gradient from the nonlinear decoder using stochastic estimators. The training begins with the weight of the Silent Gradients, w_{lin}, at 1.0 and the weight of the baseline estimator's gradient, w_{nl}, at 0.0. As training progresses, w_{lin} is gradually annealed to 0 while w_{nl} is increased to 1.0. This dynamic allows the encoder to first learns a representation guided by the noise-free, analytical signal before fine-tuning with the sample-based gradients from the full, expressive model. The complete training dynamic is detailed in Algorithm 1.

We benchmark both our standalone Silent Gradients method and the combined approach against baselines on MNIST, ImageNet, and CIFAR-10. All models were tuned for optimal hyperparameters and trained until convergence to ensure a fair comparison. Our experimental results, presented in Section 4.2, demonstrate two key findings. First, our Silent Gradients method consistently improves the performance of existing gradient estimators. In every case, combining a standard estimator with our technique results in a lower BPD score compared to the baseline alone across all tested datasets. This shows that our analytical gradient serves as a powerful and general-purpose training aid. Second, our method used as a standalone estimator is highly competitive, even outperforming the widely used REINFORCE estimator on both MNIST and ImageNet. We defer more experiment details to Appendix Appendix B, and the visualized reconstruction output is presented in Appendix C.

An analysis of the KL Divergence (KLD) offers an explanation for these performance gains. As shown in Section 4.2, models trained with Silent Gradients consistently achieve a higher KLD, which suggests that the encoder learns a more informative latent representation and better avoids posterior collapse. We hypothesize this is because the zero-variance analytical gradient provides a cleaner, more stable training signal to the encoder than the noisy gradients from the standard stochastic estimators.

It is important to note that while these results are not state-of-the-art, they are by design; our experiments use a simple decoder architecture to isolate the impact of our gradient computation technique, rather than to achieve record-breaking BPD. Our method provides a consistent and significant performance lift across all baselines, demonstrating its broad potential as a general tool for improving the training of deep generative models.

	Method	MNIST		ImageNet		CIFAR-10	
		Without SG	With SG	Without SG	With SG	Without SG	With SG
Continuous	None	–	330.77	–	478.43	–	550.70
	Reparameterization	155.15	**165.76**	382.87	**533.20**	427.79	**577.25**
Discrete	None	–	91.82	–	534.29	–	243.69
	Gumbel-Softmax	94.64	**96.24**	368.02	**404.57**	446.75	442.58
	REINFORCE	109.67	**128.55**	294.88	**381.27**	303.33	**367.10**

Table 4: KL Divergence (KLD) comparison for models with learnable variance. The KLD for the combined method is in bold when it is higher than its corresponding baseline. The results consistently show a higher KLD when a baseline estimator is combined with our Silent Gradient technique, which suggests the encoder learns a more informative latent representation.

5 Related Work

VAEs. Variational Autoencoders (VAEs) are generative models that learn a latent representation of data through an encoder-decoder framework (Kingma and Welling 2014). They can be categorized by their latent space: VAEs with continuous latent variables typically use Gaussian distributions and are widely applied to tasks like image modeling (Kingma and Welling 2014), while VAEs with discrete latent spaces have become an active research area, as discrete representations can offer better interpretability and computational efficiency (van den Oord, Vinyals, and Kavukcuoglu 2018; Jang, Gu, and Poole 2017). This line of work contains various architectures of the discrete latent space, such as the use of vector quantization in VQ-VAE (van den Oord, Vinyals, and Kavukcuoglu 2018) and relaxed Boltzmann priors in DAVE# (Vahdat, Andriyash, and Macready 2018).

Gradient Estimation Techniques. A key challenge in training VAEs is propagating gradients through stochastic sampling layers. In the continuous case, the reparameterization trick, which separates the stochasticity into a fixed noise source and a deterministic function, is widely used (Kingma and Welling 2014). Although unbiased, reparameterization still introduces variance that impedes optimization.

In the discrete case, two main lines of techniques are used. The first is the use of the REINFORCE technique, or score function estimator, which provides a general and unbiased gradient estimate applicable to both discrete and continuous latent variables. It rewrites the gradient of the expectation as: $\nabla_\phi \mathbb{E}_{q_\phi(\boldsymbol{z})} f(\boldsymbol{z}) = \mathbb{E}_{q_\phi(\boldsymbol{z})}[f(\boldsymbol{z}) \nabla_\phi \log q_\phi(\boldsymbol{z})]$ (Williams 1992). However, this estimator is often hindered by high variance, which has led to the development of variance reduction techniques such as control variates, (Mnih and Gregor 2014; Kool, van Hoof, and Welling 2019).

The second line of research strives to make discrete variables compatible with low-variance reparameterization trick. The straight-through (ST) estimator approximates the discrete sampling in the backward pass with a differentiable function, such as using the mean value for a Bernoulli variable (Bengio, Léonard, and Courville 2013). Another approach is to relax discrete variables into a continuous distribution; the Concrete (Maddison, Mnih, and Teh 2017) or Gumbel-Softmax (Jang, Gu, and Poole 2017) distribution, for instance, achieves this by adding Gumbel noise to the logits of a softmax function, enabling reparameterization. More recent techniques such as SIMPLE (Ahmed et al. 2023), IndeCateR (Smet, Sansone, and Martires 2023), and Implicit Maximum Likelihood Estimation (IMLE) (Niepert, Minervini, and Franceschi 2021) offer alternative strategies to derive low-variance gradient estimates for generative models with discrete latent variables.

Linear VAEs. Linear VAEs are a cornerstone in various contexts. First, their analytical tractability makes them an ideal setting for theoretical investigation. For example, Lucas et al. (2019) used linear VAEs to show that posterior collapse can be an inherent issue of the marginal log-likelihood objective, not a problem caused by the ELBO approximation. Other work uses them to investigate the implicit bias of gradient descent, showing how training dynamics can recover the ground-truth data manifold (Koehler et al. 2022). Additionally, linear decoders are also crucial in tasks such as learning sparse and interpretable features from complex data (Lu et al. 2025; Vafaii, Galor, and Yates 2024). This broad utility motivates our method, which allows for analytic gradient estimation for any VAE with a linear decoder. Having demonstrated its effectiveness in image modeling, Silent Gradients could directly enhance these other applications.

Analytic ELBO. Lucas et al. (2019) derive an analytical ELBO for linear VAEs under assumptions of a fixed scalar output variance, a Gaussian latent space and a linear encoder. In contrast, our method is more general, that supports a learnable variance for output Gaussian distribution, applies to any latent distribution with tractable central moments, and makes no assumptions about the encoder architecture.

6 Conclusion

In this work, we introduced Silent Gradients, a new approach to training VAEs without the problem brought by variance in gradient estimation. Instead of improving stochastic estimators, we leverage specific decoder architectures to analytically compute a zero-variance gradient signal. We provided a derivation for this method and demonstrated its effectiveness empirically. Our experiments show that Silent Gradients not only outperforms standard estimators in a controlled setting but also consistently improves their performance when combined through a novel training dynamic in general VAEs.

7 Acknowledgments

This work was funded in part by the DARPA ANSR, CODORD, and SAFRON programs under awards FA8750-23-2-0004, HR00112590089, and HR00112530141, NSF grant IIS1943641, and gifts from Adobe Research, Cisco Research, and Amazon. Approved for public release; distribution is unlimited. The authors would like to thank Benjie Wang for useful discussion on the proofs.

References

Ahmed, K.; Zeng, Z.; Niepert, M.; and den Broeck, G. V. 2023. SIMPLE: A Gradient Estimator for k-Subset Sampling. In *The Eleventh International Conference on Learning Representations*.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. arXiv:1308.3432.

Bohrnstedt, G. W.; and Goldberger, A. S. 1969. On the Exact Covariance of Products of Random Variables. *Journal of the American Statistical Association*, 64(328): 1439–1442.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.

Deng, L. 2012. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine*, 29(6): 141–142.

Gauss, C. F. 1877. *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*, volume 7. FA Perthes.

Jang, E.; Gu, S.; and Poole, B. 2017. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*.

Jordan, M. I.; Ghahramani, Z.; Jaakkola, T. S.; and Saul, L. K. 1999. An Introduction to Variational Methods for Graphical Models. *Mach. Learn.*, 37(2): 183–233.

Kingma, D. P.; and Welling, M. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Koehler, F.; Mehta, V.; Zhou, C.; and Risteski, A. 2022. Variational autoencoders in the presence of low-dimensional data: landscape and implicit bias. In *International Conference on Learning Representations*.

Kool, W.; van Hoof, H.; and Welling, M. 2019. Buy 4 REINFORCE Samples, Get a Baseline for Free! In *DeepRL-StructPred@ICLR*.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario.

Lu, Y.; Zhu, X.; He, T.; and Wipf, D. 2025. Sparse Autoencoders, Again? In *Forty-second International Conference on Machine Learning*.

Lucas, J.; Tucker, G.; Grosse, R.; and Norouzi, M. 2019. *Don't blame the ELBO! a linear VAE perspective on posterior collapse*. Red Hook, NY, USA: Curran Associates Inc.

Maddison, C. J.; Mnih, A.; and Teh, Y. W. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations*.

Mnih, A.; and Gregor, K. 2014. Neural Variational Inference and Learning in Belief Networks. In Xing, E. P.; and Jebara, T., eds., *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, 1791–1799. Bejing, China: PMLR.

Niepert, M.; Minervini, P.; and Franceschi, L. 2021. Implicit MLE: Backpropagating Through Discrete Exponential Family Distributions. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In Xing, E. P.; and Jebara, T., eds., *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, 1278–1286. Bejing, China: PMLR.

Smet, L. D.; Sansone, E.; and Martires, P. Z. D. 2023. Differentiable Sampling of Categorical Distributions Using the CatLog-Derivative Trick. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Teh, Y.; Newman, D.; and Welling, M. 2006. A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation. In Schölkopf, B.; Platt, J.; and Hoffman, T., eds., *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

Vafaii, H.; Galor, D.; and Yates, J. L. 2024. Poisson Variational Autoencoder. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Vahdat, A.; Andriyash, E.; and Macready, W. G. 2018. DVAE#: discrete variational autoencoders with relaxed boltzmann priors. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, 1869–1878. Red Hook, NY, USA: Curran Associates Inc.

van den Oord, A.; Vinyals, O.; and Kavukcuoglu, K. 2018. Neural Discrete Representation Learning. arXiv:1711.00937.

Vergari, A.; Choi, Y.; Liu, A.; Teso, S.; and Van den Broeck, G. 2021. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 13189–13201. Curran Associates, Inc.

Williams, R. J. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.*, 8(3–4): 229–256.

Winkelbauer, A. 2014. Moments and Absolute Moments of the Normal Distribution. arXiv:1209.4340.

A Derivation

In this section, we provide the step-by-step derivation for Equation (4), Proposition 1, and Theorem 1.

A.1 Equation (4) ($\mathbb{E}||\mathbf{W}_\mu \boldsymbol{z}||_2^2$)

We wish to prove:

$$\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2] = ||\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]||_2^2 + \sum_i ||\mathbf{w}_{\mu,i}||_2^2 \text{Var}(z_i). \tag{9}$$

Derivation. To begin with, we shall expand $\mathbb{E}||\mathbf{W}_\mu \boldsymbol{z}||_2^2$ using summations:

$$\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2] = \mathbb{E}\left[||\sum_i \mathbf{w}_{\mu,i} z_i||_2^2\right], \tag{10}$$

$$= \mathbb{E}\left[\sum_i \sum_j (\mathbf{w}_{\mu,i} z_i)^T (\mathbf{w}_{\mu,j} z_j)\right], \tag{11}$$

$$= \sum_i \sum_j \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \mathbb{E}[z_i z_j]. \tag{12}$$

where $\mathbf{w}_{\mu,i}$ is the ith column of $\mathbf{W}_\mu$. Notably, $\mathbb{E}[z_i z_j] = \mathbb{E}[z_i]\mathbb{E}[z_j] + \text{Cov}(z_i, z_j)$, by the fundamental identity relating the second moment of a random vector to its mean and covariance. Using this identity, we further split the expression into:

$$\begin{aligned} &= \sum_i \sum_j \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \mathbb{E}[z_i]\mathbb{E}[z_j] \\ &\quad + \sum_i \sum_j \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \text{Cov}(z_i, z_j). \end{aligned} \tag{13}$$

The first term, containing the expectations, can be factored into the square of a sum: $||\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]||_2^2$. The second term, involving the covariance is simplified by decomposing the summation into two cases. The first case is when the indices are equal ($i = j$), and the second is when they are not ($i \neq j$):

$$\begin{aligned} &\sum_i \sum_j \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \text{Cov}(z_i, z_j) \\ &= \sum_i ||\mathbf{w}_{\mu,i}||_2^2 \text{Cov}(z_i, z_i) + \sum_{i \neq j} \mathbf{w}_{\mu,i}^T \mathbf{w}_{\mu,j} \text{Cov}(z_i, z_j). \end{aligned} \tag{14}$$

By definition, the covariance of a variable with itself is its variance: $\text{Cov}(z_i, z_i) = \text{Var}(z_i)$. Additionally, we assume the components of the latent vector z are independent. A standard property of independent random variables is that their covariance is 0. Therefore, for all $i \neq j$, $\text{Cov}(z_i, z_j) = 0$, which cancels out the second summation entirely. By combining the simplified expectation and covariance terms, the final analytical expression for the quadratic term is:

$$\mathbb{E}[||\mathbf{W}_\mu \boldsymbol{z}||_2^2] = ||\sum_i \mathbf{w}_{\mu,i}\mathbb{E}[z_i]||_2^2 + \sum_i ||\mathbf{w}_{\mu,i}||_2^2 \text{Var}(z_i). \tag{15}$$

A.2 Proposition 1

Proposition 1. Let $\boldsymbol{z} \in \mathbb{R}^d$ be a random vector with independent components z_i. The first four central moments of each component, $\mathbb{E}[\tilde{z}_i] := \mathbb{E}[(z_i - \mathbb{E}[z_i])^k]$ for $k \in \{1, 2, 3, 4\}$, can be computed in closed form of the parameters of its distribution if z_i follows:

1. A Gaussian distribution, $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$.
2. A Bernoulli distribution, $z_i \sim \text{Bern}(p_i)$.

Derivation. 1. Let z_i be a random variable following a Gaussian distribution, $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$.

The kth central moment is defined as $\mathbb{E}[(z_i - \mathbb{E}[z_i])^k]$. Winkelbauer (2014) introduces the formula to calculate central moments for a Gaussian distribution:

$$\mathbb{E}[(z_i - \mu_i)^k] = \begin{cases} \sigma_i^k (k-1)!! & \text{if } k \in \mathbb{N}_+ \text{ is even,} \\ 0 & \text{if } k \in \mathbb{N}_+ \text{ is odd.} \end{cases} \tag{16}$$

where $(k-1)!!$ is the double factorial. Using this formula, we can state the first four central moments:

- $k = 1$. $\mathbb{E}[z_i - \mu_i] = 0$ since k is odd.
- $k = 2$. $\mathbb{E}[(z_i - \mu_i)^2] = \sigma_i^2 (2-1)!! = \sigma_i^2$ since k is even. Notably, the result is the variance of this Gaussian distribution.
- $k = 3$. $\mathbb{E}[(z_i - \mu_i)^3] = 0$ since k is odd.
- $k = 4$. $\mathbb{E}[(z_i - \mu_i)^4] = \sigma_i^4 (4-1)!! = 3\sigma_i^4$ since k is even.

Therefore, all four central moments are closed-form functions of the distribution's variance σ_i^2.

2. Let z_i be a random variable following a Bernoulli distribution, $z_i \sim \text{Bern}(p_i)$. The variable z_i takes the value 1 with probability p_i and 0 with probability $1 - p_i$. The mean is $\mathbb{E}[z_i] = p_i$.

The kth central moment is defined as $\mathbb{E}[(z_i - \mathbb{E}[z_i])]$. We consider the two possible outcomes for z_i:

- If $z_i = 1$, then $(z_i - p_i)^k = (1 - p_i)^k$.
- If $z_i = 0$, then $(z_i - p_i)^k = (-p_i)^k$.

We can compute the central moments using the definition of expectation.

$$\mathbb{E}[(z_i - p_i)^k] = (1 - p_i)^k \cdot p_i + (-p_i)^k \cdot (1 - p_i). \tag{17}$$

Now we can compute the first four central moments:

- $k = 1$. $\mathbb{E}[z_i - p_i] = (1 - p_i)^1 p_i + (-p_i)^1 (1 - p_i) = p_i - p_i^2 - p_i + p_i^2 = 0$.
- $k = 2$.

$$\begin{aligned} \mathbb{E}[(z_i - p_i)^2] &= (1 - p_i)^2 p_i + (-p_i)^2 (1 - p_i), \\ &= p_i - 2p_i^2 + p_i^3 + p_i^2 - p_i^3, \\ &= p_i - p_i^2 = p_i(1 - p_i). \end{aligned} \tag{18}$$

- $k = 3$.

$$\begin{aligned} \mathbb{E}[(z_i - p_i)^3] &= (1 - p_i)^3 p_i + (-p_i)^3 (1 - p_i), \\ &= (1 - 3p_i + 3p_i^2 - p_i^3) p_i - p_i^3 (1 - p_i), \\ &= p_i - 3p_i^2 + 3p_i^3 - p_i^4 - p_i^3 + p_i^4, \\ &= p_i - 3p_i^2 + 2p_i^3 \\ &= p_i(1 - p_i)(1 - 2p_i). \end{aligned} \tag{19}$$

- $k = 4$.

$$\begin{aligned}\mathbb{E}[(z_i - p_i)^4] &= (1-p_i)^4 p_i + (-p_i)^4(1-p_i),\\ &= (1 - 4p_i + 6p_i^2 - 4p_i^3 + p_i^4)p_i\\ &\quad + p_i^4(1-p_i),\\ &= p_i - 4p_i^2 + 6p_i^3 - 4p_i^4 + p_i^5 + p_i^4 - p_i^5,\\ &= p_i - 4p_i^2 + 6p_i^3 - 3p_i^4\\ &= p_i(1-p_i)(1 - 3p_i + 3p_i^2). \end{aligned} \tag{20}$$

Therefore, all four central moments are closed-form functions of the parameter p_i.

A.3 Theorem 1

Theorem 1. Let $\mathbf{W}_\mu \boldsymbol{z}$ and $\mathbf{W}_\alpha \boldsymbol{z}$ be two linear projections of a random vector $\boldsymbol{z}$ whose components z_i are independent. The covariance terms $\mathrm{Cov}(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||^2)$ and $\mathrm{Cov}(||\mathbf{W}_\mu \boldsymbol{z}||_2^2, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)$ can be expressed as a linear combination of the first four central moments of the components z_i. The coefficients of this linear combination are polynomials in the entries of the matrices $\mathbf{W}_\mu$ and $\mathbf{W}_\alpha$.

Proof. For simplicity in writing, we define:

$$u_1 = u_2 = \mathbf{W}_\mu \boldsymbol{z}, \quad v_1 = v_2 = \mathbf{W}_\alpha \boldsymbol{z}, \tag{21}$$

$$\Delta u_1 = \Delta u_2 = \mathbf{W}_\mu \boldsymbol{z} - \mathbb{E}[\mathbf{W}_\mu \boldsymbol{z}] = \mathbf{W}_\mu(\boldsymbol{z} - \mathbb{E}[\boldsymbol{z}]), \tag{22}$$

$$\Delta v_1 = \Delta v_2 = \mathbf{W}_\alpha \boldsymbol{z} - \mathbb{E}[\mathbf{W}_\alpha \boldsymbol{z}] = \mathbf{W}_\alpha(\boldsymbol{z} - \mathbb{E}[\boldsymbol{z}]). \tag{23}$$

And we denote $\tilde{\boldsymbol{z}} = \boldsymbol{z} - \mathbb{E}[\boldsymbol{z}]$, thus $\mathbb{E}[\tilde{\boldsymbol{z}}] = 0$, and $\mathbb{E}[(\tilde{\boldsymbol{z}})^2] = \mathrm{Var}(\boldsymbol{z})$.

Bohrnstedt and Goldberger (1969) introduces the formula to compute covariance between the products of independent variables as follows:

$$\begin{aligned}\mathrm{Cov}(u_1, v_1 v_2) = \mathbb{E}[v_1]\mathrm{Cov}(v_2, u_1) + \mathbb{E}[v_2]\mathrm{Cov}(v_1, u_1)\\ + \mathbb{E}[(\Delta v_1)(\Delta v_2)(\Delta u_1)].\end{aligned} \tag{24}$$

Identical to the fixed variance case, we can derive $\mathrm{Cov}(v_1, u_1) = \mathrm{Cov}(v_2, u_1) = \sum_i \mathbf{w}_{\alpha,i}^T \mathbf{w}_{\mu,i} \mathrm{Var}(z_i) = \mathbf{W}_\alpha^T \mathbf{W}_\mu \mathbb{E}[(\tilde{\boldsymbol{z}})^2]$. And to compute the last term, we expand it as follows, with $\odot$ denoting Hadamard product:

$$\mathbb{E}\big[(\Delta v_1)(\Delta v_2)(\Delta u_1)\big] \tag{25}$$

$$= \mathbb{E}\big[(\mathbf{W}_\mu \tilde{\boldsymbol{z}}) \odot (\mathbf{W}_\alpha \tilde{\boldsymbol{z}}) \odot (\mathbf{W}_\alpha \tilde{\boldsymbol{z}})\big], \tag{26}$$

$$= \mathbb{E}\big[\sum_i \sum_j \sum_k (\mathbf{w}_{\mu,i}\tilde{z}_i) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,k} \tilde{z}_j \tilde{z}_k)\big], \tag{27}$$

$$= \sum_i \sum_j \sum_k \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,k} \mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_k]. \tag{28}$$

Notably, $\mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_k]$ is nonzero only if $i = k = j$. In other generic cases, for example, $i = j \neq k$, we can always separate $\tilde{z}_i \tilde{z}_j$ from $\tilde{z}_k$. In fact, because of the constraint, we can simplify the expression:

$$\mathbb{E}[\tilde{z}_i \tilde{z}_i \tilde{z}_k] = \mathbb{E}[(\tilde{z}_i)^2 \tilde{z}_k] \tag{29}$$

$$= \mathbb{E}[(\tilde{z}_i)^2]\mathbb{E}[\tilde{z}_k] \tag{30}$$

$$= 0 \tag{31}$$

Therefore, the only case we need to consider is when $i = j = k$, and thus we can write:

$$\mathbb{E}\big[(\Delta v_1)(\Delta v_2)(\Delta u_1)\big] = \sum_i \sum_j \sum_k \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,k}$$

$$\mathbb{E}[\tilde{z}_i \tilde{z}_i \tilde{z}_i] \tag{32}$$

$$= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,k} \mathbb{E}[(\tilde{z}_i)^3] \tag{33}$$

Piecing all together, we derive the expression for the covariance term $\mathrm{Cov}(\mathbf{W}_\mu \boldsymbol{z}, (\mathbf{W}_\alpha \boldsymbol{z})^2)$:

$$\begin{aligned}\mathrm{Cov}(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2) &= (\mathbf{W}_\alpha \mathbb{E}[\boldsymbol{z}]) \odot (\mathbf{W}_\alpha \odot \mathbf{W}_\mu \mathbb{E}[(\tilde{\boldsymbol{z}})^2])\\ &\quad + (\mathbf{W}_\alpha \mathbb{E}[\boldsymbol{z}]) \odot (\mathbf{W}_\alpha \odot \mathbf{W}_\mu \mathbb{E}[(\tilde{\boldsymbol{z}})^2])\\ &\quad \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,k} \mathbb{E}[(\tilde{z}_i)^3],\\ &= 2(\mathbf{W}_\alpha \mathbb{E}[\boldsymbol{z}]) \odot (\mathbf{W}_\alpha \odot \mathbf{W}_\mu \mathbb{E}[(\tilde{\boldsymbol{z}})^2])\\ &\quad + \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,k} \mathbb{E}[(\tilde{z}_i)^3].\end{aligned} \tag{34}$$

Bohrnstedt and Goldberger (1969) also introduced the formula to calculate the covariance between two products of random variables:

$$\begin{aligned}&\mathrm{Cov}(u_1 u_2, v_1 v_2)\\ &= \mathbb{E}(u_1)\,\mathbb{E}(v_1)\,\mathrm{Cov}(u_2, v_2) + \mathbb{E}(u_1)\,\mathbb{E}(v_2)\,\mathrm{Cov}(u_2, v_1)\\ &\quad + \mathbb{E}(u_2)\,\mathbb{E}(v_1)\,\mathrm{Cov}(u_1, v_2) + \mathbb{E}(u_2)\,\mathbb{E}(v_2)\,\mathrm{Cov}(u_1, v_1)\\ &\quad + \mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_1\,\Delta v_2\big] + \mathbb{E}(u_1)\,\mathbb{E}\big[\Delta u_2\,\Delta v_1\,\Delta v_2\big]\\ &\quad + \mathbb{E}(u_2)\,\mathbb{E}\big[\Delta u_1\,\Delta v_1\,\Delta v_2\big] + \mathbb{E}(v_1)\,\mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_2\big]\\ &\quad + \mathbb{E}(v_2)\,\mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_1\big] - \mathrm{Cov}(u_1, u_2)\,\mathrm{Cov}(v_1, v_2).\end{aligned} \tag{35}$$

Following the derivation earlier, we can compute the terms $\mathbb{E}(u_1)\,\mathbb{E}\big[\Delta u_2\,\Delta v_1\,\Delta v_2\big]$, $\mathbb{E}(u_2)\,\mathbb{E}\big[\Delta u_1\,\Delta v_1\,\Delta v_2\big]$, $\mathbb{E}(v_1)\,\mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_2\big]$, $\mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_1\big]$. And similarly, we wish to consider all nonzero cases in $\mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_1\,\Delta v_2\big]$, and they are: $i = j = k = l, i = j \neq k = l, i = k \neq j = l, i = l \neq k = j$.

$$\mathbb{E}\big[\Delta u_1\,\Delta u_2\,\Delta v_1\,\Delta v_2\big] \tag{36}$$

$$= \mathbb{E}\big[\mathbf{W}_\mu \tilde{z} \odot \mathbf{W}_\mu \tilde{z} \odot \mathbf{W}_\alpha \tilde{z} \odot \mathbf{W}_\alpha \tilde{z}\big] \tag{37}$$

$$= \mathbb{E}\big[\sum_i \sum_j \sum_k \sum_l [(\mathbf{w}_{\mu,i}\tilde{z}_i) \odot (\mathbf{w}_{\mu,j}\tilde{z}_j)] \odot$$

$$[(\mathbf{w}_{\alpha,k}\tilde{z}_k) \odot (\mathbf{w}_{\alpha,l}\tilde{z}_l)]\big] \tag{38}$$

$$= \sum_i \sum_j \sum_k \sum_j \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,k} \odot \mathbf{w}_{\alpha,l} \mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_k \tilde{z}_l] \tag{39}$$

Consider the case where $i = j = k = l$,

$$\begin{aligned}&\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[\tilde{z}_i \tilde{z}_i \tilde{z}_i \tilde{z}_i]\\ &= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^4]\end{aligned} \tag{40}$$

And consider the case where $i = j \neq k = l$

$$\sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[\tilde{z}_i \tilde{z}_i \tilde{z}_j \tilde{z}_j] \tag{41}$$

$$= \sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_i)^2 (\tilde{z}_j)^2] \tag{42}$$

$$= \sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_i)^2] \mathbb{E}[(\tilde{z}_j)^2] \tag{43}$$

$$= \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \mathbb{E}[(\tilde{z}_i)^2] \sum_{j,j\neq i} (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_j)^2] \tag{44}$$

$$= \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \mathbb{E}[(\tilde{z}_i)^2] \left(\sum_j (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_j)^2] - (\mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i}) \mathbb{E}[(\tilde{z}_i)^2] \right) \tag{45}$$

$$= \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \mathbb{E}[(\tilde{z}_i)^2] \sum_j (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_j)^2] - \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i}) \mathbb{E}[(\tilde{z}_i)^2] (\mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i}) \mathbb{E}[(\tilde{z}_i)^2] \tag{46}$$

This holds because we know $i \neq k$. Similarly, when $i = k \neq j = l$,

$$\sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j}) \odot (\mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_i \tilde{z}_j] \tag{47}$$

$$= \sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j}) \odot (\mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_i)^2 (\tilde{z}_j)^2] \tag{48}$$

$$= \sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j}) \odot (\mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,j}) \mathbb{E}[(\tilde{z}_i)^2] \mathbb{E}[(\tilde{z}_j)^2] \tag{49}$$

$$= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_{j,j\neq i} \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \tag{50}$$

$$= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] - \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \tag{51}$$

When $i = l \neq k = j$,

$$\sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j}) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,i}) \mathbb{E}[\tilde{z}_i \tilde{z}_j \tilde{z}_j \tilde{z}_i] \tag{52}$$

$$= \sum_i \sum_j (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j}) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,i}) \mathbb{E}[(\tilde{z}_i)^2 (\tilde{z}_j)^2] \tag{53}$$

$$= \sum_i \sum_{j,j\neq i} (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,j}) \odot (\mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,i}) \mathbb{E}[(\tilde{z}_i)^2] \mathbb{E}[(\tilde{z}_j)^2] \tag{54}$$

$$= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_{j,j\neq i} \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \tag{55}$$

$$= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \left(\sum_j \mathbf{w}_{\mu,j}^T \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] - \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \right) \tag{56}$$

$$= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] - \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \tag{57}$$

(58)

Since these four cases are mutually exclusive, we could rewrite the full term as:

$$\begin{aligned}
&\mathbb{E}\big[\Delta u_1 \, \Delta u_2 \, \Delta v_1 \, \Delta v_2\big] \\
&= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^4] \\
&\quad + \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\alpha,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \\
&\quad - \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \\
&\quad + 2 \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \\
&\quad - 2 \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2]) (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2]),
\end{aligned} \tag{59}$$

$$\begin{aligned}
&= \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^4] \\
&\quad + \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_j)^2] \\
&\quad + 2 \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \\
&\quad - 3 \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2]) (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2]).
\end{aligned} \tag{60}$$

Putting this back to Equation (35), we can write the final expression as:

$$\begin{aligned}
&\text{Cov}(u_1u_2, v_1v_2) = \text{Cov}(||\mathbf{W}_\mu z||_2^2, ||\mathbf{W}_\alpha z||_2^2) \\
&= 4(\mathbf{W}_\mu \odot \mathbf{W}_\mu \odot \mathbf{W}_\alpha \odot \mathbf{W}_\alpha (\mathbb{E}[\boldsymbol{z}])^2 \mathbb{E}[(\tilde{\boldsymbol{z}})^2] \\
&\quad + \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^4] \\
&\quad + 2 \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2] \sum_j \mathbf{w}_{\mu,j} \odot \mathbf{w}_{\alpha,j} \mathbb{E}[(\tilde{z}_j)^2] \\
&\quad - 3 \sum_i (\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2])(\mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^2]) \\
&\quad + 2\mathbf{W}_\mu \mathbb{E}[z] \odot \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^3] \\
&\quad + 2\mathbf{W}_\alpha \mathbb{E}[z] \odot \sum_i \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\mu,i} \odot \mathbf{w}_{\alpha,i} \mathbb{E}[(\tilde{z}_i)^3]. \quad (61)
\end{aligned}$$

Therefore, we show that the covariance terms $\text{Cov}(\mathbf{W}_\mu \boldsymbol{z}, ||\mathbf{W}_\alpha \boldsymbol{z}||^2)$ (cf. Eq. 34) and $\text{Cov}(||\mathbf{W}_\mu \boldsymbol{z}||_2^2, ||\mathbf{W}_\alpha \boldsymbol{z}||_2^2)$ (cf. Eq. 61) can be expressed as a linear combination of the first four central moments of the components z_i. The coefficients of this linear combination are polynomials in the entries of the weight matrices $\mathbf{W}_\mu$ and $\mathbf{W}_\alpha$.

B Experiment Details

B.1 Uniform Dequantization

Our model's decoder defines a likelihood over continuous values using a Gaussian distribution. However, the image datasets we use, such as MNIST, consist of discrete pixels values. To bridge this gap, we employ uniform dequantization. This standard technique adds a small amount of uniform noise to each discrete pixel value, transforming data into a continuous variable that is compatible with our model's likelihood function.

Specifically, for discrete data $\boldsymbol{x}_{\text{int}}$ with values in $\{0, 1, \ldots, 255\}$, the dequantized data is defined as $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{u}$, where $\boldsymbol{x} = \frac{\boldsymbol{x}_{\text{int}}}{256}$, $\boldsymbol{u} \sim \mathcal{U}[0, \frac{1}{256})$. This process maps each discrete pixel value to a unique continuous bin of width $\frac{1}{256}$ within $[0, 1)$.

The true probability of a discrete pixel value under our continuous model is thus defined as:

$$P_{\text{model}}(X = \boldsymbol{x}_{\text{int}}) = \int_{\boldsymbol{x}_{\text{int}}/256}^{(\boldsymbol{x}_{\text{int}}+1)/256} p_{\text{model}}(\boldsymbol{y}) d\boldsymbol{y} \quad (62)$$

By applying Jensen's inequality, we can establish a formal relationship:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{u}}[\log p_{\text{model}}(\boldsymbol{y})] &\leq \log\left(\mathbb{E}_{\boldsymbol{u}}[p_{\text{model}}(\boldsymbol{y})]\right), \quad (63) \\
&= \log P_{\text{model}}(X = \boldsymbol{x}_{\text{int}}) - \log(256). \quad (64)
\end{aligned}$$

Rearranging this gives us a lower bound on the discrete log-likelihood:

$$\log P_{\text{model}}(X = \boldsymbol{x}_{\text{int}}) \geq \mathbb{E}_{\boldsymbol{u}}[\log p_{\text{model}}(\boldsymbol{y})] + \log(256). \quad (65)$$

Therefore, to ensure we are optimizing a valid lower bound on the true log-likelihood of the discrete data, we apply a correction to the pixel-wise reconstruction log-likelihood by adding a constant $\log(256)$ to it.

B.2 Model Architecture

In this section, we detail the model architecture used in the experiments in section 3 and 4.

Fixed Variance Experiment In section 3, we conduct a controlled experiment with a fixed output variance, the VAE consists of a convolutional encoder and a simple linear decoder. The encoder architecture, which is shared across both continuous and discrete latent space models, is detailed in Table 5. The decoder is a single fully-connected linear layer that maps the latent variable $\boldsymbol{z}$ directly to the flattened output image pixels. Equivalent to a learnable bias, we augment the latent vector $\boldsymbol{z}$ by concatenating it with an additional dimension fixed at a constant value of 1.

Table 5: Encoder architecture for the fixed and learnable variance experiments on MNIST.

Layer	Kernel Size	Stride	Padding	Activation
Conv2d	3x3	1	1	ReLU
Conv2d	3x3	1	1	ReLU
Conv2d	3x3	1	1	-
Flatten	-	-	-	-
Linear	-	-	-	-

Learnable Variance Experiment

MNIST. The encoder for the MNIST experiments is consistent with fixed-variance experiment, as presented in Table 5. The nonlinear decoder mirrors this structure with a linear layer followed by several convolutional layers. The architecture is detailed in Table 6. The linear decoder is a single fully-connected layer without any activations.

Table 6: Nonlinear Decoder architecture for the learnable variance experiments on MNIST.

Layer	Kernel Size	Stride	Padding	Activation
Linear	-	-	-	-
Reshape	-	-	-	-
Conv2d	3x3	1	1	ReLU
Conv2d	3x3	1	1	ReLU
Conv2d	3x3	1	1	ReLU
Conv2d	3x3	1	1	ReLU
Conv2d	1x1	1	0	-

ImageNet Architecture and CIFAR-10 For the more complex CIFAR-10 and ImageNet datasets, we use deeper, strided convolutional architectures for both encoder and the nonlinear decoder, with batch normalization after each convolutional layers. The linear decoder remains a single fully-connected layers. The architectures are detailed in Table 7 and Table 8.

Table 7: Encoder architecture for the learnable variance experiments on CIFAR-10 and ImageNet.

Layer	Kernel Size	Stride	Padding	Activation
Conv2d	4x4	2	1	ReLU
BatchNorm2d	-	-	-	-
Conv2d	4x4	2	1	ReLU
BatchNorm2d	-	-	-	-
Conv2d	4x4	2	1	ReLU
BatchNorm2d	-	-	-	-
Conv2d	4x4	1	0	ReLU
BatchNorm2d	-	-	-	-
Flatten	-	-	-	-
Linear	-	-	-	-

Table 8: Nonlinear Decoder architecture for the learnable variance experiments on CIFAR-10 and ImageNet.

Layer	Kernel Size	Stride	Padding	Activation
Linear	-	-	-	-
Reshape	-	-	-	-
ConvTranspose2d	4x4	1	0	ReLU
BatchNorm2d	-	-	-	-
ConvTranspose2d	4x4	2	1	ReLU
BatchNorm2d	-	-	-	-
ConvTranspose2d	4x4	2	1	ReLU
BatchNorm2d	-	-	-	-
ConvTranspose2d	4x4	2	1	-

B.3 Training Details

Data Preprocessing For all experiments, the input images are first transformed into PyTorch tensors. Before being passed to the model, the image data is scaled by $\frac{255}{256}$, in preparation for uniform dequantization.

Baselines For our baseline models used throughout the following experiments, we use standard implementations for the Gumbel-Softmax and the reparameterization trick in VAEs. For the REINFORCE, we implement a baseline to reduce variance. Specifically, we use the running average of the reconstruction loss.

Fixed Variance Experiment All models in this experiment are trained using the AdamW optimizer with betas set to $(0.9, 0.95)$. We performed a hyperparameter search for the learning rate over the values $\{1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-5}, 5 \times 10^{-5}\}$. For each model, the best performing rate was selected based on the final BPD score, which was evaluated on the validation set at the end of Epoch 500. All reported metrics in Section 3.2 are likewise evaluated on the validation set at this same epoch.

The output variance of the decoder was fixed for these experimented. We tested σ^2 values of $\{0.1, 0.05, 0.01\}$ and found that a fixed variance $\sigma^2 = 0.01$ yielded the best results across all models. The models are trained with a batch size of 64, and no gradient clipping was applied. Additionally, we did not use KL annealing; the β parameter for KLD is fixed at 1.0 throughout training.

Gradient Variance Calculation The gradient variance with respect to the encoder parameters reported in Section 2 is measured empirically. To isolate the variance only from the latent variable sampling, we first perform a single forward pass through the encoder on a fixed batch of data to obtain the parameters of the latent distribution, which is to avoid the randomness introduced by the uniform noise we add for dequantization. With these parameters held constant, we then draw 100 latent samples from this fixed distribution. For each sample, we compute the corresponding reconstruction loss and backpropagate to get a gradient vector with respect to the encoder's parameters. The total gradient variance is computed by first calculating the variance for each individual parameter in the encoder across the 100 gradient samples. These per-parameter variances are then summed together to produce the final scalar value that is reported in Section 2.

Learnable Variance Experiment The training scheme is similar to the fixed variance experiment. All models are trained using AdamW optimizer(s) with betas of $(0.9, 0.95)$, and we selected the best learning rate for each baseline from the set $\{1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-5}, 5 \times 10^{-5}\}$. For our combined methods that integrate Silent Gradients, we built upon the best baseline learning rates and introduced separate optimizers for the linear decoder's μ and α components. We perform a hyperparameter search for the annealing rate from $\{1 \times 10^{-2}, 5 \times 10^{-2}, 1 \times 10^{-3}, 5 \times 10^{-3}\}$, and the encoder freeze epoch (cut-off) from $\{50, 80, 100, 150, 200\}$.

The training duration and batch sizes varied by dataset:

MNIST. Models are trained with a batch size of 64. The REINFORCE models are trained for 300 epochs, while all others are trained for 200 epochs.

ImageNet. Models are trained for 100 epochs with a batch size of 128.

CIFAR-10. Models are trained for 2000 epochs with a batch size of 256.

At the end of training, all models are guaranteed to be converged. And like the fixed variance experiment, no KL annealing or β parameter for KLD other than 1.0 is used.

Computing Resources All experiments included were conducted on 8 NVIDIA GeForce RTX 4090 GPUs.

C Visualized Output

In this section, we provide the visualization of reconstructed images using the trained model at the epoch where the metrics are reported.

The visualizations are generated by taking a fixed batch of images from the validation set of each respective dataset. These images are passed through the train encoder to obtain a latent variable $\boldsymbol{z}$, which is then passed to the corresponding decoder to produce the reconstruction. For the learnable variable experiment in which a dual-decoder setting is introduced, only nonlinear decoder is used to generate the reconstruction. The linear decoder used for computing the Silent Gradient, in this case, is not used.

C.1 Fixed Variance Experiment

The visualizations show the original images and the corresponding reconstructed mean, as in Figure 2.

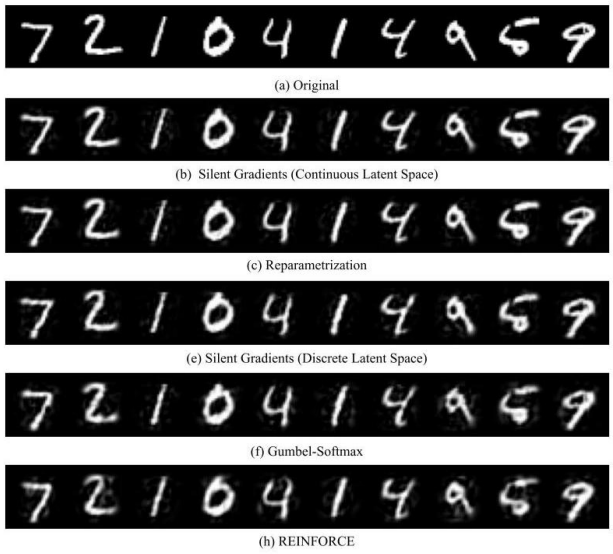

Figure 2: Visual comparison of reconstructions for the fixed variance experiment on MNIST. The top row (a) displays original images from the validation set. Subsequent rows show the reconstructed means from our Silent Gradients method and the baseline estimators for both continuous and discrete latent spaces.

C.2 Learnable Variance Experiment

In addition to the reconstructed mean, the visualizations include an additional row displaying the learned standard deviation for each pixel. For visualization purposes, the standard deviation is normalized to the range $[0, 1]$ to be displayed as an image, as shown in Figure 3, Figure 4, and Figure 5.

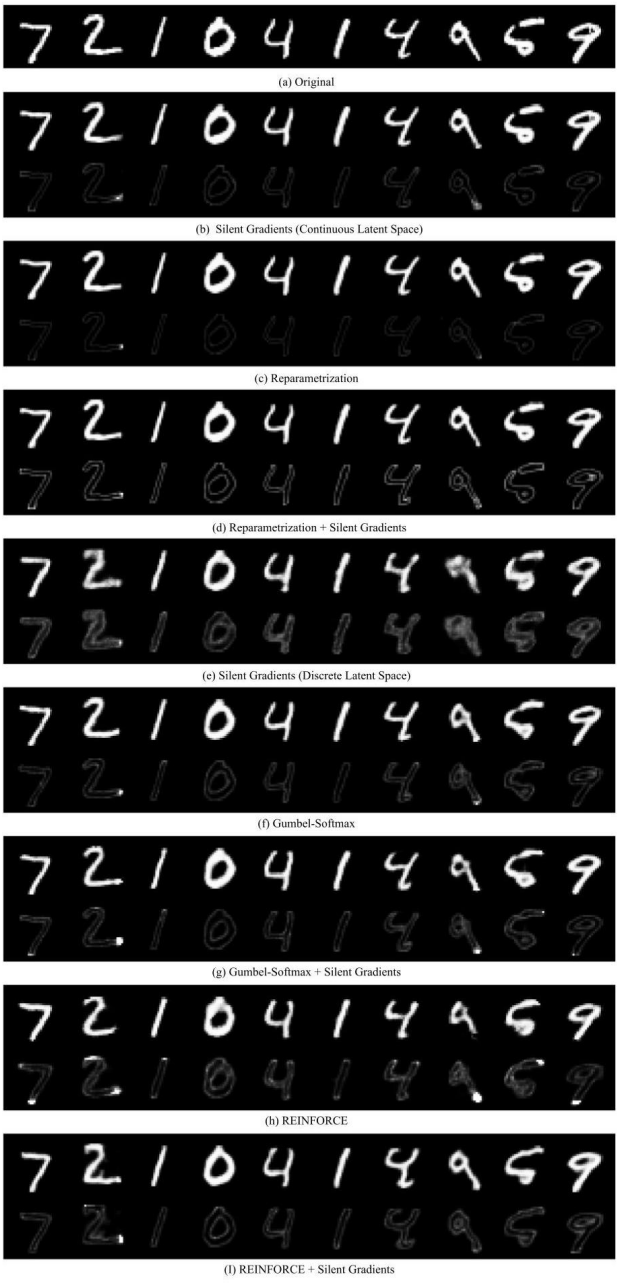

Figure 3: Reconstructions on the MNIST dataset in learnable variance experiment. The images are the output of the nonlinear decoder.

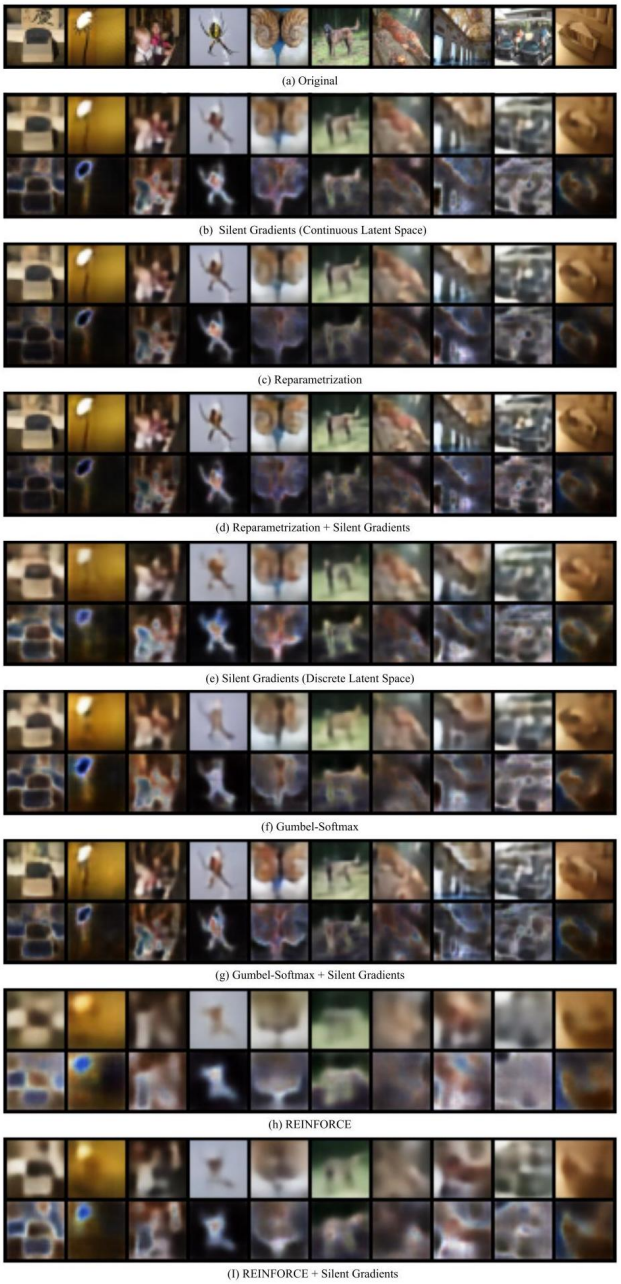

Figure 4: Reconstructions on the ImageNet dataset in learnable variance experiment.

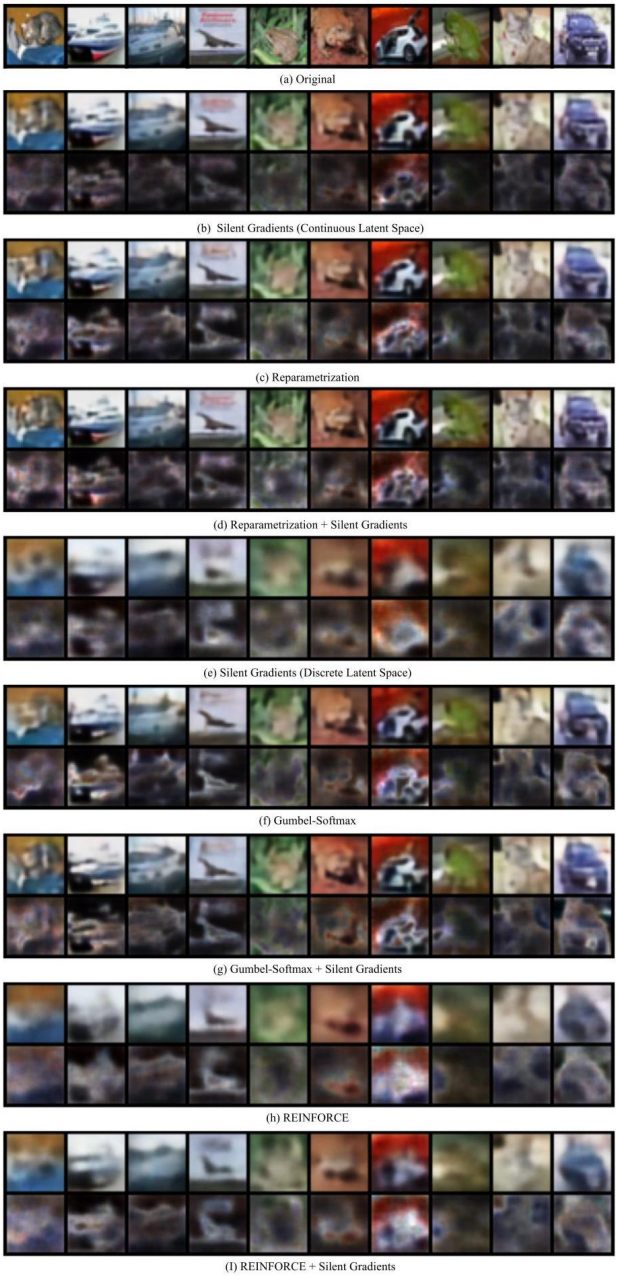

Figure 5: Reconstructions on the CIFAR-10 dataset in learnable variance experiment.