Density-Informed VAE (DiVAE): Reliable Log-Prior Probability via Density Alignment Regularization

Michele Alessi[1,2], **Alessio Ansuini**[2], **Alex Rodríguez**[1]

[1]Department of Mathematics, Informatics and Geosciences, University of Trieste, Italy
[2]Laboratory of Data Engineering, Area Science Park, Trieste, Italy

michele.alessi@phd.units.it
alessio.ansuini@areasciencepark.it
alejandro.rodriguezgarcia@units.it

Abstract

We introduce **Density-Informed VAE (DiVAE)**, a lightweight, data-driven regularizer that aligns the VAE log-prior probability $\log p_Z(z)$ with a log-density estimated from data. Standard VAEs match latents to a simple prior, overlooking density structure in the data-space. DiVAE encourages the encoder to allocate posterior mass in proportion to data-space density and, when the prior is learnable, nudges the prior toward high-density regions. This is realized by adding a robust, precision-weighted penalty to the ELBO, incurring negligible computational overhead. On synthetic datasets, DiVAE (i) improves distributional alignment of latent log-densities to its ground truth counterpart, (ii) improves prior coverage, and (iii) yields better OOD uncertainty calibration. On MNIST, DiVAE improves alignment of the prior with external estimates of the density, providing better interpretability, and improves OOD detection for learnable priors.

1 Introduction

Variational autoencoders (VAEs) typically assume a simple prior p_Z (e.g., $\mathcal{N}(0, I)$) [1]. This choice encourages the variational posterior $q_\phi(z \mid x)$ to place latent codes near the origin and to distribute mass according to the prior rather than the data's *empirical density*. As a result, the log-prior probability $s(z) = \log p_Z(z)$ may be poorly aligned with how samples concentrate in data space, reducing its usefulness for anomaly detection, out-of-distribution (OOD) scoring, and uncertainty estimation. It also contributes to well-known issues such as the *hole problem* –regions of high prior mass unsupported by the theoretically optimal prior: the aggregated variational posterior [2, 3].

To address the limitations of a simple, fixed prior described above, we introduce **Density-Informed VAE (DiVAE)**, which injects a per-sample density signal ρ –an estimate of log-density measured in a data-derived projection– into training via a regularization term. Acting as a *density teacher*, DiVAE adds an alignment penalty that pushes the encoder to place latent codes in proportion to ρ. Consequently, even with a non-learnable prior (e.g., $\mathcal{N}(0, I)$), the latent space recovers the data-space density structure, improving the usefulness of $s(z)$ without changing the prior itself. For a learnable prior (e.g., a Gaussian mixture model, GMM), the same density signal simultaneously (i) guides the encoder toward regions whose mass reflects the estimate and (ii) pulls the prior parameters toward those high-density regions, thereby improving agreement between prior mass and empirical structure. Concretely, we augment the ELBO with a robust, precision-weighted alignment loss (Section 2)

EurIPS 2025 Workshop on Principles of Generative Modeling (PriGM).

that incurs negligible computational overhead (Table 12) and requires no changes to the encoder or decoder architectures.

We implement this alignment in two forms: a *direct* variant that matches the model's latent log-prior to the external log-density estimate, and a *flow-corrected* variant that learns a normalizing flow to account for the change-of-variables (Jacobian) between latent and data-derived projections.

We show that on synthetic datasets (in which we know the ground truth), DiVAE (Sec. 3): (i) improves distributional alignment of latent log-densities to the ground truth log-density, shifting the mean toward the true scale; (ii) improves prior coverage, and (iii) yields better OOD uncertainty calibration—stronger separation, higher posterior entropy. These gains hold under both standard-normal and learnable GMM priors; FLOW attains the strongest separation, whereas DIRECT provides the best calibration trade-off. In MNIST, where the ground truth is unknown, we find stable improvements in alignment between several types of prior and an external estimate of the density, with latent codes that are significantly most typical under the learned prior (see Table 2); for the OOD, when the prior is learnable, there are consistent improvements in adopting a density regularizer (4) .

Related Work A major line of work improves VAE priors to better approximate the (theoretically optimal) aggregated posterior. Building on the observation that the aggregate is typically a gaussian mixture [2], VampPrior introduces learnable pseudo-inputs so that the prior becomes a mixture of variational posteriors that explicitly targets this distribution [4]. Flow– and autoregressive–based priors further increase flexibility and improve likelihoods/compression [5, 6, 7, 8, 9].

In this work, rather than only enriching the prior, we directly align the density around data points –measured in the PCA-reduced space– with the latent density, enforcing local agreement between empirical structure and prior mass.

2 Methods

Setup Let $x \in \mathcal{X}$ be an input, $q_\phi(z \mid x) = \mathcal{N}\big(\mu_\phi(x), \mathrm{diag}(\sigma^2_\phi(x))\big)$ is the encoder, $p_\theta(x \mid z)$ the decoder, and $p_Z(z) = p_Z(z;\psi)$ a prior (fixed or learnable) with parameters ψ (for a standard Gaussian prior, $\psi = \varnothing$).

We apply a dimensionality-reduction step used *only* for density estimation and for comparing densities between data and latent spaces. Specifically, we project the data from the original space $\mathcal{X}$ onto the linear subspace $\mathcal{U} \subseteq \mathcal{X}$ spanned by the top d principal components, where $d = \dim(\mathcal{Z})$ matches the latent dimensionality. Let $P : \mathcal{X} \to \mathcal{U}$ denote the PCA projector; for each training point x_i, we write $u_i = P\,x_i$ for its projection.

We assume that we have access to a *external* (i.e., model-independent) log-density estimate ρ_i in the $\mathcal{U}$ space of the PCA projected data, with a standard error σ_i expressing the estimator's uncertainty at u_i. In practice, this estimate will be performed with standard techniques such as Kernel-Density Estimation (KDE), or Point Adaptive k-Nearest Neighbors (PAk) [10, 11].

We define the *log-prior probability* for x_i, expressing the log probability assigned by the prior to the encoding of a data point x_i, as $s_i = \mathbb{E}_{z\sim q_\phi(\cdot|x_i)}\big[\log p_Z(z)\big]$, and estimate it with Monte Carlo sampling, where in practice we use one-sample estimates. The central idea of our work is to nudge the log-prior probabilities $\{s_i\}$ towards $\{\rho_i\}$ using a penalty, which we call the *alignment loss*. This loss comes in two flavours, depending on the aligner type: 1) a direct aligner and 2) a flow-corrected aligner.

Direct aligner The direct aligner operates with a robust, precision-weighted loss term that acts as a regularizer, and compares the model's prior log-densities $\{s_i\}$ to the external log-densities $\{\rho_i\}$. We define precision weights from the estimator's uncertainties $w_i = \sigma_i^{-2}$.

Given a mini–batch $\{x_i\}_{i=1}^B$ the direct alignment loss is written as:

$$\mathcal{L}_{\text{direct-align}} = \sqrt{\frac{1}{B}\sum_{i=1}^{B} w_i\,\Gamma_\delta\big(s_i - \rho_i\big)}, \tag{1}$$

		k=4		k=8		k=12	
Prior	Method	KS↓	KL↓	KS↓	KL↓	KS↓	KL↓
GMM	NONE	0.469	114.184	0.501	20.877	0.380	**3.728**
GMM	DIRECT	**0.102**	69.620	**0.073**	6.447	**0.124**	6.823
GMM	FLOW	0.946	**4.419**	0.753	**5.334**	0.934	4.582
STANDARD	NONE	**0.116**	-	0.322	-	0.418	-
STANDARD	DIRECT	0.228	-	**0.231**	-	0.384	-
STANDARD	FLOW	0.338	-	0.315	-	**0.255**	-

Table 1: KS statistic and coverage divergence $\mathrm{KL}(p_2 \,\|\, p_Z(\cdot;\psi))$ on synthetic Gaussian-mixture datasets with k=4, 8, 12 components. Means over three seeds. External densities are estimated using PAk. KL is omitted for the standard-normal prior since p_Z is fixed and identical across methods.

where Γ_δ is the Huber loss. The Huber loss [12] behaves quadratically within a range $\delta > 0$, and linearly outside this range, to mitigate the influence of outliers (see D). In our experiments we set $\delta = 1$, corresponding to $\approx \frac{1}{2}$ probability for the residual to fall in the linear regime.

Training objective We optimize the (negative) ELBO plus the alignment term $\mathcal{L} = -\mathcal{L}_{\text{ELBO}} + \gamma_t \mathcal{L}_{\text{direct-align}}$, with a warm-up weight γ_t, practically chosen as a linear warm-up from 0 to a 1 over training epochs. Unless stated otherwise, we *do not* detach encoder gradients in (1); this allows the regularizer to shape the variational posterior q_ϕ.

Flow-corrected aligner The direct aligner supervises the log-prior probability $s(z) = \log p_Z(z)$ but, by itself, cannot account for the change of variables between the latent space $\mathcal{Z}$ and $\mathcal{U}$, i.e., it omits the log-determinant Jacobian term. We therefore learn an invertible map $f : \mathcal{Z} \to \mathcal{U}$, parameterized as a normalizing flow [9]. The change of variables formula implies $\log p_U(u) = \log p_Z\big(f^{-1}(u)\big) + \log \big|\det J_{f^{-1}}(u)\big|$. We train f by maximum likelihood on observed projections u_i:

$$\mathcal{L}_{\text{flow-ML}} = -\frac{1}{B}\sum_{i=1}^{B}\Big[\log p_Z\big(f^{-1}(u_i)\big) + \log\big|\det J_{f^{-1}}(u_i)\big|\Big]. \tag{2}$$

For alignment in $\mathcal{Z}$-space, the ideal target would be $s(z) = \log p_U\big(f(z)\big) + \log\big|\det J_f(z)\big|$. Since p_U is unknown, we substitute the external density estimate at the corresponding projected point. Assuming local coherence $f(z_i) \approx u_i$, we obtain the practical approximation $s_i \approx \rho_i + \log\big|\det J_f(z_i)\big|$. The flow-corrected alignment loss then reads

$$\mathcal{L}_{\text{flow-align}} = \sqrt{\frac{1}{B}\sum_{i=1}^{B} w_i\,\Gamma_\delta\Big(s_i - \rho_i - \underbrace{\log\big|\det J_f(z_i)\big|}_{\text{stop-grad w.r.t. } f}\Big)} \tag{3}$$

where the log-determinant term provides the correct Jacobian correction, and we stop its gradient into f during alignment (so the encoder/prior is driven by the density signal while the flow is learned by (2)). The overall objective becomes $\mathcal{L}_{\text{total}} = -\mathcal{L}_{\text{ELBO}} + \gamma_t \mathcal{L}_{\text{flow-align}} + \mathcal{L}_{\text{flow-ML}}$ optimized with two parameter blocks: (ϕ, ψ, θ) for the VAE and flow parameters for f.

3 Results

We evaluate DiVAE on three synthetic datasets under both a fixed standard-normal prior and a learnable GMM prior, using several external density estimators (details in Appendix A and B). For each configuration we compare a non-regularized baseline (NONE) to the proposed alignment variants (DIRECT, FLOW). We report: (i) the mean prior log-density $\bar{s}$; (ii) distributional alignment between model and external or ground-truth log densities using the two-sample Kolmogorov–Smirnov (KS) test [13] and the 1D Wasserstein distance (W) [14]; (iii) coverage between the true mixture and the learned prior via $\mathrm{KL}(\text{true} \,\|\, \text{prior})$; and (iv) the mean stochastic ELBO $\overline{\mathcal{L}}$. For notation clarity, we summarize all quantities and definitions in Tab. 5. The reasoning behind our selection of metrics is discussed in Section C.

Prior	Method	$\bar{\ell} \uparrow$	$\bar{s} \uparrow$	KS $\downarrow$	W $\downarrow$
STANDARD	NONE	**-101.214 ± 29.087**	-14.209 ± 1.750	**0.1102**	0.8374
STANDARD	DIRECT	-101.517 ± 28.980	**-13.958 ± 1.629**	0.1133	**0.6475**
STANDARD	FLOW	-101.585 ± 28.660	-16.102 ± 2.053	0.4540	2.7300
GMM	NONE	-100.087 ± 29.314	-20.633 ± 2.510	0.9008	7.2613
GMM	DIRECT	**-99.793 ± 29.419**	**-13.568 ± 3.216**	**0.0815**	**0.3992**
GMM	FLOW	-101.309 ± 29.402	-41.539 ± 2.024	1.0000	28.1673
VAMP	NONE	**-99.548 ± 29.393**	-20.833 ± 2.875	0.8873	7.4612
VAMP	DIRECT	-99.895 ± 29.503	**-13.510 ± 3.167**	**0.0698**	**0.3460**
VAMP	FLOW	-102.175 ± 30.056	-56.126 ± 1.757	1.0000	42.7541

Table 2: MNIST validation results across priors and regularization methods. Values are averaged over three seeds. The external PAk estimate has mean $\bar{s}^{\text{ext}} = -13.372 \pm 2.711$.

Prior	Method	$\bar{s}\downarrow$	$KL(q, p_2)\uparrow$	$\mathcal{H}(q)\uparrow$
GMM	NONE	-2.378	12.975	-5.785
GMM	DIRECT	-3.365	12.477	-5.216
GMM	FLOW	**-7.397**	**16.567**	**-2.654**
STANDARD	NONE	-3.428	18.247	-5.474
STANDARD	DIRECT	-3.236	18.581	-5.645
STANDARD	FLOW	**-4.850**	**19.683**	**-5.326**

Table 3: Out-of-distribution experiment on synthetic data. Models are trained on a k=4 GMM and evaluated on an unseen k=8 GMM (dim=50). External densities for alignment are estimated using PAk.

We also evaluate DiVAE on the MNIST dataset [15], where ground-truth log-densities are not available. External densities are therefore estimated with the PAk method [11]. We again compare NONE, DIRECT, and FLOW across three priors: standard normal, learnable GMM, and VampPrior. For each model we report: (i) the mean prior log-density $\bar{s}$, interpreted as latent log-likelihood; (ii) the ELBO $\bar{\ell}$; and (iii) alignment with the PAk log-densities via KS and W.

Table 1 summarizes results on the synthetic datasets, which consist of high-dimensional Gaussian mixtures with k=4, 8, 12 components (Appendix A). DIRECT consistently improves KS and coverage KL under a learnable GMM prior compared to NONE, whereas FLOW achieves the lowest KL but with substantially higher KS, indicating over-correction and shape mismatch. Under a fixed standard prior, DIRECT generally improves KS, while FLOW is beneficial primarily for more complex mixtures (k=12). Complete numerical results are provided in Tables 6–11.

Table 2 reports MNIST results. Across all priors, DIRECT improves alignment with external PAk densities (lower KS and W) while preserving or slightly improving the ELBO and $\bar{s}$ relative to NONE. The FLOW variant often deteriorates both KS and W, indicating that its stronger transformations can misalign the latent structure in the absence of ground truth. Overall, DIRECT provides the most stable improvements across both synthetic and real data regimes.

Out-of-distribution (OOD) We first assess OOD sensitivity on synthetic data by treating a k=8 GMM as OOD for models trained on k=4 (Table 3). We report: (i) the mean prior log-density $\bar{s}$, which decreases when the model assigns lower latent likelihood to OOD samples; (ii) the posterior–true prior divergence $\mathrm{KL}(q, p_2)$, capturing latent mismatch under OOD inputs; and (iii) the posterior entropy $\mathcal{H}(q)$, reflecting uncertainty. Both DIRECT and FLOW improve OOD separation over NONE (lower $\bar{s}$). FLOW yields the strongest separation but also shows substantially larger $\mathrm{KL}(q, p_2)$ and higher entropy, indicating aggressive rejection and over-correction. DIRECT provides a more balanced trade-off: improved separation with moderate increases in divergence and competitive entropy.

We further evaluate OOD detection on real data by training models on MNIST and testing on FashionMNIST [16] (Table 4). Here we report shifts between OOD and in-distribution metrics: ELBO ($\Delta\bar{\ell}$), prior log-density ($\Delta\bar{s}$), latent prior mismatch shift ($\Delta\mathrm{KL}$), and posterior entropy shift

Prior	Method	$\Delta\bar{\ell} \downarrow$	$\Delta\bar{s} \downarrow$	$\Delta\mathrm{KL} \uparrow$	$\Delta\mathcal{H} \uparrow$
STANDARD	NONE	**-1058.67**	0.504	**3.165**	-2.810
STANDARD	DIRECT	-1025.35	**0.016**	1.554	**-1.577**
STANDARD	FLOW	-1033.58	0.402	2.556	-3.169
GMM	NONE	-1073.48	-2.761	2.945	**0.450**
GMM	DIRECT	-996.18	**-3.608**	3.476	0.091
GMM	FLOW	**-1133.04**	0.400	**3.615**	-4.202
VAMP	NONE	-1063.15	**-3.065**	2.055	0.725
VAMP	DIRECT	-1071.64	-2.790	0.916	**1.686**
VAMP	FLOW	**-1239.54**	0.730	**10.138**	-9.706

Table 4: OOD results on MNIST. Models are trained on MNIST and evaluated on FashionMNIST. External densities for alignment are estimated using PAk.

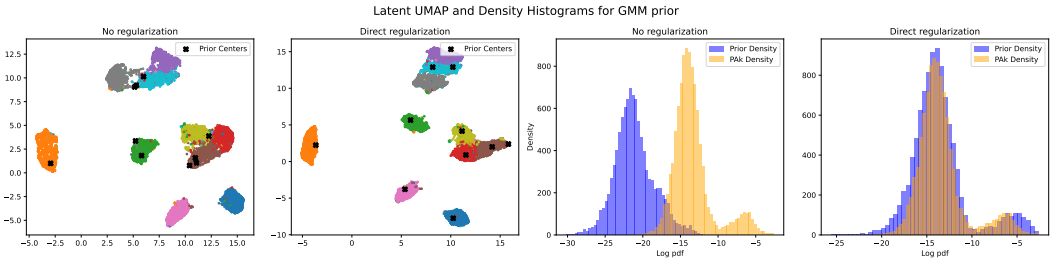

Figure 1: UMAP representation and density alignments on the MNIST latent space. The regularized model places the prior components directly onto the data clusters (as evidenced by the coloring by ground-truth labels), whereas the unregularized model tends to collapse several prior centers onto the same region of the latent space. Moreover, the regularizer enforces a strong alignment between the learned log prior density and an external, non–black-box density proxy, providing a clear and interpretable correspondence between the latent geometry and the data distribution.

($\Delta\mathcal{H}$). Across all priors, DIRECT yields more stable and calibrated OOD shifts, with moderate ELBO and $\bar{s}$ reductions and controlled increases in $\Delta\mathrm{KL}$ and $\Delta\mathcal{H}$. In contrast, FLOW produces the largest separation (e.g. most negative $\Delta\bar{\ell}$ and largest $\Delta\mathrm{KL}$ for GMM and VampPrior) but often at the cost of excessive mismatch and inflated entropy. Overall, DIRECT consistently improves OOD robustness especially on learnable priors, without the over-rejection effects seen in the stronger FLOW transformation.

4 Conclusion

We introduced *Density-Informed VAE (DiVAE)*, a lightweight, data-driven regularizer that aligns the VAE prior log-density with a log-density estimate derived from data via suitable aligners. On synthetic benchmarks with known ground truth, DiVAE improves prior log-density calibration under both standard-normal and GMM priors with negligible overhead. On MNIST, where the ground truth is unknown, we find similar benefits when we adopt a learnable prior (Figure 1). These findings, although preliminary, suggest that explicitly coupling the prior to data-density structure can be a simple and effective route for developing new generative models or improving existing ones. Future directions include testing in real-world datasets and a deeper analysis of the promising flow approach.

Limitations

The results we report are limited to simple datasets and small architectures: scaling to larger models and more challenging datasets remains to be demonstrated.

Acknowledgements

The authors acknowledge the AREA Science Park supercomputing platform ORFEO made available for conducting the research reported in this paper, and the technical support of the Laboratory of Data Engineering staff. Alessio Ansuini was supported by the project "Supporto alla diagnosi di malattie rare tramite l'intelligenza artificiale" CUP: F53C22001770002, from the project "Valutazione automatica delle immagini diagnostiche tramite l'intelligenza artificiale", CUP: F53C22001780002, and by the European Union – NextGenerationEU within the project PNRR "PRP@CERIC" IR0000028 - Mission 4 Component 2 Investment 3.1 Action 3.1.1.

References

[1] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[2] Jakub M Tomczak. *Deep Generative Modeling*. Springer Cham, 2024.

[3] Danilo Jimenez Rezende and Fabio Viola. Taming vaes, 2018.

[4] Jakub Tomczak and Max Welling. Vae with a vampprior. In *International conference on artificial intelligence and statistics*, pages 1214–1223. PMLR, 2018.

[5] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, pages 1530–1538, 2015.

[6] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

[7] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

[8] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. Pixelvae: A latent variable model for natural images. In *International Conference on Learning Representations (ICLR)*, 2017.

[9] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

[10] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[11] Alex Rodriguez, Maria d'Errico, Elena Facco, and Alessandro Laio. Computing the free energy without collective variables. *Journal of Chemical Theory and Computation*, 14(3):1206–1215, 2018. PMID: 29401379.

[12] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pages 492–518. Springer, 1992.

[13] Frank J. Massey. The kolmogorov–smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951.

[14] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2008.

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

[16] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. arXiv:1412.6980.

[18] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. arXiv:1602.02282.

[19] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[20] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *International Conference on Learning Representations (ICLR)*, 2017.

[21] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

A Datasets

From 2D GMM to D-dimensional data via padding and rotation

Setup. Let $u \in \mathbb{R}^2$ be drawn from a 2D mixture density $p_2(u)$ (the ground-truth Gaussian mixture in the main text). For a target ambient dimension $D \geq 2$, set $m = D - 2$ and define

$$z \;=\; \begin{bmatrix} u \\ w \end{bmatrix} \in \mathbb{R}^D, \qquad w \sim \mathcal{N}\big(0,\, \sigma_{\text{pad}}^2 I_m\big).$$

Let $R \in \mathbb{R}^{D\times D}$ be an orthogonal matrix ($R^\top R = I$) with $\det R = +1$, and set

$$x \;=\; Rz, \qquad z \;=\; R^\top x.$$

By independence of (u, w), the D-dimensional density factorizes as

$$p_Z(z) \;=\; p_2(u)\, p_W(w) \;=\; p_2(u) \prod_{j=1}^{m} \mathcal{N}\big(w_j; 0, \sigma_{\text{pad}}^2\big).$$

Since $x = Rz$ with orthogonal R, the change-of-variables determinant is unity ($|\det R| = 1$), hence

$$p_X(x) \;=\; p_Z\big(R^\top x\big).$$

Writing $z = (u, w) = R^\top x$, we obtain the oracle log-density

$$l^{\text{oracle}}(x) = \log p_X(x) \;=\; \log p_2\big(u\big) \;+\; \sum_{j=1}^{m} \log \mathcal{N}\big(w_j; 0, \sigma_{\text{pad}}^2\big). \tag{4}$$

Recovering the 2D generator density Let $R \in \mathbb{R}^{D\times D}$ be the rotation used to construct the dataset and define the selector

$$S \;=\; \begin{bmatrix} I_2 & 0_{2\times(D-2)} \end{bmatrix} \in \mathbb{R}^{2\times D}.$$

Given any observed point $x \in \mathbb{R}^D$, the unrotated coordinates are $z = R^\top x$, and the *generator (ancestor) coordinates* in $\mathbb{R}^2$ are

$$u \;=\; S\, z \;=\; SR^\top x.$$

Equivalently, with the fixed linear map $\Pi = SR^\top \in \mathbb{R}^{2\times D}$,

$$u \;=\; \Pi x, \qquad \Pi^\top \Pi \;=\; RS^\top SR^\top \;\text{ (projection onto the generator subspace).}$$

Since the global rotation is orthogonal, the 2D density of the latent ancestor corresponding to x is simply the mixture density evaluated at u:

$$p_{\text{ancestor}}(x) \;=\; p_2\big(u\big) \;=\; p_2\big(\Pi x\big). \tag{5}$$

B Experimental setup

Synthetic data We construct three synthetic datasets as follows. We first sample from a 2D GMM with $k \in \{4, 8, 12\}$ mixture components, then append $(D-2)$ i.i.d. "filler" dimensions drawn from $\mathcal{N}(0, 0.02^2)$. Subsequently, we apply a random orthogonal rotation in $\mathbb{R}^D$ as described in Appendix A. We set D=50 for the first two datasets (k=4 and k=8) and D=30 for the third dataset (k=12). Each dataset contains 60,000 training and 10,000 validation points.

Priors We assess two priors for the VAE latent variable z: (i) the isotropic standard Gaussian $\mathcal{N}(0, I)$; and (ii) a learnable k-component Gaussian mixture,

$$p_Z(z) = \sum_{k=1}^{k} \pi_k\, \mathcal{N}\big(z;\, \mu_k, \operatorname{diag}(\sigma_k^2)\big),$$

with $k \in \{4, 8, 12\}$ chosen to match the generator's component count. In the mixture prior, the component means $\{\mu_k\}$ and diagonal scales $\{\sigma_k\}$ are learned.

Density estimates We associate to each training sample its log-density estimate ρ_i used by the aligner as a supervising signal: (i) *Oracle*—the ground-truth $\log p_X(x)$ from the data generator (details on the Oracle are given in A, see in particular eq. 4); (ii) *PAk* estimator [11] computed after projecting the data onto the top d=2 principal components (matching the latent dimensionality). The PAk routine also returns a standard error, which we use as precision weight.

Training For the synthetic datasets, all models are VAEs with latent dimension $d = 2$, a Gaussian decoder with fixed observation noise $\sigma_x = 0.02$, and shallow linear MLPs. The encoder maps $D \to D/2 \to (\mu, \log \sigma^2) \in \mathbb{R}^2$, and the decoder maps $2 \to D/2 \to D$. For MNIST, all models are VAEs with latent dimension $d = 10$, a Bernoulli decoder, and a shallow MLP with one hidden layer with 300 neurons, both in the encoder and the decoder.

In both settings, we train with Adam ($lr = 10^{-3}$) [17], batch size 128, for 100 epochs. We apply a KL warm-up [18] from 0.1 to 1.0 over the first half of training. We compare NONE (no alignment), DIRECT (direct aligner), and FLOW (flow aligner). For the flow aligner, we use a small normalizing flow with coupling layers [19, 20, 21], with 5 layers and hidden width 16. Experiments are repeated over multiple random seeds.
Our code is publicly available at `https://github.com/alessimichele/DiVAE-EurIPS`.

C Definitions and explanation of the metrics

Why KS/W. We evaluate *distributions* of log-densities through KS and W rather than relying on $\bar{s}$ alone, because means can mask "holes" (mass misplacement/missing modes): two models may match $\bar{s}$ yet allocate probability very differently. KS captures the worst-case gap between empirical CDFs, while W measures average transport. Together they provide complementary diagnostics of distribution-level calibration (lower is better).

Why $\mathrm{KL}(p_2 \,\|\, p_Z)$. We favor KL(*true* ‖ *prior*) because it penalizes under-allocation of probability where the *true* generator places mass (poor coverage). Lower values indicate better coverage of the true support.

Why $\mathrm{KL}(q \,\|\, p_2)$. It quantifies how incompatible the latent encoding of x is with the *true* generator. For OOD inputs, a *higher* value is desirable: it indicates the model correctly identifies that the posterior mass lies off the true latent manifold (i.e., it does not spuriously project OOD inputs onto plausible generator modes).

Why entropy $\mathcal{H}(q)$. It measures the encoder's uncertainty about the latent representation. OOD inputs should induce *higher* entropy (broader posteriors), reflecting appropriate uncertainty rather than overconfident misassignments.

Definition and motivation for $\Delta\mathrm{KL}$. We define a *latent mismatch shift* ($\Delta\mathrm{KL}$) as the difference between the expected posterior–prior divergence on OOD and in-distribution inputs:

$$\Delta\mathrm{KL} \;=\; \mathbb{E}_{x\sim p_{\mathrm{OOD}}}\big[\mathrm{KL}\big(q(z \mid x) \,\|\, p(z)\big)\big] \;-\; \mathbb{E}_{x\sim p_{\mathrm{IN}}}\big[\mathrm{KL}\big(q(z \mid x) \,\|\, p(z)\big)\big]. \tag{6}$$

This metric measures how much does the posterior–prior mismatch increase when I move from in-distribution (MNIST) inputs to OOD (Fashion-MNIST) inputs, thus positive values of $\Delta\mathrm{KL}$ indicate that the posterior–prior mismatch is systematically larger on OOD data than on in-distribution data.

Definition and motivation for $\Delta\mathcal{H}$. We define an entropy-based *uncertainty shift* as the difference between the expected posterior entropy on OOD and in-distribution inputs:

$$\Delta\mathcal{H} \;=\; \mathbb{E}_{x\sim p_{\mathrm{OOD}}}\big[\mathcal{H}\big(q(z \mid x)\big)\big] \;-\; \mathbb{E}_{x\sim p_{\mathrm{IN}}}\big[\mathcal{H}\big(q(z \mid x)\big)\big], \tag{7}$$

where $q(z \mid x)$ is the approximate posterior and $\mathcal{H}(\cdot)$ denotes its entropy.

We hypothesize that, in a good model, the encoder becomes noticeably more uncertain when it sees OOD data. Therefore, positive values of $\Delta\mathcal{H}$ are to be preferred.

D Model details

Huber loss The Huber loss, defined as:

$$\Gamma_\delta(e) = \begin{cases} \frac{1}{2}e^2, & |e| \le \delta, \\ \delta\big(|e| - \frac{1}{2}\delta\big), & |e| > \delta. \end{cases}$$

is quadratic when the residual's magnitude e is below a threshold a, and linear beyond that, with a continuous derivative at a. It is preferred to the squared error when performing robust regression for it is less sensitive to outliers [12].

E Tables

Notation	Definition	Notes
$\widehat{\mathcal{L}}_i$	$\log p_\theta(x_i \mid z_i) - \mathrm{KL}\big(q_\phi(z \mid x_i) \,\|\, p_Z(z;\psi)\big)$	Single-point ELBO estimator; $z_i \sim q_\phi(z \mid x_i)$
ℓ_i^{oracle}	$\log p_X(x_i)$ (see Appx. A.4)	Single-point true data log-density
s_i	$\log p_Z(z_i;\psi)$	Single-point log-prior probability; $z_i \sim q_\phi(z \mid x_i)$
s_i^{oracle}	$\log p_2(u_i)$ (see Appx. A.5), $u_i = \Pi x_i$	Single-point true generator (2D) log-density
ℓ	$(\widehat{\mathcal{L}}_i)_{i=1}^{N_{\text{val}}}$	Mean: $\overline{\ell} = \frac{1}{N_{\text{val}}}\sum_i \widehat{\mathcal{L}}_i$
ℓ^{oracle}	$(\ell_i^{\text{oracle}})_{i=1}^{N_{\text{val}}}$	Mean: $\overline{\ell}^{\text{oracle}} = \frac{1}{N_{\text{val}}}\sum_i \ell_i^{\text{oracle}}$
$\mathbf{s}$	$(s_i)_{i=1}^{N_{\text{val}}}$	Mean: $\bar{s} = \frac{1}{N_{\text{val}}}\sum_i s_i$
$\mathbf{s}^{\text{oracle}}$	$(s_i^{\text{oracle}})_{i=1}^{N_{\text{val}}}$	Mean: $\bar{s}^{\text{oracle}} = \frac{1}{N_{\text{val}}}\sum_i s_i^{\text{oracle}}$
KS	$\sup_x \lvert F_{\mathbf{s}}(x) - F_{\mathbf{s}^{\text{oracle}}}(x)\rvert$	Kolmogorov-Smirnov two samples test between $\bar{s}$ and $\bar{s}^{\text{oracle}}$
W	$\int_0^1 \big\lvert F_{\mathbf{s}}^{-1}(u) - F_{\mathbf{s}^{\text{oracle}}}^{-1}(u)\big\rvert\, du$	1D Wasserstein (EMD) between $\bar{s}$ and $\bar{s}^{\text{oracle}}$
KL	$\mathrm{KL}(p_2(\cdot) \,\|\|\, p_Z(\cdot;\psi))$	KL divergence between the true 2D GMM and the VAE's prior
$KL(q, p_2)$	$\mathrm{KL}(q(\cdot \mid x) \,\|\|\, p_2(\cdot))$	KL divergence between the variational posterior and the true 2D GMM
$\mathcal{H}(q)$	$-\mathbb{E}_{z\sim q_\phi(z\mid x_i)}\big[\log q_\phi(z \mid x_i)\big]$	Entropy of the variational posterior

Table 5: Per-sample quantities and their vector/mean aggregates used in evaluation. F denotes the empirical CDF.

Prior	Method	$\overline{\ell}$	$\bar{s}$	W	KS	KL
GMM	No reg	**116.037 ± 5.193**	-1.998 ± 0.880	0.814	0.469	114.184
GMM	DIRECT	115.970 ± 5.225	**-2.589 ± 0.821**	**0.223**	**0.133**	64.259
GMM	FLOW	115.856 ± 5.241	-3.306 ± 0.806	0.516	0.398	**34.723**
STANDARD	No reg	**115.408 ± 5.162**	**-2.728 ± 0.581**	**0.233**	**0.116**	
STANDARD	DIRECT	115.225 ± 5.229	-2.663 ± 0.528	0.258	0.126	
STANDARD	FLOW	115.330 ± 5.219	-2.964 ± 0.740	0.270	0.193	

Table 6: Synthetic results for $k = 4$, dim=50. Averaged across 3 seeds. Oracle external densities. $\overline{\ell}^{\text{oracle}} = 116.831$; $\bar{s}^{\text{oracle}} = -2.81252$

Prior	Method	$\overline{\ell}$	$\bar{s}$	W	KS	KL
GMM	No reg	**116.037 ± 5.193**	-1.998 ± 0.880	0.814	0.469	114.184
GMM	DIRECT	115.933 ± 5.186	**-2.667 ± 0.759**	**0.182**	**0.102**	69.620
GMM	FLOW	114.441 ± 5.291	-6.395 ± 0.838	3.583	0.946	**4.419**
STANDARD	No reg	**115.408 ± 5.162**	**-2.728 ± 0.581**	**0.233**	**0.116**	
STANDARD	DIRECT	115.247 ± 5.182	-2.473 ± 0.396	0.361	0.228	
STANDARD	FLOW	115.082 ± 5.228	-3.286 ± 0.813	0.500	0.338	

Table 7: Synthetic results for $k = 4$, dim=50. Averaged across 3 seeds. PAk external densities. $\overline{\ell}^{\text{oracle}} = 116.831$; $\bar{s}^{\text{oracle}} = -2.81252$

Prior	Method	$\overline{\ell}$	$\bar{s}$	W	KS	KL
GMM	No reg	115.188 ± 5.300	-2.495 ± 0.902	0.870	0.501	20.877
GMM	DIRECT	**115.245 ± 5.236**	-3.078 ± 0.813	0.294	**0.187**	**8.840**
GMM	FLOW	114.755 ± 16.984	**-3.117 ± 0.913**	**0.285**	0.190	9.494
STANDARD	No reg	**114.339 ± 5.196**	-2.720 ± 0.496	0.645	0.322	
STANDARD	DIRECT	113.794 ± 17.043	-2.847 ± 0.520	0.518	0.233	
STANDARD	FLOW	113.748 ± 18.338	**-3.054 ± 0.648**	**0.310**	**0.137**	

Table 8: Synthetic results for $k = 8$, dim=50. Averaged across 3 seeds. Oracle external densities. $\overline{\ell}^{\text{oracle}} = 116.256$; $\bar{s}^{\text{oracle}} = -3.36459$

Prior	Method	$\overline{\ell}$	$\bar{s}$	W	KS	KL
GMM	No reg	**115.188 ± 5.300**	-2.495 ± 0.902	0.870	0.501	20.877
GMM	DIRECT	115.142 ± 5.305	**-3.244 ± 0.803**	**0.151**	**0.073**	6.447
GMM	FLOW	113.506 ± 5.891	-6.325 ± 1.341	2.960	0.753	**5.334**
STANDARD	No reg	**114.339 ± 5.196**	-2.720 ± 0.496	0.645	0.322	
STANDARD	DIRECT	113.919 ± 5.332	-2.844 ± 0.504	0.521	**0.231**	
STANDARD	FLOW	113.490 ± 17.099	**-3.812 ± 0.925**	**0.493**	0.315	

Table 9: Synthetic results for $k = 8$, dim=50. Averaged across 3 seeds. PAk external densities. $\overline{\ell}^{\text{oracle}} = 116.256$; $\bar{s}^{\text{oracle}} = -3.36459$

Prior	Method	$\overline{\ell}$	$\bar{s}$	W	KS	KL
GMM	NONE	**64.961 ± 4.118**	-4.265 ± 0.925	0.633	0.380	3.728
GMM	DIRECT	64.813 ± 4.141	**-3.988 ± 0.773**	**0.334**	**0.238**	3.349
GMM	FLOW	64.833 ± 4.155	-5.049 ± 0.869	1.288	0.606	**3.238**
STANDARD	NONE	**64.363 ± 4.069**	-2.866 ± 0.680	0.896	0.418	
STANDARD	DIRECT	64.156 ± 4.164	-2.955 ± 0.723	0.808	0.388	
STANDARD	FLOW	64.272 ± 4.141	**-3.199 ± 0.896**	**0.564**	**0.358**	

Table 10: Synthetic results for $k = 12$, dim=30. Averaged across 3 seeds. Oracle external densities. $\overline{\ell}^{\text{oracle}} = 66.045$; $\bar{s}^{\text{oracle}} = -3.76285$

Prior	Method	$\bar{\ell}$	$\bar{s}$	W	KS	KL
GMM	NONE	**64.961 ± 4.118**	-4.265 ± 0.925	0.633	0.380	**3.728**
GMM	DIRECT	64.837 ± 4.051	**-3.549 ± 0.608**	**0.239**	**0.124**	6.823
GMM	FLOW	63.194 ± 4.300	-8.195 ± 1.290	4.432	0.934	4.582
STANDARD	NONE	**64.363 ± 4.069**	-2.866 ± 0.680	0.896	0.418	
STANDARD	DIRECT	64.099 ± 4.173	-2.996 ± 0.717	0.767	0.384	
STANDARD	FLOW	63.967 ± 4.233	**-3.824 ± 1.264**	**0.551**	**0.255**	

Table 11: Synthetic results for $k = 12$, dim=30. Averaged across 3 seeds. PAk external densities. $\bar{\ell}^{\text{oracle}} = 66.045$; $\bar{s}^{\text{oracle}} = -3.76285$

Prior	Method	t_{init}	$\bar{t}_{\text{batch}}$	$\bar{t}_{\text{epoch}}$
STANDARD	NONE	-	0.0040	1.898
STANDARD	DIRECT	12.08	0.0051	2.392
STANDARD	FLOW	10.68	0.0258	12.111
GMM	NONE	-	0.0053	2.508
GMM	DIRECT	12.34	0.0073	3.404
GMM	FLOW	13.00	0.0295	13.849
VAMP	NONE	-	0.0060	2.802
VAMP	DIRECT	16.07	0.0088	4.133
VAMP	FLOW	11.36	0.0309	14.482

Table 12: **Timing results.** Direct regularization introduces essentially no computational overhead during training: both per-batch and per-epoch times remain comparable to the unregularized baseline, with a modest increase in initialization time due solely to the $O(N \log N)$ preprocessing step required to compute the nearest-neighbour graph over the N training samples. In contrast, the flow-based regularizer incurs substantial overhead in both batch and epoch times, as it requires training an additional normalizing flow module jointly with the VAE.