The VampPrior Mixture Model

Andrew A. Stirn
Columbia University

David A. Knowles
Columbia University & New York Genome Center

Abstract

Widely used deep latent variable models (DLVMs), in particular Variational Autoencoders (VAEs), employ overly simplistic priors on the latent space. To achieve strong clustering performance, existing methods that replace the standard normal prior with a Gaussian mixture model (GMM) require defining the number of clusters to be close to the number of expected ground truth classes a-priori and are susceptible to poor initializations. We leverage VampPrior concepts (Tomczak & Welling, 2018) to fit a Bayesian GMM prior, resulting in the *VampPrior Mixture Model* (VMM), a novel prior for DLVMs. In a VAE, the VMM attains highly competitive clustering performance on benchmark datasets. Integrating the VMM into scVI (Lopez et al., 2018), a popular scRNA-seq integration method, significantly improves its performance and automatically arranges cells into clusters with similar biological characteristics.

1 INTRODUCTION

Extracting meaningful knowledge from complex, high-dimensional data is a central promise of data science. A striking example is single-cell RNA sequencing (scRNA-seq), which has become ubiquitous in biomedical research, enabling genome-wide profiling of gene expression for millions of cells in one experiment. However, single-cell datasets increasingly contain many samples collected under different conditions using different methodologies, leading to complex nested batch effects (Luecken et al., 2022) that often drive larger variance than the biological signals of interest. The goal of scRNA-seq integration is to remove these batch effects while conserving biological variation such as cell type identity, disease, and perturbation effects.

The modus operandi of scRNA-seq analysis is to first integrate the observed data X, a N cells $\times$ G genes matrix, by mapping to an embedded space Z, with lower dimension $p \ll G$. Z is further reduced to visualizable dimensions with t-SNE (Van der Maaten & Hinton, 2008), UMAP (McInnes et al., 2018), or MDE (Agrawal et al., 2021). When the embedding function does not account for systematic shifts in expression profiling between datasets and/or batches that use different scRNA-seq technologies, misleading structure can arise, confounding standard analysis pipelines. Accordingly, Lähnemann et al. (2020) identify atlas-level integration as one of the grand challenges of single-cell data science.

Luecken et al. (2022) benchmark 12 integration methods on 13 atlas-level integration tasks and find that Scanorama (Hie et al., 2019) as well as DLVMs using amortized variational inference (VI) such as scVI (Lopez et al., 2018) and its derivatives (Svensson et al., 2020; Xu et al., 2021) are the most performant, outperforming popular methods such as Harmony (Xu et al., 2021). scVI provides the encoder and decoder a one-hot encoded batch identifier to encourage a biological embedding space that is disentangled from confounding batch effects. With a fixed $\mathcal{N}(0, I)$ prior on the embedding, scVI's generative process does not explicitly encourage clustering of similar cells. We propose replacing scVI's $\mathcal{N}(0, I)$ prior with a more flexible mixture prior that both improves its integration performance and provides robust clustering capabilities.

To achieve simultaneous integration and clustering we first investigate a method, VaDE (Jiang et al., 2017), that replaces the $\mathcal{N}(0, I)$ prior of the standard Variational Autoencoder (VAE) (Kingma & Welling, 2014; Rezende et al., 2014) with a Gaussian mixture model (GMM). We extend VaDE in two ways. First, we reformulate the mixture as a Bayesian GMM with hyper-priors on component parameters and a Dirichlet hyper-prior on mixing proportions; this allows us to instantiate an arbitrarily large number of compo-

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

nents to closely approximate a Dirichlet process (DP) GMM (Edward, 2000) (i.e. a truncated DP) enabling our model to automatically prune unnecessary components. Second, rather than learning point-estimates for cluster centers, we fit their distributions with the encoder network operating on trainable pseudo-inputs analogously to the VampPrior (Tomczak & Welling, 2018). Initializing these pseudo-inputs with randomly selected training data ensures latent cluster centers are well initialized resolving VaDE's susceptibility to poor initializations. We call our approach the *VampPrior Mixture Model* (VMM), which we optimize using an algorithm that alternates between amortized VI steps (maximizing the variational objective with fixed prior parameters) and Empirical Bayes steps (fitting just prior parameters).

Since the VMM merely replaces a $\mathcal{N}(0, I)$ prior, it is applicable to any DLVM with continuous latent variables. Section 4 employs a VMM within the VAE's generative process for image clustering. The VMM not only outperforms VAE-based clustering methods but also approaches state-of-the-art unsupervised classification performance. Section 5 integrates the VMM into scVI and finds significant improvements to both batch correction and biological conservation during scRNA-seq integration, while also enabling unsupervised cell-type annotation. Numerous other DLVMs extending scVI have been developed for scATAC-seq, multimodal single-cell, and spatial transcriptomic data analyses that all use $\mathcal{N}(0, I)$ priors in their generative processes. Integrating the VMM into these tools, among others, would be straightforward, amplifying this work's potential impact.

2 BACKGROUND

2.1 The Variational Autoencoder

DLVMs parameterize data generating distributions with neural networks operating on latent variables. The VAE (Kingma & Welling, 2014) is a DLVM with generative process,

$$z_i \sim \mathcal{N}(0, I) \quad \forall\ i \in [N] \tag{1}$$

$$x_i | z_i \sim p_\theta(x_i|z_i) \triangleq p(x_i|f(z_i;\theta)) \quad \forall\ i \in [N], \tag{2}$$

where f is the decoding neural network with parameters θ. The VAE employs amortized VI, defining the mean and covariance of the variational family $q_\phi(z_i; x_i) \triangleq \mathcal{N}(z_i; g(x_i;\phi))$ as outputs of neural network g with parameters ϕ operating on observed x_i, and black-box VI (Ranganath et al., 2014) with reparameterization gradients (Williams, 1992) to maximize the evidence lower bound (ELBO) w.r.t. θ and ϕ,

$$\mathcal{L}(\mathcal{D};\theta,\phi) = \sum_{x\in\mathcal{D}} \mathop{\mathbb{E}}_{q_\phi(z;x)} \big[\log p_\theta(x|z)\big] - \mathrm{KL}\left(q_\phi(z;x)||p(z)\right). \tag{3}$$

2.2 The VampPrior

Tomczak and Welling (2018) identify the prior that maximizes eq. (3) as the aggregate posterior $p^*(z) = N^{-1}\sum_{i=1}^N q_\phi(z; x_i)$, over the N training points. To approximate the aggregate posterior, they replace the VAE's prior eq. (1) with their VampPrior,

$$p(z) \triangleq \frac{1}{K}\sum_{j=1}^K q_\phi(z; u_j),$$

where pseudo-inputs $u_1, \ldots, u_K$ are trainable prior parameters initialized with $K \ll N$ randomly selected training points for efficiency. Inference proceeds by maximizing eq. (3) w.r.t. θ, ϕ, and now u.

2.3 Gaussian Mixture Models

A GMM on z can be represented as,

$$z_1, \ldots, z_N | \pi, \mu, \Lambda \sim \sum_{j=1}^K \pi_j\, \mathcal{N}(z|\mu_j, \Lambda_j^{-1}), \tag{4}$$

with mixing proportions π, cluster means μ, and cluster precisions Λ. In a Bayesian framework, we place priors on all parameters,

$$\alpha \sim \mathrm{InverseGamma}(1, 1), \tag{5}$$

$$\pi|\alpha \sim \mathrm{Dirichlet}\big(\alpha K^{-1}\mathbf{1}_K^T\big), \tag{6}$$

$$\mu_k \sim \mathcal{N}(0, I) \qquad \forall\ k \in [K], \tag{7}$$

$$\Lambda_k \sim \mathrm{Wishart}\left(p+2, \frac{K^{\frac{1}{p}}}{p+2} I\right) \qquad \forall\ k \in [K], \tag{8}$$

where $p \equiv \dim(z)$. Taking $K \to \infty$ results in a Dirichlet process (DP) GMM (Edward, 2000). Ishwaran and James (2002) prove that an upper bound for the absolute difference between the marginal densities of a K-component GMM and the ∞-component GMM integrated over $\Re^p$ decays exponentially w.r.t. K. Our Bayesian mixtures, which use large K, will therefore well approximate the limiting DP mixture and eliminate clusters in a similar manner.

3 METHODS

Before introducing the VMM, we first show how to use eq. (4)-eq. (8) as a hierarchical prior that replaces eq. (1) in the VAE. We then describe our inference algorithm that alternates between VI and Empirical Bayes steps. From there, the VMM is a straightforward change.

Algorithm 1 Alternating VI and Empirical Bayes

while not converged **do**
  Sample batch: $\mathcal{B} \leftarrow \{x_1, \dots, x_M \sim \hat{\Pr}(\mathcal{D})\}$
  VI: $(\phi, \theta) \leftarrow (\phi, \theta) + \gamma_1 \nabla_{\phi,\theta} \mathcal{L}(\mathcal{B}; \theta, \phi)$
  Sample posterior: $z_i \sim q_\phi(z_i; x_i) \quad \forall\ i \in [M]$
  EM: $\psi \leftarrow \psi + \gamma_2 \nabla_\psi \mathbb{E}_{q(c)}[\log p(z, c, \pi, \mu, \Lambda, \alpha)]$
end while

3.1 A Bayesian GMM Prior

Fraley and Raftery (2007)′s recommended Wishart parameters for GMMs with observed z_i′s motivated our choice in eq. (8). In our setting, z_i′s are latent, so we replace the empirical precision term in the Wishart's scale matrix with the identity. We found that further normalizing the scale matrix by $p + 2$ to ensure $\mathbb{E}[\Lambda_j] = K^{\frac{1}{p}} I$ makes eq. (8) more tolerant to different latent dimensions p.

For a VAE, defining $p(z)$ as a GMM changes only the Kullback-Leibler (KL) divergence term in eq. (3) to,

$$\mathrm{KL}\left(q_\phi(z; x) \middle|\middle| \sum_{j=1}^{K} \pi_j \mathcal{N}(z|\mu_j, \Lambda_j^{-1})\right). \tag{9}$$

3.2 Alternating Inference Algorithm

We use a Bayesian GMM prior to achieve a clustered latent representation without knowing the true number of classes a-priori. In this scenario, we found jointly optimizing variational and prior parameters was suboptimal. Instead, we partition all parameters into distinct variational (θ and ϕ) and prior ($\alpha, \pi, \mu, \Lambda$) parameter sets and then alternate between variational inference and Empirical Bayes gradient steps[1]. We perform VI (optimizing eq. (3) w.r.t. θ and ϕ) to fit variational parameters and perform maximum a posteriori (MAP) estimation (optimizing eq. (4)-(8) w.r.t. to $\psi \triangleq \{\alpha, \pi, \mu, \Lambda\}$) to fit prior parameters.

Algorithm 1 shows how we alternate between these two separate inference procedures. We first perform a step of stochastic amortized VI for the VAE as usual. We then sample the aggregate posterior and perform a step of gradient based MAP estimation via expectation maximization (EM), learning point estimates for $\alpha, \pi, \mu, \Lambda$. EM requires representing the GMM as,

$$c_i|\pi \sim \text{Categorical}(\pi), \quad z_i|c_i, \mu, \Lambda \sim \mathcal{N}(z|\mu_{c_i}, \Lambda_{c_i}^{-1}).$$

The E-step is $q(c_i = j) \propto \log p(z_i|c_i = j, \mu, \Lambda) + \log \pi_j$. We repeat this process until convergence, i.e. when validation set performance plateaus.

[1]Direct optimization of θ can be viewed as part of VI by considering a uniform prior and defining $q(\theta)$ as a Dirac delta.

Cluster granularity and thereby performance depends on the strength of the hyper-priors in eqs. (5) to (8). We found that adjusting the batch size and/or the Empirical Bayes learning rate γ_2 was effective at regulating the strength of the hyper-priors and simpler than adjusting their many parameters. A larger batch size up-weights the prior likelihood eq. (4) relative to the hyper-priors during Empirical Bayes EM.

3.3 The VMM

The VMM retains the DP-GMM's generative process eq. (4)-(8), but rather than learning point estimates for $\mu_1, \dots, \mu_K$, it fits distributions over cluster centers $\mu_1, \dots, \mu_K$ analogously to the VampPrior by defining their variational distributions as,

$$\mu_j \sim q_\phi(\mu_j; u_j) \triangleq \mathcal{N}(\mu_j; g(u_j; \phi)) \quad \forall\ j \in [K]. \tag{10}$$

Thus, the standard normal prior on $\mu_1, \dots, \mu_K$ eq. (7) still regularizes cluster dispersion but indirectly via $u_1, \dots, u_K$ rather than directly on $\mu_1, \dots, \mu_K$. Like the VampPrior, the VMM uses the encoder network operating on pseudo-inputs $g(u_j; \phi)$. Where the VampPrior uses the encoder network to parameterize a mixture prior over the latent embedding z directly, the VMM uses the encoder network to parameterize variational cluster distributions during Empirical Bayes.

Fitting distributions $q_\phi(\mu_j; u_j)$ instead of point estimates only requires substituting eq. (9) with,

$$\mathrm{KL}\left(q_\phi(z; x) \middle|\middle| \sum_{j=1}^{K} \pi_j \mathop{\mathbb{E}}_{q_\phi(\mu_j; u_j)} \left[\mathcal{N}(z|\mu_j, \Lambda_j^{-1})\right]\right) \tag{11}$$

during VAE inference, which we calculate analytically. Equation (11) approaches that of the VampPrior for $\pi_j \to \frac{1}{K}$ and $\Lambda_j^{-1} \to 0$.

Following Tomczak and Welling (2018), we initialize pseudo-inputs with randomly sampled training data. After replacing μ with u in parameter set ψ, algorithm 1 adjusts easily to the VMM, requiring only that the E-step now also be w.r.t. $q_\phi(\mu; u)$ (the joint distribution for $\mu_1, \dots, \mu_K$),

$$\mathop{\mathbb{E}}_{q(c) q_\phi(\mu; u)} \left[\log p(z, c, \pi, \mu, \Lambda, \alpha)\right],$$

which we also calculate analytically. The M-step back propagates through $g(u_j; \phi)$, treating ϕ as a constant, to compute gradients w.r.t. u_j. For a complete presentation of VMM updates, see section B.

We selected MAP EM for prior inference to preserve Gaussian amortized variational families and maintain compatibility with the many non-standard VAE-based methods used in scRNA-seq analysis (Lopez et al.,

2018; Svensson et al., 2020; Xu et al., 2021). We seek only to replace the VAE's $\mathcal{N}(0, I)$ prior with a more flexible GMM prior (where the $\mathcal{N}(0, I)$ is now over cluster centers). For example, had we chosen VI instead of MAP EM for prior inference, a Normal-Wishart $q(\mu_j, \Lambda_j)$ is required for analytic component distribution computation, $\mathbb{E}[\mathcal{N}(z|\mu_j, \Lambda_j^{-1})]$, in eq. (11).

Prior selection can be seen as a spectrum with the unimodal $\mathcal{N}(0, I)$ prior at one extreme and the exact VampPrior (i.e. the aggregate posterior) with a component for each training data point at the other. The VMM falls between these extremes, identifying an appropriate number of modes to accurately model the data. In the limit of a single cluster, the VMM can approach an unimodal $\mathcal{N}(0, I)$ prior. In the limit of N clusters (or K for the approximate VampPrior), the VMM approaches the VampPrior. In this sense, the computational complexity of the VMM lies between the VAE and the VampPrior. For added efficiency, the VMM can turn off gradient computation for eliminated clusters, which is not possible for the VampPrior as it tends to use all K components as sections 4 and 5 demonstrate.

3.4 Related Work

Our alternating inference procedure is similar to Lee et al. (2020), who alternate VI and EM steps for a non-Bayesian GMM in the context of meta-learning. Our work is distinct from theirs in that we use a Bayesian mixture and the VampPrior to parameterize the cluster center distributions. Also, our intended applications are the latent clustering of images and scRNA-seq data integration.

Zhou et al. (2025) survey and taxonomize various deep clustering approaches, of which VAE-based methods are a subset. Our focus on VAEs is driven by their success in single-cell genomics and our hypothesis that these methods will benefit from a more flexible prior given the multimodal nature of biological data.

The DLGMM (Nalisnick et al., 2016) uses a GMM prior eq. (4) but with fixed component means (at equally spaced points on the line) and variances (one), learning only the prior weights π eq. (6). They have $q_\phi(z_{ij}; x_i)$ for $j \in [K]$, which requires evaluating the decoder K times and averaging the result according to $\pi_i \sim q_\phi(\pi_i; x_i)$, a Kumaraswamy stick-breaking variational family (Nalisnick & Smyth, 2017). Stirn et al. (2019) show this construction is non-exchangeable and thereby has limited approximation capacity. We attempted to scale their approach beyond one-dimensional z_i's with limited success but include their reported performance.

The GMVAE (Dilokthanakul et al., 2016) is,

$$z_i \sim \mathcal{N}(0, I), \qquad c_i \sim \text{Categorical}(\pi),$$
$$y_i|z_i, c_i \sim \mathcal{N}(f_{c_i}(z_i; \theta_{c_i})), \quad x_i|y_i \sim p(x_i; f_x(y_i; \theta_x))$$

for observed x_i, utilizing K decoders $(f_1, \ldots, f_K)$. Their chosen variational family, $q(z_i, y_i, c_i) \triangleq$

$$q(z_i; g_z(x_i; \phi_z))q(y_i; g_y(x_i; \phi_y))q(c_i|z_i, y_i),$$

sets $q(c_i|z_i, y_i)$ to the true posterior. Given their computational complexity, we rely on their reported performance.

VaDE (Jiang et al., 2017) is closest to our proposals. They optimize a GMM for the VAE's $p(z)$,

$$c_i \sim \text{Categorical}(\pi), \qquad z_i|c_i \sim \mathcal{N}(\mu_{c_i}, \Sigma_{c_i}).$$

They define $q(c_i; x_i)$ as we do in our E-step and $q_\phi(z_i; x_i)$ in the usual way and optimize their ELBO w.r.t. both variational and prior parameters. For small K fixed a-priori in the non-Bayesian setting, VaDE cannot automatically discover an appropriate number of clusters (as our Bayesian GMM and VMM do), often utilizing all available, and is highly susceptible to poor initializations. To avoid poor initializations, they pre-train a deterministic autoencoder, fit a GMM to the latent space, and use those GMM parameters to initialize VaDE's GMM parameters. Both our GMM and VMM do not require such pre-training.

Hrovatin et al. (2024) and Lopez et al. (2022) leverage the VampPrior for scRNA-seq and spatial transcriptomic data analysis, respectively. In contrast, our VMM, a DP-approximating GMM, leverages ideas from the VampPrior to parameterize the variational cluster distributions.

Both the VMM and VampPrior approximate the Empirical Bayes solution (i.e. the aggregate posterior), but the VMM provides additional representational flexibility and contains the VampPrior as a special case. In particular, the precision matrices Λ_j in our DP-GMM provide an additional degree of freedom when learning cluster widths. This flexibility proved critical for the VMM's clustering performance. The VampPrior burdens the encoder's amortized parameter map with the conflicting tasks of producing both low variance posteriors $q_\phi(z; x)$ when operating on data x to drive predictive performance and high variance prior components $q_\phi(z; u)$ when operating on pseudo-inputs u to drive the marginal likelihood. To the best of our knowledge, this paradox has not yet been recognized in the literature. The VMM resolves this paradox by learning the additional covariance terms Λ_j^{-1}.

Table 1: Clustering performance on benchmark datasets. Reported results appear in the top partition, where a '–' denotes missing. We collected all results in the bottom partition. We bold the best score for each column and any other scores whose $p \geq 0.05$ under a one-sided Mann-Whitney U test.

	mnist w/ $\dim(z) = 10$			fashion mnist w/ $\dim(z) = 30$		
Method	Utilized clusters	Accuracy	NMI	Utilized clusters	Accuracy	NMI
DLGMM (Nalisnick et al., 2016)	10	91.58	–	–	–	–
GMVAE (Dilokthanakul et al., 2016)	10 ± 0.00	0.778 ± 0.06	–	–	–	–
	16 ± 0.00	0.851 ± 0.02	–	–	–	–
	30 ± 0.00	0.928 ± 0.02	–	–	–	–
VampPrior (Tomczak & Welling, 2018)	100 ± 0.00	0.945 ± 0.011	0.627 ± 0.003	100 ± 0.00	**0.775 ± 0.016**	0.529 ± 0.003
VaDE (Jiang et al., 2017)	10 ± 0.00	0.857 ± 0.067	0.832 ± 0.033	5.5 ± 2.51	0.352 ± 0.066	0.499 ± 0.113
Our GMM	10.7 ± 2.31	0.902 ± 0.043	0.876 ± 0.013	16.1 ± 3.14	0.513 ± 0.046	0.509 ± 0.037
Our VMM	13.9 ± 2.13	**0.960 ± 0.006**	**0.899 ± 0.015**	16.5 ± 2.92	0.712 ± 0.009	**0.653 ± 0.015**

4 VAE EXPERIMENTS

While our intended application is scRNA-seq integration, we first test our methods on the familiar MNIST and Fashion MNIST datasets. We rescale images to $[-1, 1]$, set eq. (2) to $\mathcal{N}(x_i|f(z_i;\theta), \sigma^2 I)$, and learn σ^2 during VAE inference. Following Jiang et al. (2017), we use a three-layer MLP with hidden dimensions [500, 500, 2000] for the encoder and the reverse for the decoder, but anecdotally report that our methods had strong clustering performance for both smaller MLPs and small CNNs. Our variational $q(z; g(x; \phi))$ has full rank covariance. We set $K = 10$ for VaDE (as they do) and $K = 100$ for the VampPrior, our GMM, and the VMM. Reported results average over 10 different trials. Within each trial, data folds and neural network initializations are identical for the VampPrior, VaDE, our GMM, and the VMM. Section C contains additional details.

4.1 Clustering Performance

Table 1 compares clustering performance for VAE-based methods. Because we are in the unsupervised setting, we measure clustering performance on the combined training and validation sets after early stopping on validation set performance (see section C for details). Computing accuracy for models whose cluster count differs from the number of ground truth classes necessitates matching cluster assignments to ground truth labels. To accomplish this, we assign cluster members the label of its most probable member. However, this method trivially achieves 100% accuracy when placing an atom on each datapoint. Thus, the exact VampPrior (the aggregate posterior) over all N data, having a component for each datapoint, will also trivially achieve nearly perfect accuracy. By contrast, normalized mutual information (NMI) penalizes a model for using too many clusters since it compares a model's cluster assignments directly against labels. Therefore, NMI is a more appropriate metric for evaluating clustering performance in this setting. Winter et al. (2022) and Yang et al. (2020) report NMI values for vanilla GMMs and DP-GMMs that are far lower than our VMM's. We also calculated the Adjusted Rand Index (ARI) and found it had perfect Spearman correlation with NMI, so we report NMI only. We report accuracy for familiarity and consistency with other work.

For both datasets, our VMM identifies an appropriate number of clusters (knowing there are 10 labeled classes) and always has the best NMI. For Fashion MNIST, the VampPrior has the best accuracy, but uses all available clusters as it did for MNIST. In contrast, our VMM uses a reasonable number of clusters and has the best accuracy for Fashion MNIST as reported by papers-with-code. Our VaDE implementation's maximum accuracy of 0.936 on MNIST is nearly identical to their reported value of 0.945, suggesting our implementation is correct. Interestingly, VaDE's clustering performance does not generalize to Fashion MNIST. We suspect their pre-training procedure failed to provide a good initialization because of higher latent dimensions and/or Fashion MNIST's increased heterogeneity. Our Bayesian GMM uses no pre-training and outperforms VaDE on both datasets, confirming the benefit of our choice to use a DP-approximating mixture. The VMM outperforming our GMM corroborates the benefit of using VampPrior concepts to fit variational cluster centers in the GMM as opposed to non-amortized point estimates.

In fig. 1, each of the first 10 clusters (columns) represents one of the 10 digits and comprise 99.6% of the data according to the VMM's cluster proportions π (printed below each column). The next largest cluster (11th column) has $\pi_j = 0.3\%$ and captures an alternative way to write a '7', known as the slashed 7. An anthropologist unfamiliar with our numerical representation system would likely find the 11th cluster interesting. Utilizing 11 of the 100 available clusters to represent 99.9% of MNIST suggests the VMM excels at identifying a semantically representative set of clusters. We see similar results for Fashion MNIST in

Table 2: VAE model performance for different priors on Fashion MNIST. We bold the best score for each column and any other scores whose $p \geq 0.05$ under a one-sided Mann-Whitney U test.

Prior	$\log p(x)$	$\mathbb{E}_q[\log p(x\|z)]$	$\mathrm{KL}(q(z\|x)\|\|p(z))$	$\mathrm{KL}(q(z)\|\|p(z))$	$\mathbb{I}[z;n]$
$\mathcal{N}(0, I)$	46.9 ± 2.48	59.1 ± 2.98	**28 ± 0.15**	**18.8 ± 0.16**	9.21 ± 0.00
VampPrior	**165 ± 1.42**	**197 ± 1.90**	50.6 ± 0.46	41.4 ± 0.46	9.21 ± 0.01
Our GMM	53.1 ± 7.88	65.2 ± 8.44	**28.1 ± 1.21**	**18.9 ± 1.21**	9.21 ± 0.00
Our VMM	104 ± 7.50	124 ± 8.90	35.2 ± 1.59	25.8 ± 2.19	9.21 ± 0.01

section C.

Comparing samples from the marginal of a VMM and a VampPrior (fig. 1 and supplement fig. 4), we see that the diversity of samples from the VMM's cluster components is markedly higher than those from the VampPrior. The VampPrior uses $q_\phi(z; u_j)$ to fully specify a cluster, whereas the VMM uses $q_\phi(\mu; u_j)$ to specify just the cluster's mean and uses Λ_j to specify its width in each latent dimension, affording it the ability to generate more diversity from a single cluster and thereby use fewer clusters. While diverse, each cluster in the VMM produces semantically similar samples.

4.2 Model Comparison

In table 2, we report the marginal likelihood, the negative distortion and rate (Alemi et al., 2018), and the marginal KL divergence and mutual information (MI) between latent embedding z and index code n (Hoffman & Johnson, 2016) as measured on the validation fold. These quantities obey,

$$\underbrace{\log p(x)}_{\text{log marginal}} \geq \underbrace{\mathop{\mathbb{E}}_{q}[\log p(x|z)]}_{-\text{distortion}} - \underbrace{\mathrm{KL}(q(z|x)||p(z))}_{\text{rate}}$$

$$\underbrace{\mathrm{KL}(q(z|x)||p(z))}_{\text{rate}} = \underbrace{\mathrm{KL}(q(z)||p(z))}_{\text{marginal KL}} + \underbrace{\mathbb{I}[z;n]}_{\text{MI}}. \quad (12)$$

Unsurprisingly, the VampPrior has the highest evidence since it is, by design, approximately optimal w.r.t. the ELBO (section 2.2). This approximation gap narrows as $K \to N$. Tomczak and Welling (2018) observe ELBO improvements as K increases. In contrast, our VMM works to reduce the VampPrior's component usage to achieve a more interpretable clustering and therefore should have a lower marginal likelihood.

Hoffman and Johnson (2016) show that $\mathbb{I}[z;n] \leq \log N$ and approaches this upper bound (9.21 for validation set size) for $p \geq 10$. Thus, by eq. (12), the model with the highest rate will also have the largest marginal KL divergence. The VampPrior's ELBO maximization trades an increased rate for superior distortion, leading to a rather paradoxical result: despite the VampPrior setting $p(z)$ to approximate the aggregate posterior $q(z)$, it has the largest $\mathrm{KL}(q(z)||p(z))$, suggesting it is ineffective at regulating the structure of the latent embedding. Table 2 shows the VMM has substantially more modelling power than the GMM, despite utilizing the same number of clusters (table 1) and exhibits performance commensurate with it lying between the extremes of the unimodal standard normal and the VampPrior.

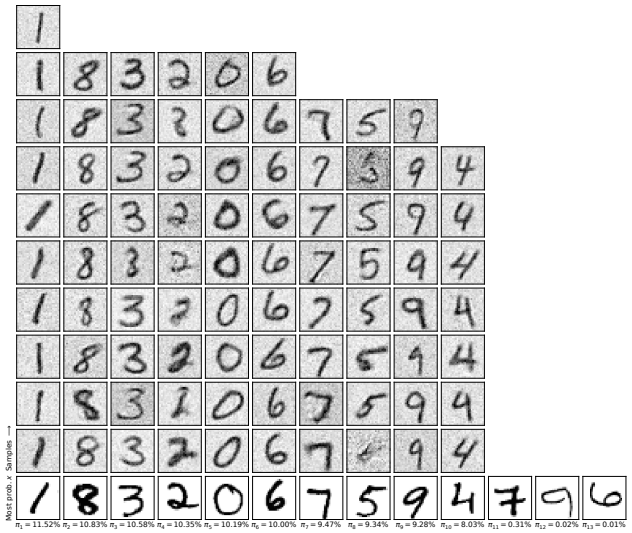


Figure 1: VMM prior predictive samples for MNIST. The number of columns equals the number of utilized clusters. The bottom row shows the data with the highest probability of belonging to the cluster, under which we print the cluster's probability π_j. The rows above are samples from the corresponding cluster. We sample $\mathrm{round}(10 \cdot \pi_j / \max(\pi))$ images from each component j to help visualize cluster proportions. No samples indicate this value rounded to zero.

5 SINGLE CELL EXPERIMENTS

We now integrate the VampPrior, our GMM, and our VMM into scVI (Lopez et al., 2018). In its default configuration, scVI models scRNA-seq read counts using amortized VI and a DLVM with generative process,

$$z_i \sim \mathcal{N}(0, I), \quad x_i|z_i, l_i, s_i \sim p\big(x_i; f_\theta(z_i, l_i, s_i)\big).$$

For each cell $i \in [N]$, we observe the read counts for G genes $x_i \in \mathbb{N}^G$, library size $l_i = \sum_j x_{ij}$ as the to-

Table 3: scRNA-seq clustering and integration performance. We bold the best score for each column and any other scores whose $p \geq 0.05$ under a one-sided Mann-Whitney U test.

			Clustering Performance			Luecken et al. (2022) Metrics		
Dataset	Model	Prior	Utilized Clusters	Accuracy	NMI	Batch correction	Bio conservation	Total
cortex	scVI tools	$\mathcal{N}(0, I)$	–	–	–	–	0.696 ± 0.020	–
	scVI (our code)	$\mathcal{N}(0, I)$	–	–	–	–	0.694 ± 0.019	–
		VampPrior	100 ± 0.00	**0.880 ± 0.022**	0.492 ± 0.005	–	**0.760 ± 0.007**	–
		GMM	19.7 ± 1.64	0.851 ± 0.037	**0.652 ± 0.016**	–	0.740 ± 0.007	–
		VMM	28.9 ± 2.88	0.833 ± 0.041	**0.663 ± 0.009**	–	**0.755 ± 0.007**	–
	Harmony	N/A	–	–	–	–	0.645 ± 0.000	–
	Scanorama	N/A	–	–	–	–	0.645 ± 0.000	–
pbmc	scVI tools	$\mathcal{N}(0, I)$	–	–	–	0.860 ± 0.011	0.661 ± 0.020	0.741 ± 0.012
	scVI (our code)	$\mathcal{N}(0, I)$	–	–	–	0.840 ± 0.013	0.669 ± 0.016	0.737 ± 0.010
		VampPrior	100 ± 0.00	**0.943 ± 0.020**	0.507 ± 0.003	**0.883 ± 0.007**	0.740 ± 0.014	**0.797 ± 0.008**
		GMM	12.2 ± 1.69	0.729 ± 0.109	0.756 ± 0.020	0.866 ± 0.007	0.722 ± 0.016	0.779 ± 0.009
		VMM	14.1 ± 0.74	**0.933 ± 0.014**	**0.817 ± 0.030**	**0.886 ± 0.009**	0.739 ± 0.011	**0.798 ± 0.007**
	Harmony	N/A	–	–	–	0.868 ± 0.001	0.734 ± 0.000	0.787 ± 0.001
	Scanorama	N/A	–	–	–	0.486 ± 0.007	**0.756 ± 0.009**	0.648 ± 0.005
split-seq	scVI tools	$\mathcal{N}(0, I)$	–	–	–	0.864 ± 0.005	0.598 ± 0.008	0.705 ± 0.005
	scVI (our code)	$\mathcal{N}(0, I)$	–	–	–	0.875 ± 0.006	0.612 ± 0.007	0.717 ± 0.004
		VampPrior	100 ± 0.00	**0.865 ± 0.013**	0.536 ± 0.004	**0.876 ± 0.005**	0.638 ± 0.010	**0.733 ± 0.006**
		GMM	14.1 ± 1.85	0.636 ± 0.067	0.600 ± 0.024	**0.876 ± 0.008**	0.631 ± 0.006	0.729 ± 0.006
		VMM	28.4 ± 2.72	**0.827 ± 0.060**	**0.641 ± 0.008**	**0.878 ± 0.004**	0.639 ± 0.004	**0.734 ± 0.003**
	Harmony	N/A	–	–	–	0.790 ± 0.002	**0.650 ± 0.002**	0.706 ± 0.002
	Scanorama	N/A	–	–	–	0.552 ± 0.000	0.632 ± 0.000	0.600 ± 0.000
lung atlas	scVI tools	$\mathcal{N}(0, I)$	–	–	–	**0.620 ± 0.006**	0.608 ± 0.015	0.613 ± 0.010
	scVI (our code)	$\mathcal{N}(0, I)$	–	–	–	0.616 ± 0.006	0.600 ± 0.008	0.607 ± 0.006
		VampPrior	100 ± 0.00	**0.627 ± 0.022**	0.548 ± 0.010	0.538 ± 0.014	**0.688 ± 0.015**	**0.628 ± 0.012**
		GMM	10.2 ± 1.32	0.305 ± 0.089	0.447 ± 0.037	0.611 ± 0.007	0.628 ± 0.019	**0.621 ± 0.013**
		VMM	20 ± 6.32	0.580 ± 0.040	**0.641 ± 0.020**	0.562 ± 0.012	0.674 ± 0.012	**0.629 ± 0.008**
	Harmony	N/A	–	–	–	0.553 ± 0.003	0.678 ± 0.002	**0.628 ± 0.002**
	Scanorama	N/A	–	–	–	0.328 ± 0.006	**0.694 ± 0.009**	0.548 ± 0.007

tal counts for cell i, and an integer batch (e.g. donor or patient) identifiers s_i. Here, $f_\theta(z_i, l_i, s_i)$ is a neural network that parameterizes a zero-inflated negative binomial (ZINB) distribution for all datasets except the lung atlas dataset, for which the scVI authors recommend a negative binomial. scVI defines the variational family $q_\phi(z_i; x_i, s_i)$ as a normal distribution with diagonal covariance. Thus, scVI is simply a VAE with a $\mathcal{N}(0, I)$ prior and a ZINB likelihood except that it additionally uses s_i to parameterize the variational and likelihood distributions.

Single-cell RNA-seq integration seeks a latent representation for biological variation that is disentangled from batch-specific technical variation. Batch variables s serve only as batch-specific calibration factors for the data generating distribution. Giving both the encoder and decoder access to s_i encourages scVI to reserve z_i's limited channel capacity for just biological variability, thus promoting disentanglement. Ideally, the encoder uses s_i to map x_i onto z_i, a batch-effect-free subspace, from which the decoder can reconstruct x_i given its access to s_i.

Incorporating our Bayesian GMM and VMM into scVI follows section 3, with one exception for the VampPrior and our VMM. Encoder network inputs x_i and s_i are non-negative integers and one-hot encoded batch IDs, respectively. For the part of each pseudo-input u corresponding to counts x, we relax integer constraints and fit a non-negative real G-vector via softplus-reparameterization. For the part of each pseudo-input corresponding to batch s, we fit a probability vector via a softmax-reparameterization.

We use scVI's API to download the cortex, PBMC, and lung atlas datasets and process them according to Lopez et al. (2018). For each dataset, we use scVI's recommended architecture, latent dimension, and learning rate for amortized VI (γ_1 in algorithm 1). Reported results are averages from 10 different trials. Within each trial, data folds are identical for all methods and neural network initializations are identical for our scVI implementations. See section D for additional details.

The left set of metrics in table 3 compares clustering performance. On all datasets the VMM achieves the best clustering performance according to NMI, suggesting strong agreement with the datasets' cell type annotations. The VampPrior's tendency to use all available clusters allows it to achieve good accuracy but hamstrings its NMI, suggesting over-clustering. Since the $\mathcal{N}(0, I)$ doesn't explicitly cluster, we do not report its clustering performance.

Table 3 (right) compares scRNA-seq integration using Luecken et al. (2022)'s benchmarking suite, which produces three summary scores: "batch correction"

assesses the removal of technical variation, "bio conservation" assesses the preservation of biological variation, and "total" is a weighted combination thereof. Batch correction and bio conservation scores are each a composite of five different metrics (supplement table 4). Luecken et al. (2022) scale all metrics to $[0, 1]$ such that larger values denote better performance.

The cortex dataset only has a single batch and thus lacks batch correction scores. Because of this, Harmony and Scanorama scores are from using PCA directly and thus identical. Our VMM has (or is tied with) the best total score on all datasets, suggesting the VMM is near state of the art. The VMM also has or matches the top batch correction score on each dataset except the lung atlas dataset, which has 16 donors and 17 cell types–a true atlas-level integration task. Examining the batch correction component scores (supplement table 4), we find that PCR (principal component regression) is the batch correction metric dragging down the VMM's batch correction score. PCR measures the relative reduction in total variance explained by the batch variable S for the integrated representation Z w.r.t. the raw count data X as $\frac{\text{Var}(X|S)-\text{Var}(Z|S)}{\text{Var}(X|S)}$. The score attains its maximal value when $\text{Var}(Z|S) = 0$, which is an unreasonable expectation for the lung atlas dataset since each donor has a different distribution of cell types with some donors only having a subset of the possible cell types. Therefore, we expect batch variable S to explain some amount of representation Z. In fact, it is concerning that scVI with a $\mathcal{N}(0, I)$ prior does so well here. This possibly suggests *over integration* as evidenced by its poorly structured Minimum-Distortion Embedding (MDE) plot in fig. 2. In contrast, the VMM has a well-structured MDE and finds meaningful clusters when comparing the cell-type annotations to VMM cluster assignments, supporting our case for simultaneous integration and clustering. The GMM's PCR metric and its qualitative MDE structure lies between those for the $\mathcal{N}(0, I)$ and VMM. Section D contains the remaining MDE comparison plots.

To confirm the VMM's batch performance scores on lung atlas were an artifact of the dataset, we use the SPLiT-seq dataset (Shabestari et al., 2022) as a control, because there every cell is sequenced twice using two slightly different sequencing library preparation steps. Thus, the cells' biological variability should be identical across the two batches (one for each library preparation). Indeed, we find the VMM is the top performer on this dataset, allaying concerns it hampered batch correction on the lung atlas data.

6 CONCLUSIONS

Advocating for the simultaneous integration and clustering of scRNA-seq, we developed the VampPrior Mixture Model (VMM) and an inference procedure that are adaptable to any DLVM model with continuous latent variables. The VMM produces well-clustered latent representations for both scRNA-seq and natural images.

The VMM outperforms all VAE-based clustering methods and attains highly competitive image clustering performance relative to all methods. To the best of our knowledge, we are the first to observe the VampPrior's paradoxical marginal KL divergence behavior and its ineffectiveness at finding semantically meaningful latent structure.

Replacing scVI's (Lopez et al., 2018) $\mathcal{N}(0, I)$ prior with the VMM significantly elevates its scRNA-seq integration performance, improving both batch correction and biological conservation. There are various extensions of scVI for other biological data modalities, which use $\mathcal{N}(0, I)$ priors in their generative processes. We conjecture these methods would derive substantial benefit by replacing the $\mathcal{N}(0, I)$ with the VMM and employing our alternating inference procedure. Furthermore, these methods would inherit the VMM's clustering ability, thereby simplifying downstream analysis.

The VMM is adept at clustering data into a semantically meaningful number of clusters when tuned to match the number of ground truth classes. We hope that the ability to control the VMM's cluster granularity via the batch size and/or the Empirical Bayes learning rate will help data scientists make novel discoveries in settings where ground truth labels are unknown.

Acknowledgements

We thank David M. Blei for his helpful comments and the anonymous reviewers for their insightful questions and suggestions. This work was made possible by support from the MacMillan Family via the MacMillan Center for the Study of the Non-Coding Cancer Genome at the New York Genome Center. This material is based upon work supported by the National Science Foundation under CAREER Grant No. DBI2146398. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

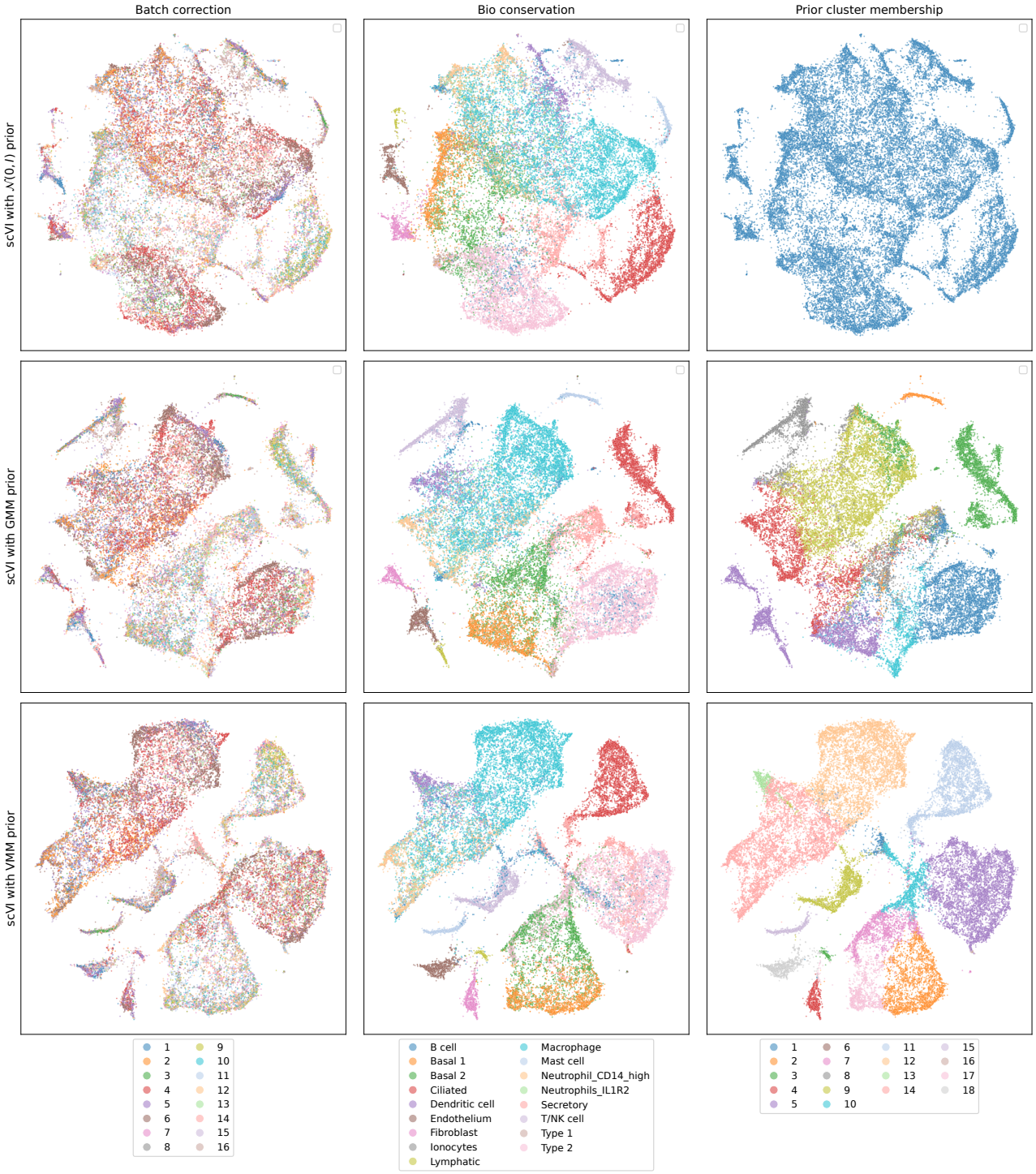


Figure 2: MDE comparison for the lung atlas dataset. Each row has the same embedding across columns for a tested prior. Columns respectively label points by the technical batch identifier, the annotated cell type, and the prior's cluster assignment.

References

Agrawal, A., Ali, A., & Boyd, S. (2021). Minimum-distortion embedding. *Foundations and Trends in Machine Learning*, *14*(3).

Alemi, A., Poole, B., Fischer, I., Dillon, J., Saurous, R. A., & Murphy, K. (2018). Fixing a broken ELBO. *Proceedings of Machine Learning Research*.

Dilokthanakul, N., Mediano, P. A., Garnelo, M., Lee, M. C., Salimbeni, H., Arulkumaran, K., & Shanahan, M. (2016). Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*.

Edward, R. C. (2000). The infinite gaussian mixture model. *Advances in neural information processing systems*.

Fraley, C., & Raftery, A. E. (2007). Bayesian regularization for normal mixture estimation and model-based clustering. *Journal of classification*, *24*(2).

Hie, B., Bryson, B., & Berger, B. (2019). Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nature Biotechnology*, *37*(6). https://doi.org/10.1038/s41587-019-0113-3

Hoffman, M. D., & Johnson, M. J. (2016). Elbo surgery: Yet another way to carve up the variational evidence lower bound. *Neural information processing systems (NeurIPS)*, *1*.

Hrovatin, K., Moinfar, A. A., Zappia, L., Lapuerta, A. T., Lengerich, B., Kellis, M., & Theis, F. J. (2024). Integrating single-cell RNA-seq datasets with substantial batch effects. *bioRxiv*.

Ishwaran, H., & James, L. F. (2002). Approximate Dirichlet Process Computing in Finite Normal Mixtures: Smoothing and Prior Information. *Journal of Computational and Graphical Statistics*, *11*(3). https://doi.org/10.1198/106186002411

Jiang, Z., Zheng, Y., Tan, H., Tang, B., & Zhou, H. (2017). Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. https://doi.org/10.24963/IJCAI.2017/273

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. *International Conference on Learning Representations (ICLR)*.

Lähnemann, D., Köster, J., Szczurek, E., McCarthy, D. J., Hicks, S. C., Robinson, M. D., Vallejos, C. A., Campbell, K. R., Beerenwinkel, N., & Mahfouz, A. (2020). Eleven grand challenges in single-cell data science. *Genome biology*, *21*(1).

Lee, D. B., Min, D., Lee, S., & Hwang, S. J. (2020). Meta-gmvae: Mixture of gaussian vae for unsupervised meta-learning. *International Conference on Learning Representations*.

Lopez, R., Li, B., Keren-Shaul, H., Boyeau, P., Kedmi, M., Pilzer, D., Jelinski, A., Yofe, I., David, E., Wagner, A., Ergen, C., Addadi, Y., Golani, O., Ronchese, F., Jordan, M. I., Amit, I., & Yosef, N. (2022). DestVI identifies continuums of cell types in spatial transcriptomics data. *Nature Biotechnology*, *40*(9). https://doi.org/10.1038/s41587-022-01272-8

Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., & Yosef, N. (2018). Deep generative modeling for single-cell transcriptomics. *Nature methods*, *15*(12).

Luecken, M. D., Büttner, M., Chaichoompu, K., Danese, A., Interlandi, M., Müller, M. F., Strobl, D. C., Zappia, L., Dugas, M., & Colomé-Tatché, M. (2022). Benchmarking atlas-level data integration in single-cell genomics. *Nature methods*, *19*(1).

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Nalisnick, E., Hertel, L., & Smyth, P. (2016). Approximate inference for deep latent gaussian mixtures. *Workshop on Bayesian Deep Learning, NeurIPS 2016*.

Nalisnick, E., & Smyth, P. (2017). Stick-Breaking Variational Autoencoders. *5th International Conference on Learning Representations, (ICLR) 2017*.

Ranganath, R., Gerrish, S., & Blei, D. (2014). Black box variational inference.

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models.

Shabestari, S. K., Morabito, S., Danhash, E. P., McQuade, A., Sanchez, J. R., Miyoshi, E., Chadarevian, J. P., Claes, C., Coburn, M. A., Hasselmann, J., Hidalgo, J., Tran, K. N., Martini, A. C., Rothermich, W. C., Pascual, J., Head, E., Hume, D. A., Pridans, C., Davtyan, H., ... Blurton-Jones, M. (2022). Absence of microglia promotes diverse pathologies and early lethality in Alzheimer's disease mice.

Cell reports, *39*(11). https://doi.org/10.1016/j.celrep.2022.110961

Stirn, A., Jebara, T., & Knowles, D. (2019). A New Distribution on the Simplex with Auto-Encoding Applications. *Advances in Neural Information Processing Systems*.

Svensson, V., Gayoso, A., Yosef, N., & Pachter, L. (2020). Interpretable factor models of single-cell RNA-seq via variational autoencoders. *Bioinformatics*, *36*(11).

Tomczak, J., & Welling, M. (2018). VAE with a VampPrior. *Proceedings of Machine Learning Research*.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, *9*(11).

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*.

Winter, V., Dinari, O., & Freifeld, O. (2022). Common Failure Modes of Subcluster-based Sampling in Dirichlet Process Gaussian Mixture Models–and a Deep-learning Solution. *arXiv preprint arXiv:2203.13661*.

Xu, C., Lopez, R., Mehlman, E., Regier, J., Jordan, M. I., & Yosef, N. (2021). Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Molecular systems biology*, *17*(1).

Yang, L., Fan, W., & Bouguila, N. (2020). Clustering analysis via deep generative models with mixture models. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(1).

Zhou, S., Xu, H., Zheng, Z., Chen, J., Li, Z., Bu, J., Wu, J., Wang, X., Zhu, W., & Ester, M. (2025). A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions. *ACM computing surveys*, *57*(3).

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. **Yes**
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. **Yes**
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. **Yes**

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. **Not Applicable**
   (b) Complete proofs of all theoretical results. **Not Applicable**
   (c) Clear explanations of any assumptions. **Not Applicable**

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). **Yes**
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). **Yes**
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). **Yes**
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). **Yes**

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. **Yes**
   (b) The license information of the assets, if applicable. **Not Applicable**
   (c) New assets either in the supplemental material or as a URL, if applicable. **Not Applicable**
   (d) Information about consent from data providers/curators. **Not Applicable**
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. **Not Applicable**

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. **Not Applicable**
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. **Not Applicable**
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. **Not Applicable**

The VampPrior Mixture Model: Supplementary Materials

A SOFTWARE AVAILABILITY AND COMPUTE INFRASTRUCTURE

Our code is available at https://github.com/astirn/VampPrior-Mixture-Model. We performed all experiments on a laptop with 16GB of RAM connected to an external Nvidia 3090 GPU.

B VMM OPTIMIZATION

Adapting algorithm 1 to the VMM results in the following procedure:

$$
\begin{aligned}
&\text{sample } x_1, \dots, x_M \sim \mathcal{D} \\
&\underset{\theta,\phi}{\text{maximize}} \sum_{i=1}^{M} \mathop{\mathbb{E}}_{q_\phi(z;x_i)} \left[\log p_\theta(x_i|z)\right] - \mathrm{KL}\left(q_\phi(z;x_i) \middle|\middle| \sum_{j=1}^{K} \pi_j \mathop{\mathbb{E}}_{q_\phi(\mu_j;u_j)} \left[\mathcal{N}(z|\mu_j, \Lambda_j^{-1})\right]\right) \\
&\text{sample } z_i \sim q_\phi(z;x_i)\ \forall i \in [M] \\
&\quad \text{set } q(c_i;z_i) \coloneqq \text{softmax}\left(\log \pi + \mathop{\mathbb{E}}_{q_\phi(\mu;u)} \left[\log \mathcal{N}(z_i|\mu,\Lambda)\right]\right)\ \forall i \in [M] \\
&\underset{\alpha,\pi,u,\Lambda}{\text{maximize}} \mathop{\mathbb{E}}_{q(c;z)q_\phi(\mu;u)} \Big[\log p(\alpha) + \log p(\pi|\alpha) + \log p(\mu) + \log p(\Lambda) + \log \mathcal{N}(z_i|\mu,\lambda)\Big],
\end{aligned}
$$

where $p(\alpha)$, $p(\pi|\alpha)$, $p(\mu)$, and $p(\Lambda)$ correspond to eqs. (5) to (8).

C VAE EXPERIMENTS

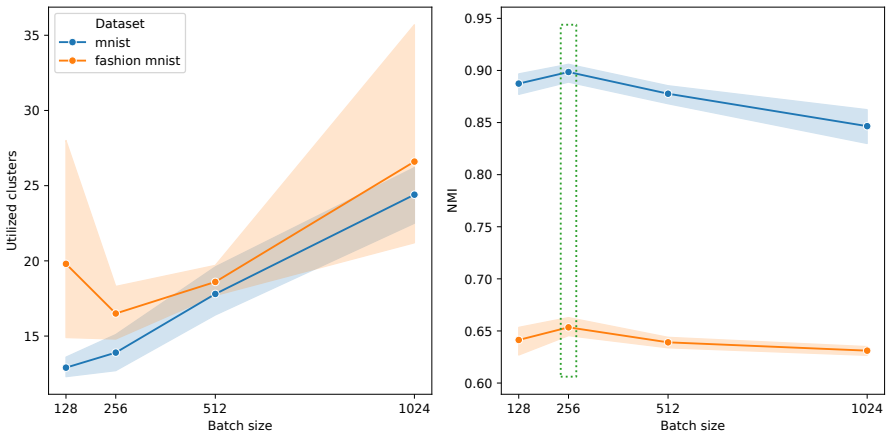


Figure 3: VMM cluster utilization and NMI performance for different batch sizes. Shading denotes 95% CIs. The dotted green rectangle marks the peak NMI performance between class labels and cluster assignments.

Section 3.2 discusses how the VMM's batch size setting serves as a simple proxy for adjusting the DP's concentration, avoiding the need to adjust hyper-prior parameters. Increasing the data available to the Empirical Bayes EM step in algorithm 1 weakens the effect of the hyper-priors resulting in more clusters being utilized. We employ this strategy for the MNIST and Fashion MNIST experiments in section 4. For various batch size settings, we compute how many clusters the VMM utilizes $|\{\forall\ i \in [N] : \arg\max_j\ q(c_i = j|x_i)\}|$ and

Table 4: scRNA-seq integration performance. We bold the best score for each column and any other scores whose $p \geq 0.05$ under a one-sided Mann-Whitney U test.

		Metric Type	Aggregate score			Batch correction					Bio conservation				
Dataset	Model	Prior	Batch correction	Bio conservation	Total	Graph connectivity	KBET	PCR comparison	Silhouette batch	iLISI	Isolated labels	KMeans ARI	KMeans NMI	Silhouette label	cLISI
cortex	scVI tools	$\mathcal{N}(0, I)$	–	0.696 ± 0.020	–	–	–	–	–	–	0.599 ± 0.002	0.641 ± 0.059	0.649 ± 0.040	0.595 ± 0.002	0.994 ± 0.001
	scVI (our code)	$\mathcal{N}(0, I)$	–	0.694 ± 0.019	–	–	–	–	–	–	0.604 ± 0.004	0.626 ± 0.050	0.650 ± 0.040	0.595 ± 0.004	0.994 ± 0.001
		VampPrior	–	**0.760 ± 0.007**	–	–	–	–	–	–	**0.661 ± 0.005**	**0.744 ± 0.028**	**0.731 ± 0.011**	**0.665 ± 0.004**	**0.999 ± 0.000**
		GMM	–	0.740 ± 0.007	–	–	–	–	–	–	0.638 ± 0.004	0.715 ± 0.021	0.709 ± 0.010	0.641 ± 0.004	0.998 ± 0.000
		VMM	–	**0.755 ± 0.007**	–	–	–	–	–	–	0.647 ± 0.004	**0.752 ± 0.023**	**0.725 ± 0.013**	0.652 ± 0.004	0.999 ± 0.000
	Harmony	N/A	–	0.645 ± 0.000	–	–	–	–	–	–	0.593 ± 0.000	0.480 ± 0.000	0.552 ± 0.000	0.605 ± 0.000	0.996 ± 0.000
	Scanorama	N/A	–	0.645 ± 0.000	–	–	–	–	–	–	0.593 ± 0.000	0.480 ± 0.000	0.552 ± 0.000	0.605 ± 0.000	0.996 ± 0.000
pbmc	scVI tools	$\mathcal{N}(0, I)$	0.860 ± 0.011	0.661 ± 0.020	0.741 ± 0.012	**0.893 ± 0.011**	0.885 ± 0.057	0.810 ± 0.042	**0.967 ± 0.003**	0.743 ± 0.006	0.572 ± 0.003	0.491 ± 0.062	0.659 ± 0.034	0.585 ± 0.005	0.999 ± 0.000
	scVI (our code)	$\mathcal{N}(0, I)$	0.840 ± 0.013	0.669 ± 0.016	0.737 ± 0.010	**0.895 ± 0.013**	0.855 ± 0.049	0.778 ± 0.054	0.959 ± 0.003	0.714 ± 0.007	0.575 ± 0.003	0.514 ± 0.051	0.670 ± 0.027	0.588 ± 0.003	0.999 ± 0.000
		VampPrior	**0.883 ± 0.007**	0.740 ± 0.014	**0.797 ± 0.008**	0.871 ± 0.024	0.916 ± 0.035	0.900 ± 0.024	0.963 ± 0.003	**0.766 ± 0.007**	**0.632 ± 0.006**	0.629 ± 0.051	0.753 ± 0.018	**0.686 ± 0.008**	**1.000 ± 0.000**
		GMM	0.866 ± 0.007	0.722 ± 0.016	0.779 ± 0.009	0.879 ± 0.015	0.919 ± 0.012	0.829 ± 0.026	0.959 ± 0.003	0.746 ± 0.005	0.611 ± 0.004	0.610 ± 0.057	0.728 ± 0.026	0.659 ± 0.004	**1.000 ± 0.000**
		VMM	**0.886 ± 0.009**	0.739 ± 0.011	**0.798 ± 0.007**	0.866 ± 0.011	0.909 ± 0.032	**0.937 ± 0.021**	0.961 ± 0.003	0.757 ± 0.006	0.614 ± 0.003	0.657 ± 0.034	0.754 ± 0.023	0.668 ± 0.007	**1.000 ± 0.000**
	Harmony	N/A	0.868 ± 0.001	0.734 ± 0.000	0.787 ± 0.001	0.738 ± 0.001	**0.965 ± 0.004**	0.908 ± 0.003	**0.967 ± 0.000**	0.761 ± 0.000	0.568 ± 0.000	0.764 ± 0.000	0.728 ± 0.000	0.609 ± 0.000	0.999 ± 0.000
	Scanorama	N/A	0.486 ± 0.007	**0.756 ± 0.009**	0.648 ± 0.005	0.373 ± 0.009	0.474 ± 0.029	0.695 ± 0.011	0.843 ± 0.001	0.043 ± 0.000	0.585 ± 0.000	**0.800 ± 0.025**	**0.770 ± 0.021**	0.627 ± 0.000	0.999 ± 0.000
split-seq	scVI tools	$\mathcal{N}(0, I)$	0.864 ± 0.005	0.598 ± 0.008	0.705 ± 0.005	**0.861 ± 0.005**	0.721 ± 0.020	0.963 ± 0.007	0.956 ± 0.002	0.822 ± 0.005	0.559 ± 0.003	0.317 ± 0.017	0.575 ± 0.017	0.541 ± 0.004	1.000 ± 0.000
	scVI (our code)	$\mathcal{N}(0, I)$	0.875 ± 0.006	0.612 ± 0.007	0.717 ± 0.004	**0.863 ± 0.002**	0.745 ± 0.023	**0.982 ± 0.003**	**0.959 ± 0.002**	0.825 ± 0.004	0.563 ± 0.003	0.347 ± 0.018	0.604 ± 0.012	0.545 ± 0.005	1.000 ± 0.000
		VampPrior	**0.876 ± 0.005**	0.638 ± 0.010	**0.733 ± 0.006**	0.855 ± 0.006	**0.773 ± 0.011**	0.953 ± 0.011	0.953 ± 0.003	**0.848 ± 0.003**	**0.577 ± 0.003**	0.375 ± 0.029	**0.647 ± 0.016**	**0.591 ± 0.004**	1.000 ± 0.000
		GMM	**0.876 ± 0.008**	0.631 ± 0.006	0.729 ± 0.006	0.858 ± 0.004	**0.766 ± 0.025**	0.965 ± 0.009	0.952 ± 0.003	0.839 ± 0.005	0.569 ± 0.005	0.375 ± 0.015	0.632 ± 0.013	0.580 ± 0.005	1.000 ± 0.000
		VMM	**0.878 ± 0.004**	0.639 ± 0.004	**0.734 ± 0.003**	0.855 ± 0.004	**0.773 ± 0.016**	0.960 ± 0.005	0.953 ± 0.002	**0.846 ± 0.004**	0.568 ± 0.003	0.389 ± 0.015	**0.648 ± 0.007**	**0.588 ± 0.004**	1.000 ± 0.000
	Harmony	N/A	0.790 ± 0.002	**0.650 ± 0.002**	0.706 ± 0.002	0.854 ± 0.004	0.606 ± 0.007	0.977 ± 0.001	0.949 ± 0.001	0.563 ± 0.004	0.568 ± 0.000	**0.443 ± 0.010**	**0.652 ± 0.000**	**0.589 ± 0.000**	**1.000 ± 0.000**
	Scanorama	N/A	0.552 ± 0.000	0.632 ± 0.000	0.600 ± 0.000	0.604 ± 0.000	0.294 ± 0.000	0.662 ± 0.000	0.858 ± 0.000	0.341 ± 0.000	0.568 ± 0.000	0.395 ± 0.000	0.619 ± 0.000	0.580 ± 0.000	1.000 ± 0.000
lung atlas	scVI tools	$\mathcal{N}(0, I)$	**0.620 ± 0.006**	0.608 ± 0.015	0.613 ± 0.010	0.845 ± 0.018	0.354 ± 0.015	0.928 ± 0.009	0.815 ± 0.012	0.157 ± 0.012	0.675 ± 0.022	0.319 ± 0.035	0.528 ± 0.025	0.537 ± 0.012	0.980 ± 0.007
	scVI (our code)	$\mathcal{N}(0, I)$	0.616 ± 0.006	0.600 ± 0.008	0.607 ± 0.006	0.840 ± 0.011	0.322 ± 0.011	**0.952 ± 0.009**	0.821 ± 0.012	0.145 ± 0.008	0.684 ± 0.022	0.289 ± 0.025	0.511 ± 0.009	0.537 ± 0.007	0.980 ± 0.003
		VampPrior	0.538 ± 0.014	**0.688 ± 0.015**	**0.628 ± 0.012**	0.823 ± 0.004	**0.530 ± 0.012**	0.449 ± 0.066	0.708 ± 0.010	**0.178 ± 0.004**	0.618 ± 0.035	**0.522 ± 0.041**	0.679 ± 0.020	**0.633 ± 0.013**	0.990 ± 0.003
		GMM	0.611 ± 0.007	0.628 ± 0.019	**0.621 ± 0.013**	**0.854 ± 0.010**	0.360 ± 0.011	0.888 ± 0.029	0.812 ± 0.009	0.141 ± 0.007	0.696 ± 0.035	0.341 ± 0.052	0.551 ± 0.032	0.564 ± 0.011	0.988 ± 0.002
		VMM	0.562 ± 0.012	0.674 ± 0.012	**0.629 ± 0.008**	**0.860 ± 0.011**	0.478 ± 0.020	0.544 ± 0.059	0.767 ± 0.008	0.160 ± 0.003	0.686 ± 0.031	0.461 ± 0.022	0.638 ± 0.013	0.599 ± 0.009	0.988 ± 0.002
	Harmony	N/A	0.553 ± 0.003	0.678 ± 0.002	**0.628 ± 0.002**	0.753 ± 0.004	**0.536 ± 0.010**	0.459 ± 0.004	**0.881 ± 0.001**	0.139 ± 0.000	0.577 ± 0.003	**0.536 ± 0.009**	**0.699 ± 0.005**	0.579 ± 0.001	0.996 ± 0.000
	Scanorama	N/A	0.328 ± 0.006	**0.694 ± 0.009**	0.548 ± 0.007	0.303 ± 0.021	0.337 ± 0.015	0.088 ± 0.007	0.830 ± 0.002	0.081 ± 0.001	**0.744 ± 0.004**	0.463 ± 0.032	**0.696 ± 0.010**	0.570 ± 0.003	**0.997 ± 0.000**

the NMI between class labels and cluster assignments $\arg\max_j q(c_i = j|x_i)$. Since MNIST and Fashion MNIST have the same number of true classes and approximately uniform class distributions, it is reassuring that batch size's effect on cluster utilization is similar (fig. 3, left) and that NMI peaks in both cases with a batch size of 256 (fig. 3, right), which we use for all VAE experiments in section 4.

We use Adam (Kingma & Ba, 2014) with a 1e-4 learning rate for both VI (γ_1, algorithm 1) and Empirical Bayes (γ_2, algorithm 1) steps. Stochastic gradient ascent's appearance in algorithm 1 is for illustrative purposes only. Following Jiang et al. (2017), we use ReLU activations for all hidden layers. For table 1, we allow all models to train for a maximum of 10,000 epochs, which no model comes close to using since we stop training early if the validation set's NMI has not improved for 100 epochs. For table 2, we allow all models to train for a maximum of 500 epochs, but stop training early if the validation set's ELBO has not improved for 20 epochs. In both cases, we restore the model's parameters from the epoch with the best validation set metric of interest. Please see our provided code for additional implementation details.

Figure 4 complements fig. 1 from the main text. Panel (a), shows the VMM's prior predictive samples for the Fashion MNIST dataset. Here, the VMM uses 13 clusters to represent 99.9% of the data, which has 10 ground truth classes. The VMM's most populous cluster (column 1) seemingly combines the 'pullover' and 'coat' classes into an outerwear cluster. Additionally, the VMM splits the 'bag' class into a handbags cluster (column 7) and shoulder bags (column 8) and the 'sandal' class into flat (column 9) and high-heeled (column 10) variants. The VMM also seems to partition graphic variants of the 'pull-over' and 't-shirt/top' classes into a separate clusters (columns 12 and 13). Not only is this clustering semantically justifiable, it is also best in class for Fashion MNIST (Yang et al., 2020).

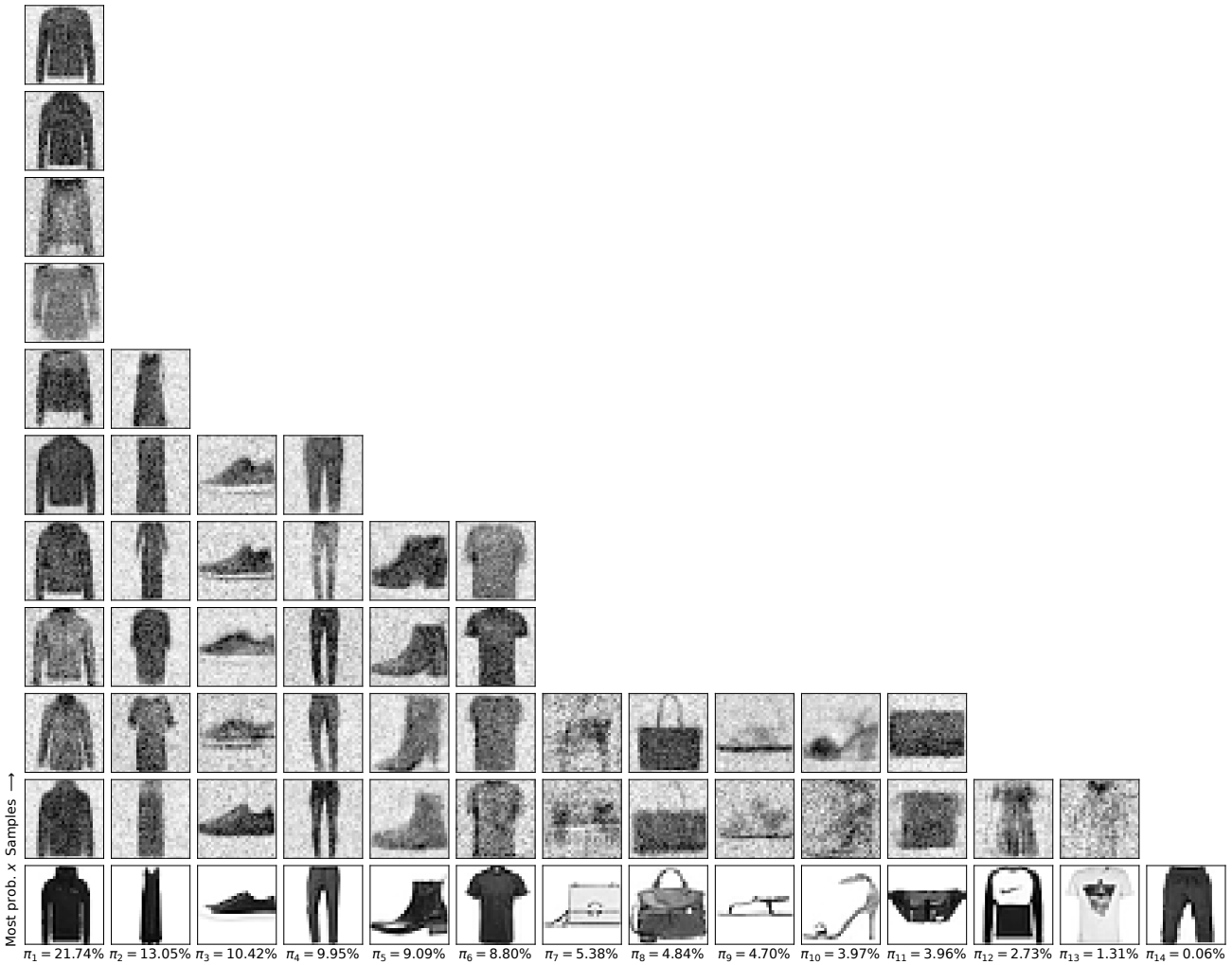
D SINGLE CELL EXPERIMENTS

Our scVI implementation is in TensorFlow, whereas the scVI tools version is part of their Python package and based in PyTorch. It was easier to build scVI with a VMM from scratch than from within the scvi-tools Python package given the complexity of the existing framework. We faithfully ensure that both implementations are equivalent (e.g. network architecture, optimizer hyperparameter, scVI options, etc.) hence the statistically equivalent performance ($p \geq 0.05$) in table 4.

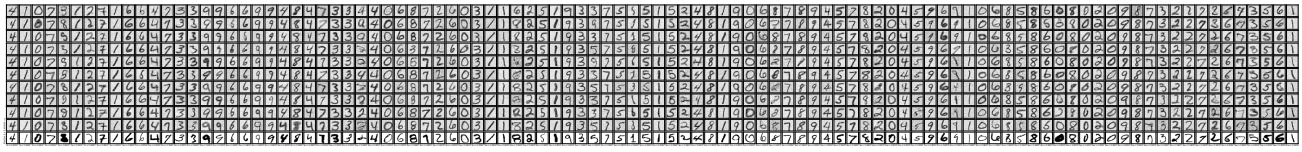
We use Lopez et al. (2018)′s recommended learning rates for scVI's variational inference steps. For the VMM, we tuned the batch size and Empirical Bayes learning rate (γ_2, algorithm 1) to optimize Luecken et al. (2022)′s aggregate total score; per section 3.2, this amounts to tuning the strength of the hyper-priors. For all scRNA-seq results, we set $K = 100$ as we did for the VAE and allow all models to train for a maximum of 10,000 epochs. Again, no model comes close to using this many epochs since we halt training early if the validation set's ELBO has not improved for 100 epochs. As before, we restore model parameters from the best epoch. Please see our provided code for additional details.

Table 4 reports Luecken et al. (2022)′s aggregate scores from table 3 alongside the scores composing the batch

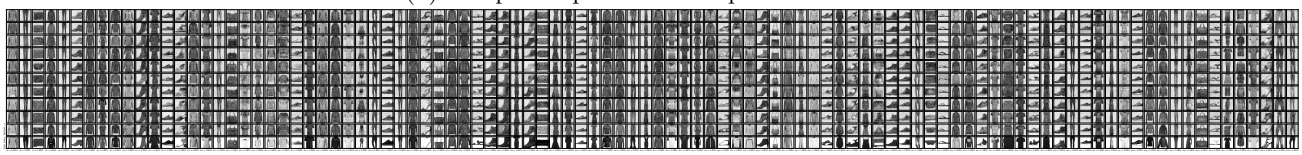
correction and the bio conservation aggregate scores. They rescale all scores to $[0, 1]$ such that larger values denote better performance. Please refer to their manuscript for further details.



(a) VMM prior predictive samples for Fashion MNIST



(b) VampPrior predictive samples for MNIST



(c) VampPrior predictive samples for Fashion MNIST

Figure 4: Prior predictive samples. The number of columns equals the number of utilized clusters. The bottom row shows the data with the highest probability of belonging to the cluster, under which we print the cluster's probability π_j. The rows above are samples from the corresponding cluster. We sample $\text{round}(10 \cdot \pi_j / \max(\pi))$ images from each component j to help visualize cluster proportions. No samples indicate this value rounded to zero. The VampPrior has uniform prior class probabilities since it specifies $\pi_j = K^{-1}$ and does not fit π during inference.

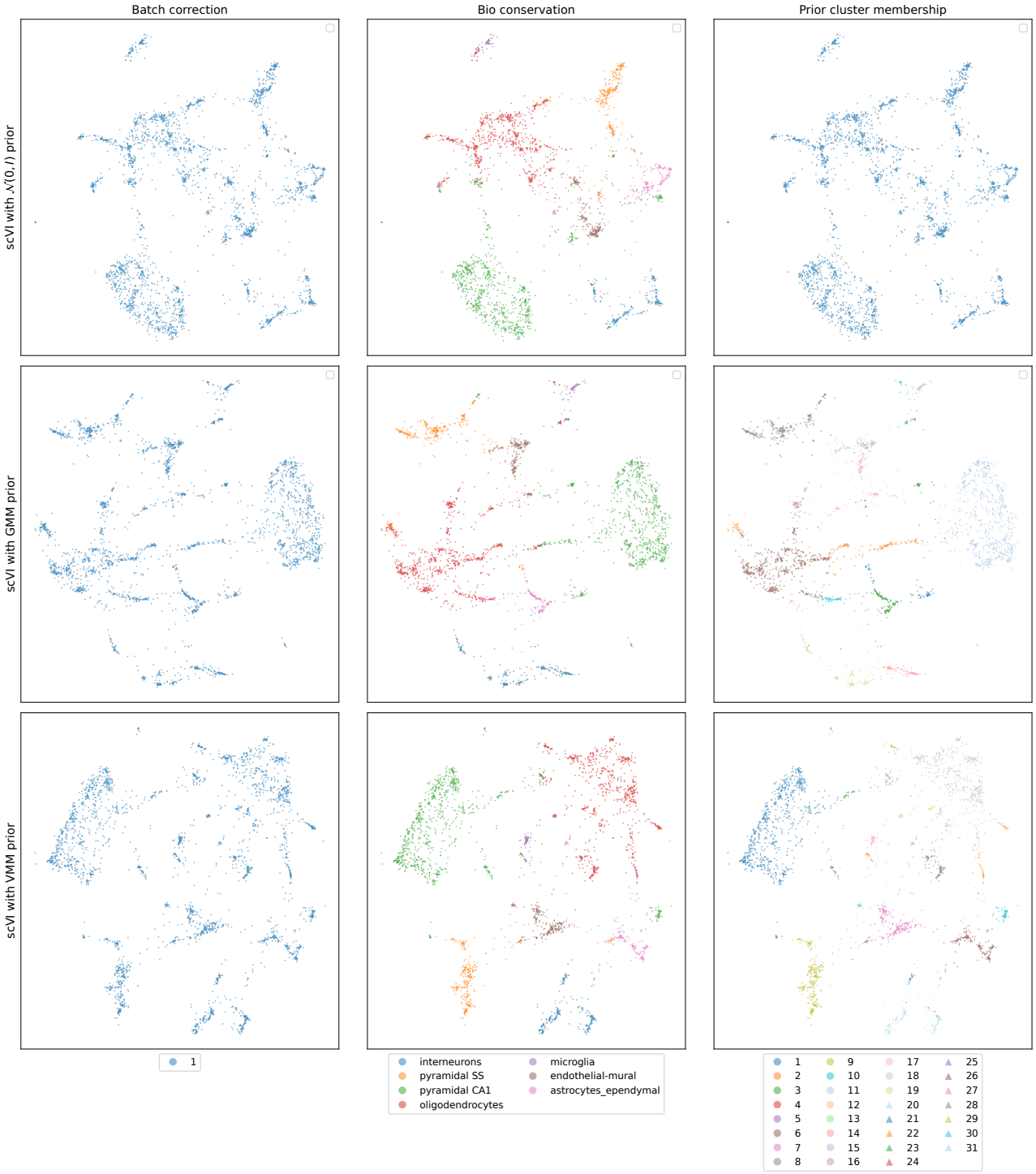


Figure 5: MDE comparison for the cortex dataset. Each row has the same embedding across columns for a tested prior. Columns respectively label points by the technical batch identifier, the annotated cell type, and the prior's cluster assignment.

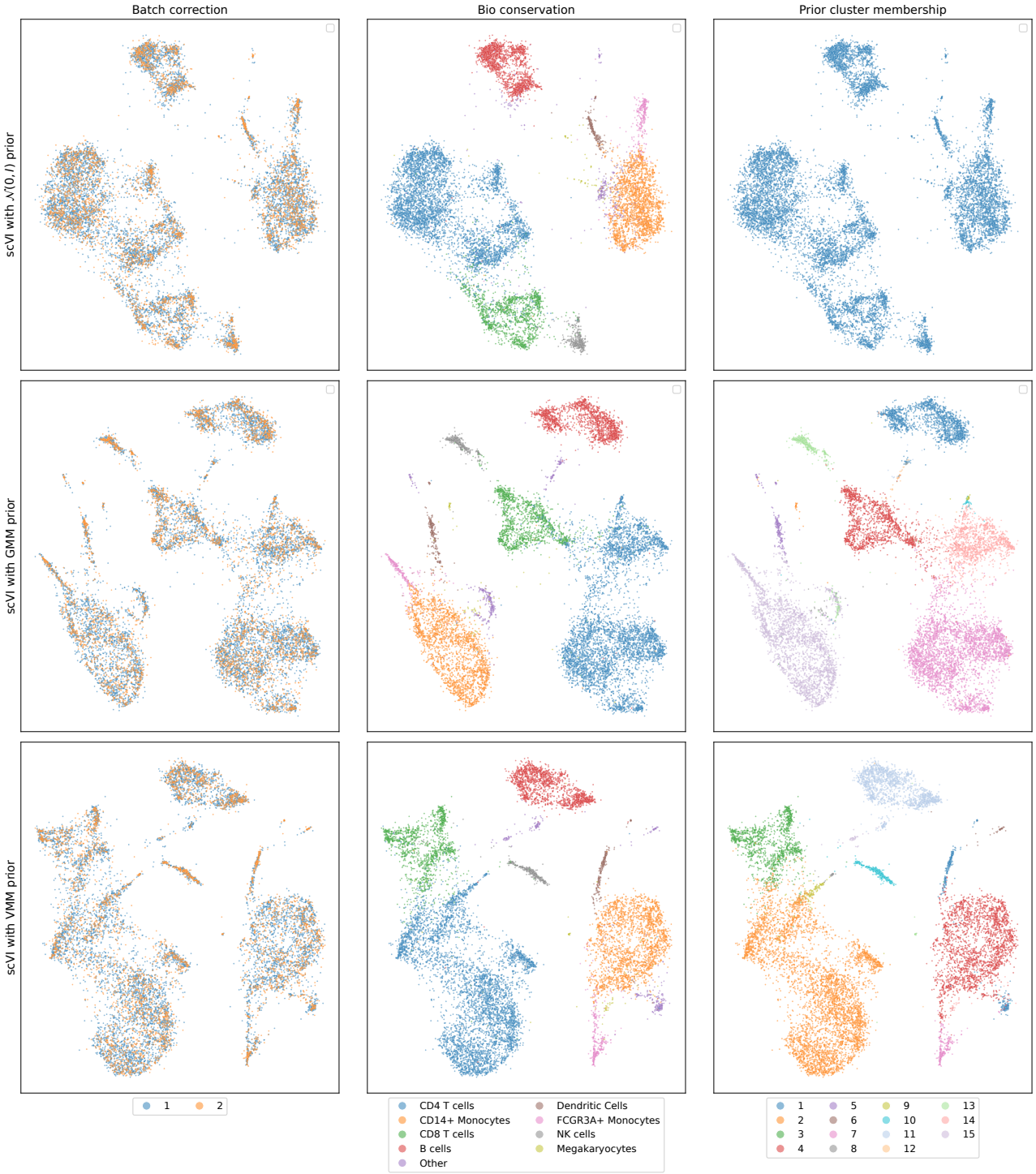


Figure 6: MDE comparison for the PBMC dataset. Each row has the same embedding across columns for a tested prior. Columns respectively label points by the technical batch identifier, the annotated cell type, and the prior's cluster assignment.

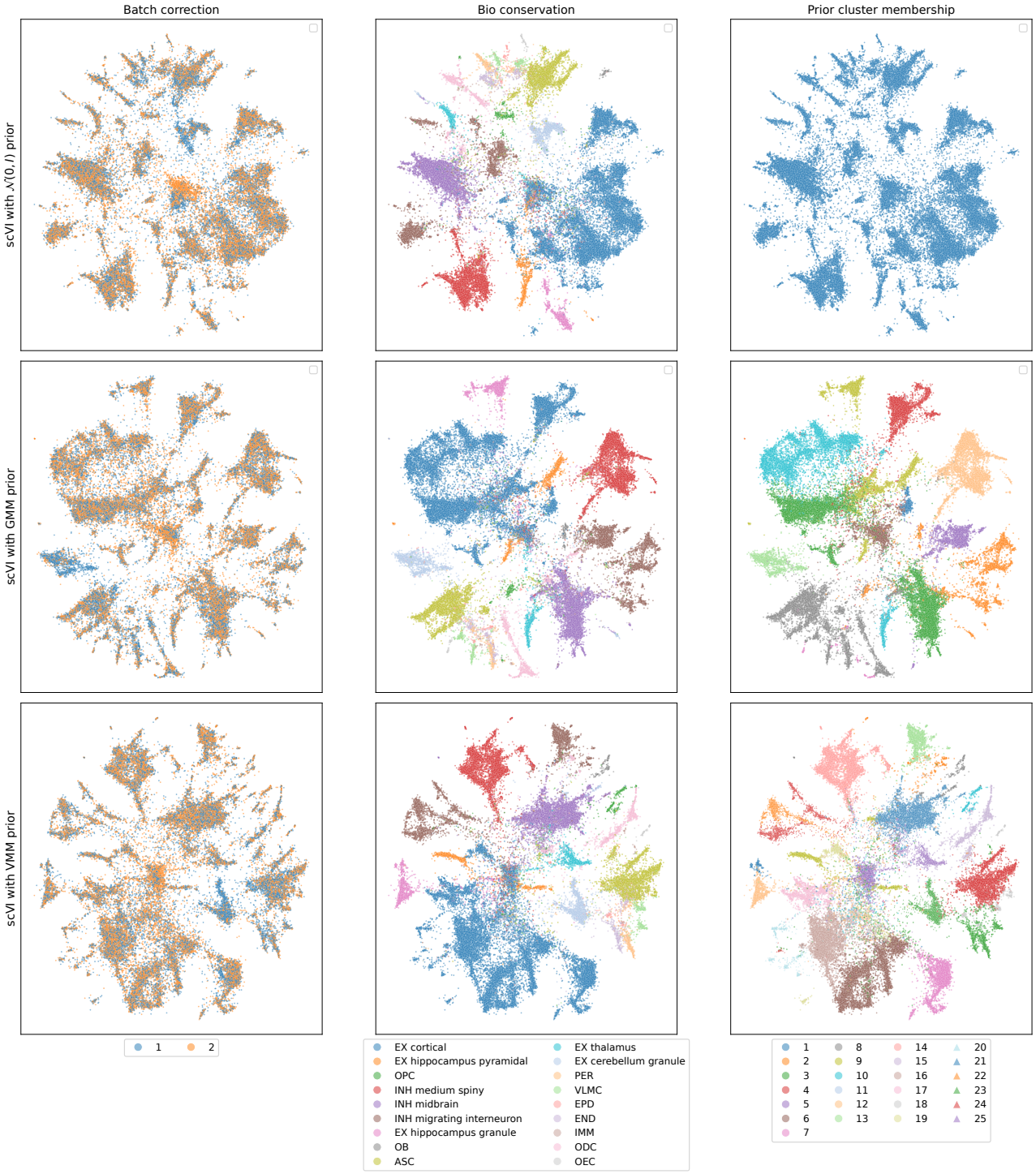


Figure 7: MDE comparison for the split-seq dataset. Each row has the same embedding across columns for a tested prior. Columns respectively label points by the technical batch identifier, the annotated cell type, and the prior's cluster assignment.