

VAEs with Learnable Priors for Learning a Robust Latent Space Representation of Single-Cell RNA Sequencing Data

Bachelor's Thesis

Leander Diaz-Bone

ldiazbone@ethz.ch

Computational Cancer Genomics
ETH Zürich

Supervisors:
Florian Barkmann,
Prof. Valentina Boeva

August 14, 2023

Abstract

In recent years, single-cell RNA sequencing data analysis has allowed to gain insides into many mechanisms of diseases on a cellular level. Analyzing this data has been complicated by so-called batch effects, non-biological factors influencing the experimental readout. Removing these effects is crucial in order to analyze the biological signals in the data.

One method used successfully to achieve this, is using Variational Autoencoders learning a latent representation of the data. These models have been shown to have limitations using the fixed Standard Normal prior, while learnable priors have been identified as a promising approach to mitigate to these problems.

The main goal of this work was to investigate the potential of learnable priors to improve the latent space learned by Variational Autoencoders with regard to batch integration and biological variance conservation. In order to do this, we implemented the Mixture of Gaussians, Vamp, and Normal Flow priors for the scVI model and benchmarked the learned latent space. In addition to the main goal, we investigated the behavior of the latent space during the training and the impact of scaling of the KL-term and prior parameters on benchmarks.

All in all, our work demonstrated that the learnable priors achieve better batch integration while achieving similar performance on biological variance conservation compared to the Standard Normal prior. Additionally, the learnable priors were shown to enable the model to improve batch integration continuously during training. Furthermore, we were able to utilize the number of epochs trained and the scaling of the KL-term to trade off biological variation conservation and batch integration. Overall, we can conclude that learnable priors, in particular, the Mixture of Gaussians prior can significantly improve the learned latent space and should be further investigated to further single-cell RNA sequencing data analysis.

Our implementation of the priors is available at
`https://github.com/LeanderDiazBone/scvi-tools`

All implementations of the experiments are available at
`https://github.com/LeanderDiazBone/BachelorThesis`

Contents

1 Introduction

Single-cell RNA sequencing (scRNA-seq) has been making important contributions to many significant fields of research over the last few years. These fields include cancer research [PTT+14, SLR+21], where investigating the tumor at a single-cell level impacted the understanding of important processes in the disease. Furthermore, scRNA-seq has found applications in autoimmune research [ZYZ+22], where it helped understanding the mechanism of disease occurrence and development. In general, scRNA-seq helps to understand how diseases develop and explains events on a single-cell level, which can lead to valuable insights into the underlying biological mechanisms. It is worthwhile improving the scRNA-seq data analysis pipeline itself, as it may improve results leading towards a deeper understanding of diseases and biology.

High-throughput sequencing, allowing sequencing of a vast amount of cells in a short amount of time, made many of these breakthroughs possible [DLW22]. On the other hand, it presents problems such as batch effects [SSZ+17] or dropout in the data [RZLA20]. Preprocessing the data by learning a lower-dimensional representation, referred to as latent representation, has been a focus of recent research [XWY+21]. The goal of the latent representation is to reduce noise while conserving biological meaningful signals.

One of the most recent approaches for learning a latent representation of scRNA-seq data is using Variational Autoencoders [Lop18]. These are generative machine learning models learning the distribution of the data, using a latent representation. The machine learning literature has found that the Standard Normal prior, which is the usual choice for these models has problems and might not be optimal for learning a meaningful latent space [RV18]. These problems might be mitigated by using learnable priors, as recent papers suggest [TW18, DKFT20].

Our main contribution to the field of scRNA-seq data analysis is the implementation and evaluation of different learnable priors for Variational Autoencoders. Furthermore, we will benchmark the learned latent representation on different metrics related to batch integration and conservation of biological variation.

We will begin by defining what scRNA-seq data is, the problems that arise during sequencing, and the goals of the tasks we will be performing. Afterwards, we will explain Variational Autoencoders with a focus on the role of the prior. Lastly, we will introduce the research that has already been conducted on using Variational Autoencoders for scRNA-seq data analysis.

1.1 Single-cell RNA Sequencing Data Analysis

In this section, we explain what we refer to when speaking about single-cell RNA sequencing data and what issues remain in the data after sequencing.

1.1.1 Single-cell RNA Sequencing Data

Single-cell RNA sequencing (scRNA-seq) data refers to the number of times each gene is expressed in each cell. This data is captured in the gene expression matrix containing genes in the rows and cells in the columns. The entry (i, j) of the expression matrix represents how many times the gene i is expressed in the cell j [ea23a]. The expression matrix is the result of sequencing and preprocessing and will be the starting point of our method. Below is a visualization of an anndata [VRT+21] object. Anndata is a library implementing storage of annotated expression matrices and will be the format we use for data storage.

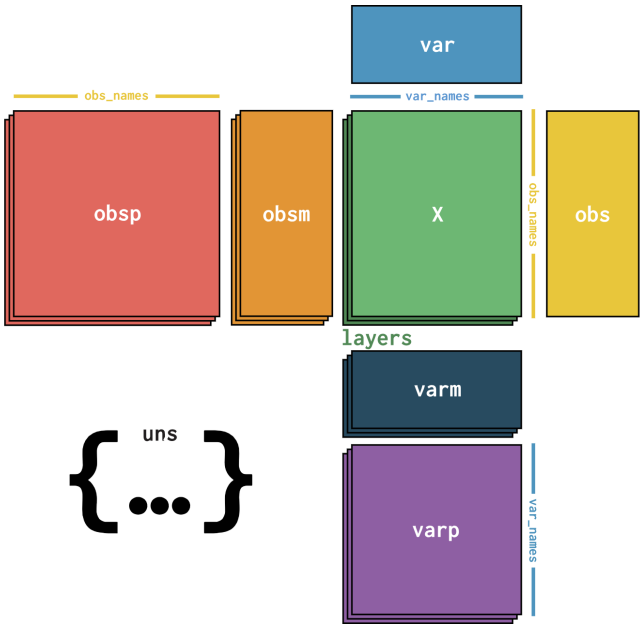


Figure 1: Annotaded scRNA data [VRT+21].

Storing the gene expression of each cell has the advantage that we can learn about the changes that happen in different cell populations.

1.1.2 Issues and Preprocessing in scRNA-seq Data

High-throughput sequencing provides a large number of cells that we can analyze, but there remain issues in the data such as low-quality reads, dropout, and batch

effects [ea23a]. Low-quality reads refer to measurements that don't provide the biological signal from an intact cell, which downstream tasks are expecting. There are many causes for it such as cells, which are dying, empty droplets, doublets, or ambient mRNA captured in a droplet. These have to be filtered during Quality Control. Dropout refers to genes being expressed in one cell, but not in other cells of the same cell type [Qiu20]. These are due to a low amount of mRNA in individual cells, that makes the expression matrics highly sparse. This issue can be addressed by highly-variable gene selection and dimensionality reduction, which we will explain in later sections in detail.

Batch effects are one major issue, complicating downstream analysis. Batch effects refer to the difference in the distribution of measured expression levels caused by handling cells in different subsets (batches) [ea23a]. These can be caused by different measuring conditions, such as environmental factors or differences in instruments, but there are many more causes. To remove these batch effects by performing batch integration is crucial to enable the joint analysis of different batches, as many tools and statistical models assume a common structure. During batch integration, a trade-off is made. Integrating batches more can lead to a loss of biological variation [ea23a]. When we are talking about biological variation we are referring to biological properties of the cell, which influence the expression of genes. There are many ways to perform batch integration, but we will focus on dimensionality reduction using Variational Autoencoders. How these models work will be discussed in the following section.

1.2 Variational Autoencoders and their Priors

Variational Autoencoders (VAEs) [KW22] are unsupervised, generative machine learning models, that learn a latent representation of the data. VAEs were first introduced by Kingma and Welling [KW22] and have found applications in many fields such as Natural Language Processing [BVV+16] or Vision [Chi21]. We will introduce an application in biology in the next section in more depth.

In this section, we explain how VAEs work, while emphasizing the role of the prior in the model.

1.2.1 Variational Autoencoder Training

The goal of a generative model is to learn the distribution $p(x)$ of the data. The idea for learning this distribution in the VAE is to introduce a lower-dimensional latent variable z, which is supposed to capture hidden features in the data [Tom23b]. Intuitively, the assumption is that a data point x can be explained by different factors, which can be captured in a lower-dimensional space. The joint distribution of the data and the latent variables $p(x, z)$ can be factorized as follows: $p(x, z) = p(x|z)p(z)$. Since we can only use x for the training, we need to marginalize out the latent variable.

$$p(x) = \int p(x, z)\, dz = \int p(x|z)p(z)\, dz$$

To compute this integral, which is untractable in the general case, variational inference can be used. This is done by considering the amortized variational posterior $\{q_\phi(z|x)\}_\phi$ parameterized by ϕ. The log-likelihood can be computed with respect to the variational posterior. Before deriving the log-likelihood equation let us introduce the Kullback-Leibler Divergence [SK51] and the empirical distribution [TW18], which we will use in the derivation.

Definition 1.1. Let P and Q be two probability measures on a measurable space X and let P be absolutely continuous with respect to Q. The Kullback-Leibler (KL) Divergence of P and Q is defined as

$$D_{KL}(P||Q) = \int P(x) \ln \frac{P(x)}{Q(x)} dx$$

Definition 1.2. Let $X = \{x_1, ..., x_n\}$ be a dataset of n data points. The empirical distribution $q(x)$ over the data is defined as

$$q(x) = \frac{1}{N} \sum_{i=1}^{N} \delta(x - x_i)$$

where δ is the Kronecker delta.

Using these definitions we can derive the evidence lower bound (ELBO) of the log-likelihood [KW22], which can be optimized to learn the distribution $p(x)$.

$$
\begin{aligned}
\mathbb{E}_{x\sim q(x)}[\ln p(x)] &= \mathbb{E}_{x\sim q(x)}[\int q_\phi(z|x) \ln p(x)\ dz] \\
&= \mathbb{E}_{x\sim q(x)}[\int q_\phi(z|x) \ln \frac{p(x|z)p_\lambda(z)}{p(z|x)}\ dz] \\
&= \mathbb{E}_{x\sim q(x)}[\int q_\phi(z|x) \ln \frac{p(x|z)p_\lambda(z)q_\phi(z|x)}{p(z|x)q_\phi(z|x)}\ dz] \\
&= \mathbb{E}_{x\sim q(x)}[\int q_\phi(z|x) \ln p(x|z)\ dz - \int q_\phi(z|x) \ln \frac{q_\phi(z|x)}{p_\lambda(z)}\ dz \\
&+ \int q_\phi(z|x) \ln \frac{q_\phi(z|x)}{p(z|x)}\ dz] \\
&= \mathbb{E}_{x\sim q(x)}[\mathbb{E}_{z\sim q_\phi(z|x)}[\ln p(x|z)] - D_{KL}(q_\phi(z|x)||p_\lambda(z)) + D_{KL}(q_\phi(z|x)||p(z|x))] \\
&\geq \mathbb{E}_{x\sim q(x)}[\mathbb{E}_{z\sim q_\phi(z|x)}[\ln p(x|z)] - D_{KL}(q_\phi(z|x)||p_\lambda(z))]
\end{aligned}
$$

To learn the distribution $\ln p(x)$ we need to parameterize the variational posterior $q_\phi(z|x)$ and the conditional distribution $p_\theta(x|z)$, this is done in the VAE using two Neural Networks, the Encoder and the Decoder Network. The usual choice for the variational posterior is a k-dimensional normal distribution, where k is the dimension of the latent variable and the conditional distribution depends on the data type, e.g. a categorical distribution for images.

In a training step of the VAE, x is transformed by the Encoder Network into the parameters of the variational posterior distribution $q_\phi(z|x)$. Now the latent variable z is sampled from the distribution $q_\phi(z|x)$ and decoded using the Decoder Network to obtain the parameters θ of the conditional distribution $p_\theta(x|z)$. Lastly, the gradients of the loss function (ELBO) can be calculated and all Neural Networks and prior parameters are updated accordingly. A pictorial description of the process is shown below.

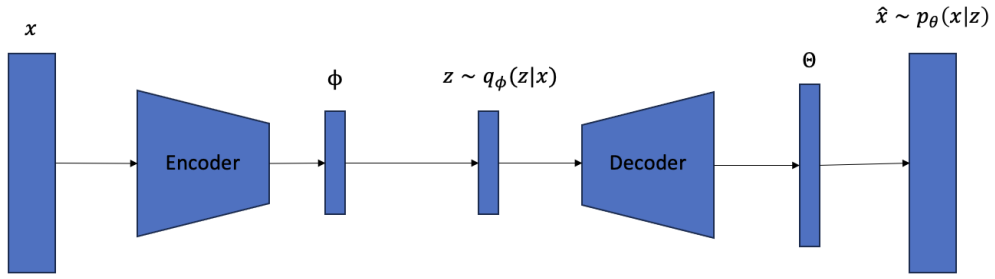


Figure 2: The general architecture of a Variational Autoencoder.

1.2.2 Variational Autoencoder Data Generation

To generate new data, we first sample z_g from the prior $p_\lambda(z)$. Next, we decode the latent variable z_g using the Decoder to obtain the parameters θ of the conditional distribution $p_\theta(x|z_g)$. Afterward, we can sample x_g from the conditional distribution to generate a new data point. The process looks pictorially as follows.

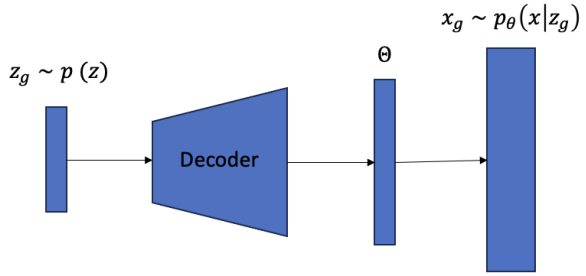


Figure 3: Data generation in a Variational Autoencoder.

1.2.3 The Role of the Prior

If we investigate the ELBO in more depth we can see that the first part of the ELBO can be intuitively seen as the reconstruction error and the second part, as the difference between the aggregated posterior and the prior [Tom23a], we will derive the second part in the following. Let us first define the aggregated posterior.

Definition 1.3. Let $X = \{x_1, ..., x_n\}$ be a dataset of n data points and let $q_\phi(z|x_i)$ denote the posterior distribution with respect to the data point x_i. The aggregated posterior is defined as

$$q_\phi(z) = \frac{1}{N} \sum_{i=1}^{N} q_\phi(z|x_n)$$

With this definition, we can now derive the previously explained intuition [Tom23a].

$$
\begin{aligned}
\mathbb{E}_{x\sim p(x)}[D_{KL}(q_\phi(z|x)||p_\lambda(z))] &= \mathbb{E}_{x\sim p(x)}[\mathbb{E}_{z\sim q_\phi(z|x)}[\ln q_\phi(z|x) - \ln p_\lambda(z)]] \\
&= \int\int p(x)q_\phi(z|x)(\ln p_\lambda(z) - \ln q_\phi(z|x))dzdx \\
&= \int\int \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)q_\phi(z|x)(\ln p_\lambda(z) - \ln q_\phi(z|x))dzdx \\
&= \int \frac{1}{N}\sum_{i=1}^{N} q_\phi(z|x_i)(\ln p_\lambda(z) - \ln q_\phi(z|x_i))dz \\
&= \int q_\phi(z)\ln p_\lambda(z)dz - \int \ln q_\phi(z|x)\frac{1}{N}\sum_{i=1}^{N} q_\phi(z|x_i)dz \\
&= -C\mathbb{E}[q_\phi(z)||p_\lambda(z)] + H[q_\phi(z|x)]
\end{aligned}
$$

Minimizing the cross entropy $C\mathbb{E}[q_\phi(z)||p_\lambda(z)]$ corresponds to matching the aggregated posterior and the prior [Tom23a]. On the other hand, the entropy $H[q_\phi(z|x)]$ is maximized for a Normal posterior when the variance is maximized, but as we are also optimizing the reconstruction error, which tries to make the variation posterior as peaky as possible, this does not happen. Therefore, maximizing the ELBO corresponds to minimizing the reconstruction error and maximizing the matching of the prior and aggregated posterior. Noteworthy, the difference between the ELBO and the actual log-likelihood is given by $D_{KL}(q_\phi(z|x)||p(z|x))$, which intuitively corresponds to the similarity of the variational posterior and the actual posterior. As the actual posterior is not known we can't compute the log-likelihood, but need to optimize the ELBO instead.

The standard choice for the prior is the Standard Normal prior: $\mathcal{N}(0, I)$ [Lop18, DKFT20]. Its advantage is that the KL divergence has a closed-form solution for the Standard Normal prior, but recent work has shown that this prior is not optimal [RV18]. This is the case, as the prior is static and forces the aggregated posterior to match the Standard Normal distribution. This is not perfectly possible, as the aggregated posterior also needs to reconstruct the data well. This leads to the hole problem [RV18]. Holes are regions where the prior assigns high probability, while the aggregated posterior assigns low probability. This can lead to poor generative performance. Additionally, the reconstruction of the data might be worse, as the Standard Normal prior forces the data to be concentrated close to the origin in the latent space.

These problems suggest that a flexible and learnable prior might be able to fit the aggregated posterior better and does not force it to concentrate close to the origin, which might benefit reconstruction [TW18, Tom23a]. Therefore, in this thesis we will investigate if different learnable priors will perform better than the Standard Normal prior. These learnable priors will be explained in depth in the Methods section.

1.3 VAEs for scRNA-seq Data Analysis

Variational Autoencoders learn a latent representation of the data, which is supposed to capture important underlying, hidden features of the data that are able to explain the data well. This is very useful for scRNA-seq data analysis, as the latent representation is potentially less noisy than the original data and captures important biological signals [Lop18]. These representations can then be used for downstream analysis such as clustering or differential expression analysis.

In this section, we will present the scVI model, which is a VAE used for scRNA-seq data. Additionally, we will present the research that has already been conducted on the use of learnable priors, for learning a latent space, for scRNA-seq data.

1.3.1 The scVI Model

The scVI model introduced by Lopez et al. [Lop18] is based on a Variational Autoencoder and learns a latent representation of scRNA-seq data with the goals of efficiency, batch integration, and conservation of biological variation. The evaluation of the model showed that it was able to improve different benchmarks such as differential expression and retaining biological signals in the latent space. It is a widely used model with a publicly available implementation in the `scvi-tools` library [ea23b]. This library can also be used to apply the model conveniently on many different analysis tasks.

The main differences between the scVI model and a standard VAE are the following: the scVI model, models library size and batch effects explicitly. The batch effects are modeled by giving the model the batch id, additionally to the expression vector of the cell. The intention is that the model can learn the batch effect, such that it won't be captured in the latent space. scVI models the conditional distribution as a zero-inflated negative binomial (zinb) distribution, which has been shown to work better than just using a negative binomial distribution for scRNA-seq data [RPG+18]. Below there is a visualization of the scVI model.

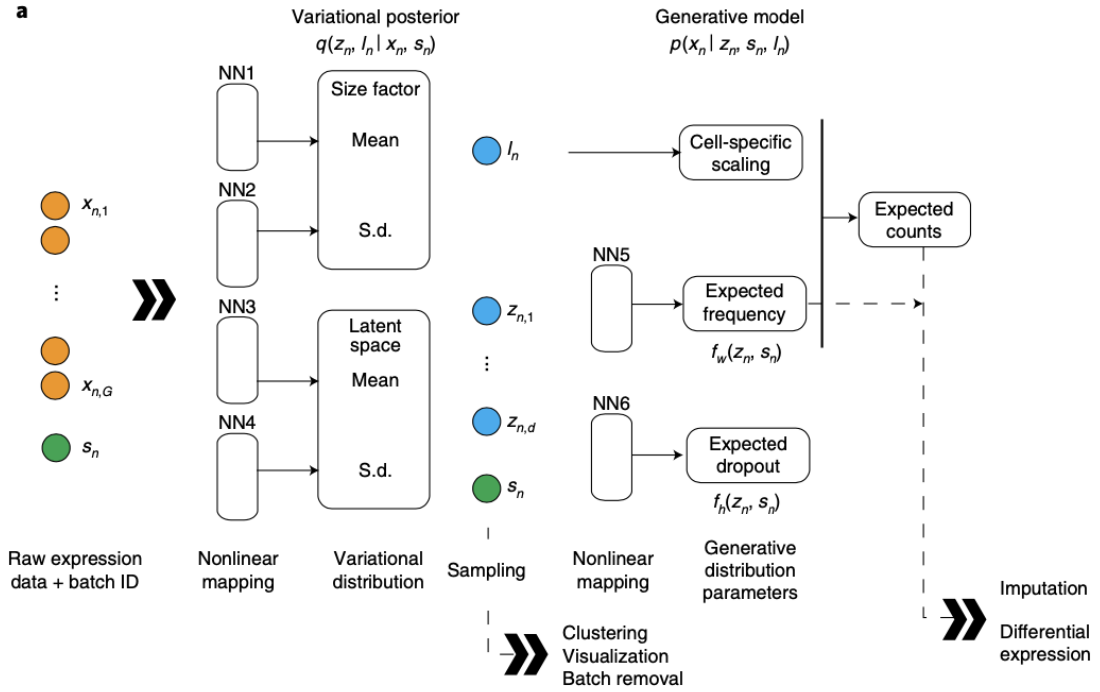


Figure 4: The scVI model [Lop18].

We will build up our implementation of the different priors on the scVI model, as scVI is based on a VAE, is widely used, performs well on many analysis tasks, and has a publicly available, easily extendable code base. We will explain how we extend the library in the methods section.

1.3.2 VAEs with Learnable Priors for scRNA-seq Data

The authors Dony et al. investigate the effects of using flexible priors for learning a robust latent space for scRNA-seq data [DKFT20]. For the flexible priors, they chose the Vamp prior [TW18] and an inverse autoregressive flow prior. The authors compared these models to a VAE with the Standard Normal prior and were able to show that flexible priors performed better on different evaluation tasks.

Firstly, they showed that the VAEs with flexible priors achieve lower reconstruction loss compared to the VAE with the Standard Normal prior. Secondly, they showed that the Vamp prior was able to capture more biological variation in the latent space. They do this by showing a UMAP [MHM20] of the learned latent space, which showed that the VAMP prior separated cells much better, based on their cell type in the latent space, compared to the VAEs using the Flow and the Standard Normal prior. Furthermore, the authors investigated the inactive latent unit problem, which was shown for VAEs using a Standard Normal prior [BGS16]. They find that the VAEs with more flexible priors didn't exhibit this problem. Lastly, to measure the conservation of biological variation in the latent space they used the average

silhouette width [Rou87], measuring how well the same cell type groups together in the latent space. The best result for this metric was achieved for the Vamp prior. All these results indicate that it is useful to further investigate flexible, learnable priors to learn a robust latent space, which will be the focus of this work.

2 Method

In this section, we will describe the priors we use and how they are implemented. Afterwards, we will discuss the benchmark and the data, which we will use for model evaluation. Finally, we will describe how the training is performed and how the experiments we conducted were designed.

2.1 Priors

As already explained in the Introduction, using more flexible, learnable priors might improve the latent representation of the scRNA-seq data with respect to biological variation conservation as well as the reduction of batch effects. In the following, we will introduce the three different priors we use, explain how they work, and how we implemented them inside the `scvi-tools` library [ea23b]. All the priors are implemented using the `pytorch.distributions` library [PGM+19]. Using this library is very convenient, as it provides a `sample` and a `log_prob` function.

2.1.1 Standard Normal Prior

The Standard Normal prior is the simplest prior and has the advantage that we have a closed-form solution for the KL divergence [ZLC+23], which the other priors do not have. We introduce it first, as it is the simplest and most widely used prior for VAEs.

Definition 2.1. Let $z \in \mathbb{R}^k$ be the latent variable. The Standard Normal prior over the latent variable z is defined as

$$p_{SD}(z) = \mathcal{N}(0, I_k)$$

As both the mean and the variance of the prior are fixed, there is nothing to learn. The Standard Normal prior is implemented using the predefined Normal distribution in the `pytorch.distributions` library.

2.1.2 Mixture of Gaussians Prior

The Mixture of Gaussians prior is a simple first choice for a flexible, learnable prior, as it has a simple structure and contains learnable parameters. Furthermore, it makes sense to model the prior as a Mixture of Gaussians, as the choice for the posterior $q_\phi(z|x)$ is a Gaussian distribution and the goal of the prior is matching the aggregated posterior.

Definition 2.2. Let $z \in \mathbb{R}^k$ be the latent variable, $\mu_i \in \mathbb{R}^k, \sigma_i \in \mathbb{R}^k_+$ means and variances and w a Categorical distribution. The Mixture of Gaussians prior over the latent variable z is defined as

$$p_{MG}(z) = \sum_{i=1}^{d} w_i \mathcal{N}(\mu_i, diag(\sigma_i^2))$$

The Mixture of Gaussians prior has three learnable components. The categorical distribution w, the means, and the standard deviations of the Normal distributions. We define all these components as `torch.nn.Parameters` such that they will be updated during the gradient steps. For the categorical distribution, we learn a vector of weights and use the softmax function to turn the weights into probabilities for the `pytorch.distributions` Categorical distribution. The Normal distributions are defined as before by using the internal Normal distribution with the learned means and variances. The mixture distribution can be defined using the MixtureSameFamility distribution in the library.

2.1.3 Vamp Prior

The main idea of the Vamp prior, first introduced by Tomczak and Welling [TW18], is that the prior should approximate the aggregated posterior. Hence the authors define the Vamp Prior as a mixture of posteriors, which corresponds to the aggregated posterior. Instead of summing over all data points, which would be too expensive during training [TW18], we sum over a constant number d of components and learn pseudo-inputs x_i.

Definition 2.3. Let $z \in \mathbb{R}^k$ be the latent variable, x_i for $i \in \{1, .., d\}$ pseudo-inputs, and w a Categorical distribution. The Vamp prior over the latent variable z is defined as

$$p_{VP}(z) = \sum_{i=1}^{d} w_i q_\phi(z|x_i)$$

The implementation of the Vamp prior is more complex than the implementation of the Mixture of Gaussians prior, as we have to determine how the pseudo-inputs are learned. This is not as simple as learning the means and variances of the Mixture of Gaussians prior, as the pseudo-inputs are supposed to have a similar structure as the actual data points. For scRNA-seq data the input data is count data. This is difficult to learn as it is discrete. To avoid having to learn count data we instead learn the $log(x+1)$ transformed count data. And instead of learning pseudo-inputs directly, we parameterize them using a Neural Network, which learns to transform the Identity matrix into the pseudo-inputs. This pseudo-transformer uses two linear layers and ReLU activation functions after each layer. The learned pseudo-inputs are then transformed by the encoder of the VAE to obtain the posterior distributions

$q_\phi(z|x_i)$. These are then combined in the same manner as the Normal distributions in the Mixture of Gaussians prior.

2.1.4 Normal Flow Prior

Normal Flow priors for variational inference introduced by Rezende and Mohamed [RM16] are another way to implement a learnable prior. Their advantage is that they are able to model almost any distribution through consecutive transformations and therefore might be able to approximate the aggregated posterior well. On the other hand, this makes them the most complex to learn.

Definition 2.4. Let $z \in \mathbb{R}^D$ be a random variable with density $q(z)$ and $f_i : \mathbb{R}^D \to \mathbb{R}^D$ for $i \in \{1, .., k\}$ invertable functions, with inverses g_i. We obtain the density function $q_k(z_k)$, by applying f to z with $f = f_1 \circ ... \circ f_k$. The path formed by the successive distributions q_k is a normalizing flow. The Normal Flow prior over the latent variable z is defined as

$$p_{NF}(z) = q_k(z_k)$$

Theorem 2.1. The logarithm of the density $q_k(z_k)$ is given by:

$$\ln q_k(z_k) = \ln q_0(z_0) - \sum_{i=1}^{k} \ln |\det \frac{\partial f_i}{\partial z_{i-1}}|$$

To implement the Normal Flow prior we use the `normflows` library [SLC+23], which offers a simple way to piece together the distributions and functions. Now we only have to choose how we want to define the base distribution $q(z)$ and the functions f_i for transforming the distribution. For the base distribution, we choose a Normal distribution with a diagonal covariance matrix and learnable mean and variance. We leave the parametrization of the functions f_i as an input to the prior and as the default, we choose a maximum likelihood predictor Neural Networks with two hidden layers, with 64 units each, and use this for every layer.

2.2 Benchmark

We want to evaluate the latent space, which our model learns in two domains: Firstly, how well batch integration is performed, and secondly, how well biological variance is conserved in the latent representation. For the evaluation, we use the benchmark designed by Luecken et al. [LBC+22], as its intention is to capture exactly these two properties. Additionally, the benchmark is provided in the open-source library `scib-metrics`, which makes its use very convenient. Another reason to use this benchmark is to make our results reproducible and comparable to other works. In the following, we will describe how the benchmark works. Below there is a pictorial description of how the benchmark is designed.

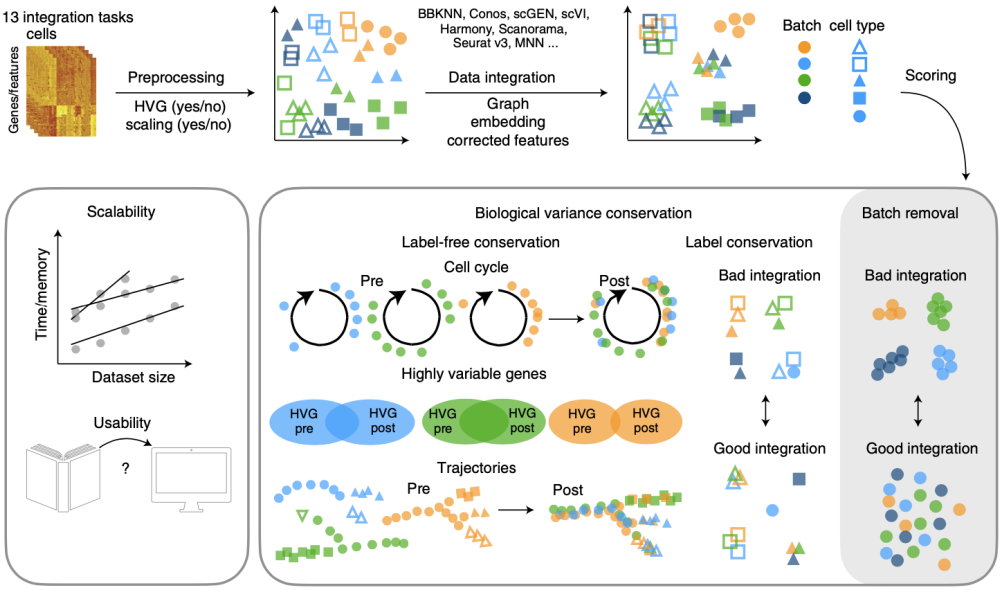


Figure 5: Benchmarking scRNA seq data integration and biological variation conservation [LBC+22].

This benchmark divides different metrics into two categories. The first category measures how well batch integration is performed and the second category measures biological variation conservation in the data. Now we will present the metrics that we will focus on to evaluate the learned latent space. The remaining metrics, presented for completeness, are explained in the original benchmarking paper [LBC+22]. To compare our models with a baseline, we will run the benchmark on Scanorama [HBB19] and the unintegrated principle components of the data.

2.2.1 Batch Integration Metrics

kBET is a metric developed by Büttner et al. [BMW+19] that measures batch integration by performing a statistical test to evaluate if the k nearest neighbors of

a data point are well mixed among batches. kBET uses a χ^2-based statistical test. This method has been shown to perform better to detect batch effects compared to other metrics [BMW+19], hence we will focus on this metric when evaluating batch integration.

2.2.2 Biological Variation Conservation Metrics

The ARI (Adjusted Rand Index) is a corrected-for-chance version of the Rand index measuring the similarity of two clusterings, by determining the percentage of right decisions the clustering algorithm took [HA85]. The NMI (Normalized Mutual Information) is a metric, computed between two clusterings and it is based on the mutual information between these clusters. The mutual information is normalized. A score of zero means no similarity between the clustering and a score of one means equality of clusterings [MGH13].

These two metrics measure the similarity of the clustering of cell types in the original data and the clustering run on the latent space of data points. The two clustering algorithms, which can be used, are Kmeans clustering [Llo82] and the Leiden clustering [TWvE19]. The Leiden clustering is more suitable for our purpose, as the Kmeans clustering assumes the same variance across clusters, which is not a valid assumption for cell types. Hence we will focus on the Leiden ARI and NMI metrics when evaluating biological variation conservation.

2.2.3 Overall Evaluation

The original benchmark evaluates the performance of batch integration and biological variation conservation by computing the average of all benchmarked metrics in their respective category. The overall score is computed as a weighted average of the two category scores with a weight of 0.6 for biological variation conservation and 0.4 for removing the batch effect. As we focus on the kBET score for batch integration and the NMI and ARI for the Leiden clustering for biological variation conservation we will evaluate the total performance as follows:

$$\begin{aligned} Batch\_Int &= KBet \\ Bio\_Cons &= \frac{1}{2}(Leiden\_NMI + Leiden\_ARI) \\ Total &= \frac{1}{2}(Batch\_Int + Bio\_Cons) \end{aligned}$$

2.3 Data

In this section, we will present the datasets and their preprocessing that will be used for the following experiments and benchmarks.

2.3.1 Datasets

In the spirit of reproducibility and comparability, we decided to evaluate our methods on the open-source datasets [LBC+22] used by the benchmark, explained in the previous section. We will use four different datasets for the evaluation. The main idea of using multiple datasets for the experiments and benchmarks is to validate results by determining if we can observe the same results across different datasets. We will now describe the four datasets that we will be using.

The first dataset is the Lung-atlas dataset. It includes scRNA-seq data from the human lung with 32472 cells split over 16 batches and 17 cell types. The Lung-atlas dataset will be our main dataset, which we will use for most experiments, while the other ones are primarily used for comparison. The second dataset is the Pancreas dataset including scRNA-seq data from the human pancreas with 16382 cells split over 9 batches and 14 cell types. The third dataset is the immune_cell_hum dataset including scRNA-seq data from human immune cells with 33506 cells split over 10 batches and 16 cell types. The last dataset is the immune_cell_hum_mou dataset. It includes scRNA-seq data from human and mouse immune cells with 97952 cells split over 23 batches and 19 cell types. Due to its size, we will use this last dataset to evaluate how the model scales to a larger amount of data.

2.3.2 Preprocessing

We have to preprocess the data, as many genes are not informative and contain a high zero count or are similarly expressed among all cells. We only want to take the most informative genes and use these to learn the latent space. We choose to take the 4000 most highly variable genes, by applying the `highly_variable_genes` function from the scanpy [WAT18] preprocessing package with the `cell-ranger` flavor and the provided batch labels. The `highly_variable_genes` function reproduces the implementation of Cell Ranger [ZTB+17], which computes means and dispersion using the log-transformed data. Genes are placed into bins based on their mean expression. Finally, the normalized dispersion is computed. This is done for each batch separately and the highly variable genes in each batch are merged.

2.4 Training

In this section, we will discuss how the model is trained. This includes discussing what parameters are needed to train the model and what values were chosen for these and why. Furthermore, we will explain the implementation of the KL term, which has no closed-form solution for the three learnable priors.

We train all models mostly using the default parameters of the scVI model [Lop18]. We will make all other choices for the training explicit and mention again the most important choices made by the scVI model. The model is trained for 400 epochs and has one hidden layer in both, the encoder and decoder with 128 units each. A dropout rate of 0.1 is used and a zero-inflated binomial distribution is chosen for the conditional distribution. We choose the latent space dimension 10 and train the models on a workstation with an Intel(R) Core(TM) i9-990K CPU and an NVIDIA GeForce RTX 2080 Ti GPU.

2.4.1 KL-term Scaling

We use a linear warm-up of the KL-term of 300 epochs and a maximal weight of the KL-term of 5. A linear warm-up is the default in the scVI model. The idea of a linear warmup is to weigh the KL-term down at first such that the model can learn a good reconstruction first and later when the weight increases, achieve a better fit to the prior. We use a maximal weight of the KL-term of 5 instead of 1, as the KL-term reaches smaller values faster compared to the reconstruction error and is hence quickly weighted down in the optimization, but as we want the prior to still be optimized and clearly see the performance difference between priors we increase the maximal weight. Additionally, a recent paper has shown a positive effect of giving a higher weight to the KL-term [HMP+17]. The effect will be explored in the Benchmarking Scaling of the KL-term experiment.

2.4.2 Prior Parameters

We train the Mixture of Gaussian prior with 50 components. This number is chosen to make a trade-off between a too-low number of components, which might not be able to model the aggregated posterior, and a too-high number, which might overfit the training data. The Vamp prior is also trained with 50 pseudo-inputs and the reasoning behind this choice is the same as for the Mixture of Gaussians prior. For the Normal Flow prior we choose 8 layers for the transforming Neural Network. The intention is to again have a prior, which is expressive enough, but will not overfit the training data. The effect of using different values will be explored in the Benchmarking Prior Parameters experiment.

2.4.3 KL-term Loss Implementation

As already mentioned in the introduction the KL-term for the three flexible priors has no closed-form solution in contrast to the Standard Normal prior. For the learnable priors, we approximate the KL-term using the Monte Carlo Estimate, as already explained in the paper introducing the Vamp prior [TW18].

$$\begin{aligned} D_{KL}(q_\phi(z|x)||p(z)) &= \mathbb{E}_{z\sim q_\phi(z|x)}[\ln q_\phi(z|x) - \ln p_\lambda(z)] \\ &\approx \frac{1}{L}\sum_{i=1}^{L} \ln q_\phi(z_\phi^{(i)}|x) - \ln p_\lambda(z_\phi^{(i)}) \end{aligned}$$

To approximate now the KL-term we only need to sample L points from the posterior $q_\phi(z|x)$ and compute the log-likelihood of $q_\phi(z_\phi^{(i)}|x)$ with respect to these samples. Also, we compute the log-likelihood of the prior with respect to the samples. In the implementation, we choose a sample size L of 1.

2.5 Experiments

In this section, we will discuss the different experiments we will perform and why they are relevant.

2.5.1 Prior and Posterior Visualization

In this experiment, we will visualize the aggregated posterior and the prior. We train the model with each prior and a two-dimensional latent space. After training, we visualize the prior and aggregated posterior in the latent space in two ways. Firstly, by plotting a contour plot of the prior and sampling once from the posterior each of 1000 randomly chosen data points. We color the samples from the posterior based on the cell type of the original data point. For the second visualization, we visualize the prior by drawing 1000 samples from the prior and plotting the posterior as before. We perform this experiment to verify if we can observe the hole problem noticed by [RV18] and determine if the learnable priors are able to mitigate this problem and are able to fit the aggregated posterior better than the Standard Normal prior. This experiment can also be seen as verification if the priors we implemented are able to learn the distribution of the aggregated posterior.

2.5.2 Relating Metrics to Training Epochs

In this experiment, we are interested in the benchmark metrics and how they evolve over the training epochs. In order to evaluate this we train the model with each prior as discussed in the Training section and checkpoint the model every 10 epochs. The checkpointed models are then evaluated on the benchmark. We will only evaluate the latent space of the model on the ARI and NMI clustering scores for the Leiden clustering to score the biological variation conservation and the kBET score for the evaluation of the batch integration. To check if the trends in the metrics observed are related to actual changes in the latent space we will plot the UMAP [MHM20] of the latent space for epochs $9, 199, 399$ and color them based on cell type, batch and Leiden clustering.

To checkpoint the model we need to implement our own callback, as the pytorch lightning callback doesn't produce checkpoints, which can be loaded with the scVI model. The implementation extends the pytorch lightning `ModelCheckpoint` callback and manually saves the model every time the `_save_checkpoint` method is called. The implementation of the callback can be found in the GitHub repository mentioned in the abstract. The goal of this experiment is to evaluate how the latent space behaves during training and how the length of the training influences the result.

2.5.3 Benchmarking Scaling the KL-term

As already mentioned in the Training section we will perform an experiment evaluating how the scaling of the maximal value of the KL-term (β) influences the benchmark metrics. We achieve this by training each prior with values $0, 1, 5, 10, 25$ and evaluating the benchmark on each of these models. The intention of this experiment is to understand how the importance of the prior relates to the benchmark metrics.

2.5.4 Benchmarking Prior Parameters

Again we mentioned in the Training section that we will be looking into evaluating the priors on the benchmark metrics based on different choices for their parameters. For the Mixture of Gaussians prior we will evaluate the model based on different number of components (number of Normal Distributions) that are used. For the Vamp prior we train the model using different numbers of pseudo-inputs. The values used for both Mixture of Gaussian prior and the Vamp prior are $2, 10, 50, 100$ and 500. Another parameter, which could be evaluated for the Vamp prior, is the architecture of the Neural Network learning the pseudo-inputs. This is not done in this work but could be explored later on. The parameter we are investigating for the Normal Flow prior is the number of layers, which are used to transform the base distribution into the final prior. Again, for the Normal Flow prior, we could also use different hyperparameters such as the architecture of the different layers or the base distribution, but these are not explored in this thesis. The values we evaluate for the Normal Flow prior are $1, 4, 8, 16, 32$ and 64.

3 Results

In this section, we will present all results we gathered. This includes the experiments described in the last section as well as the results of the benchmark, the training and validation losses recorded during the training for the benchmark.

3.1 Prior and Posterior Visualization

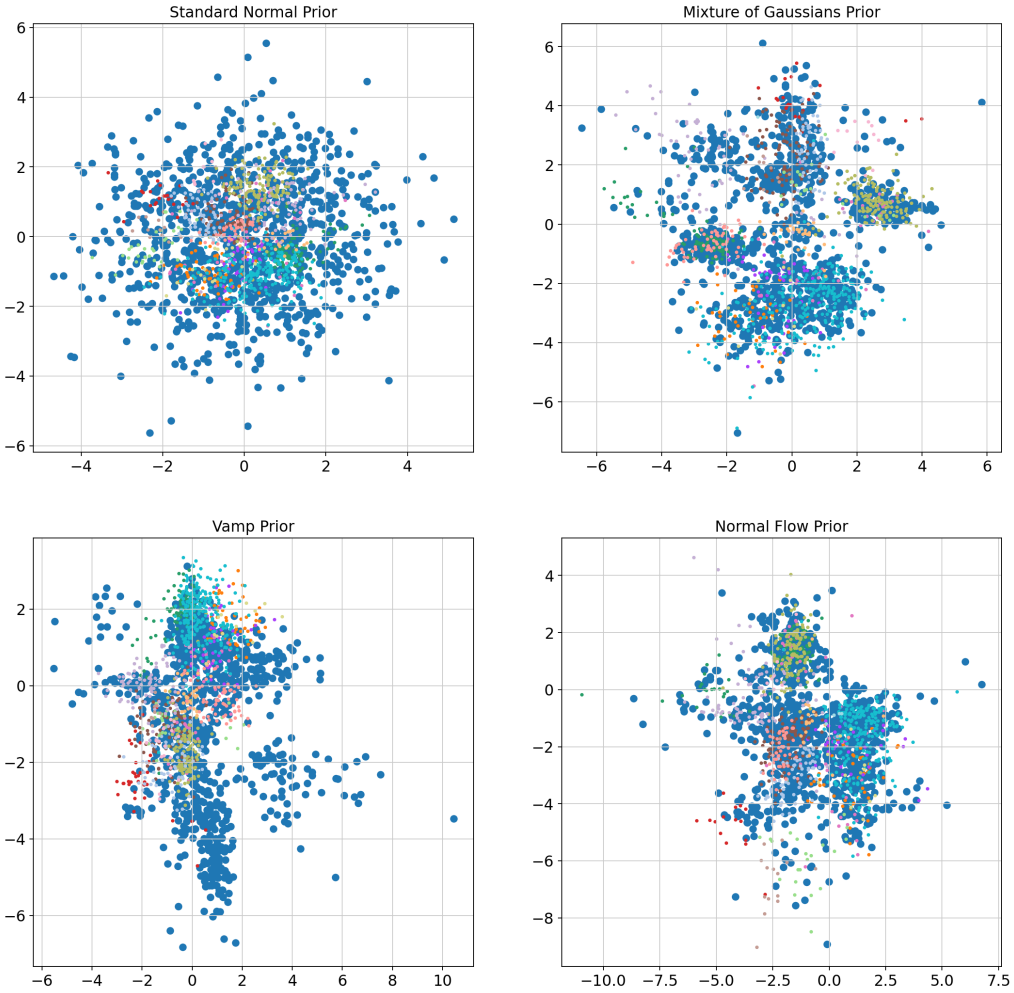


Figure 6: Samples drawn from the prior distribution (blue) and samples from the posterior distributions colored depending on the cell type of the input data point.

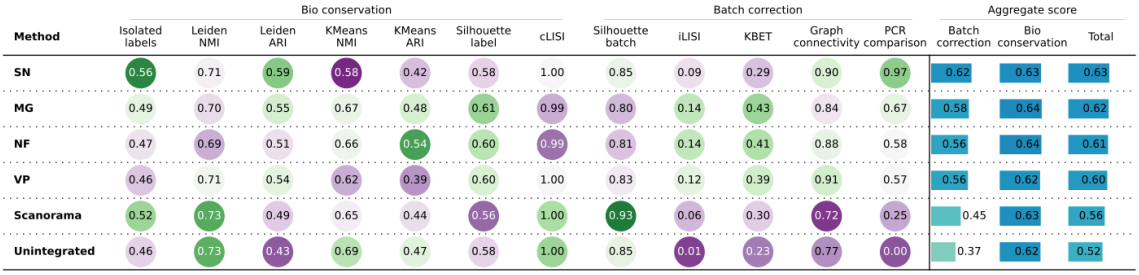
The visualization shows that the Standard Normal prior forces the latent representation of all cells to stay very close to the origin. This leads to the different cell types being very close together in the latent space and mixing a lot, which should make reconstruction harder. The hole problem described in [RV18] is not very dominant, because close to the origin there are no larger regions, which contain no samples from the aggregated posterior. The Mixture of Gaussians prior doesn't force the

latent space representation to be close to the origin but instead builds clusters in the latent space corresponding to one or multiple components of the prior. This allows the latent space to separate different cell types better. A perfect separation is definitely not achieved as there are often multiple cell types clustered together. The fit of the prior to the aggregated posterior seems to work well, as there are no larger regions, which contain only samples from the prior or the aggregated Posterior. Similar properties are shown by the Normal Flow prior with the difference that clusters stay closer together compared to the Mixture of Gaussians prior. The Vamp prior doesn't seem to fit the aggregated posterior well, as there is a large region containing many samples from the prior, but no samples from the aggregated Posterior. The plot visualizing the prior as a contour plot instead of samples drawn from the prior can be found in the supplementary section and shows similar results.

3.2 Benchmark Results

The next results we want to present are the benchmark results achieved by the VAEs with different priors.

3.2.1 Lung Atlas Benchmark

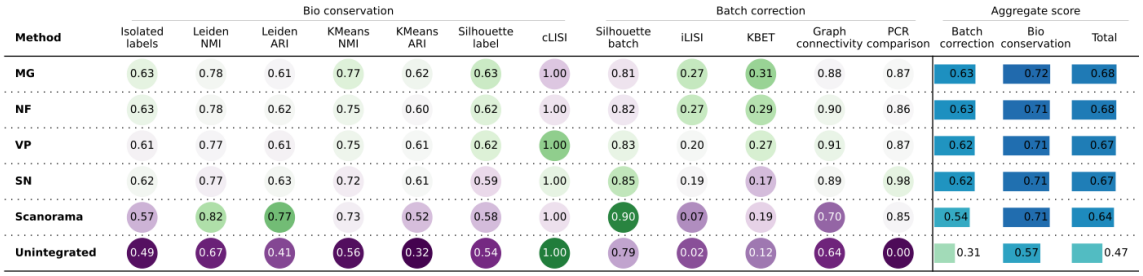


	Bio conservation							Batch correction					Aggregate score		
Method	Isolated labels	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	cLISI	Silhouette batch	iLISI	KBET	Graph connectivity	PCR comparison	Batch correction	Bio conservation	Total
SN	0.56	0.71	0.59	0.58	0.42	0.58	1.00	0.85	0.09	0.29	0.90	0.97	0.62	0.63	0.63
MG	0.49	0.70	0.55	0.67	0.48	0.61	0.99	0.80	0.14	0.43	0.84	0.67	0.58	0.64	0.62
NF	0.47	0.69	0.51	0.66	0.54	0.60	0.99	0.81	0.14	0.41	0.88	0.58	0.56	0.64	0.61
VP	0.46	0.71	0.54	0.62	0.39	0.60	1.00	0.83	0.12	0.39	0.91	0.57	0.56	0.62	0.60
Scanorama	0.52	0.73	0.49	0.65	0.44	0.56	1.00	0.93	0.06	0.30	0.72	0.25	0.45	0.63	0.56
Unintegrated	0.46	0.73	0.43	0.69	0.47	0.58	1.00	0.85	0.01	0.23	0.77	0.00	0.37	0.62	0.52

Figure 7: Benchmark Results for the Lung atlas dataset. SN: Standard Normal prior, MG: Mixture of Gaussians prior, VP: Vamp prior, NF: Normal Flow prior

The biological variation conservation metrics, which we are most interested in are the Leiden NMI and ARI. The Standard Normal prior achieves a slightly better performance across these two metrics compared to the learnable priors. One thing to note is that the Scanorama method, as well as the Unintegrated components, achieve a better NMI score than the VAEs, but a worse ARI. This is different from the expected result of the NMI and ARI scores being highly correlated. Additionally, we would expect the VAEs to learn a better latent space with respect to biological variation conservation than the two methods we are comparing to. For the batch correction metrics, we are most interested in the kBET score. We can witness a large increase from the Standard Normal to the learnable priors in the kBET score. Furthermore, the learnable priors perform much better than Scanorama and the Unintegrated components. The integration performance across the learnable priors is similar. The Mixture of Gaussians prior performs slightly better than the Normal Flow prior and the Vamp prior has the worst performance among the learnable priors.

3.2.2 Pancreas Benchmark



	Bio conservation							Batch correction					Aggregate score		
Method	Isolated labels	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	cLISI	Silhouette batch	iLISI	KBET	Graph connectivity	PCR comparison	Batch correction	Bio conservation	Total
MG	0.63	0.78	0.61	0.77	0.62	0.63	1.00	0.81	0.27	0.31	0.88	0.87	0.63	0.72	0.68
NF	0.63	0.78	0.62	0.75	0.60	0.62	1.00	0.82	0.27	0.29	0.90	0.86	0.63	0.71	0.68
VP	0.61	0.77	0.61	0.75	0.61	0.62	1.00	0.83	0.20	0.27	0.91	0.87	0.62	0.71	0.67
SN	0.62	0.77	0.63	0.72	0.61	0.59	1.00	0.85	0.19	0.17	0.89	0.98	0.62	0.71	0.67
Scanorama	0.57	0.82	0.77	0.73	0.52	0.58	1.00	0.90	0.07	0.19	0.70	0.85	0.54	0.71	0.64
Unintegrated	0.49	0.67	0.41	0.56	0.32	0.54	1.00	0.79	0.02	0.12	0.64	0.00	0.31	0.57	0.47

Figure 8: Benchmark Results for the Pancreas dataset.

To compare if the results on the last benchmark for the Lung atlas dataset are reproducible on other datasets we investigate the Pancreas dataset. For the biological conservation metrics, we see that the results are very similar among all priors. The Scanorama method again performs better than the VAEs, while the Unintegrated components perform worse. On the other hand, on the batch correction metrics we can witness again the same trend, the learnable priors achieve better performance than the Standard Normal prior. The ordering among the learnable priors stays the same. These results indicate that the flexible priors achieve better batch integration while achieving comparable performance on biological variation conservation compared to the Standard Normal prior. The further benchmark results on the other datasets can be found in the supplementary section. While the immune_cell_hum dataset shows similar results, the Standard Normal prior performs better in biological variance conservation than the learnable priors on the immune_cell_hum_mou dataset.

3.3 Training Losses and Times

After presenting the benchmark results, we will now show the training losses, validation losses and training times recorded during the training for the benchmark. We present the losses for the Lung Atlas dataset. The training and validation losses for the remaining data sets can be found in the supplementary section, but all behave similarly.

3.3.1 Training Losses

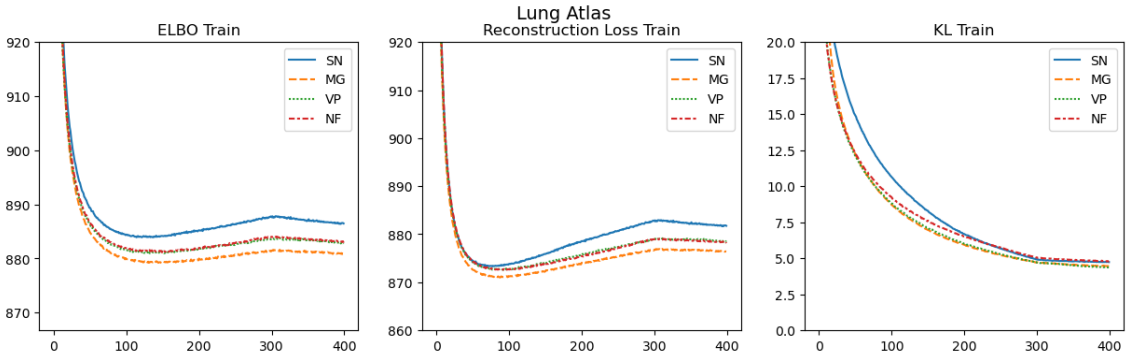


Figure 9: Training losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (Lung Atlas dataset).

The plots show that the learnable priors achieve a lower ELBO compared to the Standard Normal prior. While the KL-term reaches a similar value for all priors the reconstruction error of the flexible priors is lower than the reconstruction error of the Standard Normal prior. When comparing the learnable priors among each other we see that the Vamp prior and the Normal Flow prior achieve a similar reconstruction error and the Mixture of Gaussians prior has a slightly lower reconstruction error compared to the other two. After about 100 epochs the ELBO starts increasing again until the 300th epoch. After the 300 epoch, the ELBO decreases again. This time can be related to the end of the KL-term warmup. When the warmup ends the weight of the KL-term is not constantly increased anymore and hence the reconstruction error becomes the focus again leading to it being reduced again.

3.3.2 Validation Losses

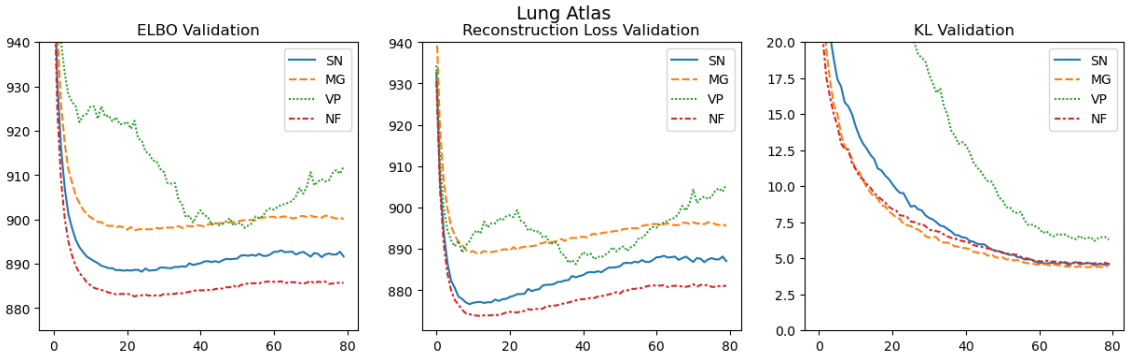


Figure 10: Validation losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (Lung Atlas dataset). The x-labels represent the epochs divided by 5.

The Validation loss behaves differently than the Training loss. The reconstruction validation loss of the Vamp prior is unstable compared to the other priors and starts increasing faster than the others after the 250th epoch. The validation loss of the other priors resembles the Training loss with the difference that the Mixture of Gaussians prior has a higher reconstruction validation loss than the Standard Normal prior and the Normal Flor prior has the lowest reconstruction validation loss among all priors. For the KL-term validation Loss, the Vamp prior has the highest value compared to other priors, while the KL-term validation loss of all other priors ends up at a similar value. The validation losses indicate that the Vamp prior overfits to the training data, which is not the case for the other priors.

3.3.3 Timing

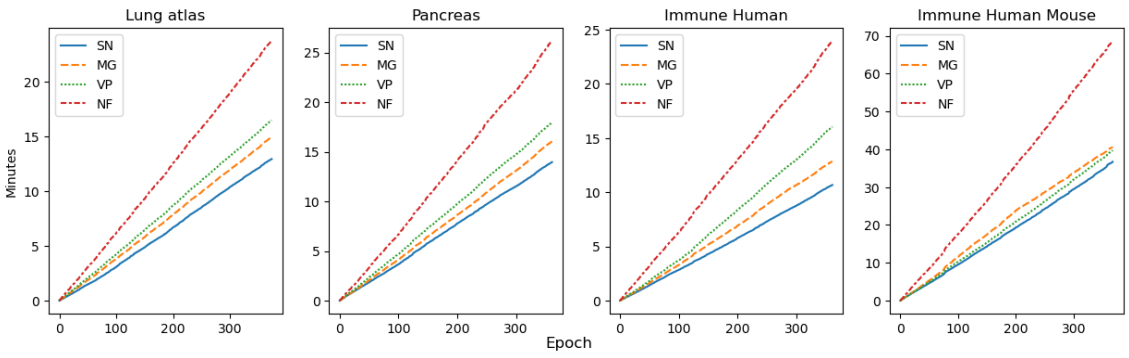


Figure 11: Time recorded each training epoch.

The Timing plots clearly show that the learnable priors take longer to train than the Standard Normal prior, as they also have to update parameters inside the prior and

not only in the Encoder and Decoder. For the learnable priors, the Normal Flow prior takes considerably longer to train than the other priors, which are much more similar in their training time. This is the case, as we train 8 layers of Neural Network components, that transform the base distribution compared to fewer parameters in the other priors.

3.4 Relating Metrics to Training Epochs

After presenting the benchmark results and training losses we now want to show the results of the experiment relating benchmark metrics to training epochs.

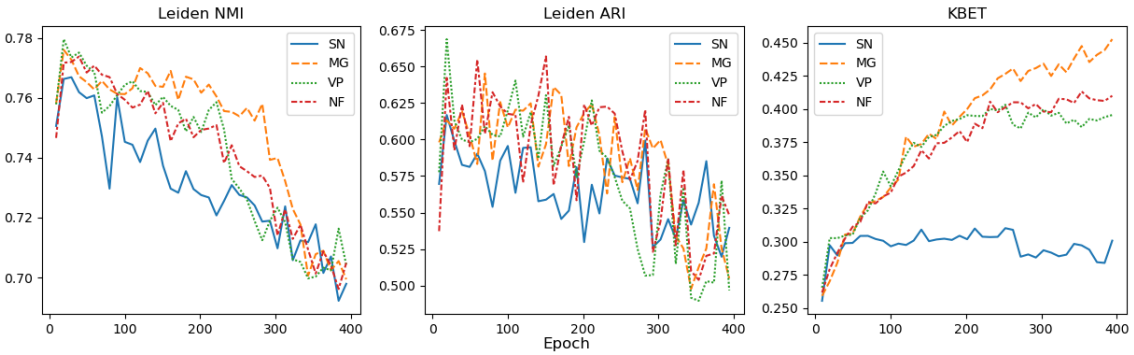


Figure 12: Benchmark Metrics plotted over Training epochs.

The two biological variation conservation metrics show that across all priors the maximal values are achieved in the first 20 epochs. Afterwards, the metrics scores decrease over the training. For the batch integration metrics we find that the kBET score increases continuously for the learnable priors, while it doesn't increase for the Standard Normal prior after the first 20 epochs. We also observe that there is a high variance in the Leiden metrics. Particularly, the Leiden ARI is unstable across the training. The best kBET score is achieved for the Mixture of Gaussians prior, which matches the results we obtained before. We will now look at the UMAPs of the latent space at different points during the training to see if we can relate the trend in the metrics to changes in the latent space.

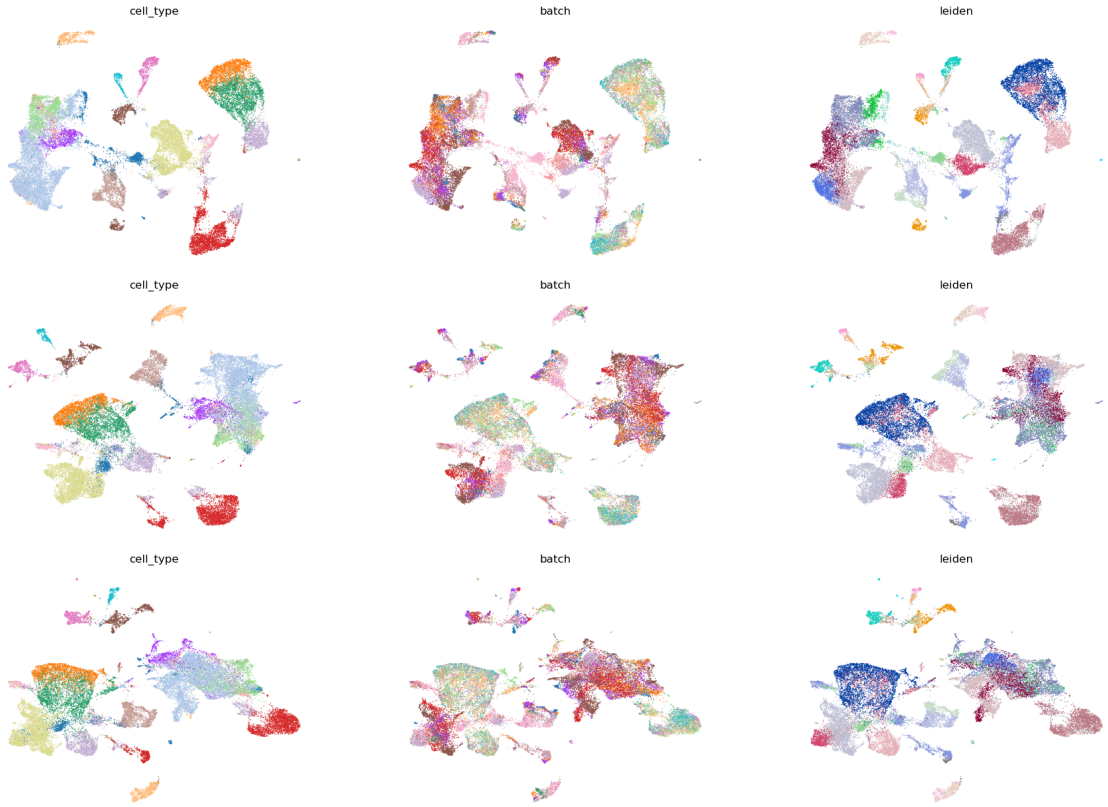


Figure 13: UMAPs of latent space for epochs $9, 199, 399$ **for the Standard Normal prior.**

The first three plots show the UMAPs of the latent space at the epochs $9, 199$ and 399 for the Standard Normal prior. We can see that already after the first 9 epochs the cell types are well separated into clusters and the Leiden clustering is able to match the cell types. On the other hand, the batch integration is still very bad, as we can see that batches are still separated in the latent space. In the epochs 199 and 399 we can see that the batches are better integrated, but we can still see that the batches are not perfectly mixed. During these steps, the clusters of cell types are moving closer together and start mixing more.

Figure 14: UMAPs of latent space for epochs $9, 199, 399$ for the Mixture of Gaussians prior.

The next three plots show the UMAPs of the latent space at epochs $9, 199$ and 399 for the Mixture of Gaussians prior. The plots show similar results to the ones of the Standard Normal prior. The separation of cell types is already achieved in the 9th epoch, while the batches are still separated. The development through the epochs is different in the sense that the cell type clusters are merging even more in comparison to the Standard Normal prior, and the batches seem to be mixed better compared to the Standard Normal prior. In general, the UMAPs correspond to the trends that we observe in the benchmark metrics. The biological variation is best conserved in the beginning after only a few epochs and while batch integration is performed the cell types also merge and hence the clustering of the cell types becomes worse. The UMAP plots of the other two learnable priors can be found in the supplementary section and show similar behavior as the Mixture of Gaussians prior.

3.5 Benchmarking Scaling the KL-term

The next experiment we want to show is the scaling of the maximal value of the KL term and how the scaling influences the benchmark metrics. We will refer to the maximal value of the KL-term scaling as β.

3.5.1 Standard Normal Prior

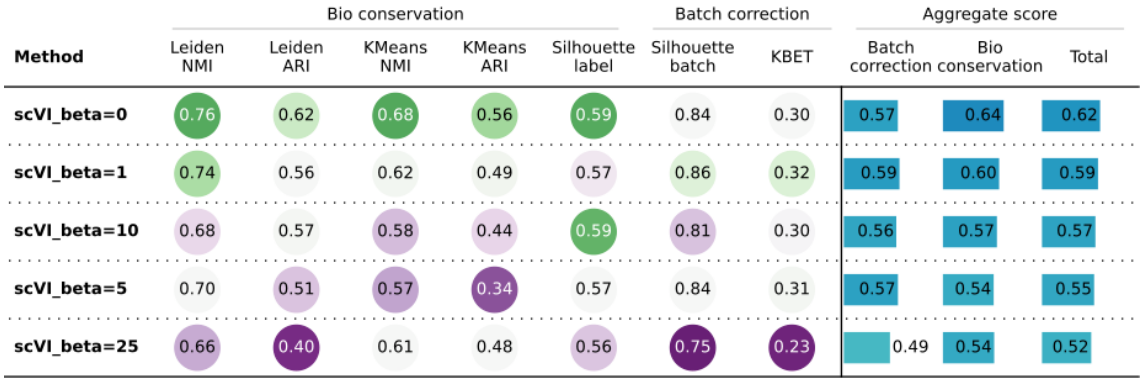


Figure 15: Benchmark Metrics with different maximal values for the KL-term (beta) for the Standard Normal prior.

We use the values $0, 1, 5, 10$ and 25 for β. The metrics measuring biological variation conservation show that the higher we choose β the worse the performance gets. The batch integration score kBET on the other hand stays very similar across different choices for β except for the largest one, which decreases performance by a lot.

3.5.2 Mixture of Gaussians Prior

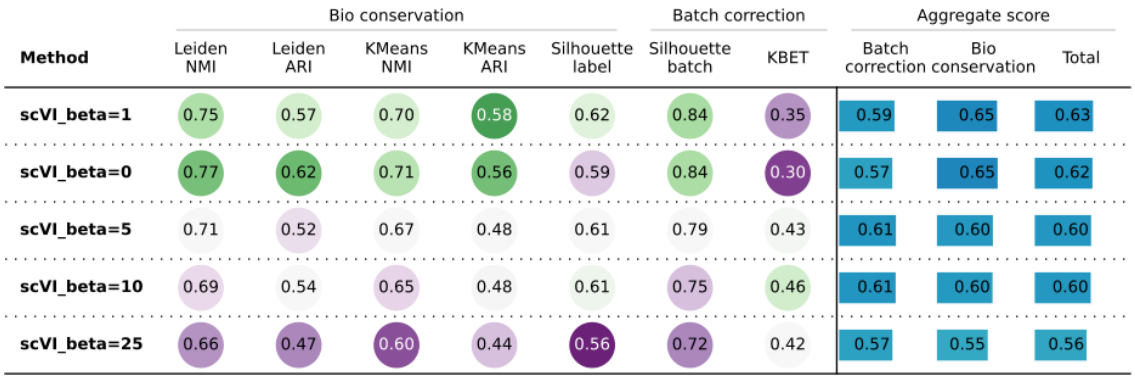


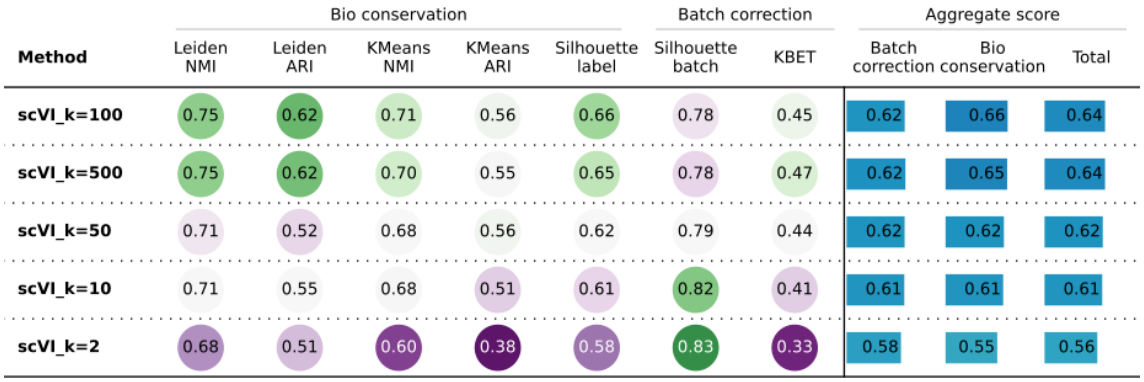
Figure 16: Benchmark Metrics with different maximal values for the KL-term (beta) for the Mixture of Gaussians prior.

The biological variation conservation metrics behave similarly to the ones of the Standard Normal prior. The performance decreases when we choose higher values for β. The batch integration on the other hand compared to the Standard Normal prior improves significantly when increasing β. The largest value for β performs worse again. The results for the other two priors can again be found in the supplementary section and show similar results to the Mixture of Gaussians prior.

3.6 Benchmarking Prior Parameters

The last result we will present are the results of the experiment benchmarking different prior parameters.

3.6.1 Mixture of Gaussians Prior

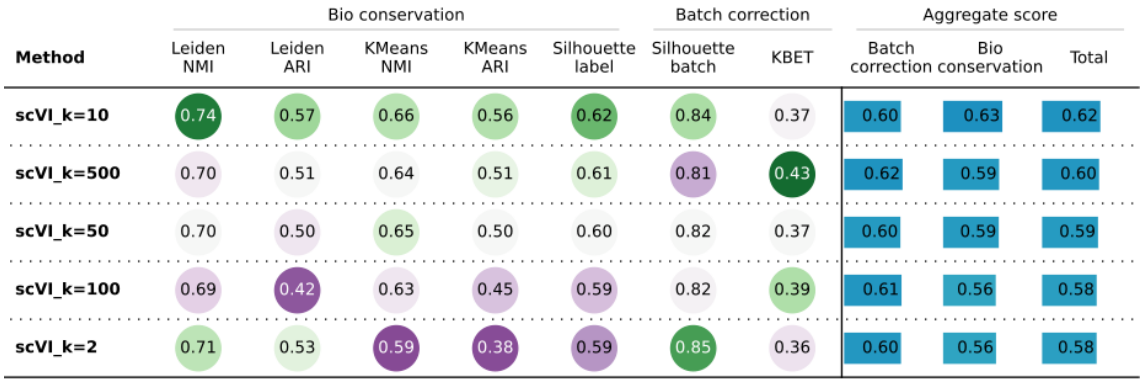


	Bio conservation					Batch correction		Aggregate score		
Method	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	Silhouette batch	KBET	Batch correction	Bio conservation	Total
scVI_k=100	0.75	0.62	0.71	0.56	0.66	0.78	0.45	0.62	0.66	0.64
scVI_k=500	0.75	0.62	0.70	0.55	0.65	0.78	0.47	0.62	0.65	0.64
scVI_k=50	0.71	0.52	0.68	0.56	0.62	0.79	0.44	0.62	0.62	0.62
scVI_k=10	0.71	0.55	0.68	0.51	0.61	0.82	0.41	0.61	0.61	0.61
scVI_k=2	0.68	0.51	0.60	0.38	0.58	0.83	0.33	0.58	0.55	0.56

Figure 17: Mixture of Gaussians prior benchmarked with a different number of components (k).

For the Mixture of Gaussians prior, all metrics improve when scaling up the number of components k that are used for the prior. The results we obtain for 500 components have the best score for batch integration and biological variation conservation.

3.6.2 Vamp Prior



	Bio conservation					Batch correction		Aggregate score		
Method	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	Silhouette batch	KBET	Batch correction	Bio conservation	Total
scVI_k=10	0.74	0.57	0.66	0.56	0.62	0.84	0.37	0.60	0.63	0.62
scVI_k=500	0.70	0.51	0.64	0.51	0.61	0.81	0.43	0.62	0.59	0.60
scVI_k=50	0.70	0.50	0.65	0.50	0.60	0.82	0.37	0.60	0.59	0.59
scVI_k=100	0.69	0.42	0.63	0.45	0.59	0.82	0.39	0.61	0.56	0.58
scVI_k=2	0.71	0.53	0.59	0.38	0.59	0.85	0.36	0.60	0.56	0.58

Figure 18: Vamp prior benchmarked with a different number of pseudo-inputs (k).

When increasing the number of pseudo-inputs of the Vamp prior we improve performance on batch integration. The best performance on biological variance conser-

vation is reached when choosing a number of 10 pseudo-inputs and the performance decreases for both, more and less pseudo-inputs.

For the Normal Flow prior when increasing the number of layers the performance on both, biological variation conservation and batch integration doesn't change significantly. The results can be found in the supplementary section.

4 Discussion

In the following section, we will discuss the obtained results, the challenges we encountered, and recommendations for further research directions.

4.1 Main Results

Below there are visualizations of the main results we found.

	Benchmark			Training			Parameters	
Prior	Bio Cons	Batch Int	Total	Train Loss	Val Loss	Time	KL Scal	Prior Par
MG_opt	0.68	0.47	0.58	nan	nan	nan	5.0	500.0
MG	0.62	0.43	0.53	880.0	900.0	15.0	5.0	50.0
NF_opt	0.64	0.41	0.53	nan	nan	nan	5.0	8.0
VP_opt	0.6	0.43	0.52	nan	nan	nan	5.0	500.0
VP	0.62	0.39	0.51	882.0	911.0	16.5	5.0	50.0
NF	0.6	0.41	0.5	882.0	885.0	23.8	5.0	8.0
SN_opt	0.69	0.3	0.5	nan	nan	nan	0.0	nan
SN	0.65	0.29	0.47	886.0	891.0	13.0	5.0	nan

Figure 19: Overview of the obtained results for the Lung-atlas dataset. $Bio\_Cons = \frac{1}{2}(Leiden\_NMI + Leiden\_ARI)$, $Batch\_int = kBET$, $Total = \frac{1}{2}(Bio\_Cons + Batch\_int)$**. The optimal values are taken from the experiment that performed best regarding the total score. The other values are taken from the first benchmark and the parameters correspond to the parameters used for training.**

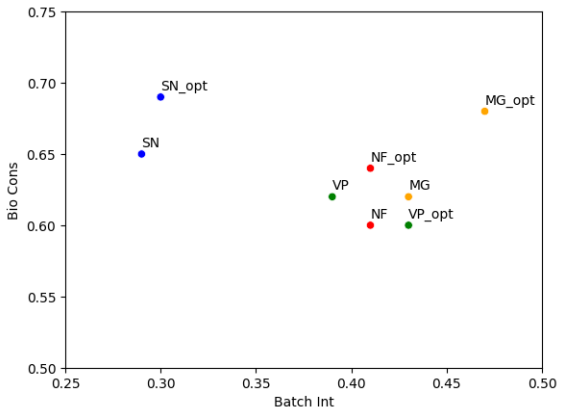


Figure 20: Summarized results visualizing biological conservation and batch integration.

Based on the visualization of the priors and posteriors (Sec. 3.1), we can validate the base assumption that learnable priors can fit the aggregated posterior well and avoid becoming overly focused on the origin like the Standard Normal prior. The Mixture of Gaussians and the Normal Flow prior achieved the best fit to the aggregated posterior, which the Vamp prior was not able to do.

Relating metrics to training epochs (Sec. 3.4) showed that already after a few epochs, the VAEs can learn a latent representation of the data separating cell types well, while the batches integration is ignored. Over further training epochs, the models learned the batch effect and achieved better integration. Here, the choice of the prior has a positive impact on the overall performance, as only the learnable priors significantly improve their batch integration after the first few epochs. During the integration, the different cell type clusters, which had been already well separated in the beginning, moved closer together and merged, which leads to decreasing biological variation conservation metrics. The number of training epochs hence gives us another trade-off between batch integration and biological variance conservation when using learnable priors.

The experiment evaluating the scaling of the maximal weight of the KL-term (β) (Sec. 3.5) indicates that the higher we choose β, the better we achieve batch integration and the more biological variation we lose during training. By selecting a higher value for β, we can focus on the matching of the prior to the aggregated posterior. This aligns with the results discussed before. As we used a linear warmup of the KL-term, in the first epochs we prioritize on a good reconstruction allowing us to capture a lot of variation. When scaling up the KL-term we then give more priority to the matching and achieve better batch integration. This shows a relationship between the matching of the prior and aggregated posterior and the batch integration. Therefore, the choice of β gives us another way to trade off biological variation conservation and batch integration.

The last experiment (Sec. 3.6) conducted showed that making the Mixture of Gaussians and Vamp prior more flexible improves batch integration. We can increase the flexibility by adding more components to the Mixture of Gaussians prior and pseudo-inputs to the Vamp prior. The same result was not observable for the Normal Flow prior, as increasing the number of layers did not improve batch integration. One reason might be that both Mixture of Gaussians and Vamp prior are the same type of distribution as the aggregated posterior. This makes it easier to fit the aggregated posterior well. Increasing flexibility had no clear effect on biological variation conservation. The Mixture of Gaussians prior was able to achieve better scores with a higher number of components, while the Vamp prior lost performance.

Generally, the investigation of the benchmark results indicates that learnable priors can achieve significantly better batch integration than the Standard Normal prior. On the other hand, the Standard Normal prior achieves slightly higher biological variation conservation scores, but this is outweighed by the substantial improvements in batch integration.

4.2 Challenges

Most notably, some of the benchmark results are unexpected, as Scanorama and sometimes even the Unintegrated components perform similarly or even better regarding biological variation conservation metrics. We assume that this is due to the datasets being of relatively low complexity. Hence, even simpler methods are able to capture biological variability well. The more complex scVI-based models are also able to capture this variability, but as they focus on improving batch integration they lose some variability in the latent space during training, as shown in the relating metrics to training epochs experiment. The weak performance on the large dataset compared to the Standard Normal prior might be due to hyperparameters not being well adjusted to the size of the dataset.

Additionally, we note that the benchmark metrics can have a large variance. We observed a particularly high variance for the Leiden ARI with a difference of 0.8 between experiments with the same parameters, compared to 0.1 for Leiden NMI, and 0.2 for kBET. This is caused by randomized components used in the training and evaluation. The best way to address this would be to perform experiments multiple times and compute the mean and standard deviation of the results, which was not done in this work due to time limitations.

One potential issue with the Vamp prior's performance might be the implementation of learning the pseudo-inputs. Also, the choices we made for the parametrization of the Neural Network learning the pseudo-inputs as well as the parametrization of the functions of the Normal Flow prior could be chosen in a different way, which might influence performance positively.

Finally, the data used was limited, as we ran the benchmark only on 4 datasets and the experiments only on one dataset. Furthermore, the way we performed data preprocessing might have influenced performance. Choosing a different number of highly-variable genes and the way these are determined play an important role.

4.3 Conclusion

During this thesis, we were able to achieve our main goal of implementing three learnable priors for the scVI model. By benchmarking the latent space, we showed that learnable priors improved batch integration significantly. Further experiments expanding the main goal of this work were able to provide insights into the underlying learning mechanisms of the latent space. In particular, we were able to show that these priors did not constrain the latent space to a distribution around the origin like the Standard Normal prior, and allow the models to improve batch integration over the course of training. We have been able to control the trade-off between biological variation conservation and batch integration by variating β as well as the number of epochs trained. Additionally, this work demonstrated the importance of

scaling the KL-term as well as choosing fitting prior parameters.

To conclude, this work underlined the potential of learnable priors for learning a robust latent space representation. The Mixture of Gaussians prior was identified as best performing prior achieving strongest scores on both biological variation conservation and batch integration.

4.4 Further Research

Generally, we were able to show that learnable priors can have a positive influence on latent space learning for scRNA-seq data, and should therefore be further explored. Based on the results of this work, there are a few avenues to research further. First and foremost, it would be interesting to explore the effect of different implementations of the priors. Additionally, a promising approach to improve performance is to conduct a larger hyperparameter search for the priors as well as other model parameters such as β or the number of trained epochs. Furthermore, the way how the warmup is performed can be varied and the impact of improvement in batch integration on downstream analysis could be evaluated.

References

[BGS16] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders, 2016.

[BMW+19] Maren Büttner, Zhichao Miao, F. Alexander Wolf, Sarah A. Teichmann, and Fabian J. Theis. A test metric for assessing single-cell rna-seq batch correction. *Nature Methods*, 16(1):43–49, Jan 2019.

[BVV+16] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space, 2016.

[Chi21] Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images, 2021.

[DKFT20] Leander Dony, Martin König, D Fischer, and Fabian J Theis. Variational autoencoders with flexible priors enable robust distribution learning on single-cell rna sequencing data. In *ICML 2020 Workshop on Computational Biology (WCB) Proceedings Paper*, volume 37, 2020.

[DLW22] Nunzio D'Agostino, Wenli Li, and Dapeng Wang. High-throughput transcriptomics. *Scientific Reports*, 12(1):20313, Nov 2022.

[ea23a] L. Heumos et al. *Best practices for single-cell analysis across modalities.* Nat Rev Genet, 2023.

[ea23b] Philipp Angerer et al. scvi-tools: Deep probabilistic analysis of single-cell omics data, 2023.

[HA85] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec 1985.

[HBB19] Brian Hie, Bryan Bryson, and Bonnie Berger. Efficient integration of heterogeneous single-cell transcriptomes using scanorama. *Nature Biotechnology*, 37(6):685–691, Jun 2019.

[HMP+17] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

[KW22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[LBC+22] Malte D. Luecken, M. Büttner, K. Chaichoompu, A. Danese, M. Interlandi, M. F. Mueller, D. C. Strobl, L. Zappia, M. Dugas, M. Colomé-Tatché, and Fabian J. Theis. Benchmarking atlas-level data integration in single-cell genomics. *Nature Methods*, 19(1):41–50, Jan 2022.

[Llo82] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[Lop18] R. et al.. Lopez. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15:1053–1058, 2018.

[MGH13] Aaron F. McDaid, Derek Greene, and Neil Hurley. Normalized mutual information to evaluate overlapping community finding algorithms, 2013.

[MHM20] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[PTT+14] Anoop P. Patel, Itay Tirosh, John J. Trombetta, Alex K. Shalek, Shawn M. Gillespie, Hiroaki Wakimoto, Daniel P. Cahill, Brian V. Nahed, William T. Curry, Robert L. Martuza, David N. Louis, Orit Rozenblatt-Rosen, Mario L. Suvà, Aviv Regev, and Bradley E. Bernstein. Single-cell rna-seq highlights intratumoral heterogeneity in primary glioblastoma. *Science*, 344(6190):1396–1401, 2014.

[Qiu20] Peng Qiu. Embracing the dropouts in single-cell rna-seq analysis. *Nature Communications*, 11(1):1169, Mar 2020.

[RM16] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016.

[Rou87] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[RPG+18] Davide Risso, Fanny Perraudeau, Svetlana Gribkova, Sandrine Dudoit, and Jean-Philippe Vert. A general and flexible method for signal extraction from single-cell rna-seq data. *Nature Communications*, 9(1):284, Jan 2018.

[RV18] Danilo Jimenez Rezende and Fabio Viola. Taming vaes, 2018.

[RZLA20] Di Ran, Shanshan Zhang, Nicholas Lytal, and Lingling An. scDoc: correcting drop-out events in single-cell RNA-seq data. *Bioinformatics*, 36(15):4233–4239, 05 2020.

[SK51] R. A. Leibler S. Kullback. On information and sufficiency, 1951.

[SLC+23] Vincent Stimper, David Liu, Andrew Campbell, Vincent Berenz, Lukas Ryll, Bernhard Schölkopf, and José Miguel Hernández-Lobato. normflows: A pytorch package for normalizing flows. *Journal of Open Source Software*, 8(86):5361, 2023.

[SLR+21] Guangshun Sun, Zhouxiao Li, Dawei Rong, Hao Zhang, Xuesong Shi, Weijun Yang, Wubin Zheng, Guoqiang Sun, Fan Wu, Hongyong Cao, Weiwei Tang, and Yangbai Sun. Single-cell rna sequencing in cancer: Applications, advances, and emerging challenges. *Molecular Therapy - Oncolytics*, 21:183–206, 2021.

[SSZ+17] Uri Shaham, Kelly P Stanton, Jun Zhao, Huamin Li, Khadir Raddassi, Ruth Montgomery, and Yuval Kluger. Removal of batch effects using distribution-matching residual networks. *Bioinformatics*, 33(16):2539–2546, 04 2017.

[Tom23a] Jakub M. Tomczak. Priors in vaes, 2023.

[Tom23b] Jakub M. Tomczak. Variational auto-encoders, 2023.

[TW18] Jakub M. Tomczak and Max Welling. Vae with a vampprior, 2018.

[TWvE19] V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1):5233, Mar 2019.

[VRT+21] Isaac Virshup, Sergei Rybakov, Fabian J. Theis, Philipp Angerer, and F. Alexander Wolf. anndata: Annotated data. *bioRxiv*, 2021.

[WAT18] F. Alexander Wolf, Philipp Angerer, and Fabian J. Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):15, Feb 2018.

[XWY+21] Ruizhi Xiang, Wencan Wang, Lei Yang, Shiyuan Wang, Chaohan Xu, and Xiaowen Chen. A comparison for dimensionality reduction methods of single-cell rna-seq data. *Frontiers in Genetics*, 12, 2021.

[ZLC+23] Yufeng Zhang, Wanwei Liu, Zhenbang Chen, Ji Wang, and Kenli Li. On the properties of kullback-leibler divergence between multivariate gaussian distributions, 2023.

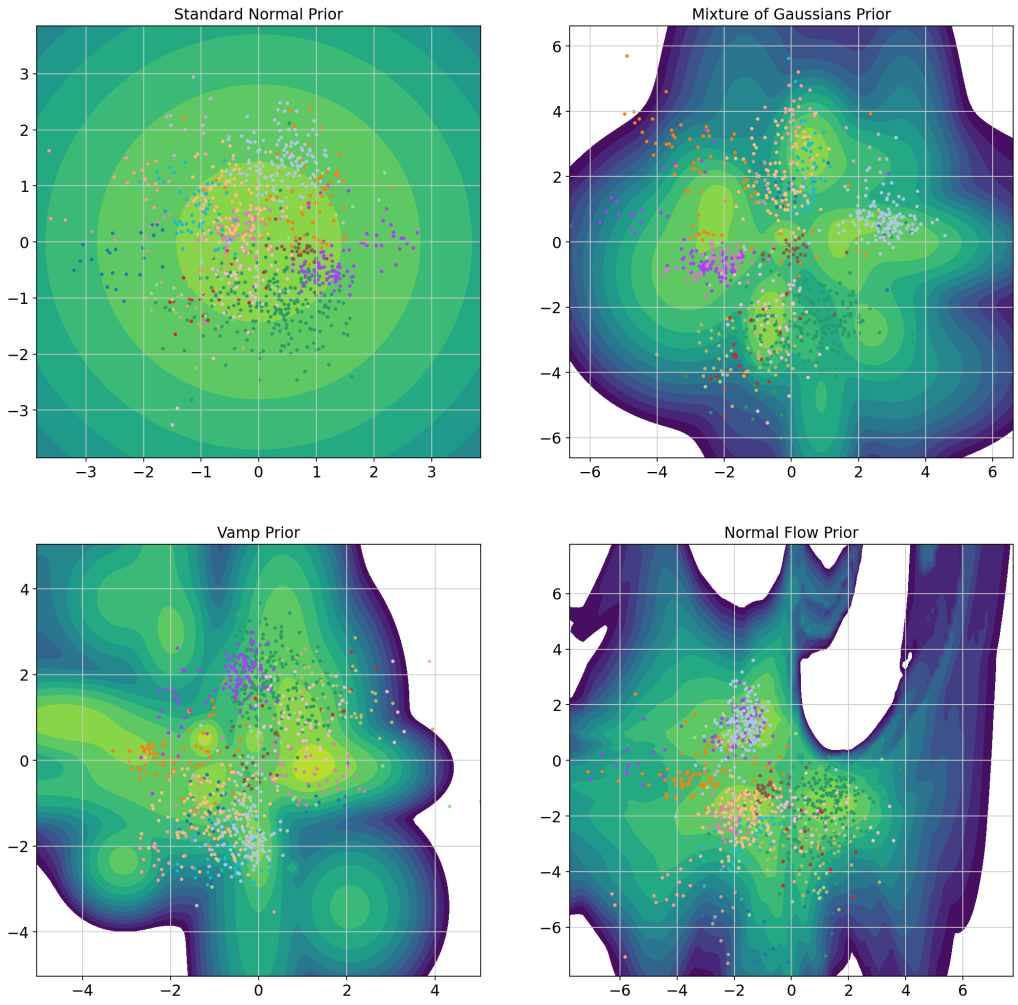
[ZTB+17] Grace X. Y. Zheng, Jessica M. Terry, Phillip Belgrader, Paul Ryvkin, Zachary W. Bent, Ryan Wilson, Solongo B. Ziraldo, Tobias D. Wheeler, Geoff P. McDermott, Junjie Zhu, Mark T. Gregory, Joe Shuga, Luz Montesclaros, Jason G. Underwood, Donald A. Masquelier, Stefanie Y. Nishimura, Michael Schnall-Levin, Paul W. Wyatt, Christopher M. Hindson, Rajiv Bharadwaj, Alexander Wong, Kevin D. Ness, Lan W. Beppu, H. Joachim Deeg, Christopher McFarland, Keith R. Loeb,

William J. Valente, Nolan G. Ericson, Emily A. Stevens, Jerald P. Radich, Tarjei S. Mikkelsen, Benjamin J. Hindson, and Jason H. Bielas. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8(1):14049, Jan 2017.

[ZYZ+22] Liuting Zeng, Kailin Yang, Tianqing Zhang, Xiaofei Zhu, Wensa Hao, Hua Chen, and Jinwen Ge. Research progress of single-cell transcriptome sequencing in autoimmune diseases and autoinflammatory disease: A review. *Journal of Autoimmunity*, 133:102919, 2022.

Supplementary

A Further Prior and Posterior Visualizations



The first results we want to present are the Posterior and Prior Visualizations.

Figure 21: Contourplots of the Prior distributions with samples from the posterior distributions colored depending on the cell type of the input data point.

B Further Benchmark Results

	Bio conservation							Batch correction					Aggregate score		
Method	Isolated labels	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	cLISI	Silhouette batch	iLISI	KBET	Graph connectivity	PCR comparison	Batch correction	Bio conservation	Total
NF	0.62	0.77	0.62	0.76	0.61	0.63	1.00	0.82	0.27	0.30	0.90	0.87	0.63	0.71	0.68
MG	0.65	0.79	0.62	0.73	0.51	0.64	1.00	0.82	0.27	0.30	0.89	0.87	0.63	0.71	0.68
VP	0.61	0.77	0.64	0.71	0.52	0.62	1.00	0.85	0.20	0.22	0.91	0.90	0.61	0.70	0.66
SN	0.62	0.76	0.61	0.69	0.54	0.59	1.00	0.85	0.19	0.17	0.90	0.97	0.62	0.69	0.66
Scanorama	0.57	0.82	0.78	0.73	0.52	0.58	1.00	0.90	0.07	0.19	0.70	0.85	0.54	0.71	0.64
Unintegrated	0.49	0.66	0.41	0.56	0.32	0.54	1.00	0.79	0.02	0.12	0.64	0.00	0.31	0.57	0.47

Figure 22: Benchmark Results for the immune_cell_hum dataset.

	Bio conservation							Batch correction					Aggregate score		
Method	Isolated labels	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	cLISI	Silhouette batch	iLISI	KBET	Graph connectivity	PCR comparison	Batch correction	Bio conservation	Total
SN	0.40	0.64	0.47	0.47	0.28	0.50	1.00	0.86	0.06	0.07	0.83	0.95	0.55	0.54	0.54
Scanorama	0.49	0.67	0.51	0.61	0.38	0.50	1.00	0.93	0.05	0.09	0.57	0.33	0.39	0.60	0.51
MG	0.38	0.57	0.31	0.53	0.29	0.47	0.99	0.81	0.08	0.11	0.83	0.75	0.52	0.51	0.51
NF	0.37	0.54	0.29	0.48	0.26	0.45	0.99	0.82	0.08	0.11	0.81	0.77	0.52	0.48	0.50
VP	0.36	0.55	0.28	0.45	0.24	0.46	0.99	0.83	0.07	0.09	0.79	0.71	0.50	0.48	0.48
Unintegrated	0.40	0.61	0.31	0.55	0.42	0.42	1.00	0.81	0.02	0.03	0.57	0.00	0.29	0.53	0.43

Figure 23: Benchmark Results for the immune_cell_hum_mou dataset.

C Further Training Losses

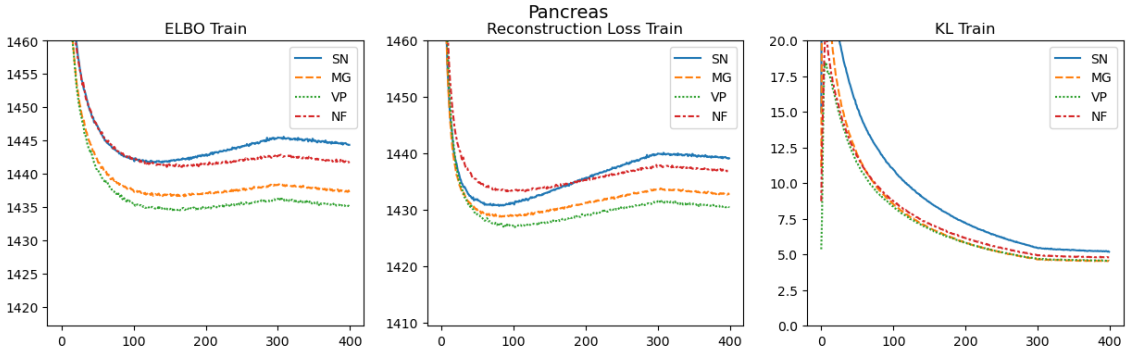


Figure 24: Training losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (Pancreas dataset).

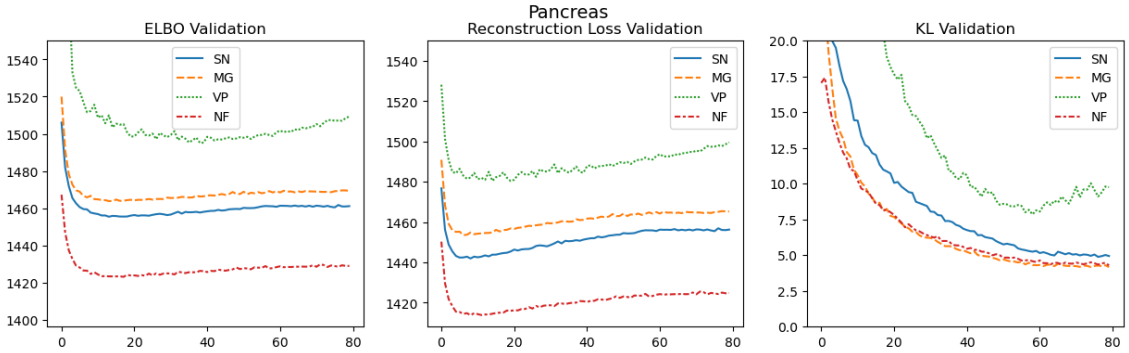


Figure 25: Validation losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (Pancreas dataset).

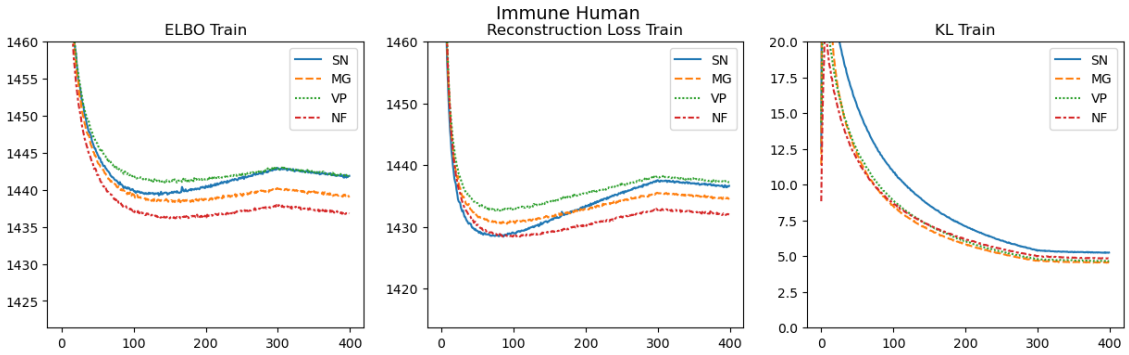


Figure 26: Training losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (immune_cell_hum dataset).

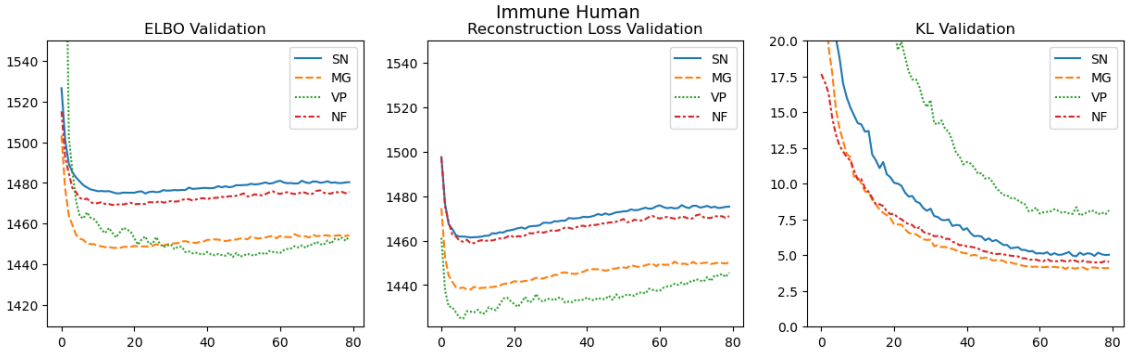


Figure 27: Validation losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (immune_cell_hum dataset).

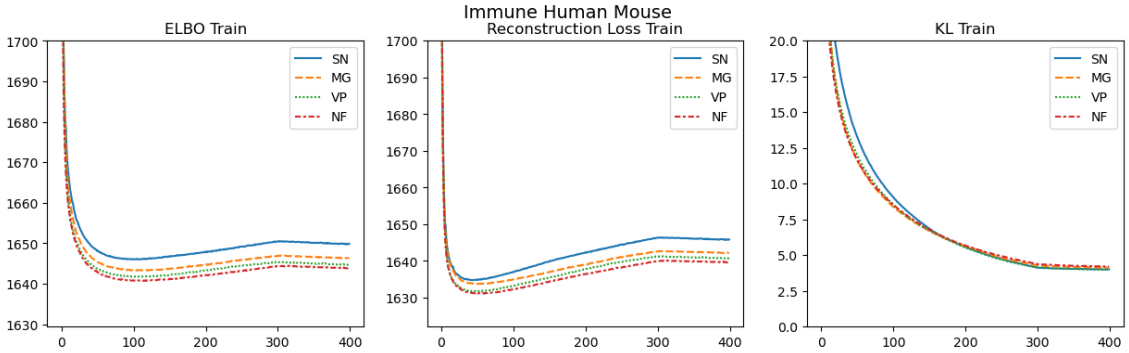


Figure 28: Training losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (immune_cell_hum_mou dataset).

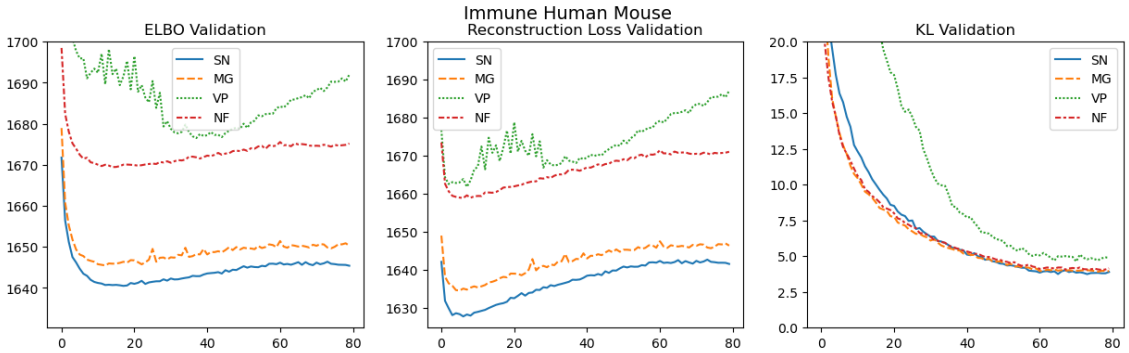


Figure 29: Validation losses divided into reconstruction and KL-term recorded during learning for benchmark evaluation (immune_cell_hum_mou dataset).

D Further Umaps Metric Epoch Results

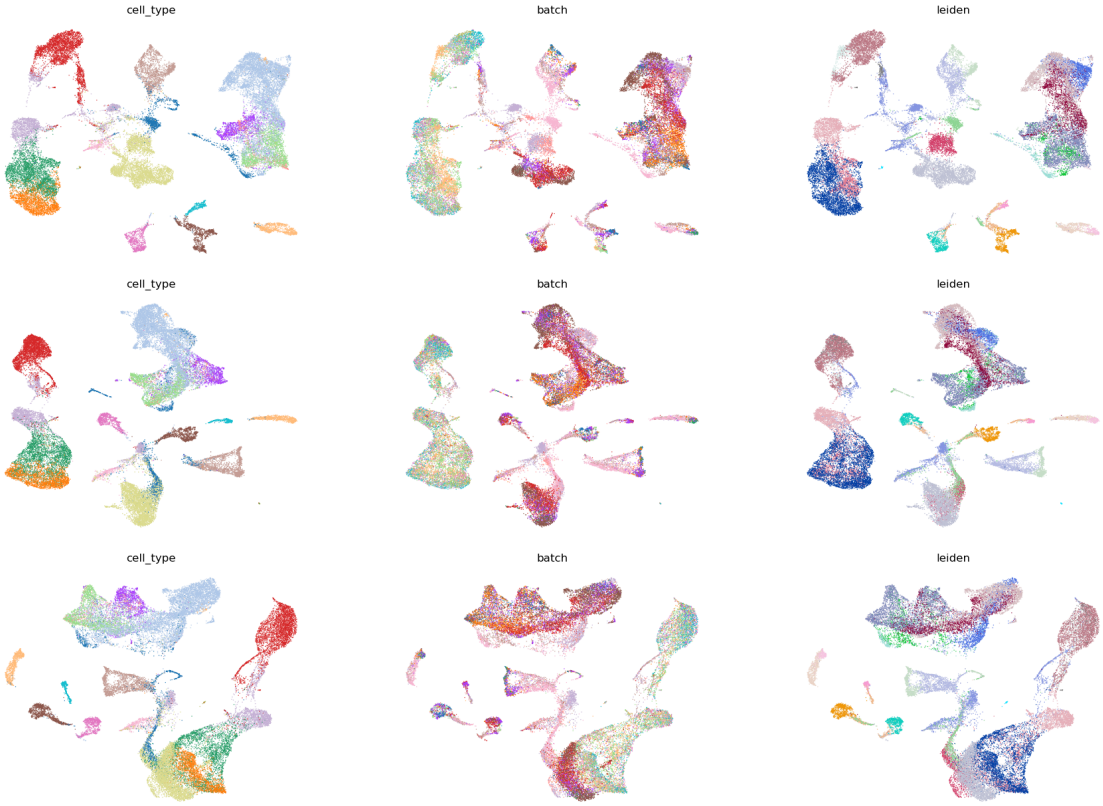


Figure 30: Umaps of latent space for epochs $9, 199, 399$ for the Vamp Prior.

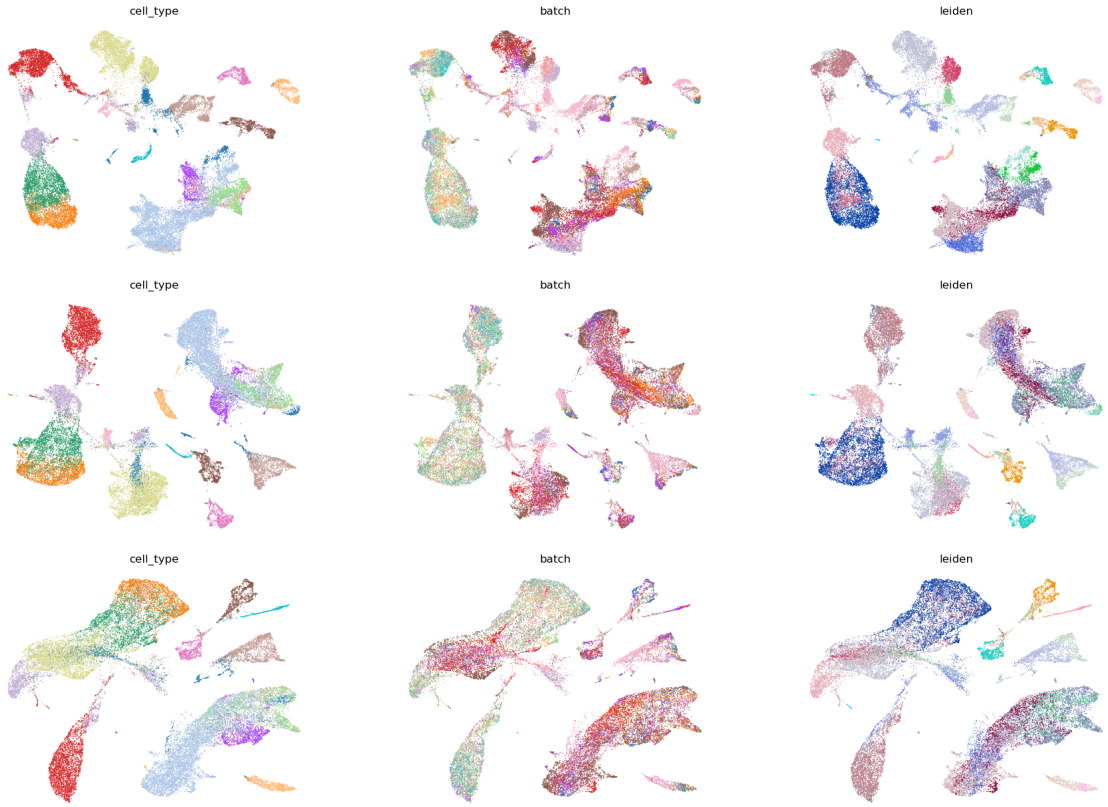


Figure 31: Umaps of latent space for epochs $9, 199, 399$ for the Normal Flow Prior.

E Further KL-term Scaling Results

Method	Bio conservation					Batch correction		Aggregate score		
	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	Silhouette batch	KBET	Batch correction	Bio conservation	Total
scVI_beta=1	0.76	0.60	0.68	0.55	0.61	0.86	0.35	0.61	0.64	0.63
scVI_beta=10	0.71	0.53	0.65	0.50	0.59	0.82	0.39	0.61	0.60	0.60
scVI_beta=0	0.75	0.60	0.66	0.49	0.58	0.85	0.32	0.58	0.61	0.60
scVI_beta=5	0.69	0.55	0.63	0.44	0.60	0.83	0.41	0.62	0.58	0.60
scVI_beta=25	0.55	0.36	0.50	0.29	0.53	0.81	0.39	0.60	0.45	0.51

Figure 32: Benchmark Metrics with different maximal values for the KL-term (beta) for the Vamp Prior.

	Bio conservation					Batch correction		Aggregate score		
Method	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	Silhouette batch	KBET	Batch correction	Bio conservation	Total
scVI_beta=1	0.75	0.66	0.71	0.58	0.62	0.83	0.36	0.60	0.66	0.64
scVI_beta=0	0.76	0.60	0.66	0.54	0.59	0.85	0.31	0.58	0.63	0.61
scVI_beta=5	0.70	0.57	0.66	0.52	0.60	0.80	0.40	0.60	0.61	0.61
scVI_beta=10	0.70	0.51	0.63	0.37	0.59	0.78	0.41	0.59	0.56	0.57
scVI_beta=25	0.57	0.35	0.54	0.34	0.50	0.68	0.38	0.53	0.46	0.49

Figure 33: Benchmark Metrics with different maximal values for the KL-term (beta) for the Normal Flow Prior.

F Further Benchmarking Prior Parameter Results

	Bio conservation					Batch correction		Aggregate score		
Method	Leiden NMI	Leiden ARI	KMeans NMI	KMeans ARI	Silhouette label	Silhouette batch	KBET	Batch correction	Bio conservation	Total
scVI_k=32	0.70	0.58	0.67	0.54	0.61	0.79	0.41	0.60	0.62	0.61
scVI_k=8	0.71	0.58	0.67	0.51	0.61	0.80	0.41	0.60	0.61	0.61
scVI_k=1	0.72	0.56	0.67	0.51	0.60	0.82	0.37	0.59	0.61	0.60
scVI_k=64	0.70	0.53	0.63	0.41	0.60	0.80	0.41	0.60	0.57	0.58
scVI_k=4	0.68	0.47	0.64	0.44	0.60	0.78	0.41	0.60	0.57	0.58
scVI_k=16	0.69	0.48	0.64	0.40	0.59	0.78	0.40	0.59	0.56	0.57

Figure 34: Normal Flow Prior benchmarked with a different number of transformation layers (k).

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Eigenständigkeitserklärung

Die unterzeichnete Eigenständigkeitserklärung ist Bestandteil jeder während des Studiums verfassten Semester-, Bachelor- und Master-Arbeit oder anderen Abschlussarbeit (auch der jeweils elektronischen Version).

Die Dozentinnen und Dozenten können auch für andere bei ihnen verfasste schriftliche Arbeiten eine Eigenständigkeitserklärung verlangen.

Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuer und Betreuerinnen der Arbeit.

Titel der Arbeit (in Druckschrift):

VAEs with learnable Priors for learning a Robust latent Space Representation of Single-Cell RNA Sequencing Data

Verfasst von (in Druckschrift):

Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich.

Name(n):	**Vorname(n):**
Diaz-Bone	Leander Paul

Ich bestätige mit meiner Unterschrift:

- Ich habe keine im Merkblatt „Zitier-Knigge“ beschriebene Form des Plagiats begangen.
- Ich habe alle Methoden, Daten und Arbeitsabläufe wahrheitsgetreu dokumentiert.
- Ich habe keine Daten manipuliert.
- Ich habe alle Personen erwähnt, welche die Arbeit wesentlich unterstützt haben.

Ich nehme zur Kenntnis, dass die Arbeit mit elektronischen Hilfsmitteln auf Plagiate überprüft werden kann.

Ort, Datum	**Unterschrift(en)**
Zürich, 12.08.2023	L. Diaz-Bone

Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich. Durch die Unterschriften bürgen sie gemeinsam für den gesamten Inhalt dieser schriftlichen Arbeit.