scVAE: Variational auto-encoders for single-cell gene expression data

Christopher H Grønbech[1], Maximillian F Vording[1], Pascal N Timshel[2,3], Capser K Sønderby[4], Tune H Pers[2,3], and Ole Winther[1,4]

Abstract

We propose a novel variational auto-encoder-based method for analysis of single-cell RNA sequencing (scRNA-seq) data. It avoids data preprocessing by using raw count data as input and can robustly estimate the expected gene expression levels and a latent representation for each cell. We show for several scRNA-seq data sets that our method outperforms recently proposed scRNA-seq methods in clustering cells. Our software tool scVAE has support for several count likelihood functions and a variant of the variational auto-encoder has a priori clustering in the latent space.

Keywords: gene expression; scRNA-seq; count data; machine learning; deep learning; Bayesian inference; generative modelling; variational auto-encoder.

Background

Single-cell RNA sequencing (scRNA-seq) enables measurement of gene expression levels at the cell level and thus provides a new framework to understand dysregulation of disease at the cell-type level [1]. To date, at number of methods have been developed to process gene expression data to normalise the data and cluster cells into putative cell types [2]. Seurat [3] is a popular method which is a multi-step process of normalisation, transformation, decomposition, embedding, and clustering of the gene expression levels. This can be cumbersome, and a more automated approach is desirable. Two recent methods, cellTree [4] and DIMM-SC [5], model gene expression levels directly as counts using either the latent Dirichlet allocation or Dirichlet-mixture generative models, respectively.

Here, we show that expressive deep generative models, leveraging the recent progress in deep neural networks, provide a powerful framework for modelling the data distributions of raw count data. We show that these models can learn biologically plausible representations of scRNA-seq experiments using only the highly sparse raw count data as input entirely skipping the normalisation and transformation steps needed in previous methods. Our approach is based on the variational auto-encoder (VAE) framework [6, 7]. These models learn compressed latent representations of the input data without any supervisory signals. We extend these models with likelihood (link) functions suitable for working with count data from RNA sequencing (RNA-seq) and perform extensive experiments on several data sets to analyse the strength and weaknesses of the models and the likelihood functions. Variational auto-encoders have been examined and extended [8–11], and they have been used in a variety of cases to, e.g., generate sentences [12], transfer artistic style of paintings [13], and create music [14]. Recently, a VAE was also used to model traditional multi-cell RNA-seq data [15].

Compared to traditional auto-encoders, VAEs have the advantage, because they are probabilistic models, that performance can be quantified in terms of the likelihood, which is especially useful for model comparison. Traditional auto-encoders also learn a latent representation from data that is used to reconstruct the same as well as new unseen data. The latent representation is low-dimensional, so the model is forced to only estimate the most important features of the data. Different auto-encoder models have previously been used to model normalised (or binarised) traditional gene expression levels: de-noising auto-encoders [16–18], sparse auto-encoders [19], and robust auto-encoders [20]. A generative adversarial network [21], which is a related model, was also recently used to model normalised single-cell gene expression levels [22].

Our contributions are threefold: 1) We have de-

[1]Section for Cognitive Systems, Department of Applied Mathematics and Computer Science, Technical University of Denmark
[2]The Novo Nordisk Foundation Center for Basic Metabolic Research, Faculty of Health and Medical Sciences, University of Copenhagen
[3]Department of Epidemiology Research, Statens Serum Institut
[4]Bioinformatics Centre, Department of Biology, University of Copenhagen

veloped new generative models based on the VAE framework for directly modelling raw counts from RNA-seq data; 2) we show that our Gaussian-mixture VAE (GMVAE) learns biologically plausible groupings of scRNA-seq data of higher quality than previous methods; and 3) we provide a publicly available framework for unsupervised modelling of count data from RNA-seq experiments.

Results

Both the standard VAE and the GMVAE have been used to model both single-cell and traditional RNA-seq data sets (Table 1). The performance of different likelihood functions have been investigated, and the latent spaces of the models have been examined.

scRNA-seq data of purified immune cells

The first data set we modelled was of single-cell gene expression levels of peripheral blood mononuclear cells (PBMC) [23]. We also modelled two subsets of the PBMC data set: one of lymphocytes (PBMC-L) and another of T cells (PBMC-T).

To assess the optimal network architectures for these data sets, we first trained VAE models using a negative binomial (NB) likelihood function (VAE-NB models) with different network architecture on the three data sets with all genes included. This was carried out as grid searches of number of hidden units and latent dimensions (see Additional file 1: Figure S1). We found that using two smaller hidden layers (of 100 units each) in the generative and inference processes yielded better results than only using one hidden layer for all data sets. For the full PBMC data set, a high-dimensional latent space of 100 dimensions result in the highest test marginal log-likelihood lower bound, whereas for the two smaller subsets, a lower-dimensional latent space of 25 dimensions gave the best lower bound.

With these network architectures different discrete likelihood functions was examined for each data set. The marginal log-likelihood lower bound as well as the adjusted Rand index for the test sets are listed in Table 2. From this table it is clear that the using the negative binomial distribution yielded the highest lower bound for all data sets followed closely by its zero-inflated version (ZINB). For the Rand indices, however, using the constrained Poisson (CP) distribution gave the highest values for the subsets, especially for the T cells, while a straight Poisson distribution yield the highest index for the full data set with the constrained version as a close second. It should be noted, however, that a high lower bound does not necessarily result in a high Rand index. For instance were the lower bounds and Rand indices for the full data set not strongly correlated with, e.g., a sample correlation coefficient of $r = -0.0903$ (see Additional file 1: Figure S2).

To test whether a more complex model is required or a simpler model is sufficient in modelling the data, the network architecture and likelihood function yielding the highest marginal log-likelihood lower bound for the VAE model for each data set were used to train GMVAE models as well as factor-analysis (FA) models. The resulting lower bounds and Rand indices are listed in Table 2. The FA models had worse lower bounds compared to their VAE equivalents, but the Rand indices are significantly higher (except for the PBMC-L data set), whereas the GMVAE models improved both the lower bounds and the Rand indices. This shows that non-linear models really makes a difference for the marginal log-likelihood of the data. The latent space of the GMVAE-NB model for the test set of the full PBMC data set can be seen in Figure 1, which shows different clusters corresponding to distinct cell types, while more similar cell types are clustered together.

To see if we can achieve better results by limiting the genes we use in our models, we also trained and tested models using only the 100 most varying genes for all data sets and the 800 most varying genes for the PBMC and the PBMC-T data sets using the same procedure as above. Different network architectures were examined (see Additional file 1: Figures S3–S4), and different likelihood functions and models were investigated using the optimal network architectures (see Additional file 1: Tables S1–S2). Using a constrained Poisson distribution as the likelihood function produced the highest lower bounds, whereas different likelihood functions produced the best Rand indices. The GMVAE models have again higher lower bounds than their corresponding VAE and FA models, but also higher Rand indices.

Sun *et al.* [5] have reported adjusted Rand indices for both PBMC-L and PBMC-T data sets using variations of their DIMM-SC model as well as the Seurat [3], cellTree [4], and k-means clustering models (see Additional file 1: Table S1–S2 for performance of these models). For the PBMC-L data set, the highest Rand index achieved by these models was a DIMM-SC model was 0.990 using the 100 most varying genes. Our GMVAE-CP model yielded a Rand index of 0.9990 using the same genes (see Additional file 1: Figure S5 for its latent space), and two models, the VAE-ZINB and the VAE-CP models, resulted in Rand indices of 1.000 using all genes. The highest Rand index that the other models achieved for the PBMC-T data set using the 100 most varying genes was

Table 1: Overview of gene expression data sets.

Data set	Selection	Examples	Features	Classes	Sparsity [%]
Purified immune cells	All cells	94 655	32 738	10	98.06
	T cells	32 695	32 378	3	98.39
	Lymphocytes	28 733	32 378	3	98.16
Mouse brain cells	All cells	1 306 127	27 998	—	92.82
	20k	20 000	27 998	—	92.82
TCGA	All gene IDs	10 830	60 498	29	53.83
	All gene names	10 830	58 581	29	52.47

Examples refer to cells or cell samples, features refer to gene names or gene IDs, while classes refer to either cell types or tissue sites. The T cells data subset contains CD8+/CD45RA+ naïve cytotoxic T cells, CD4+/CD25+ regulatory T cells, and CD4+/CD45RA+/CD25- naïve T cells, while the lymphocytes data subset contains CD56+ natural killer cells, CD19+ B cells, and CD4+/CD25+ regulatory T cells.

Table 2: Test metrics for the purified immune cells data sets.

		PBMC		PBMC-T		PBMC-L	
Model	Likelihood function	$\mathcal{L}^{\text{test}}$	$R^{\text{test}}_{\text{adj}}$	$\mathcal{L}^{\text{test}}$	$R^{\text{test}}_{\text{adj}}$	$\mathcal{L}^{\text{test}}$	$R^{\text{test}}_{\text{adj}}$
VAE	Poisson (P)	−2074.1	**0.5635**	−1774.5	0.5485	−1998.4	0.9737
	Negative binomial (NB)	**−2066.2**	0.4276	**−1769.6**	0.5449	**−1989.7**	0.9917
	Zero-inflated Poisson (ZIP)	−2068.7	0.5297	−1771.7	0.5929	−1992.4	0.9990
	Zero-inflated negative binomial (ZINB)	−2066.4	0.5319	−1770.3	0.5577	−1990.0	**1.000**
	Piece-wise categorical Poisson (PCP)	−2079.6	0.4819	−1778.3	0.4824	−2001.8	0.9928
	Constrained Poisson (CP)	−2100.3	0.5543	−1795.1	**0.8240**	−2019.2	**1.000**
FA	Negative binomial	−2083.0	**0.6207**	−1785.0	**0.8146**	−2010.2	0.9969
GMVAE	Negative binomial	**−2064.5**	0.4664	**−1768.8**	0.5563	**−1986.9**	**0.9990**

The test marginal log-likelihood lower bounds (in nats), $\mathcal{L}^{\text{test}}$, and the adjusted Rand indices, $R^{\text{test}}_{\text{adj}}$, of the PBMC data sets evaluated using different likelihood functions for the standard VAE and using the most promising likelihood function for the FA and GMVAE models. The highest lower bounds and Rand indices for each data set and model have been highlighted in bold.

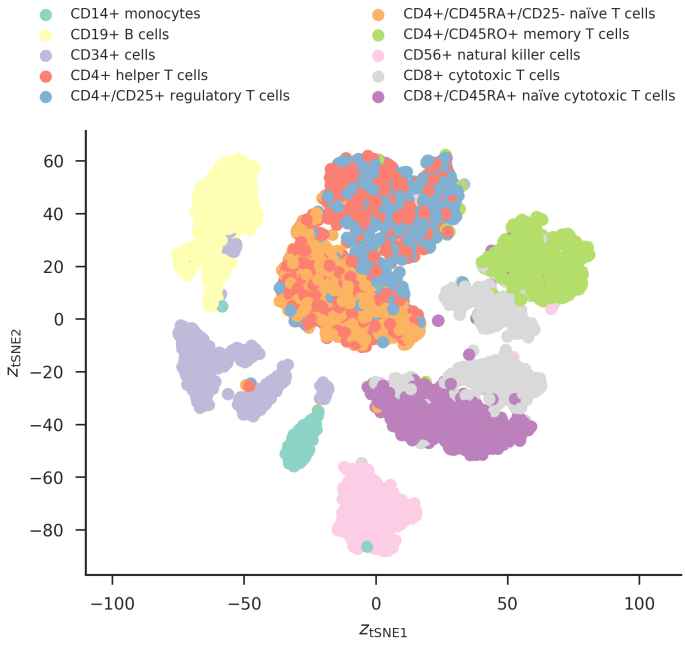

Figure 1: Latent space of the GMVAE-NB trained and evaluated on the PBMC data set. The encoding of the cells in the latent space has been embedded in two dimensions using t-SNE and are colour-coded using their cell types. Clear separations can be seen corresponding to different cell types, but some similar cell types are also clustered close together or mixing together.

Table 3: Test metrics for the mouse brain cells data sets.

Data set	Model	Likelihood	$\mathcal{L}^{\text{test}}$
MB-20k	VAE	P	−5553.01
		NB	−5517.0
		ZIP	−5538.3
		ZINB	**−5516.4**
		PCP	−5586.7
		CP	−5595.3
	GMVAE	ZINB	**−5508.6**
MB-1M	VAE	ZINB	−5398.2
	GMVAE	ZINB	**−5396.3**

The test marginal log-likelihood lower bound (in nats), $\mathcal{L}^{\text{test}}$, for the MB-1M data sets evaluated using different likelihood distributions for the standard VAE and using the most promising likelihood function for the GMVAE. The highest lower bounds for each data set and model have been highlighted in bold.

Table 4: Test metrics for the TCGA data set.

Model	Likelihood	$\mathcal{L}^{\text{test}}/10^5$	$R^{\text{test}}_{\text{adj}}$
VAE	P	−337.84	0.6517
	NB	**−1.6187**	0.6176
	ZIP	−338.08	0.6470
	ZINB	−1.6220	0.6673
	PCP	−337.98	0.6421
	CP	−72.615	**0.6995**
GMVAE	NB	−1.6194	0.3611

The test marginal log-likelihood lower bound (in nats), $\mathcal{L}^{\text{test}}$, as well as the adjusted Rand index, $R^{\text{test}}_{\text{adj}}$, for the feature-mapped TCGA data set evaluated using different likelihood distributions for the standard VAE and using the most promising likelihood function for the GMVAE. The highest lower bound and Rand index have been highlighted in bold.

0.408 by a DIMM-SC model. Our GMVAE-CP model yielded a Rand index of 0.7076 using the same genes. Using the 800 most varying genes, the highest Rand index attained by the other models was 0.556 for the best DIMM-SC model, while it was 0.8452 for the GMVAE-CP model (see Additional file 1: Figure S6 for its latent space).

scRNA-seq data of mouse brain cells

Next, we considered the second single-cell gene expression data set [24], which consists of gene expression levels for 1.3 million mouse brain cells (MB-1M). Since no cell labels are available for this data set, the adjusted Rand index cannot be evaluated.

We can scale the model to train on the full MB-1M data set without any modifications. However, because of the large number of cells, we used the smaller subset of 20 000 cells (MB-20k) to perform a network architecture grid search (see Additional file 1: Figure S7) as well as to test the different likelihood functions (see Table 3) following the same procedure as in the previous section. The highest test marginal log-likelihood lower bound was achieved using the zero-inflated negative binomial distribution with the regular negative binomial as a close second. A GMVAE model with 10 components (clusters) was also trained on the subset using this configuration, and this model improved upon the lower bound (see Additional file 1: Figure S8 for its latent space).

For the full data set a VAE and a GMVAE model were trained using the optimal configuration found for the subset, and the lower bounds using these models are listed in Table 3. The GMVAE model achieved the highest lower bound of the two, and its latent space have been plotted in Figure 2, which shows somewhat clear regions for each cluster and even some separation for some clusters. To train each model one epoch on the full data set took approximately 8 minutes for the VAE model and approximately 26 minutes for the GMVAE model on a GeForce GTX 1080 Ti graphics card.

Traditional RNA-seq data of human cancer cells

Lastly, we modelled the traditional RNA-seq data set of human cancer cells from TCGA [25]. As with the previous data sets different network architectures (see Additional file 1: Figure S9) and likelihood functions (see Table 4) were investigated. There was a clear difference in marginal log-likelihood lower bound between using either of the negative binomial distributions from any of the Poisson distributions. The constrained Poisson distribution, however, achieved a significantly higher lower bound than the other Poisson distributions. The adjusted Rand indices were much closer together in value for the different likelihood functions with the constrained Poisson distribution having the highest index. A GMVAE model with negative binomial distribution was also trained, but it did not produce better results than its VAE equivalent, especially not for the Rand index. The latent space of the GMVAE-NB model can be seen in Figure 3, where samples belonging to multiple tissue sites are clearly separated. The latent space for the VAE-NB model looks very similar despite the difference in Rand index (see Additional file 1: Figure S10).

Discussion

We show that VAEs can be used to model scRNA-seq data. The GMVAE model achieves better marginal log-likelihood lower bounds as well as higher Rand indices compared to both a corresponding VAE model as well as previous results,

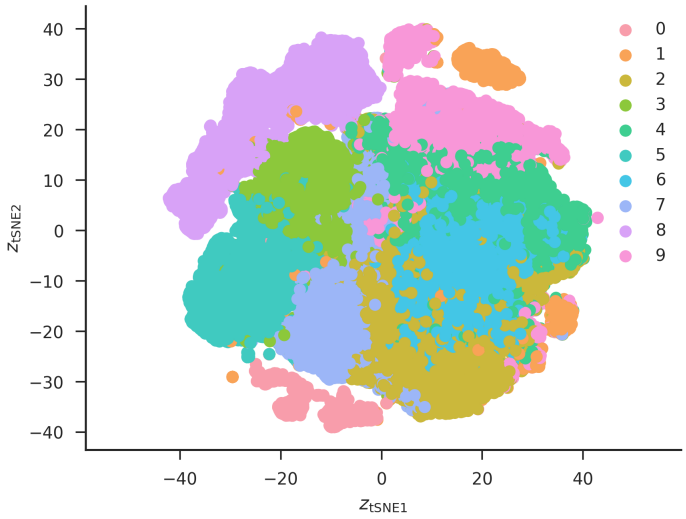

Figure 2: Latent space of the GMVAE-ZINB trained and evaluated on the MB-1M data set. Ten clusters was used in the model, and the encoding of the cells in the latent space has been embedded in two dimensions using t-SNE and are colour-coded using the clusters to which they belong. Different regions can be seen with even some separation for some clusters.

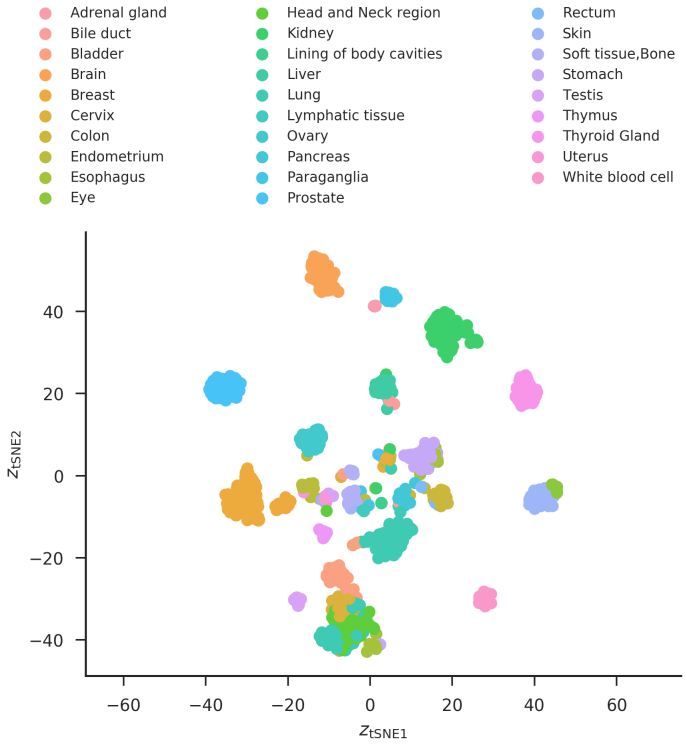

Figure 3: Latent space of the GMVAE-NB trained and evaluated on the feature-mapped TCGA data set. The encoding of the cells in the latent space has been embedded in two dimensions using t-SNE and are colour-coded using the tissue sites to which they belong. Many of the clusters are clearly separated and they correspond to different tissue sites.

and the latent spaces for the GMVAE also show clear separation according to cell types. The GMVAE model, however, struggles with traditional RNA-seq data, but this might be explained by the diversity within each data sample.

In modelling three different data sets and subsets thereof, we have found the following guidelines help in achieving a higher marginal log-likelihood lower bound score on a data set. The network architecture should adapt to the data set: The more cells in the data set, the larger the network should be to better capture more subtle co-variation between genes. The network should also be larger if several cell types are present to capture this variability, or if preprocessing leaves the data set less sparse, which is the case for, e.g., selecting the most varying genes. The models for all data sets also benefited from deeper network architectures. For the likelihood function, the negative binomial distribution (or in one case, its zero-inflated version) yielded the highest values of the lower bound when using all genes, especially for the traditional RNA-seq data set, while the constrained Poisson distribution prevailed when limiting to the most varying genes. We have also shown that using a simpler linear-factor model for the generative process lead to significantly worse lower bounds, demonstrating that non-linear transformations can more easily express more subtle co-variation patterns in the data sets.

Its built-in clustering helps the GMVAE achieve a better log likelihood lower bound than the standard VAE. Compared to the DIMM-SC model by Sun *et al.* [5], the GMVAE models and some VAE models achieve a higher adjusted Rand index. We found, however, that the correlation between lower bounds and Rand indices to be weak. We also showed that both models are able to scale up to very large data sets like the data set of 1.3 million mouse brain cells from 10x Genomics.

During the submission process and independent of our work, two recent articles [26, 27] have proposed methodology quite similar to ours. Both focus on the zero-inflated negative binomial likelihood function. One [26] uses variational inference like we propose here, and the other [27] uses a bottleneck non-probabilistic auto-encoder.

Conclusions

We show that the two proposed variational auto-encoders are able to model gene expression counts using appropriate discrete probability distribution as likelihood (link) functions and provide a software implementation. These models are probabilistic and put a lower bound on the marginal likelihood of the data sets, enabling us to examine and compare different link functions. We have applied both models successfully to both single-cell and traditional RNA-seq data sets. Building clustering into the Gaussian-mixture variational auto-encoder, we have a model that can sort cells into cell populations with little human interaction compared to earlier more involved models. Having both an inference process and a generate process, makes it possible to project new data onto an existing latent space, or even generate new data from samples in the latent space. This means that new cells can be introduced to an already trained model, and it could enable combining the latent representations of two cells to generate a cell and the transitional states in between.

In the future, we would like to make the models more flexible by adding more latent variables [10] and making the models adaptable to the number of clusters [28]. We could also use semi-supervised learning and active learning to better classify cells and identify cell populations. This would also help with transfer learning enabling modelling multiple data sets with the same model. Since these models are generative, it should also be possible to combine encodings of the cells in the latent space a produce in-between cells like Campbell and Kautz [29], who used a Gaussian-process latent-variable model on fonts [30]. We note that it is possible to apply these models to data sets with multiple modalities such as RNA-seq and exome sequencing [31].

Methods

We have developed generative models for directly modelling the raw read counts from scRNA-seq data. In this section, we describe the models as well as the different data likelihood (link) functions for this task.

Latent-variable models

We take a generative approach to modelling the data-generating process of the count data vector $\mathbf{x}$. We assume that $\mathbf{x}$ is drawn from the distribution $p_{\boldsymbol{\theta}}(\mathbf{x}) = p(\mathbf{x} \,|\, \boldsymbol{\theta})$ parameterised by $\boldsymbol{\theta}$. In this case, $\mathbf{x}$ represents a single cell and its components x_n, which are also called *features*, correspond to the gene expression count for gene n. The number of features, N, is very high, but we would still want to be able to model co-variation between the features. We achieve this by introducing a stochastic latent variable $\mathbf{z}$, with fewer dimensions than $\mathbf{x}$, and condition the data-generating process on this. The joint probability distribution of $\mathbf{x}$ and $\mathbf{z}$ is then

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{z}), \tag{1}$$

where $p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z})$ is the likelihood function and $p_{\boldsymbol{\theta}}(\mathbf{z})$ is the prior probability distribution of $\mathbf{z}$. We have

considerable freedom to choose these distributions. We will consider a deep neural network mapping from $\mathbf{z}$ to (the sufficient statistics) of $\mathbf{x}$. Marginalising out $\mathbf{z}$ results in the marginal likelihood for one data point of $\boldsymbol{\theta}$:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{z}) \,\mathrm{d}\mathbf{z}. \tag{2}$$

The log-likelihood function for all data points (also called *examples*, which in this case are cells) will be the sum of the log-likelihoods for each data point: $\mathcal{F}(\boldsymbol{\theta}) = \sum_{m=1}^{M} \log p_{\boldsymbol{\theta}}(\mathbf{x}_m)$, where M is the number of cells. We can then use maximum-likelihood estimation, $\arg\max_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta})$, to infer the value $\boldsymbol{\theta}$.

Deep generative models

For all but the simplest linear models, the marginalisation over the latent variables in equation (2) is intractable. However, we want to use flexible non-linear functions in order to model complex data. Variational auto-encoders [6, 7] solve this by using an auto-encoding variational Bayesian optimisation algorithm as described below.

The non-linear transformations used in a variational auto-encoder are most often deep neural networks (or multi-layer perceptrons, MLPs). The choice of likelihood function $p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z})$ depends on the statistical properties of the data, e.g., continuous or discrete observations, sparsity, as well as computational tractability. Contrary to most other VAE-based articles considering either continuous or categorical input data, our goal is to model discrete count data directly, which is why Poisson or negative binomial likelihood functions are natural choices. This will be discussed in more detail below. The prior over the latent variables $p(\mathbf{z})$ is usually chosen to be an isotropic standard multivariate Gaussian distribution to get the following generative process (illustrated in Fig. 4):

$$p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z}) = f(\mathbf{x}; \boldsymbol{\lambda}_{\boldsymbol{\theta}}(\mathbf{z})), \tag{3a}$$

$$p_{\boldsymbol{\theta}}(\mathbf{z}) = \mathcal{N}\big(\mathbf{z}; \mathbf{0}, \mathbf{I}\big), \tag{3b}$$

where f is a discrete distribution such as the Poisson: $f'(x; \lambda) = \mathrm{e}^{-\lambda} \frac{\lambda^x}{x!}$ and $f(\mathbf{x}; \boldsymbol{\lambda}) = \prod_k f'(x_k; \lambda_k)$. The Poisson rate parameters are functions of $\mathbf{z}$: $\boldsymbol{\lambda}_{\boldsymbol{\theta}}(\mathbf{z})$. We can for example parameterise it by a single-layer feedforward neural network:

$$\boldsymbol{\lambda}_{\boldsymbol{\theta}}(\mathbf{z}) = h\big(\mathbf{W}\mathbf{z} + \mathbf{b}\big). \tag{4}$$

Here, $\mathbf{W}$ and $\mathbf{b}$ are the weights and bias of a linear model, $\boldsymbol{\theta} = (\mathbf{W}, \mathbf{b})$, and $h(\cdot)$ is an appropriate element-wise non-linear transformation to make the rate non-negative. The Poisson likelihood function can be substituted by other probability distributions with parameters of the distribution being non-linear functions of $\mathbf{z}$ in the same fashion as above (see below). In order to make the model more expressive we can also replace the single-layer model with a deep model with one or more hidden layers. For L layers of adaptable weights we can write: $\mathbf{a}_l = h_l(\mathbf{W}_l \mathbf{a}_{l-1} + \mathbf{b}_l)$ for $l = 1, \ldots, L$, $\mathbf{a}_0 = \mathbf{z}$ and $\boldsymbol{\lambda}_{\boldsymbol{\theta}}(\mathbf{z}) = \mathbf{a}_L$ with $h_l(\cdot)$ denoting the activation function of the lth layer. For the hidden layers, the rectifier function, $\mathrm{ReLU}(x) = \max(0, x)$, is often a good choice. The reconstruction $\tilde{\mathbf{x}}$ is the mean of this probability distribution: $\tilde{\mathbf{x}} = \mathrm{E}_{\mathbf{x} \,|\, \mathbf{z}}\big[\mathbf{x}\big]$, which can be computed analytically from the parameters. The variance of the reconstruction can also be computed in similar fashion.

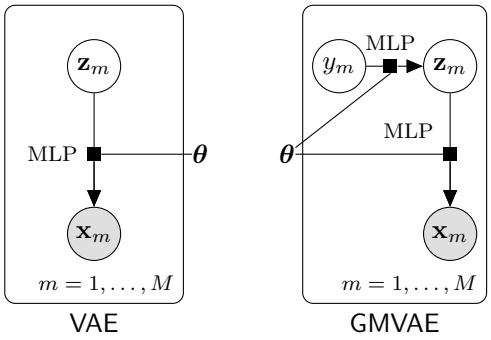

Figure 4: Model graphs of the generative processes. The generative processes of (*left*) the standard variational auto-encoder and (*right*) the Gaussian-mixture variational auto-encoder. Grey circles signify observable variables and white circles represent latent variables. The black squares denote the functions next to them with the variables connected by lines as input and the variable connected by an arrow as output.

Gaussian-mixture variational auto-encoder

Using a Gaussian distribution as the prior probability distribution of $\mathbf{z}$ only allows for one mode in the latent representation. If there is an inherent clustering in the data, like for scRNA-seq data where cells represent different cell types, it would be desirable to have multiple modes – one for every cluster or class. This can be implemented by using a Gaussian-mixture model in place of the Gaussian distribution. Following previous work [7–9, 32], a discrete latent variable y is introduced to distinguish between the clusters and $\mathbf{z}$ is conditioned on this. The joint probability distribution of $\mathbf{x}$, y, and $\mathbf{z}$ is then factorised in the following way:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, y, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{z} \,|\, y) p_{\boldsymbol{\theta}}(y). \tag{5}$$

As before, the latent variables, y and $\mathbf{z}$, can be marginalised out to give the marginal likelihood:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_y \int p_{\boldsymbol{\theta}}(\mathbf{x}, y, \mathbf{z})\, \mathrm{d}\mathbf{z}. \tag{6}$$

For $\mathbf{z}$ to follow a Gaussian-mixture model, the mean and variance should be dependent on y, and y should follow a categorical distribution. Using a Poisson likelihood again, the generative process becomes

$$p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z}) = f(\mathbf{x}; \boldsymbol{\lambda}_{\boldsymbol{\theta}}(\mathbf{z})), \tag{7a}$$
$$p_{\boldsymbol{\theta}}(\mathbf{z} \,|\, y) = \mathcal{N}\big(\mathbf{z}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(y), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(y)\mathbf{I}\big), \tag{7b}$$
$$p_{\boldsymbol{\theta}}(y) = \mathrm{Cat}\,\big(y; \boldsymbol{\pi}\big). \tag{7c}$$

Here, $\boldsymbol{\pi}$ is a K-dimensional probability vector, where K is the number of components in the Gaussian-mixture model. The component π_k of $\boldsymbol{\pi}$ is the mixing coefficient of the kth Gaussian distribution, quantifying how much this distribution contributes to the overall probability distribution. Without any prior knowledge the categorical distribution is fixed to be uniform. Fig. 4 shows the generative process for the Gaussian-mixture variational auto-encoder (GMVAE). The reconstruction $\tilde{\mathbf{x}}$ is computed for each cluster as before and then averaged using the mixing coefficients: $\tilde{\mathbf{x}} = \mathrm{E}_y\left[\mathrm{E}_{\mathbf{x}\,|\,\mathbf{z}}\left[\mathbf{x}\right]\right]$. The variance of the reconstruction can again be computed in similar fashion.

Modelling gene expression count data

Instead of normalising and transforming the gene expression data, the original transcript counts are modelled directly to take into account the total amount of genes expressed in each cell also called the *cell depth*. To model count data, the likelihood function $p_{\boldsymbol{\theta}}(\mathbf{x} \,|\, \mathbf{z})$ will need to be discrete and only have non-negative support. We will consider a number of such distributions in the following and investigate which ones are best in term of likelihood on held-out data. The Poisson (P) distribution is chosen as the first potential candidate. Gene expression data are also generally over-dispersed [33], so the negative binomial (NB) distribution is also tested.

To properly handle the sparsity of the scRNA-seq data [34], we test two approaches: a zero-inflated distribution and modelling of low counts using a discrete distribution. A zero-inflated distribution adds an additional parameter, which controls the amount of excessive zeros added to an existing probability distribution. For the Poisson distribution, $f(x; \lambda)$, the zero-inflated version (ZIP) is defined as

$$f(x; \rho, \lambda) = \begin{cases} \rho + (1-\rho) f(x; \lambda), & x = 0, \\ (1-\rho) f(x; \lambda), & x > 0, \end{cases} \tag{8}$$

where ρ is the probability of excessive zeros. The zero-inflated negative binomial (ZINB) distribution have an analogous expression. Both these zero-inflated version have also been examined.

The other method is to model low counts using a discrete distribution and using a second probability distribution to model the higher counts. Using the Poisson distribution in the latter case, leads to the following distribution:

$$f(x; \boldsymbol{\tau}, \lambda) = \begin{cases} \tau_k, & x = k, 0 \leq x < K, \\ \tau_K g(x - K; \lambda), & x \geq K, \end{cases} \tag{9}$$

where K is the cut-off between low and high counts, and $\boldsymbol{\tau}$ is a $K+1$-dimensional probability vector with components τ_k quantifying the probability of a count being equal to k for low counts or K for high counts. This distribution we call a piece-wise categorical Poisson (PCP) distribution, and it has been used to predict demand for large quantities [35].

Since the cell depth varies for cells [34], a variation of the Poisson distribution called the constrained Poisson distribution is also examined. This distribution has been used to model word counts in documents of varying length [36], which corresponds to gene expression levels in cells. For the constrained Poisson distribution, the rate parameter λ_n for each count x_n of gene n is reparameterised as

$$\lambda_n = D p_n, \tag{10}$$

where D is the cell depth and p_n is the proportional contribution for gene n to the cell depth. This ensures the cell depth is reconstructed correctly and builds normalisation directly into the model. Since $\sum_n p_n = 1$, this also couples the counts for different genes in each cells, whereas the other probability distributions model each count individually.

As described earlier, parameters for these probability distributions are modelled using deep neural networks. Since we do not know the true conditioning structure of the genes, we make the simplifying assumption that they are independent for computational reasons and therefore use feedforward neural networks.

Variational auto-encoders

In order to train and evaluate deep generative models, we need approximative inference. Here we will use variational auto-encoders which amounts to replacing the intractable marginal likelihood with its variational lower bound and estimate the intractable integrals with low-variance Monte Carlo estimates. The lower bound is optimised for different models on the training set and after training evaluated on a test set in order to perform model comparison.

Since the likelihood function $p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})$ for the standard VAE is modelled using non-linear transformations, the posterior probability distribution $p_{\boldsymbol{\theta}}(\mathbf{z}\,|\,\mathbf{x}) = p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z})/p_{\boldsymbol{\theta}}(\mathbf{x})$ becomes intractable. Variational auto-encoders use a variational Bayesian approach where $p_{\boldsymbol{\theta}}(\mathbf{z}\,|\,\mathbf{x})$ is replaced by an approximate probability distribution $q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})$ modelled using non-linear transformations parameterised by $\boldsymbol{\phi}$. Thus, the marginal log-likelihood can be written as

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \mathrm{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) \,\middle\|\, p_{\boldsymbol{\theta}}(\mathbf{z}\,|\,\mathbf{x})\right) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}). \tag{11}$$

Here, the first term is the Kullback–Leibler (KL) divergence between the true and the approximate posterior distributions. The KL divergence is a non-negative measure of the dissimilarity between two probability distributions, and it is only zero for identical probability distributions. However, since the true posterior distribution is intractable, this KL divergence cannot be evaluated, but because it is non-negative, it does put a lower bound on the marginal log-likelihood: $\log p_{\boldsymbol{\theta}}(\mathbf{x}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$, which is why $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x})$ is called the marginal log-likelihood lower bound. This can be decomposed into two terms as well:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = &\mathrm{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] \\ &- \mathrm{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,\middle\|\, p_{\boldsymbol{\theta}}(\mathbf{z})\right),\end{aligned} \tag{12}$$

where the first term is the *expected negative reconstruction error*, which measures how good the model can reconstruct $\mathbf{x}$. The second term is the *relative KL divergence* between the approximate posterior distribution $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ and the prior distribution $p_{\boldsymbol{\theta}}(\mathbf{z})$ of $\mathbf{z}$. Since both terms are integrals over $\mathbf{z}$, these are still analytically intractable, so they are evaluated using Monte Carlo sampling instead. Compared to estimating the marginal likelihood directly by for example importance sampling, the variance of these Monte Carlo estimators have much lower variance because they involve averages over *logarithms* of distributions and not the distributions themselves. The KL divergence can, however, be computed analytically for two Gaussian distributions. For the standard VAE, the approximate posterior distribution is chosen to be a multivariate Gaussian distribution with a diagonal covariance matrix:

$$q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) = \mathcal{N}\big(\mathbf{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x})\mathbf{I}\big), \tag{13}$$

where $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})$ and $\boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x})$ are non-linear transformations of $\mathbf{x}$ parameterised by $\boldsymbol{\phi}$. Using a single-layer feedforward neural network for each, they are computed as

$$\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}) = \mathbf{u}\Big(\mathbf{W}^{(\boldsymbol{\phi})}_{\boldsymbol{\mu}}\mathbf{x} + \mathbf{b}^{(\boldsymbol{\phi})}_{\boldsymbol{\mu}}\Big), \tag{14a}$$

$$\boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x}) = \mathbf{v}\Big(\mathbf{W}^{(\boldsymbol{\phi})}_{\boldsymbol{\sigma}^2}\mathbf{x} + \mathbf{b}^{(\boldsymbol{\phi})}_{\boldsymbol{\sigma}^2}\Big). \tag{14b}$$

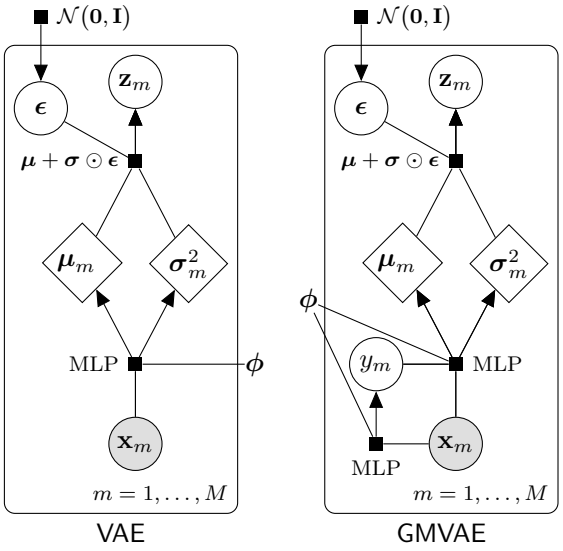

Figure 5: Model graphs of the inference processes. The inference processes of (*left*) the standard variational auto-encoder and (*right*) the Gaussian-mixture variational auto-encoder. See Fig. 4 for explanation of symbols. In addition, the rhombi symbolise deterministic variables.

As in equation (4), $\mathbf{W}^{(\boldsymbol{\phi})}$ and $\mathbf{b}^{(\boldsymbol{\phi})}$ are the weights and biases of linear models, while $\mathbf{u}(\cdot)$ and $\mathbf{v}(\cdot)$ are appropriate non-linear transformations. Equation (13) is called the inference process, and it is illustrated in Fig. 5.

Both processes are optimised simultaneously using a stochastic gradient descent algorithm, and a reparameterisation trick for sampling from $q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})$ is used that allows back-propagation of the gradients in the optimisation algorithm. The reported marginal log-likelihood lower bound is the mean over all examples.

Because of the additional latent variable, the marginal log-likelihood lower bound for the GMVAE has added complexity:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}) = &\mathrm{E}_{q_{\boldsymbol{\phi}}(y|\mathbf{x})}\Big[\mathrm{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x},y)}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] \\ &\qquad - \mathrm{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,\middle\|\, p_{\boldsymbol{\theta}}(\mathbf{z}|y)\right)\Big] \\ &- \mathrm{KL}\left(q_{\boldsymbol{\phi}}(y|\mathbf{x}) \,\middle\|\, p_{\boldsymbol{\theta}}(y)\right).\end{aligned} \tag{15}$$

The first term is the standard VAE lower bound for each class y averaged over all classes, and the second term is the KL divergence for y. The two approximate posterior distributions in the inference process are

$$q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}, y) = \mathcal{N}\big(\mathbf{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}, y), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x}, y)\mathbf{I}\big), \tag{16a}$$

$$q_{\boldsymbol{\phi}}(y|\mathbf{x}) = \mathrm{Cat}\left(y; \boldsymbol{\pi}_{\boldsymbol{\phi}}(\mathbf{x})\right). \tag{16b}$$

Here, the kth component of $\boldsymbol{\pi}_{\boldsymbol{\phi}}(\mathbf{x})$ is the responsibility of the kth Gaussian distribution for $\mathbf{x}$, quantifying how much this distribution contributes to

the overall probability of $\mathbf{x}$. The inference process for the GMVAE is shown in Figure 5.

Adjusted Rand index

The adjusted Rand index [37], R_{adj}, is used to measure the similarity between two different clusterings of the same data set. Two identical clusterings have $R_{\text{adj}} = 1$, while the expected value of the adjusted Rand index is 0 (it is not bounded below).

Because the VAE have no built-in clustering, k-means clustering is used to cluster the latent representations of the cells fro this model. For the Gaussian-mixture VAE, each cell sample can directly be a assigned to a cluster by picking the mixture component with the highest responsibility.

We note that the adjusted Rand index is not optimised during the optimisation algorithm and is computed only at the end.

Visualising the latent space

To visualise the latent space in which the latent representations reside, the mean values of the approximate posterior distribution $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ are plotted instead of the samples. This is done to better get a representation of the probability distribution for the latent representation of each cell. For the standard VAE, this is just $\bar{\mathbf{z}} = \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})$, and for the GMVAE, $\bar{\mathbf{z}} = \mathrm{E}_y\left[\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}, y)\right]$. To visualise higher dimensional latent spaces, we projected to two dimensions using t-distributed stochastic neighbour embeddings [38].

RNA-seq data sets

We model three different RNA-seq data sets as summarised in Table 1. The first data set is of single-cell gene expression levels of peripheral blood mononuclear cells (PBMC) [23]. It is published by 10x Genomics as ten data sets of different purified cell types [39–48], and we use the filtered gene–cell matrices for each data set. One of these data sets contains two different cell types, but since there are no separation of the two in the data set, we only use the ten cell types assigned by 10x Genomics. We combine these ten data sets into one single data set of purified immune cells (PBMC). Following Sun *et al.* [5], we also make two smaller subsets of the purified immune cell types: one subset of lymphocytes with three distinct cell types (PBMC-L) and one subset of T cells with three similar cell types (PBMC-T). The specific cell types for each subset are listed in Table 1. This table also shows the high sparsity of the data sets.

The second data set is also a single-cell gene expression data set made publicly available by 10x Genomics [24], and we again use the filtered gene–cell matrix. This data set contains gene expression levels for 1.3 million mouse brain cells (MB-1M), but in contrast to the PBMC data set, the cells were not purified so this data set does not contain any information about cell types. A smaller subset of 20 000 uniformly randomly selected cells (MB-20k) is also provided by 10x Genomics [24].

The last data set is a traditional RNA-seq data set made publicly available by TCGA [25, 49]. This data set consists of RSEM expected gene expression counts for samples of human cancer cells from 29 tissue sites. Gene IDs are used in this data set, so the available mapping from TCGA [49] is used to map the IDs to gene names. The difference in number of genes and sparsity from the original data set is not large as can be seen from Table 1.

Experiment setup

Each data set is divided into training, validation, and test sets using a 81 %-9 %-10 % split with uniformly random selection. The training sets are used to train the models, the validation sets are used to validate the models during training, and the test sets are used to evaluate the models after training.

For the deep neural networks, we examine different network architectures to find the optimal one for each data set. We test deep neural networks of both one and two hidden layers with 100, 250, 500, or 1000 units each. We also experiment with a latent space of both 10, 25, 50, and 100 dimensions. A standard VAE with the negative binomial distribution as the likelihood function (a VAE-NB model) is used for these experiments. Using the optimal architecture, we test the probability functions introduced earlier as likelihood function for the standard VAE.

The hidden units in the deep neural networks use the rectifier function as their non-linear transformation, while the latent units and the output units use a non-linear transformation appropriate for the parameters they model. For real, positive parameters (λ for the Poisson distribution, r for the negative binomial distribution, σ in the Gaussian distribution), we model the natural logarithm of the parameter. For the standard deviation σ in the Gaussian-mixture model, however, we use the softplus function, $\log(1+\mathrm{e}^x)$, to constrain the possible covariance matrices to be only positive-definite. The units modelling the probability parameters in the negative binomial distribution, p, and the zero-inflated distributions, π_k, use the sigmoid function, while for the categorical distributions in the constrained Poisson distribution, the piece-wise categorical distributions, as well as for the Gaussian-mixture model, the probabilities are given as logits with linear functions, which can be

evaluated as probabilities using softmax normalisation. Additionally for the piece-wise categorical distributions, we choose the cut-off K to be 4, since this strikes a good balance between the number of low and high count for all examined data sets (see Additional file 1: Figure S11 for an example of this). For the k-means clustering and the GMVAE model, the number of clusters is chosen to be equal to the number of classes, if cell types were provided.

The models are trained using one Monte Carlo sample for each example and using the Adam optimisation algorithm [50] with a mini-batch size of 100 and a learning rate of 10^{-4}. Additionally, we use batch normalisation [51] to improve convergence speed of the optimisation. We train all models for 500 epochs and use early stopping with the validation marginal likelihood lower bound to select parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. As a baseline model to test whether the non-linearity in the neural network contributes to a better performance, we try out models with the same link function, but without the hidden layers in the generative process. This corresponds to factor analysis (FA) with a generalised linear model link function (denoted FA models).

Software implementation

The models described in this sections have been implemented in Python using Google's machine-learning software library TensorFlow [52] in an open-source and platform-independent software tool called *scVAE* (*single-cell* (or *sparse count*) *variational auto-encoders*), and the source code is freely available at `https://github.com/chgroenbech/scVAE` along with a user guide and examples. The results in this article were produced using version 1.0.[53]

The data sets presented in "Results" along with others are also easily accessed using this tool. This tool also includes extensive support for sparse data sets and makes use of this sparseness to reduce its memory footprint. It can thus be used on the largest data sets currently available as demonstrated on the MB-1M data set [24].

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and materials

The PBMC [39–48] and MB-1M [24] data that support the findings of this study are available from 10x Genomics, `https://support.10xgenomics.com/single-cell-gene-expression/datasets/` (see individual references for direct links to each data set).

The TCGA data that support the findings of this study are available from UCSC Xena, `https://xenabrowser.net/datapages/?dataset=tcga_gene_expected_count&host=https://toil.xenahubs.net`.[49]

The scVAE software tool is platform independet, open source under the Apache 2.0 license (OSI-compliant) and available at `https://github.com/chgroenbech/scVAE`. The results in this article were produced using version 1.0 (`https://doi.org/10.5281/zenodo.1229188`).[53]

Competing interests

The authors declare that they have no competing interests.

Funding

OW gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. THP acknowledges support from the Lundbeck Foundation and Novo Nordisk Foundation.

Author's contributions

CHG, MFV, and OW designed the methods. CHG and MFV implemented them. CHG modelled and analysed the data sets with help from CKS for the TCGA data set. CHG and OW drafted the manuscript. All authors have read and approved the final manuscript.

Acknowledgements

We would like to thank Lars Maaløe for discussions about multimodal VAE's and technical help for implementing these. We also want to give thanks to Associate Professor Aki Vehtari for discussions about discrete probability distributions and model comparison.

References

[1] Regev, A., Teichmann, S.A., Lander, E.S., Amit, I., Benoist, C., Birney, E., Bodenmiller, B., Campbell, P., Carninci, P., Clatworthy, M., Clevers, H., Deplancke, B., Dunham, I., Eberwine, J., Eils, R., Enard, W., Farmer, A., Fugger, L., Göttgens, B., Hacohen, N., Haniffa, M., Hemberg, M., Kim, S., Klenerman, P., Kriegstein, A., Lein, E., Linnarsson, S., Lundberg, E., Lundeberg, J., Majumder, P., Marioni, J.C., Merad, M., Mhlanga, M., Nawijn, M., Netea, M., Nolan, G., Pe'er, D., Phillipakis, A., Ponting, C.P., Quake, S., Reik, W., Rozenblatt-Rosen, O., Sanes, J., Satija, R., Schumacher, T.N., Shalek, A., Shapiro, E., Sharma, P., Shin, J.W., Stegle, O., Stratton, M., Stubbington, M.J.T., Theis, F.J., Uhlen, M., van Oudenaarden, A., Wagner, A., Watt, F., Weissman, J., Wold, B., Xavier, R., and, N.Y.: The human cell atlas. eLife **6** (2017). doi:10.7554/elife.27041

[2] Davis, S., contributers: List of software packages for single-cell data analysis, including RNA-seq, ATAC-seq, etc. Accessed 23 April 2018 (2018). doi:10.5281/zenodo.1169178. `https://github.com/seandavi/awesome-single-cell`

[3] Satija, R., Farrell, J.A., Gennert, D., Schier, A.F., Regev, A.: Spatial reconstruction of single-cell gene expression data. Nature Biotechnology **33**(5), 495–502 (2015). doi:10.1038/nbt.3192

[4] duVerle, D.A., Yotsukura, S., Nomura, S., Aburatani, H., Tsuda, K.: cellTree: an r/bioconductor package to infer the hierarchical structure of cell populations from single-cell RNA-seq data. BMC Bioinformatics **17**(1) (2016). doi:10.1186/s12859-016-1175-6

[5] Sun, Z., Wang, T., Deng, K., Wang, X.-F., Lafyatis, R., Ding, Y., Hu, M., Chen, W.: DIMM-SC: a dirichlet mixture model for clustering droplet-based single cell transcriptomic data. Bioinformatics **34**(1), 139–146 (2017). doi:10.1093/bioinformatics/btx490

[6] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. ArXiv e-prints (2013). 1312.6114

[7] Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 1278–1286. PMLR, Bejing, China (2014). http://proceedings.mlr.press/v32/rezende14.html

[8] Kingma, D.P., Rezende, D.J., Mohamed, S., Welling, M.: Semi-supervised learning with deep generative models. ArXiv e-prints (2014). 1406.5298

[9] Dilokthanakul, N., Mediano, P.A.M., Garnelo, M., Lee, M.C.H., Salimbeni, H., Arulkumaran, K., Shanahan, M.: Deep unsupervised clustering with gaussian mixture variational autoencoders. ArXiv e-prints (2016). 1611.02648

[10] Sønderby, C.K., Raiko, T., Maaløe, L., Sønderby, S.K., Winther, O.: Ladder variational autoencoders. ArXiv e-prints (2016). 1602.02282

[11] Maaløe, L., Fraccaro, M., Winther, O.: Semi-supervised generation with cluster-aware generative models. ArXiv e-prints (2017). 1704.00637

[12] Bowman, S.R., Vilnis, L., Vinyals, O., Dai, A.M., Jozefowicz, R., Bengio, S.: Generating sentences from a continuous space. ArXiv e-prints (2016). 1511.06349

[13] Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style. ArXiv e-prints (2015). 1508.06576

[14] Roberts, A., Engel, J., Eck, D. (eds.): Hierarchical Variational Autoencoders for Music (2017). https://nips2017creativity.github.io/doc/Hierarchical_Variational_Autoencoders_for_Music.pdf

[15] Way, G.P., Greene, C.S.: Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. bioRxiv e-prints (2017). doi:10.1101/174474

[16] Tan, J., Ung, M., Cheng, C., Greene, C.S.: Unsupervised feature construction and knowledge extraction from genome-wide assays of breast cancer with denoising autoencoders. In: Biocomputing 2015, pp. 132–143. World Scientific, Kohala Coast, Hawaii, USA (2014). doi:10.1142/9789814644730_0014

[17] Gupta, A., Wang, H., Ganapathiraju, M.: Learning structure in gene expression data using deep architectures, with an application to gene clustering. bioRxiv e-prints (2015). doi:10.1101/031906

[18] Tan, J., Hammond, J.H., Hogan, D.A., Greene, C.S.: ADAGE-based integration of publicly available Pseudomonas aeruginosa gene expression data with denoising autoencoders illuminates microbe-host interactions. mSystems **1**(1), 00025–15 (2016). doi:10.1128/msystems.00025-15

[19] Chen, L., Cai, C., Chen, V., Lu, X.: Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model. BMC Bioinformatics **17**(S1) (2016). doi:10.1186/s12859-015-0852-1

[20] Cui, H., Zhou, C., Dai, X., Liang, Y., Paffenroth, R., Korkin, D.: Boosting gene expression clustering with system-wide biological information: A robust autoencoder approach. bioRxiv e-prints (2017). doi:10.1101/214122

[21] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. ArXiv e-prints (2014). 1406.2661

[22] Ghahramani, A., Watt, F.M., Luscombe, N.M.: Generative adversarial networks uncover epidermal regulators and predict single cell perturbations. bioRxiv e-prints (2018). doi:10.1101/262501

[23] Zheng, G.X.Y., Terry, J.M., Belgrader, P., Ryvkin, P., Bent, Z.W., Wilson, R., Ziraldo, S.B., Wheeler, T.D., McDermott, G.P., Zhu, J., Gregory, M.T., Shuga, J., Montesclaros, L., Underwood, J.G., Masquelier, D.A., Nishimura, S.Y., Schnall-Levin, M., Wyatt, P.W., Hindson, C.M., Bharadwaj, R., Wong, A., Ness, K.D., Beppu, L.W., Deeg, H.J., McFarland, C., Loeb, K.R., Valente, W.J., Ericson, N.G., Stevens, E.A., Radich, J.P., Mikkelsen, T.S., Hindson, B.J., Bielas, J.H.: Massively parallel digital transcriptional profiling of single cells. Nature Communications **8**(14049) (2017). doi:10.1038/ncomms14049

[24] 10x Genomics: 1.3 Million Brain Cells from E18 Mice – Single Cell Gene Expression Dataset by Cell Ranger 1.3.0 (2017). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons

[25] Weinstein, J.N., Collisson, E.A., Mills, G.B., Shaw, K.R.M., Ozenberger, B.A., Ellrott, K., Shmulevich, I., Sander, C., Stuart, J.M.: The cancer genome atlas pan-cancer analysis project. Nature Genetics **45**, 1113–1120 (2013). doi:10.1038/ng.2764

[26] Lopez, R., Regier, J., Cole, M.B., Jordan, M., Yosef, N.: Bayesian inference for a generative model of transcriptome profiles from single-cell RNA sequencing. bioRxiv e-prints (2018). doi:10.1101/292037

[27] Eraslan, G., Simon, L.M., Mircea, M., Mueller, N.S., Theis, F.J.: Single cell RNA-seq denoising using a deep count autoencoder. bioRxiv e-prints (2018). doi:10.1101/300681

[28] Rasmussen, C.E.: The infinite gaussian mixture model. In: Solla, S.A., Leen, T.K., Müller, K.-R. (eds.) Advances in Neural Information Processing Systems 12, pp. 554–560. MIT Press, Denver, CO, USA (2000)

[29] Campbell, N.D.F., Kautz, J.: Learning a manifold of fonts. ACM Transactions on Graphics (SIGGRAPH) **33**(4) (2014)

[30] Campbell, N.D.F., Kautz, J.: Interactive 2D Font Manifold Demonstration. Accessed 23 April 2018. http://vecg.cs.ucl.ac.uk/Projects/projects_fonts/projects_fonts.html#interactive_demo

[31] Brouwer, T., Lió, P.: Bayesian hybrid matrix factorisation for data integration. In: Singh, A., Zhu, J. (eds.) Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 54, pp. 557–566. PMLR, Fort Lauderdale, FL, USA (2017). http://proceedings.mlr.press/v54/brouwer17a.html

[32] Shu, R.: Gaussian Mixture VAE: Lessons in Variational Inference, Generative Models, and Deep Nets. Accessed 23 April 2018. http://ruishu.io/2016/12/25/gmvae/

[33] Oshlack, A., Robinson, M.D., Young, M.D.: From RNA-seq reads to differential expression results. Genome Biology **11**(12), 220 (2010). doi:10.1186/gb-2010-11-12-220

[34] Vallejos, C.A., Risso, D., Scialdone, A., Dudoit, S., Marioni, J.C.: Normalizing single-cell RNA sequencing data: challenges and opportunities. Nature Methods **14**(6), 565–571 (2017). doi:10.1038/nmeth.4292

[35] Seeger, M.W., Salinas, D., Flunkert, V.: Bayesian intermittent demand forecasting for large inventories. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 4646–4654. Curran Associates, Inc., Barcelona, Spain (2016)

[36] Salakhutdinov, R., Hinton, G.: Semantic hashing. International Journal of Approximate Reasoning **50**(7), 969–978 (2009). doi:10.1016/j.ijar.2008.11.006

[37] Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification **2**(1), 193–218 (1985). doi:10.1007/bf01908075

[38] van der Maaten, L.J.P., Hinton, G.E.: Visualizing high-dimensional data using *t*-sne. Journal of Machine Learning Research **9**, 545 (2008)

[39] 10x Genomics: CD14+ Monocytes – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/cd14_monocytes

[40] 10x Genomics: CD19+ B Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/b_cells

[41] 10x Genomics: CD34+ Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/cd34

[42] 10x Genomics: CD4+ Helper T Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/cd4_t_helper

[43] 10x Genomics: CD4+/CD25+ Regulatory T Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/regulatory_t

[44] 10x Genomics: CD4+/CD45RA+/CD25- Naive T cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/naive_t

[45] 10x Genomics: CD4+/CD45RO+ Memory T Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/memory_t

[46] 10x Genomics: CD56+ Natural Killer Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/cd56_nk

[47] 10x Genomics: CD8+ Cytotoxic T Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/cytotoxic_t

[48] 10x Genomics: CD8+/CD45RA+ Naive Cytotoxic T Cells – Single Cell Gene Expression Dataset by Cell Ranger 1.1.0 (2016). https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/naive_cytotoxic

[49] UCSC TOIL RNA-seq recompute: TCGA Pan-Cancer (PANCAN) – gene expression RNAseq – TOIL RSEM expected count (2016). https://xenabrowser.net/datapages/?dataset=tcga_gene_expected_count&host=https://toil.xenahubs.net

[50] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. ArXiv e-prints (2014). 1412.6980

[51] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. ArXiv e-prints (2015). 1502.03167

[52] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Accessed on 25 April 2018 (2015). http://tensorflow.org/

[53] Grønbech, C.H., Vording, M.F.: scVAE: Single-cell variational auto-encoders (2018). doi:10.5281/zenodo.1229188. https://github.com/chgroenbech/scVAE

Additional Files

Additional file 1 — Tables S1–S2 with descriptions; Figures S1–S11 with descriptions