PALAVA: a neural-network model for inferring genetic pathways from single-cell data

Lachlan Doig, School of Mathematics and Statistics, The University of Melbourne, with supervisor Heejung Shim

Introduction

In all living organisms, genes are expressed differently from cell to cell based on the biological processes each cell is undergoing. Understanding which groups of genes are responsible for different biological processes is key to explaining the traits present in cells and organisms.

Existing biological pathway libraries, like MSigDB or REACTOME, are of great utility in this process and were used as the basis for inferring pathways by f-scLVM [1], a linear method based on factor analysis. Here, we build on that idea to propose our own non-linear model based on variational autoencoders (VAE), taking inspiration from the output-interpretable VAE (oi-VAE) framework published by Ainsworth et al. [2]

Linear and Non-linear Factor Analysis

Factor analysis attempts to infer the underlying structure of a dataset by mapping high-dimensional data to a low-dimensional set of latent variables, or factors. Inferring gene pathways from single cell expression data can be abstracted to factor analysis, where:

- observations are cells
- attributes are genes
- factors are gene pathways: groups of genes expressed together in cells

Linear methods such as PCA model factors as linear combinations of attributes and attempt to find a matrix decomposition of the data:

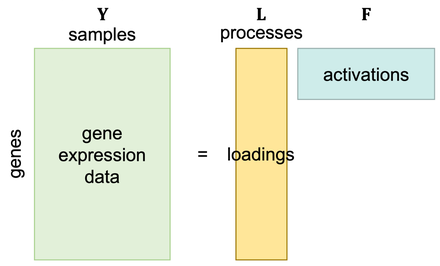


Figure 1:

The loadings matrix maps the high-dimensional attributes (genes) to low-dimensional factors (biological pathways), while the activations are the low-dimensional representations of each cell, corresponding to the active pathways in that cell. Since the pathways are just linear combinations of genes they are easily interpretable, but are limited by simplicity. Contrast this with the highly versatile, non-linear dimension reduction offered by VAE:

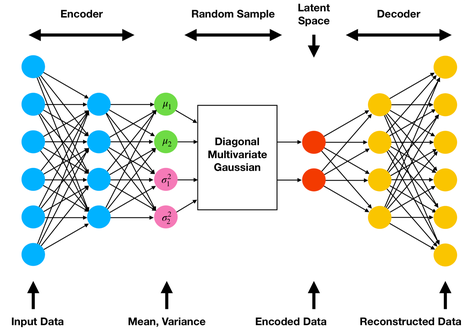


Figure 2:

Here, the encoder and decoder are neural-networks. The encoder takes the high-dimensional data-point as input and outputs the parameters to a multivariate normal distribution from which the latent representation is sampled. The decoder then attempts to reconstruct the original data-point from this latent representation. The use of neural-networks allows great flexibility in what relationships learned, although this does come at the cost of interpretability of the latent representation, since the factors are not simply linear combinations of genes and therefore no equivalent of a loadings matrix exists.

The PALAVA Model: Pathway-Annotated-LAtency Variational Autoencoder

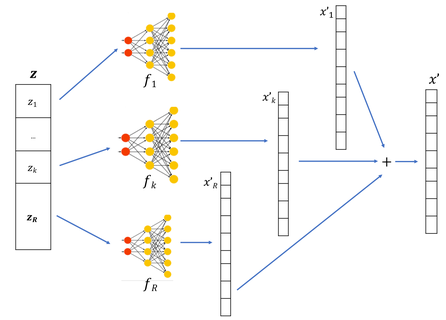
We present a VAE modification to make use of existing pathway annotations in a similar fashion to f-scLVM. The model modifies only the decoder portion of the original VAE, by assigning each pathway its own latent variable.

Consider a set of m genes $\{g_1, \dots, g_m\}$ and a set k vectors $\{\boldsymbol{P}_1, \dots, \boldsymbol{P}_k\}$, where each $\boldsymbol{P}_i$ is a Boolean vector of size m indicating which genes are in pathway i:

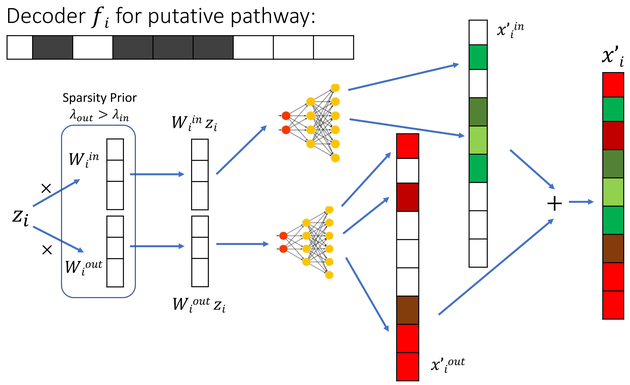
$$\boldsymbol{P}_i = \{c_i(x) \mid x \in \{1, \dots, m\}\}, \text{ where } c_i(x) = \begin{cases} 1, & \text{if gene } g_x \text{ is in pathway } i \\ 0, & \text{otherwise} \end{cases}$$

For a latent representation $\boldsymbol{z}$ with n elements, where $n > k$, the first k elements of $\boldsymbol{z}$ are each annotated with their own pathway $\boldsymbol{P_i}$ and decoder f_i, which takes in a single value and has output dimension m, representing that pathway's expression (details below). The remaining $n - k$ unannotated elements of z, named $\boldsymbol{z}_R$ are given a single decoder f_R, a standard neural-network with input dimension $n - k$ and output dimension m, representing gene expression not attributed to any pathway. The expectation of the final reconstruction $\boldsymbol{x}'$ is just the sum of all the expressions:

$$E(\boldsymbol{x}') = \sum_{i=1}^{k} f_i(z_i) \; + \; f_R(\boldsymbol{z}_R)$$



Each annotated decoder f_i consists of two column vectors $\boldsymbol{W}_i^{in}$ and $\boldsymbol{W}_i^{out}$, and two neural-networks f_i^{in} and f_i^{out}, whose input dimensions are hyperparemeters $p = |\boldsymbol{W}_i^{in}|$ and $q = |\boldsymbol{W}_i^{out}|$ respectively. Their output dimensions are the number of genes in and not in the pathway respectively i.e. the number of 1s in $\boldsymbol{P}_i$ and the number of 0s in $\boldsymbol{P}_i$. The inputs to the neural-networks are the matrix products $\boldsymbol{W}_i^{in} z_i$ and $\boldsymbol{W}_i^{out} z_i$. Their outputs represent the gene expression for genes in $\boldsymbol{P}_i$ and not in $\boldsymbol{P}_i$ respectively. These outputs are concatenated and reordered to give the expression of all m genes attributed to the ith pathway:



The same group sparsity used by oi-VAE is placed on each $\boldsymbol{W}_i^{in}$ and $\boldsymbol{W}_i^{out}$, controlled by sparsity parameters λ_{in} and λ_{out} respectively.

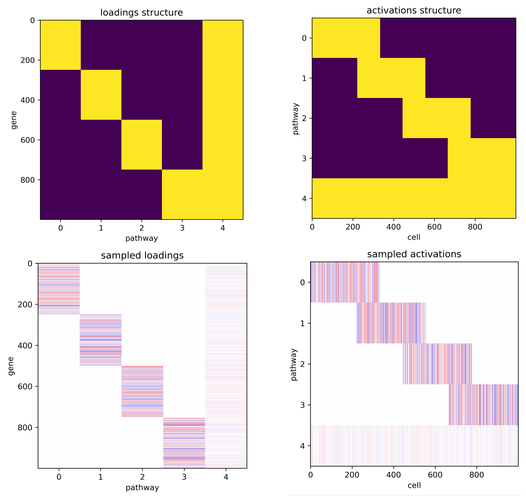
$$\gamma_{in}{}^2 \sim \text{Gamma}\left(\frac{p+1}{2}, \lambda_{in}\right) \; ; \; \gamma_{out}{}^2 \sim \text{Gamma}\left(\frac{q+1}{2}, \lambda_{out}\right)$$

$$\boldsymbol{W}_i^{in} \sim \text{N}(\boldsymbol{0}, \gamma_{in}{}^2 * \text{I}) \; ; \; \boldsymbol{W}_i^{out} \sim \text{N}(\boldsymbol{0}, \gamma_{out}{}^2 * \text{I})$$

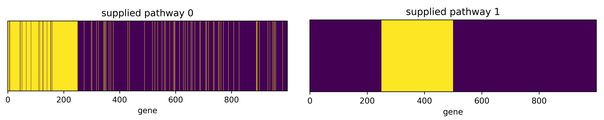
where $\text{Gamma}(\cdot,\cdot)$ has shape and rate parameters. This means that higher λ_{out} values make it more likely for a $\boldsymbol{W}_i^{out}$ to reduce to $\boldsymbol{0}$, at which point that z_i will only influence the expression of genes annotated as belonging to the ith pathway. The equivalent is true of λ_{in}. We choose higher λ_{out} values to ensure each pathway decoder contributes mostly to genes annotated as belonging to the pathway. We can then tell if the model has learned to add new genes if it learns a non-zero $\boldsymbol{W}_i^{out}$.

Experiment – more at https://github.com/ljdoig/PALAVA

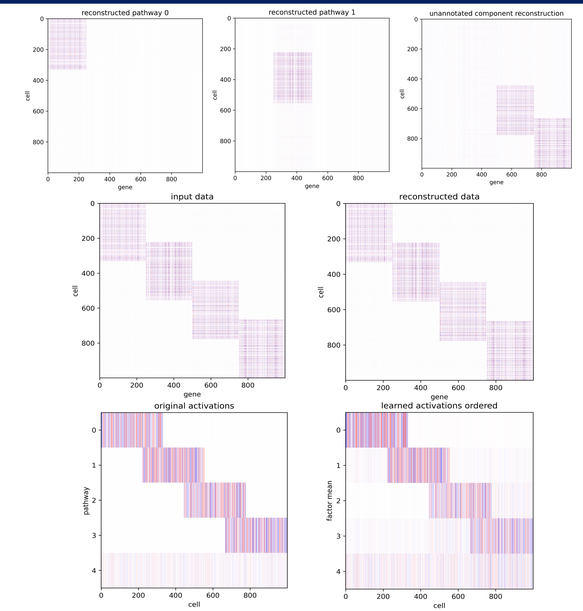
We tested the model on simulated linear data, constructed from the loadings and activations structure below with 1000 genes, 1000 cells and 5 factors. We then sampled the loadings **L** and activations **F**, from either $\text{N}(0,1)$ or $\text{N}(0, 0.3^2)$ for sparse and dense factors respectively. The remaining purple sections were sampled from $\text{N}(0, 0.01^2)$. The final data matrix used for the simulation was sampled from $\text{N}(\mathbf{L} * \mathbf{F}, 0.001^2)$.



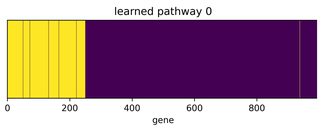
We supplied the model with the first two loading structure columns as pathways, containing the first and second quarters of genes respectively. Error was applied to pathway 0 by sampling from $\text{Bernoulli}(0.07)$ for each element, leading to 15 false negatives and 57 false positives:



As mentioned earlier, there is no equivalent of a loadings matrix for VAE, so we will use the reconstruction of the data and the learned latent representations (activations) to judge the model. Furthermore, we will be interested in which genes the model has included in a given pathway (particularly pathway 0). We used the variance of a gene's expression associated with that pathway over all cells as a scoring metric. That is, with c cells, where $\boldsymbol{x'}_i[j,k]$ gives the expected expression of gene j in cell k associated with pathway i, we calculate: $Var(\{\boldsymbol{x'}_i[j,k] \mid k \in \{1, \dots, c\}\})$



We see the data was reconstructed successfully and the specified latent representations were learned! Furthermore, even our rudimentary variance metric was able to correctly identify if genes belonged in pathway 0 or not. Using a threshold of 0.0003 produced only 5 false negatives and 1 false positive, meaning the model had removed over 91% of the errors we supplied to the pathway. We show the final pathway according to whether or not the variance metric was over the threshold:



Conclusion

PALAVA offers the flexibility of VAE, while incorporating valuable prior pathway information from pathway databases to kickstart factor-learning. Initial experimentation has verified the model's potential in understanding linear data and correcting annotation error. Further experimentation and refinement is required to ensure the model is robust to error and more complex data, and to reduce the number of hyperparameters required to train the model. Ultimately we hope to see the application of the model to genuine single-cell RNA-seq data.

References

[1] Buettner, F., Pratanwanich, N., McCarthy, D.J. et al. f-scLVM: scalable and versatile factor analysis for single-cell RNA seq. Genome Biol 18, 212 (2017). https://doi.org/10.1186/s13059-017-1334-8

[2] Ainsworth, S.K., Foti, N.J., Lee, A.K.C. & Fox, E.B.. (2018). Oi-VAE: Output Interpretable VAEs for Nonlinear Group Factor Analysis. Proceedings of the 35th International Conference on Machine Learning in Proceedings of Machine Learning Research 80:119-128 Available from https://proceedings.mlr.press/v80/ainsworth18a.html

Image sources

Figure 1: Yong See Foo, A comparison of Bayesian inference techniques for sparse factor analysis. https://vrs.amsi.org.au/student-profile/yong-see-foo/

Figure 2: Steven Flores, Variational Autoencoders are Beautiful. https://www.compthree.com/blog/autoencoder

Further Information – email me at Lachlan@Doig.org

Github for code and simulations: https://github.com/ljdoig/PALAVA

Background on neural-networks: https://www.3blue1brown.com/topics/neural-networks

Background on autoencoders and variational autoencoders: https://avandekleut.github.io/vae/