Mechanistic dissection of cellular perturbations with interpretable deep learning models

Dissertation submitted to the
Faculty of Engineering of the
Ruprecht-Karls-Universität Heidelberg
for the degree of
Doctor of Natural Sciences
(Doktor der Naturwissenschaften)

Presented by

M.Sc. Daria Ivona Dončević
born in Koblenz, Germany

Mechanistic dissection of cellular perturbations with interpretable deep learning models

Referees: Prof. Dr. Carl Herrmann

Prof. Dr. Emanuel Schwarz

Declaration of Authorship

I hereby declare that the contents of the submitted dissertation are original and written by myself, without using any sources or assistance except as stated in the acknowledgement and text or cited in the specific reference. No other additional examination procedure for any other degree or qualification at other institutions in any forms elsewhere as an examination paper and dissertation have been applied and submitted. In addition, checks for compliance with generally applicable scientific standards using electronic data processing programs on the dissertation are agreed to be made.

This work was carried out in the Bioinformatics department at the Institute of Pharmacy and Molecular Biotechnology (IPMB) in the group of Prof. Dr. Carl Herrmann.

Heidelberg, August, 2024

. . . . . .

Daria Ivona Dončević

To Aaron, who is my greatest motivation and my greatest distraction.

Acknowledgements

First of all, I would like to thank my supervisor and mentor Prof. Dr. Carl Herrmann for the opportunity of conducting my PhD research in his group. Carl, I really enjoyed and still enjoy working with you. You always made the time to discuss science and other matters whenever I needed it, and you were always understanding and supportive, especially when my personal circumstances changed. I always enjoyed our spontaneous brainstormings together, which always opened up new and exciting directions about the project. You confided in me and my work, giving me the opportunities to present at conferences and meetings. I hope you stay the enthusiastic, passionate and compassionate scientist and person that you are.

I am grateful to all my friends and colleagues that I met and made along the way in Health Data Science Unit (HDSU) and in the Bioinformatics Department, in particular Dr. Ana Luísa Costa, Dr. Andrés Quintero, Dr. Carlos Ramirez, Dr. Ashwini Kumar Sharma, Youcheng Zhang, Dr. Lin Yang, Qian-Wu Liao, Dr. Robin Droit, Albert Li, Jean Radig, Dr. Saifullah Tumrani, Eugenia Guerrero, Pablo Naranjo, and Dr. Nelida Palau for their friendship, their support and all the time spent together. You made this journey an unforgettable one and time fly! Thanks to Andrés and Ashwin for mentoring me during my first steps, and providing me advice whenever I needed it. Especially Andrés, you were never tired to hear about my project and brainstorm possible ideas with me. Thanks to Carlos, Albert, Leonie, Youcheng, and Kasimir for all the work you did with COBRA. Especially Carlos, I even used parts of your work for this thesis.

I would also like to thank Prof. Dr. Emanuel Schwarz who has been involved in this project from the very beginning, as responsible person for COMMITMENT, as part of my thesis advisory committee, and as an examiner of this thesis. You were never tired of listening to my presentations and always gave valuable suggestions how things could be improved.

Besides, I want to thank Steeve Boulant, Megan Stanifer, and Zina Uckeley for the fruitful collaboration on the interferon project and the interesting discussions held during our regular meetings.

I would also like to express my gratitude towards the other members of my thesis advisory committee, Prof. Dr. Rob Russell, and PD Dr. Christian Fufezan, and of my thesis defense committee, PD Dr. Karl Rohr, and apl. Prof. Dr. Ullrich Köthe for accompanying my studies.

My gratitude also extends to Manuela Schaefer, Sabrina Wetzel, and especially Cathrin Hollenbach from the administration staff, who always assisted with matters of bureaucracy, such as setting up the contract, or organizing a trip to a conference. Cathrin, you were a life-saver many times.

I also want to mention my friends that I made during my studies in Heidelberg and that made my time here special: Anja, Daniel, Isabelle, Umut, Masroor, Constantin, Sara, Marina, Emma, Mareike, Katharina, and Julian. Thank you for all the memories we created together. Without you, it would not have been the same.

I am extremely grateful to my parents Ognjanka and Dario who are always there for me when I need them. Without your support especially in these last months of writing I never would have been able to finish this thesis. To my brother Danimir, who always wants to discuss science with me and even managed to read and understand my publications. I am sorry that I never actually read yours. And to Kersten, who has supported me all the way, encouraging me every time I felt down, and sharing with me the good and the bad moments of this journey. Thank you for always having my back.

List of publications

Thesis related

- Doncevic, D, and Herrmann, C. 2023. Biologically informed variational autoencoders allow predictive modeling of genetic and drug-induced perturbations. Bioinformatics 39(6) https://doi.org/10.1093/bioinformatics/btad387

Other contributions

- Gartlgruber, M., Sharma, A.K., Quintero, A. et al. Super enhancers define regulatory subtypes and cell identity in neuroblastoma. Nat Cancer 2, 114–128 (2021). https://doi.org/10.1038/s43018-020-00145-w

Contributions at Conferences

- BC2 Basel Computational Biology Conference, Basel, September 2021
  Contribution: Poster
  Daria Doncevic and Carl Herrmann
  Incorporation of biological ontologies into Neural Networks to enhance model interpretability
  Award: Poster prize

- SBMC Systems Biology of Mammalian Cells Conference, Heidelberg, May 2022
  Contribution: Poster
  Daria Doncevic, Qian-Wu Liao and Carl Herrmann

Enhancing model interpretability by biologically informed decoder structures

Award: Poster prize

- SBHD Systems Biology of Human Diseases Conference, Nashville, June 2022

  Contribution: Poster and short talk

  Daria Doncevic

  From black box to grey box – How to shed light on Variational Autoencoder models

  Award: Best short talk

- e:Med Meeting, Heidelberg, November 2022

  Contribution: Talk

  Daria Doncevic

  From black box to grey box – How to shed light on Variational Autoencoder models

Abstract

Background: Deep Learning (DL) is becoming more and more state-of-the-art for the analysis of next-generation sequencing data such as RNA-seq and single-cell RNA-seq, due to its ability to capture more complex patterns in the data. In particular, variational autoencoders (VAEs) have been used for a variety of tasks ranging from batch effect removal to data integration. One disadvantage of DL lies in its limited interpretability due to the the non-linear nature of the models. However, interpretability is crucial especially in the biomedical context.

Results: In this thesis, we developed OntoVAE, an interpretable VAE model whose latent space and decoder are reflecting a biological regulatory network. OntoVAE can be installed from Pypi and is available on GitHub at `https://github.com/hdsu-bioquant/onto-vae`. We used OntoVAE to compute pathway activities and to predict the outcome of a gene knockout and of interferon treatment response. We then further developed COBRA, a tool that extends OntoVAE with an adversarial approach to disentangle the effects of different covariates. We used COBRA to study interferon response, adrenal medulla development, and schizophrenia.

Conclusion: OntoVAE and COBRA are useful VAE tools that are based on an interpretable latent space and decoder. They can compute pathway activities, but also be used for predictive modeling, and in the case of COBRA, also to extract effects otherwise overshadowed by confounders. Both tools are easy to install and easy to use, and thus a valuable resource to the scientific community.

Zusammenfassung

Hintergrund: Deep Learning (DL) entwickelt sich mehr und mehr zum Standard für die Analyse von Hochdurchsatzsequenzierungsdaten, wie zum Beispiel RNA-Sequenzierung und single-cell RNA-Sequenzierungsdaten, weil die Methoden in der Lage sind, komplexere Strukturen in den Daten zu erkennen. Vor allem Variatonal Autoencoders (VAEs) wurden für viele Aufgaben verwendet, vom Entfernen von Batch Effekten hin zu der Integration von Daten. Ein Nachteil von DL ist die limitierte Interpretabilität aufgrund der nicht-linearen Natur der Modelle. Jedoch ist die Interpretabilität essenziell vor allem im biomedizinischen Kontext.

Ergebnisse: In dieser Arbeit haben wir OntoVAE entwickelt, ein interpretierbares VAE Modell, dessen latent space und Decoder aus einem biologischen regulatorischen Netzwerk bestehen. OntoVAE kann über Pypi installiert werden und ist auf GitHub verfügbar: `https://github.com/hdsu-bioquant/onto-vae`. Wir haben OntoVAE verwendet, um die Aktivitäten von Signalwegen zu berechnen, und um Vorhersagen über das Knockout von Genen sowie die Behandlung mit Interferon zu treffen. Weiterhin haben wir COBRA entwickelt, das OntoVAE um einen adversarial Ansatz ergänzt, der es ermöglicht, die Effekte einzelner Kovariaten voneinader zu isolieren. Wir haben COBRA angewendet, um Interferonantwort, die Entwicklung des Nebennierenmarks, sowie Schizophrenie zu studieren.

Schlussfolgerungen: OntoVAE und COBRA sind nützliche VAE Modelle, die beide auf einem interpretierbaren latent space und Decoder basieren. Sie können Aktivitäten von

Signalwegen berechnen, Vorhersagen treffen, und im Fall von COBRA Effekte isolieren, die sonst von Störvariablen verdeckt werden. Beide Tools sind einfach zu installieren und anzuwenden, und daher eine nützliche Resource für die wissenschaftliche Gemeinde.

Table of Contents

List of Abbreviations

AE	Autoencoder
AI	Artificial intelligence
ANN	Artificial neural network
AUC	Area under the ROC curve
BRN	Biological Regulatory Network
COBRA	COvariate Biological Regulatory network Autoencoder
CPA	Compositional Perturbation Autoencoder
DAG	Directed acyclic graph
DL	Deep learning
DMD	Duchenne Muscular Dystrophy
EEC	Enteroendocrine cell
ELBO	evidence lower bound
ESC	Embryonic stem cell
GAN	Generative adversarial network
GSEA	Gene set enrichment analysis
GTEx	Genotype Tissue Expression
GO	Gene Ontology
GWAS	Genome-wide association studies
HPO	Human Phenotype Ontology
HTS	high-throughput sequencing
IFN	Interferon

KL	Kullback-Leibler
LGMD	Limb-girdle muscular dystrophy
MD	Muscular dystrophy
MEF	Mouse embryonic fibroblast
ML	Machine learning
MLP	Multi-Layer Perceptron
MSE	Mean squared error
NLP	Natural language processing
NO	Nitric oxide
OntoVAE	Ontology guided VAE
ood	out-of-distribution
ORA	Overrepresentation analysis
PBMC	Peripheral blood mononuclear cells
pcw	post-conception week
ReLU	Rectified Linear Unit
RNA-seq	RNA sequencing
ROC	Receiver operator characteristics
SCP	Schwann cell precursor
scRNA-seq	single-cell RNA sequencing
scVI	single-cell Variational Inference
SNP	Single nucleotide polymorphism
TA	Transit Amplifying cell
TF	Transcription factor
VAE	Variational Autoencoder
VEGA	VAE Enhanced by Gene Annotations

List of Figures

List of Tables

Chapter 1

Scope

1.1 Background

With the emergence of high-throughput sequencing techniques such as RNA-seq and single-cell RNA-seq, researchers became able to study development and disease on a molecular level (Lightbody et al. 2019). Nowadays, massive amounts of sequencing data can be collected that describe an individuals or an individual cells genotype and phenotype. The entirety of influences that affect a cells phenotype are denoted as cellular perturbations, and they comprise genetic as well as environmental stimuli (Ji et al. 2021).

Due to their good performance on large datasets, deep learning (DL) models are becoming more and more state-of-the-art in the analysis of omics data and the prediction of cellular perturbations. Especially the Variational Autoencoder (VAE) is now widely used since it can compress the high-dimensional omics data and generate an embedding which captures the essence of the data. One challenge that arises in applying these models outside of the academic area in a clinical context is their limited interpretability due to their non-linear nature, meaning that it is not possible to attribute feature importances, and thus, to explain the observed predictions (Linardatos, Papastefanopoulos, and Kotsiantis 2020).

However, as trust and understanding of model decisions are crucial in the clinic, the subfield of interpretable DL tries to come up with models that mitigate this problem. Broadly, there are post-hoc methods which are applied on a trained model to calculate feature importances, and model based methods which natively provide interpretability due to their structure, making use of biological knowledge in the form of a prior.

1.2 Aims

The main objective of this thesis was to address the problem of limited interpretability when applying VAE models on omics data in order to analyse mechanisms and make predictions. Thus, the following aims were addressed:

- Development of toolbox of user-friendly, interpretable VAE models that can be applied in the context of predictive modeling.
- Application of the tools to study different cellular perturbations, such as genetic perturbations, drug treatments, development, and disease.
- Development of a web application to share results.

Moreover, in the scope of a collaborative side project, the interferon response of the gut was analyzed and results were collected in a web application.

1.3 Major findings

In this thesis, I developed an interpretable VAE model, OntoVAE, that incorporates a biological ontology, and thus allows to extract pathway activities as well as to model perturbations. I expanded OntoVAE with an adversarial approach that furthermore allows for the separation of different covariate effects while maintaining the interpretability. As a side project, I investigated the interferon response in the gut using standard single-cell analysis techniques. I also created web applications to share results and to facilitate collaboration. The major findings and products of this work are:

- Development of OntoVAE, an interpretable VAE model that learns pathway activities. OntoVAE is implemented in pytorch, is available from GitHub at https://github.com/hdsu-bioquant/onto-vae, and can also be installed via Pypi (`pip install onto-vae`). The original manuscript was published in *Bioinformatics*.

- Development of OntoVAE Model Explorer, a Dash based web application that allows for interactive exploration of some of the results from the OntoVAE paper. The app is hosted on http://ontovaemodelexplorer.pythonanywhere.com, and the source code is available on https://github.com/daria-dc/OntoVAE-Model-Explorer.

- Further optimization of OntoVAE and development of COBRA, a tool that extends OntoVAE with an adversarial approach that encourages separation of different covariate effects. Both VAE models are now hosted in the same GitHub repository https://github.com/hdsu-bioquant/cobra-ai, which will be made available after publication.

- Application of OntoVAE on different ontologies and datasets to compute pathway activities of bulk tissues, and to predict the outcome of a disease (limb-girdle muscular dystrophy) and a treatment (interferon response).

- Application of COBRA with different biological networks on different datasets to distinguish celltype and treatment effect in interferon response, and to study adrenal medulla development and schizophrenia.

- Development of a Shiny based web application to summarize results from the analysis of Ileum and Colon organoid single-cell transcriptomic data that was treated with interferon. The app helped in studying how interferon response differed in the different cell types.

1.4 Outline of the thesis

This document is organized as follows:

- **Introduction**: The introduction showcases how deep learning is used to address questions in computational biology, and explains the concept of artificial neuronal networks and variational autoencoders in more detail. Furthermore, the concept of interpretable deep learning is outlined together with different types of methods. Finally, the concept of cellular perturbations is explained, with a focus on schizophrenia and interferon response.

- **Part I. Tool Development**: The first part describes the development and optimization of OntoVAE and its extension, COBRA. It includes detailed instructions on how to use the tools, as well as proof of concept applications on well understood datasets.

- **Part II. Wep Application development**: The second part describes the two web applications that were developed in this thesis, one to explore results from the OntoVAE manuscript, and the other to facilitate collaboration on the single-cell interferon project. In the scope of this, results are presented that were generated during the analysis of the single-cell data from interferon-treated Ileum and Colon organoids

- **Part III. Perturbation modeling**: The third part contains results that were generated with OntoVAE or COBRA on different datasets. OntoVAE was used to predict how a genetic perturbation changes pathway activities, and which genes most significantly perturb a specific pathway that is related with disease, or treatment. COBRA was used to study perturbations such as interferon response and adrenal medulla development, and to isolate this effect from the celltype effect.

- **Part IV. Final remarks**: The last part presents a summary of the results, discussing the relevance of the findings but also the limitations of the tools. It also places the work in the context of the current developments in the field, along with future perspectives of the study.

Chapter 2

Introduction

2.1 Deep Learning in Computational Biology

The emergence of high-throughput sequencing (HTS) technologies drastically reduced the costs and the time required to sequence a genome and increased the capacities compared to the traditional Sanger sequencing, and thus opened up a new era of biomedical research (Reuter, Spacek, and Snyder 2015). It is now possible to produce massive amounts of sequencing data to profile not only the genome, but also the epigenome of tissues and samples, including the transcriptome, proteome and metabolome among others. RNA-sequencing (RNA-seq) for example allows the simultaneous profiling of the expression of ten thousands of genes (Z. Wang, Gerstein, and Snyder 2009). Whole-genome bisulfite sequencing, ATAC-sequencing and ChIP-sequencing allow the profiling of DNA methylation state, chromatin accessibility and histone marks, and thus give valuable insights into epigenetic regulation. This data is collectively referred to as 'omics' data. With this, researchers can tackle different questions on a molecular level, and identify biomarkers (genes or proteins), to understand mechanisms of disease onset and progression, or predict response to a treatment (Lightbody et al. 2019). In recent years, the subsequent development of single-cell technologies such as single-cell RNA-seq (scRNA-seq) furthermore enabled researchers to monitor gene expression within individual cells and thus

study their heterogeneity and identify and characterize different cell populations (Jovic et al. 2022). For this purpose, the accurate annotation of the cell type of each cell is a prerequisite. This often relies on unsupervised clustering of the cells, followed by manual annotation using established marker genes. As this is a time-consuming task, computational methods have been developed that automate the process. One example is label transfer, which uses a prelabelled reference dataset to annotate a new target dataset based on cell-cell similarities (Stuart et al. 2019). For label transfer to succeed, a similar dataset of approximately the same celltype composition needs to be available, which is not always the case. An international collaborative effort, the Human Cell Atlas Consortium, set out to define all cell types of the human body, and create a comprehensive reference map which aids in studying developmental trajectories, physiological states, cell-cell interactions, and regulation and dysregulation in disease (Regev et al. 2017). Since the foundation of the consortium, the amount of data that is being integrated is constantly expanding, additionally including specified atlases such as the Pediatric Atlas or the Human Lung Cell Atlas. The rapid development of single-cell technologies that drastically increases the number of cells that can be profiled together with an ever growing amount of publicly available collections of single-cell data opens up new computational challenges. Computational methods are needed that can pre-proceess, normalize and analyze the data, and integrate new and existing datasets in an efficient way (Qiu 2020; Luecken et al. 2022).

Inspired from other fields such as computer vision or natural language processing (NLP), where they were applied with huge success, the so-called Deep Learning (DL) algorithms are being more and more widely adopted in the field of omics as well, especially for scRNA-seq data, as they benefit from large amounts of data (Chai et al. 2021; Angermueller et al. 2016) (see **Box 2.1** for a brief overview of Artifical Neural Networks).

Box 2.1: Artificial Neural Networks

The backbone of DL algorithms are Artificial neural networks (ANNs), which mimic the neural networks of the human brain (McCulloch and Pitts 1943; Rosenblatt 1958). The single unit of every ANN is the mathematical neuron, which processes the input X it receives, and returns some output z (**Figure B2.1a**). Mathematically, the working of the neuron can be formulated as:

$$z = \varphi(X\beta + b) \tag{2.1}$$

whereas the matrix of input weights is denoted as β, the node internal bias as b, and the (mostly non-linear) activation function as φ. When multiple of those neurons are organized in L layers and working in series, with each layer L_i receiving input from the previous layer L_{i-1}, and forwarding its output to the next layer L_{i+1}, we end up with a feed-forward architecture or multi-layer perceptron (MLP), which is the most basic ANN (**Figure B2.1b**). Usually the architecture is fully connected, meaning that each neuron in a layer receives input from every neuron in the previous layer. The layers between input and output layer are denoted as 'hidden layers' (Goodfellow, Bengio, and Courville 2016).

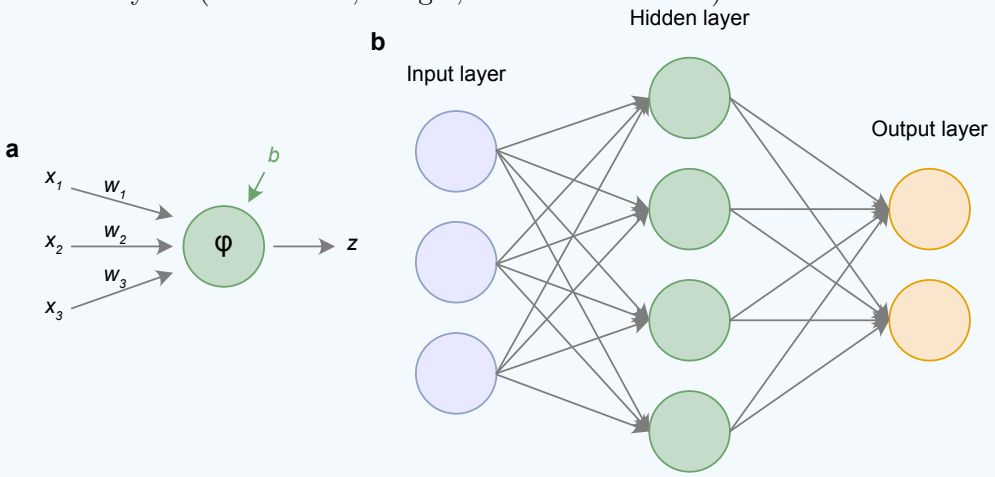

Figure B2.1: Schematic drawing of an artificial neuron (**a**) and an MLP (**b**).

The weights and biases of any ANN are learnt during model training. During this

process, samples are being passed through the network in batches, and then the prediction of the network is evaluated based on a previously defined cost or loss function. The loss function is chosen according to the task, which is mostly either regression (prediction of a continuous value, e.g. gene expression) or classification (prediction of class membership, e.g. cancer or no cancer). Based on the loss, the gradients of the network are calculated in a backward iteration starting from the last layer. This process is called 'backpropagation', and an optimization algorithm uses these gradients to compute the parameter updates. Usually a network is trained until convergence. To prevent overfitting, a scenario in which the model has captured the training data very well but cannot generalize on unseen data, the dataset is usually split into a training and validation set. Parameter updates are performed on the training set, and evaluation of the model on the validation set. Training stops when the performance on the validation set does not increase anymore (Goodfellow, Bengio, and Courville 2016).

Which DL model to use depends on the data and the task. In the context of omics data, which is high-dimensional with ten thousands of features, Variational Autoencoders (VAE) are widely used, since they perform dimension reduction and can capture the underlying distribution of the data (Svensson et al. 2020). Autoencoder based methods have successfully been applied on tasks such as cancer classification (X. Zhang et al. 2019), data denoising (Eraslan et al. 2019), batch correction (J. Wang et al. 2019), multi-domain translation (Yang et al. 2021), and treatment effect prediction (Lotfollahi, Wolf, and Theis 2019). Moreover, models are being developed that are supposed to replace complete workflows especially in the context of single-cell analysis. The scVI (single-cell Variational Inference) model explicitly models the expected counts from the data without any normalization and can perform downstream tasks such batch correction, clustering and differential expression analysis (Lopez et al. 2018). Its successor scanVI furthermore adds the possibility to annotate cell types and states (Xu et al. 2021).

2.1.1 Variational Autoencoder

The idea of autoencoders has been around for decades (Rumelhart, Hinton, and Williams 1986; Bourlard and Kamp 1988; Hinton and Zemel 1993). An autoencoder learns to compress unlabeled input observations x_i in a way that they can be faithfully reconstructed from this compressed representation. For example, if the input data is an image, the autoencoder would give as output an image that is as similar to the input as possible (Michelucci 2022). The typical architecture of an autoencoder is visualized in **Figure 2.1**. Its three main components are the encoder, the latent space, and the decoder. Encoder and decoder are usually neural networks and can be treated as functions, thus, we can define the encoder as a function g

$$h_i = g(x_i) \tag{2.2}$$

where $h_i \in \mathbb{R}^q$, the output of the encoder, is also the latent space representation (Michelucci 2022). The encoder reduces the dimension of the input data, thus, it applies $g : \mathbb{R}^n \to \mathbb{R}^q$. The decoder can be written as another generic function f of the latent space features

$$\tilde{x}_i = f(h_i) = f(g(x_i)) \tag{2.3}$$

where $\tilde{x}_i$ is the reconstructed input data. Since the decoder restores the original dimension of the data, it applies $\tilde{x}_i \in \mathbb{R}^n$. During training, the parameters of encoder $g(\cdot)$ and decoder $f(\cdot)$ have to be learnt such that

$$\arg\min_{f,g} < [\Delta(x_i, f(g(x_i)))] > \tag{2.4}$$

which means that the input and output of the autoencoder should differ as little as possible. Thus, the most common loss function that is used is the mean squared error (MSE) (**equation** (2.5)). Since the MSE is measuring how well the original data was reconstructed, it is also referred to as the reconstruction error (RE) in this context.

$$RE \equiv MSE = \frac{1}{M} \sum_{i=1}^{M} |x_i - \tilde{x}_i|^2 \tag{2.5}$$

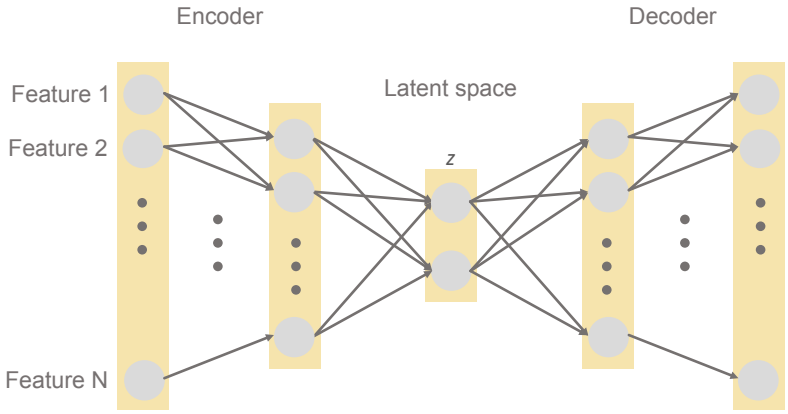

Figure 2.1: Schematic drawing of an autoencoder. The autoencoder consists of an encoder part and a decoder part. The encoder part compresses the input data to a lower dimensional latent representation z. The decoder part reconstructs the input data from z.

The main purpose of the autoencoder lies not so much in an accurate reconstruction of input data, but in the fact that, for reconstructing the data from the latent space representation, this latent space representation must capture the most important features, the essence of the data. One problem of the standard autoencoder is the disjoint latent space representation which cannot generalize to unseen variations of the training data (I. Goodfellow, Bengio, and Courville 2016).

This is where the variational autoencoder (VAE) comes into play, a generative autoencoder model that was introduced by Kingma and Welling in (2013). The main difference compared to the traditional autoencoder is that the VAE learns the parameters of a probability distribution over the latent space. This is schematically visualized in **Figure 2.2**. The encoder part of the VAE yields a posterior distribution $q_\theta(z/x)$, where x is the input data, and z the latent space. The decoder part of the VAE yields a likelihood distribution $p_\phi(x/z)$. θ and ϕ are the learnable parameters of the neural network (Diederik P. Kingma and Welling 2019). Thus, in practice, when data is passed through the VAE model during training or predictions, the latent space representation is sampled from $q_\theta(z/x)$ in a stochastic process, making the VAE a probabilistic model. The loss formula

of the VAE consists of two components:

$$\mathcal{L}_{\phi,\theta}(X) = E_{q_\phi(z/x)}[logp_\phi(x/z)] - D_{KL}(q_\phi(z/x)||p_\theta(z)) \tag{2.6}$$

The first term of **equation** (2.6) is aiming at optimizing the likelihood of the observed data under the model. In practice, we can use an MSE loss here, as for the standard autoencoder. The second term of **equation** (2.6) is the Kullback-Leibler (KL) divergence, which measures the divergence of a learnt distribution from the prior. The KL divergence is used since computation of the posterior $q_\theta(z/x)$ is intractable. The derivation of this is given in more detail in **Appendix A** (Blei, Kucukelbir, and McAuliffe 2017; Odaibo 2019). In practice, we often model the true posterior as a multivariate Gaussian, thus, a closed-form solution exists for the KL divergence and can be written as follows (Odaibo 2019):

$$\mathcal{L}_{\phi,\theta}(X) = -\sum_{j=1}^{J} \frac{1}{2}[1 + log(\sigma_i^2) - \sigma_i^2 - \mu_i^2] - \frac{1}{L}\sum_{l} \mathbb{E}_{\ q_\theta(z/x_i)}[logp(x_i/z^{(i,l)})] \tag{2.7}$$

During model training, the reparameterization trick is applied to enable backpropagation (Diederik P. Kingma and Welling 2013).

2.2 Interpretable Deep Learning

In practice, in the field of computational biology, researchers are mostly interested in the latent space embedding of the samples that is learnt by the VAE. Due to the bottleneck structure, the VAE is reducing the dimension of the high-dimensional omics data, and its latent space is capturing the most essential features of the data in a lower-dimensional embedding. However, one problem that the VAE has in common with all DL methods is the limited interpretability due to their non-linear nature. The word 'interpretability' in this context means that the user understands the predictions of a model and knows which features contributed how strongly, so that possible model biases can be faithfully detected. A good example for this comes from the field of image classification: A classifier that was trained to distinguish huskies from wolves made a wrong prediction, which could

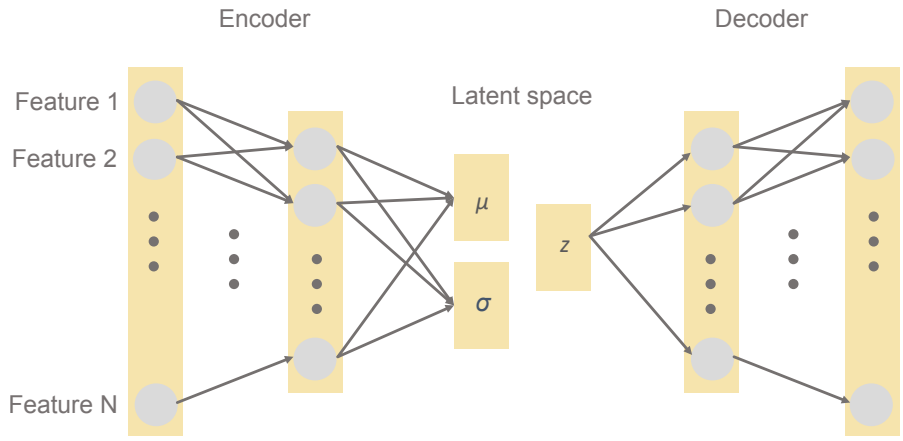

Figure 2.2: Schematic drawing of a Variational Autoencoder. The VAE consists of an encoder part and a decoder part. The encoder part learns the parameters of a distribution over the latent variables of the data, μ and σ. The lower dimenstional latent representation z is then obtained by sampling from the learnt distribution, and is then fed into decoder which reconstructs the input data.

then be explained by looking at the feature contributions, revealing that the animal in the image itself was not taken into account by the classifier, but only the surrounding snow, which the classifier seemingly mistook for fur (**Figure 2.3**)(Ribeiro, Singh, and Guestrin 2016).

For DL models, the determination of the feature contributions is not straight-forward due to the non-linearity. Thus, while DL models learn more complex patterns in the data and outperform less complex, linear models on many tasks, this comes at the cost of sacrificing interpretability (Linardatos, Papastefanopoulos, and Kotsiantis 2020). This is often referred to as the black-box problem of neural networks (X. Li et al. 2022). As already stated, in the field of 'omics', where VAEs are often applied, their learnt latent space embedding cannot be directly attributed to certain input features. In critical domains such as healthcare, however, understanding the rationale behind decision-making of the models is important in order to be able to trust their predictions. Personalized

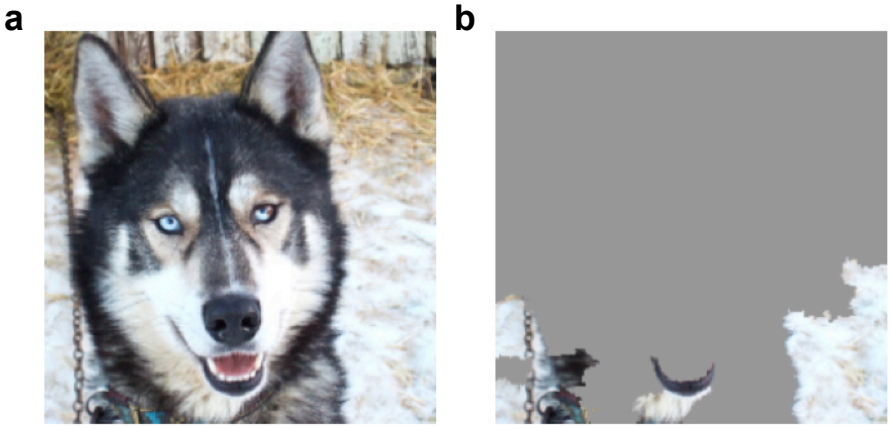

Figure 2.3: The importance of model interpretability. (**a**) Image of a husky that was classified as wolf. (**b**) Saliency map in which pixel colors correspond to their contribution to the prediction. Figure was taken from Ribeiro *et al.* (2016).

medicine is a concept of healthcare that wants to tailor medical treatment to the individual patient for earlier diagnosis and more effective treatment. The approach takes into account multiple factors such as the individuals genetic background and environment, and collects different patient-related data to understand the onset and progression of disease in the individual (Goetz and Schork 2018). Thus, for any model that is trained on this data to make predictions, it is important that these predictions are enriched with an understanding of the underlying biological processes. For example, in the case of single-cell data, the model should not only provide a prediction, but also offer insight on the molecular regulators and disease mechanisms, so that adequate conclusions can be drawn and downstream experimental validations can be performed.

To address this need for models that are high performing but still trustworthy, the subfield of Interpretable DL has emerged (Linardatos, Papastefanopoulos, and Kotsiantis 2020). The different approaches taken by the field can mainly be separated into two classes: post-hoc and model based. Post-hoc methods are applied post-hoc on a trained model, while model based methods modify the structure of the model to gain interpretability.

2.2.1 Post-hoc methods

Examples for post-hoc methods are the general neural network feature attribution methods such as Layerwise Relevance Propagation (LRP) (Bach 2015), LIME (Ribeiro, Singh, and Guestrin 2016), DeepLIFT (Shrikumar, Greenside, and Kundaje 2019) or SHAP (Lundberg and Lee 2017). LRP propagates the output back through the network using the learnt weights and activations, and a set of purposely designed propagation rules. LIME is model-agnostic and performs local model approximation with interpretable linear models and assesses how model predictions change as a consequence of feature perturbation. DeepLIFT also backpropagates the contributions of every neuron in the network to all input features, comparing their activations to a reference for the calculation of contribution scores. SHAP (SHapley Additive exPlanations) is based on Shapley values, a concept which originated in game theory, and quantifies the contribution of each feature to the model prediction by considering each possible combination of input features. One drawback of methods such as SHAP is the poor scalability, as the computational load increases with an increasing number of samples and features (Choi, Li, and Quon 2023).

2.2.2 Model based methods

Examples for model based methods in the biomedical field either introduce some linearity and/or prior biological knowledge into the model. LDVAE replaces the non-linear decoder of a VAE by a linear version to allow to assign feature weights to the different latent space dimensions (Svensson et al. 2020). VEGA also implements a single-layer linear decoder, but additionally introduces sparsity by incorporation of prior biological knowledge in the form of so-called gene module variables (GMVs) into the latent space (Seninge et al. 2021). Thus, each latent space dimension of VEGA represents a GMV, and connections are only present to the genes in the output layer that are annotated to the specific GMV, making the decoder sparse. In principle, any entity with gene annotations can be used as GMV, such as pathways or transcription factors (TFs). Binary masks are used to indicate whether connections are present or not. Modifications of this approach allow

the model some flexibility by solely imposing a penalty term on connections that are not part of the prior, so that the model is guided but not completely determined by the prior (Rybakov et al. 2020).

In line with these approaches, other methods make use not only of annotated gene sets, but of hierarchical biological networks, through their direct incorporation into model structure. These biological networks can be signaling networks, as in knowledge-primed neural networks (KPNNs) (Fortelny and Bock 2020), or biological ontologies. What the networks have in common is that they are directed acyclic graphs (DAGs), thus they fulfill two properties: first, edges of the graph are directed, and second, cycles or loops in the graph are not allowed. The most prominent biological ontology is *Gene Ontology* (GO), which has three subontologies: *biological process*, *molecular function*, and *cellular component* (Ashburner et al. 2000). GO is a hierarchical grouping of genes based on some characteristic such as their function. **Figure 2.4** illustrates the principles of GO based on an excerpt of the GO biological process graph. Each node of the graph is a GO term, and each edge represents a directed connection from a parent term to a child term. GO is organized in a way that a child term is always more specific than its parent term. Here, the root term is *metabolic process*, and it is connected to its child term *biosynthetic process* through an is_a relationship. Thus, one can say that 'biosynthetic process' *is a* 'metabolic process'. As we progress through the DAG, terms are becoming more and more specific. Thus, when looking at the set of genes that are annotated to a term, one can distinguish between directly annotated genes and genes that are annotated to children terms. When considering the distance between two nodes, the *level* describes the shortest possible path, while *depth* refers to the longest possible path.

One method that is making use of GO is DCell, a neural network structured according to GO subsets, and used for prediction of growth rates in yeast (Ma et al. 2018). DeepGONet is a neural network classifier that is encouraging the formation of connections that mirror the GO DAG by imposing regularization on model weights, and thus, favoring connections between associated GO terms (Bourgeais et al. 2021). It has been used

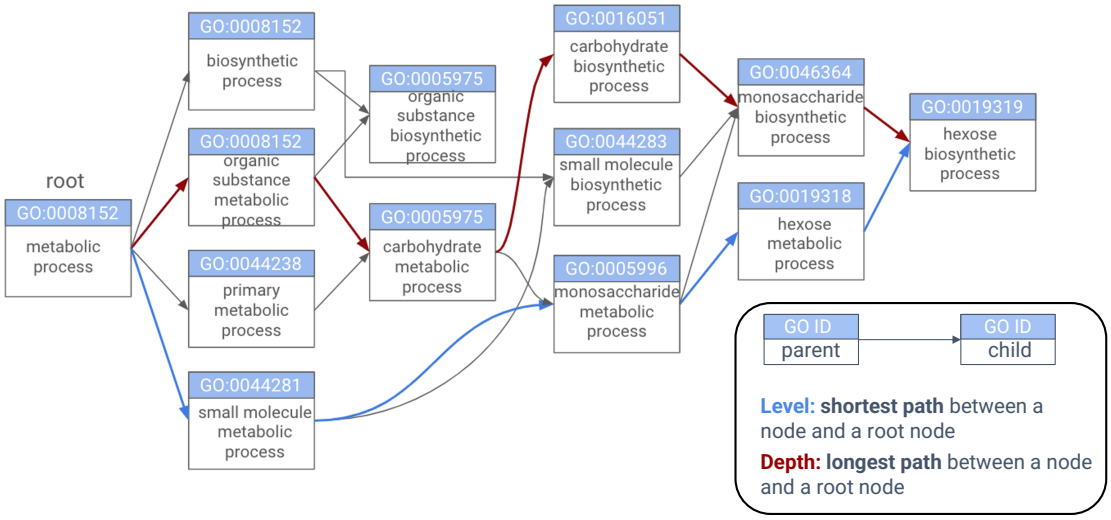

Figure 2.4: Subgraph of the GO biological process DAG. Directed arrows are connecting parent terms to children terms. With progression trough the subgraph (from left to right) starting from the root node *metabolic process* up to the leaf node *hexose biosynthetic process*, terms are becoming more and more specific. *Level* refers to the shortest path between a node and a root node (the level of *hexose biosynthetic process* is 4), while *depth* refers to the longest possible path between a node and a root node (the depth of *hexose biosynthetic process* is 5). Terms in the graph are arranged by depth, so that terms with the same depth are depicted below each other.

to provide biological explanations in a cancer classification task. The successor of this method is GraphGONet, which models the GO topology directly without regularization (Bourgeais, Zehraoui, and Hanczar 2022). The model that was developed in this thesis, OntoVAE, is a VAE that integrates the GO DAG in its latent space and decoder part (Doncevic and Herrmann 2023).

2.3 Cellular perturbations

Another task that is more and more commonly addressed by DL methods in computational biology is the modeling and prediction of cellular perturbations. The term 'cellular perturbations' has been introduced to denote the entirety of influences that cells are constantly subjected to that affect their phenotype (Ji et al. 2021). These can be of genetic nature, such as gene knockouts or overexpressions, or caused by external stimuli such as drug treatment (**Figure 2.5**). Cells constantly respond to their environment through the so-called epigenetic regulation. While every cell contains the same genetic blueprint, the body is composed of many different cell types and tissues, all with their specialized functions. Epigenetic regulation mechanisms determine the spatio-temporal patterns of gene expression and allow the cell to respond to perturbations such as developmental and environmental cues (Jaenisch and Bird 2003). In a broader sense, a disease can also be considered a perturbation as it interferes with the normal, healthy status of a cell. With the emergence of next-generation sequencing techniques such as RNA-seq and scRNA-seq, it became possible to study how the phenotype changes in response to a perturbation in a tissue or in an individual cell. Hence, cell type specific perturbation responses can be uncovered that might go unnoticed in a bulk tissue. In the following, I will briefly discuss two types of perturbations that have been investigated in the scope of this thesis: schizophrenia, a mental disorder, and interferon response, which is usually triggered upon viral infection.

2.3.1 Schizophrenia

Schizophrenia is a severe psychiatric disorder with an incidence of approximately 1% in the population worldwide, thus posing a great burden on patients and families, but also on healthcare and societies (Insel 2010; Prince et al. 2007). Due to the heterogeneity of the disease, affected individuals suffer from a wide range of symptoms which can be broadly separated into three categories: positive symptoms (hallucinations, delusions, paranoia), negative symptoms (social withdrawal, apathy), and cognitive impairment

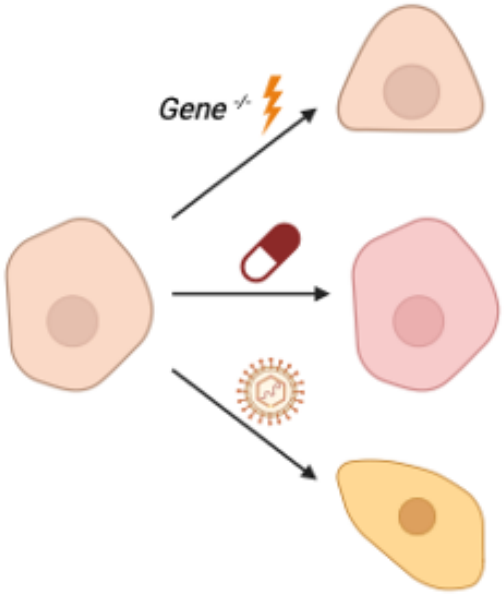

Figure 2.5: Different kinds of cellular perturbations. Perturbations that affect the phenotype of a cell include genetic knockouts, drug treatment, and disease. Figure was created with BioRender.com

affecting concentration, executive function and memory (Lu et al. 2022; Birnbaum and Weinberger 2017). Men and women are equally affected by the disease, and for most patients, the onset occurs in their adolescence (Picchioni and Murray 2007; Nishioka et al. 2012). An accurate diagnosis of schizophrenia is difficult and can often change over time, partly due to the wide range of symptoms that can manifest in affected individuals, but also due to the fact that many of the symptoms can also be indicative of related diseases or substance abuse (Schultz, North, and Shields 2007). Once the diagnosis has been made, treatment with antipsychotic drugs, usually dopamine antagonists, is initiated. Together with psychotherapy and family interventions, the medication can alleviate the condition, however, most of the patients rely on therapy for the rest of their lives. The antipsychotic drugs also have many metabolic and neurologic side effects, among them obesity and susceptibility to type 2 diabetes (P. Li, Snyder, and Vanover 2016). Type 2 diabetes together with cardiovascular diseases is one of the common comorbidities of

schizophrenia, thus making it difficult to dissect the contributions of mechanistic and drug treatment related processes that lead to the development of a comorbid disease in schizophrenia patients (Mangurian et al. 2016).

The molecular origin of schizophrenia is still not well understood. It is known that both, the genetic background and environmental factors, contribute to the development of the disease. A large genome-wide association study (GWAS) identified 108 schizophrenia-associated genetic loci, among others associated to genes involved in glutamatergic neurotransmission and synaptic plasticity, and encoding subunits of voltage-gated calcium channels (Ripke et al. 2014). Furthermore, it is known that a microdeletion in chromosome 22q11, which causes developmental defects such as DiGeorge syndome and velocardiofacial syndrome, is associated with a 30% risk of developing schizophrenia (Bassett and Chow 2008). Some of the genes in this locus were independently identified as schizophrenia susceptibility genes.

Besides, family and twin studies of schizophrenia have proven a strong genetic influence, with the lifetime incidence increasing with degree of relativity to an affected individual. For first-degree relatives, the risk of developing schizophrenia is at 6 to 17%, and for monozygotic twins at 50% (Schultz, North, and Shields 2007; Lewis and Lieberman 2000). This also highlights the fact that onset of schizophrenia cannot be solely attributed to the genetic background, but is also associated with environmental factors. Known early environmental agents are encountered mostly due to obstetric complications and involve prenatal viral infection, a premature birth, perinatal hypoxia, and reduced nutrition before and after birth (Kunugi, Nanko, and Murray 2001; Brown and Derkits 2010). Later in infancy and early adulthood, environmental factors that increase the risk for schizophrenia are social isolation, socioeconomic status, an urban environment, belonging to a minority group, and cannabis consumption (Os, Kenis, and Rutten 2010).

For patients and families, it is important to get a better mechanistic understanding of onset and development of schizophrenia in order improve diagnosis and to design novel therapies that are better tailored to the needs of the individual. Thus, in this thesis, in

the scope of COMMITMENT, I analysed a single cell dataset of schizophrenia to identify molecular pathways that are involved in the disease.

2.3.2 Interferon response

Interferons (IFNs) are a family of cytokines that alert the immune system to the presence of viral infection and activate defense mechanisms (Le Page et al. 2000). They are secreted by infected cells and immune cells and bind to cell surface receptors to trigger signal transduction cascades that culminate in the transcription of so-called interferon stimulated genes (ISGs). IFNs can be subcategorized into type I (IFNα, IFNβ), type II (IFNγ), and type III (IFNλ), with each type initiating a slightly different cascade. Type I and type III IFNs are more similar in their response, and considered to be the classical antiviral IFNs. Both lead to the activation of the same transcription factors (TFs), including STAT1, STAT2, IRF7 and IRF9, which then induce ISG transcription (Schoggins 2019).

Per definition, an ISG is a gene that is induced during IFN response, however, the complexity of the system makes the understanding and nomenclature of ISGs more difficult. Some ISGs already exhibit a baseline expression in the absence of IFNs while others do not, some are direct targets of members of the signalling cascade and can thus be induced independently of IFN signaling. For the members of the signalling cascade, some are IFN-inducible themselves, leading to multiple potential pathways that can promote transcription of a single ISG (Schoggins 2019).

Interestingly, about 10% of the human genes are potential IFN targets. With the development of next-generation sequencing technologies, studies can now systematically profile ISGs in different tissues. Viral infections affect the respiratory tract through aerosol transmission, but also the gut through oral and fecal transmission. Over the past years, intestinal organoids have emerged as surrogate models as they mimic the composition of the intestine (Triana et al. 2021). In this thesis, in collaboration with the group of Steeve Boulant, I analysed single-cell data from colon and ileum organoids that

had been treated with IFNβ, IFNλ, or both, to identify the tissue- and cell type-specific ISGs.

2.3.3 Computational analysis of cellular perturbations

As already stated, cellular perturbations change the phenotype of a cell. Gene expression, for example, can be used as a measure to characterize and quantify this change. One goal of computational biology is the development of methods that can predict the perturbation response (Lotfollahi, Wolf, and Theis 2019). Methods with high accuracy could be applied for *in silico* drug screening for example, to lower the number of potential drug candidates before validation in the wetlab. A VAE based model, scGen, was demonstrated to predict perturbation responses in single-cell data for out-of-sample data (Lotfollahi, Wolf, and Theis 2019). This model learns a mapping from unperturbed to perturbed in the latent space, and applies it on new data. Another recent method, GEARS, combines graph neural networks (GNNs) with an MLP structure, whose final output ist the gene expression prediction (Roohani, Huang, and Leskovec 2024). One GNN contains information about the gene coexpression, and the other models perturbation relationships based on GO. GEARS is also able to predict interaction mechanisms of two-gene perturbations. While these methods perform quantitative prediction at the level of gene expression, we used the model developed in this thesis, OntoVAE, for a qualitative prediction on the level of pathway activities that are encoded in the latent space and decoder of OntoVAE (Doncevic and Herrmann 2023).

The CPA (Compositional Perturbation Autoencoder) model, which was developed by Lotfollahi *et al.* (2023), is autoencoder based, and can separate drug perturbation effects from other confounders such as the celltype by factorizing the latent space. We used the approach implemented in CPA to extend our model OntoVAE, thus, I will explain it in more detail. The purpose of this model is to learn an overall drug perturbation state and then predict gene expression upon drug perturbations for cell types where only untreated

data is available. Mathematically, the authors formulate it the following way:

$$x'_i = M((x_i, d_i, c_i), d') \tag{2.8}$$

whereby (x_i, d_i, c_i) is a given dataset triplet that was perturbed with drug d and has the covariate c, and x'_i is the estimated gene expression if the data had been perturbed with d' instead of d_i. As already stated, this goal is achieved in CPA by separating the effects of drug treatment from other covariates in the latent space, a schematic drawing of this is represented in **Figure 2.6**. During model training, this separation is encouraged by the implementation of an adversarial approach, where for each covariate (celltype, drug), an auxiliary classifier is attached to the latent space, which is supposed to force the encoder to learn a representation of samples in which different celltypes and drugs cannot be distinguished anymore. The authors call this the learning of a basal state:

$$\hat{z_i}^{basal} = \hat{f}^{enc}(x_i) \tag{2.9}$$

Simultaneously, perturbation embeddings $V^{perturbation}$ and covariate embeddings V^{cov} are learned for each covariate through embedding layers, and those embeddings are then added to the basal state. Thus, the basal state is used both as an input to the auxiliary classifiers, but also for computation of the estimated perturbed state:

$$\hat{z_i} = \hat{z_i}^{basal} + \hat{V}^{perturbation}.(\hat{f}_1(d_{i,1}), ...\hat{f}_M(d_{i,M})) + \sum_{j=1,...,K} \hat{V}^{cov_j}.c_{i,j} \tag{2.10}$$

whereby $(\hat{f}_1, ..., \hat{f_M})$ are learnable dose-response curves that are implemented as neural networks themselves in CPA, and K is the number of categories a covariate j can take. Finally, a decoder estimates the gene expression from this perturbed state:

$$x'_i = \hat{f}^{dec}(\hat{z}_i) \tag{2.11}$$

Thus, three loss functions are used during CPA model training. The Gaussian negative log-likelihood is computed for the reconstruction loss function l_i as in standard autoencoders. Additionally, auxiliary loss functions are defined to remove covariate information from z^{basal}:

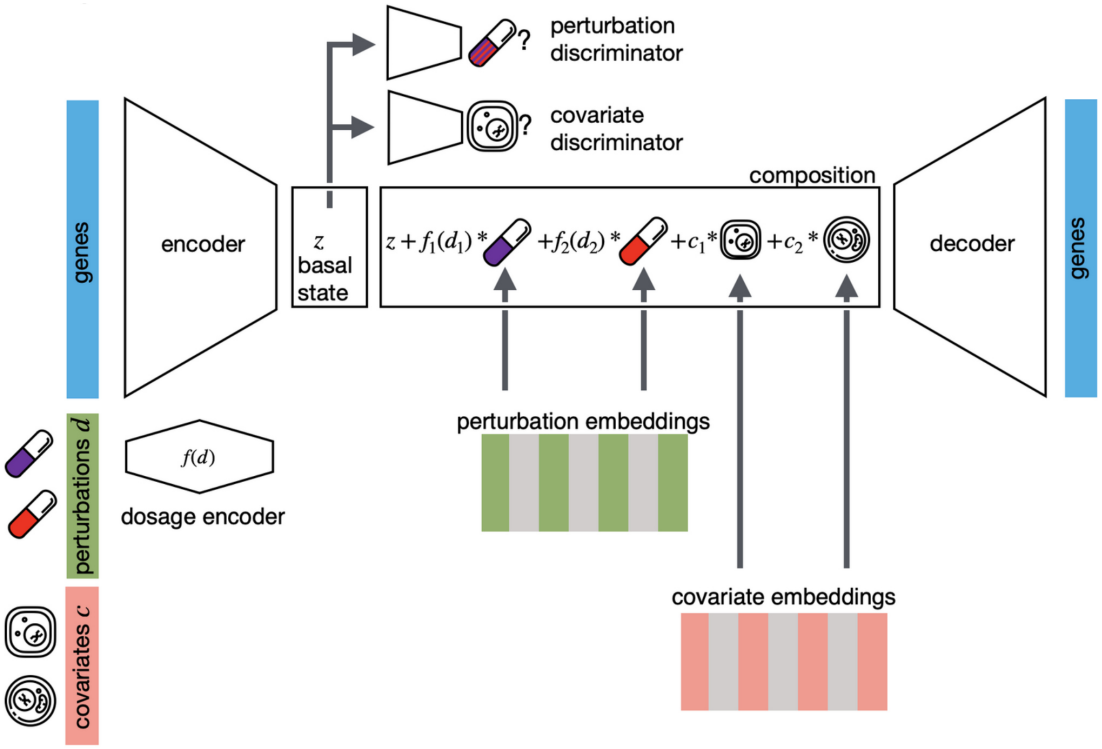

Figure 2.6: Schematic drawing of the CPA model. Figure was taken from Lotfollahi *et al.* (2023).

$$l_i^d = CrossEntropy(\hat{f}_d^{\,adv}(\hat{z}_i^{basal}), d_i)$$

$$l_{i,j}^c = CrossEntropy(\hat{f}_{c_{i,j}}^{adv}(\hat{z}_i^{basal}), c_{i,j}), \quad j = 1, ..., K$$

whereby $\hat{f}_d^{\,adv}$ and $\hat{f}_{c_{i,j}}^{adv}$ are neural network classifiers that try to predict (d_i, c_i) from the basal state $\hat{z}_i^{basal}$. The parameter optimization during training is a two-step process and follows the principle of Generative Adversarial Networks (GANs), where a generator and discriminator compete against each other, with the generator trying to generate fake samples that are as close to the real ones as possible, and the discriminator trying to discriminate between real and fake (I. J. Goodfellow et al. 2014). Thus, the authors also call this an adversarial approach. The following two steps are repeated during the training process:

- The parameters of the auxiliary classifiers $\hat{f}_d^{adv}$ and $\hat{f}_{c_{i,j}}^{adv}$ are updated to minimize $l_i^d + \sum_j l_{i,j}^c$
- The parameters of encoder $\hat{f}^{enc}$, decoder $\hat{f}^{dec}$, perturbation embeddings $\hat{V}^{perturbation}$, covariate embeddings $\hat{V}^{cov_j}$, and dose-response curve estimators $(\hat{f}_1, ..., \hat{f_M})$ are updated to minimize $l_i - \lambda(l_i^d + \sum_j l_{i,j}^c)$

The interpretable VAE model that was developed in this thesis, OntoVAE, was extended with an adversarial approach analogous to CPA, to be able to dissect the contributions of different covariates. This new COBRA model is still unpublished.

Part I

Tool Development

Chapter 3

OntoVAE: Interpretable VAE based on biological ontologies

Disclosure: The results that are presented in this section have been published in Doncevic and Herrmann (2023).

With the advance of high-throughput sequencing technologies, researchers became able to collect large amounts of biological data. For example, genomics profiles the DNA sequence, transcriptomics quantifies the abundance of RNA molecules, proteomics measures protein abundance, and so on. All these techniques can be grouped under the collective term 'omics' (Hasin, Seldin, and Lusis 2017). VAEs have been widely used to extract meaningful patterns from this high-dimensional data, and have been applied on many different tasks, including data denoising, sample clustering, batch correction, and transfer learning. However, VAEs are lacking interpretability as we cannot assign the contributions of different features to the patterns learnt in their latent space (Svensson et al. 2020). As model interpretability is a crucial aspect especially in biomedical applications, we constructed a novel interpretable VAE termed 'OntoVAE' in whose latent space and decoder any biological ontology can be incorporated (Doncevic and Herrmann 2023). Therefore, by design, the neurons in latent space and decoder correspond to terms

of the integrated ontology, and their activations can be directly interpreted as pathway or phenotype activities.

3.1 Model architecture

We named our novel VAE architecture OntoVAE (short for Ontology guided VAE) as the structure of latent space and decoder of OntoVAE is guided by a biological ontology, so that each neuron in these layers corresponds to a term of the given ontology. Hence, OntoVAE consists of a fully connected non-linear encoder which is coupled to a sparse, multi-layer linear decoder (**Figure 3.1**), whereby the sparse connections in the decoder reflect either relationships between parent and children terms of the ontology or annotations of genes to terms of the ontology. Furthermore, the decoder is organized into multiple layers based on ontology depth, meaning that each layer in the decoder represents one depth layer of the ontology and terms that have the same depth are located in the same layer. Hereby, the term 'depth' refers to the longest possible path between a given node/term and a root node. In contrast to a standard VAE where connections are only present between neighboring layers, OntoVAE also has to model connections between non-neighboring layers, since parent child term relationships are not restricted to neighboring depth layers of the ontology. Additionally, annotations have to be modelled between the genes in the output layer and ontology terms in each depth layer.

The root terms, which are at the same time the most generic terms of the ontology, are localized in the latent space layer (depth 0). With progression through the following layers of the decoder, terms are becoming more and more specific until reaching the reconstruction layer which contains the genes that could be mapped to the ontology. In general, a biological ontology, as any directed acyclic graph (DAG), has one root node. As a one-dimensional latent space would not be meaningful, we apply a pruning process to every ontology before incorporation into OntoVAE which is illustrated in **Figure 3.2**. Basically, a top and a bottom pruning threshold define the maximum and minimum number of annotated genes that is allowed. Terms with a higher number of annotated

genes than the top threshold are removed, and their children are eventually relocated to a different depth layer to ensure connectivity. This process usually removes the root node, and converts many of the children terms to new root terms. Terms with a lower number of annotated genes than the bottom threshold are removed likewise, and their annotated genes are transferred to all of their parent terms.

Due to the orientation of the ontology in the model, with information flowing from parent to children terms, two children would be completely correlated if they shared the same parent which means that the input they receive would completely overlap. In order to mitigate this correlation effect, we use three neurons for each term.

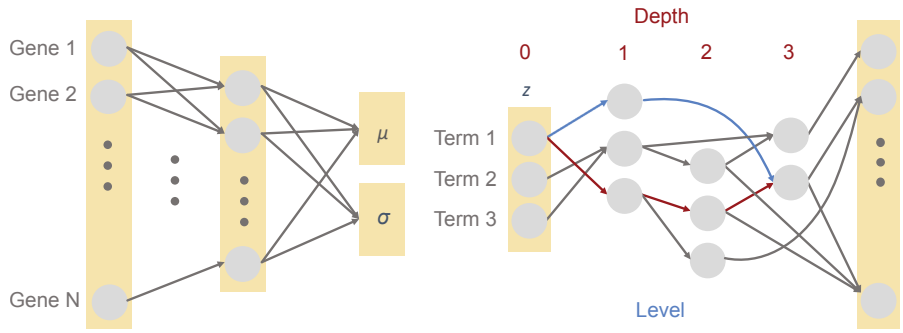

Figure 3.1: Schematic drawing of OntoVAE model. A non-linear encoder is connected to a masked, multi-layer linear decoder. The structure of latent space and decoder reflects a biological ontology, and the wiring reflects the connections between parent and children terms, and the annotation between terms and genes in the given ontology. The root terms of the ontology are located in the latent space layer, and each layer in the decoder corresponds to one depth layer of the ontology. Note that 'depth' refers to the longest possible path between a node and a root node, while 'level' refers to the shortest possible path between a node and a root node.

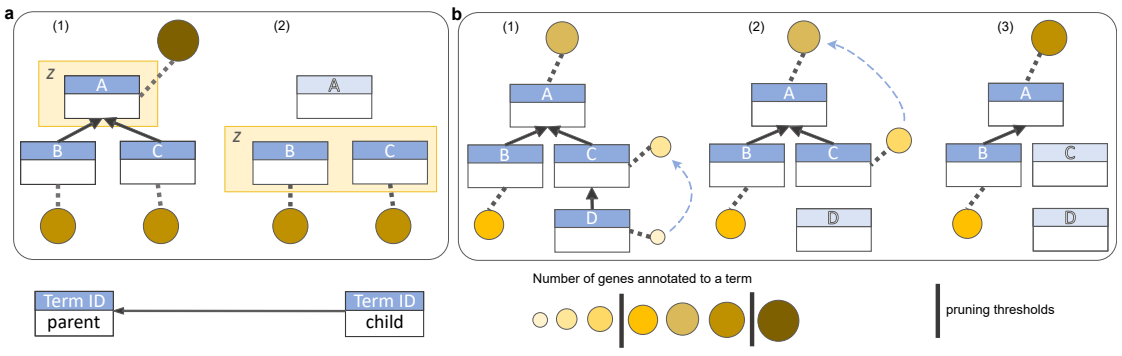

Figure 3.2: Overview of the ontology pruning process. (**a**) Top pruning. Term A is a root term and located in the latent space. Since the number of genes annotated to term A is higher than the top pruning threshold, term A is removed and its children terms B and C are relocated to the latent space. (**b**) Bottom pruning. (1) Term D is a leaf term and has less genes annotated to it than the bottom pruning threshold. Therefore, D will be removed and its annotated genes will be transferred to its parent terms, in this case term C. (2) The number of genes annotated to C is still lower than the bottom pruning threshold, so C will also be removed and its annotated genes will be transferred to its parent term A. (3) The number of genes annotated to A lies within the pruning thresholds, so pruning stops here.

3.2 Model implementation

OntoVAE was implemented in pytorch and made freely available through pip and GitHub at `https://github.com/hdsu-bioquant/onto-vae` under the GPLv3 license. The python package has two main classes, `Ontobj()` and `OntoVAE()`. The `Ontobj()` class handles the preparation of a given ontology and dataset and has predefined slots which store the necessary information, such as:

- annotation of the ontology: a dataframe containing information such as term identifier, term name, depth, number of parents, number of children, number of descendants, number of annotated genes, number of descendant genes

- genes of the ontology: an alphabetically sorted list containing the genes that could be mapped to the ontology and therefore define the input to OntoVAE
- ontology graph: a dictionary containing all mappings between children and parent terms as well as between genes and terms they are annotated to
- matched datasets: datasets whose features have been matched to the genes of the ontology

It is possible to store multiple trimmed versions with different pruning thresholds in the same `Ontobj()` instance. It is also possible to store multiple matched datasets per trimmed version in the object. The `OntoVAE()` class is initialized with an instance of the `Ontobj()` class and contains the actual implementation of the VAE. The class itself is composed of an encoder building block, and an OntoDecoder building block which contains the logic of the ontology. Alongside with that, an OntoEncoder building block, and a standard decoder were also implemented, so that the user could also create a reverse model. Tests with this were also carried out in the manuscript. The modular implementation of OntoVAE makes it very flexible. Additional functions for model training and retrieving information make the tool user-friendly, since it requires only a handful of commands to run a complete analysis. A vignette that was made available together with the OntoVAE package illustrates how to initialize and train an OntoVAE model.

The code for the ontology preparation as shown in the vignette is displayed below.

```
# import modules
import os
from onto_vae.ontobj import *
from onto_vae.vae_model import *


# initialize the Ontobj
# the description should be an identifier, e.g. the ontology used,
# here: PWO (Pathway Ontology)
```

```
pwo = Ontobj(description='PWO')

# initialize our ontology
# obo: path to an obo file
# gene_annot: path to a tab separated file with two columns:
#              Genes and Ontology IDs
pwo.initialize_dag(obo=data_path() + 'pw.obo',
                   gene_annot=data_path() + 'gene_term_mapping.txt')

# trim the ontology
pwo.trim_dag(top_thresh=1000,
             bottom_thresh=30)

# create masks for decoder initialization
pwo.create_masks(top_thresh=1000,
                 bottom_thresh=30)

# match a dataset to the ontology
# expr_path: path to the dataset (either h5ad)
pwo.match_dataset(expr_data = data_path() + 'pbmc_sample_expr.csv',
                  name='PBMC_CD4T')
```

Two files have to be provided by the user, an obo file containing the ontology and an annotation file containing the mapping from genes to ontology terms. From these files, the initial ontology graph dictionary is built. The user then chooses a top and a bottom pruning threshold to trim the ontology. In a next step, the binary masks that are needed to initialize the decoder wiring are created. This process is illustrated in more detail in **Figure 3.3**.

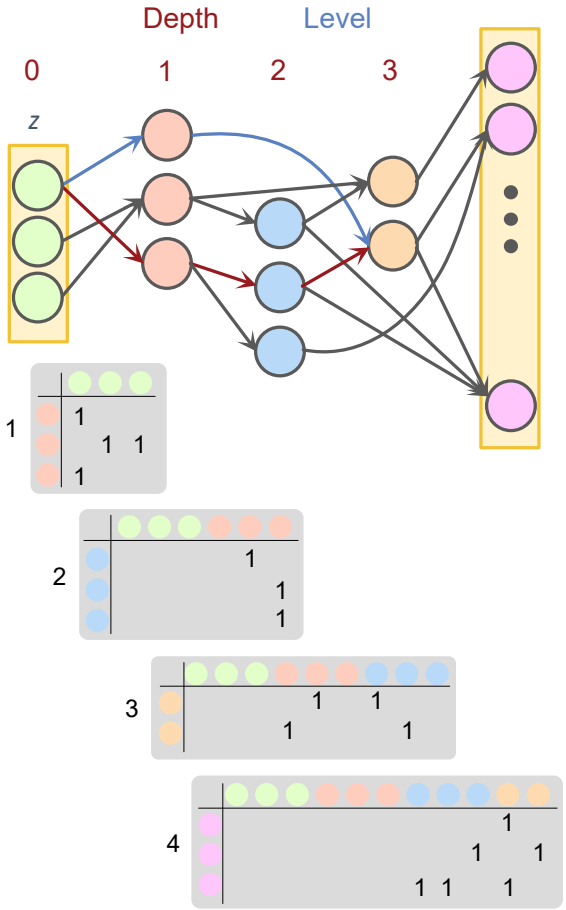

Figure 3.3: Representation of binary masks for decoder wiring initialization. The binary masks are generated through a step-by-step concatenation process. Mask 1 models the connection between parent terms in the depth 0 layer (root terms in latent space) and children terms in the depth 1 decoder layer. Mask 2 models the connections between parent terms in both, the depth 0 and the depth 1 decoder layers, and children terms in the depth 2 decoder layer. Mask 3 models connections between parent terms in depth 0, 1, and 2, decoder layers, and children terms in depth 3 decoder layer. Finally, mask 4 models the annotations of the genes (output layer) to all terms of the ontology (depth layers 0, 1, 2, and 3). The entry 1 in the binary mask indicates parent child relationships (masks 1-3) or gene annotations (mask 4).

Since connections also have to be modelled between non-neighbouring layers, at each step, the previous layer is concatenated to the current one. The final binary mask contains the annotation between the genes in the output layer and all terms. Finally, the datasets that we want to use for model training or passing through a trained model are matched to the ontology genes and stored in a separate slot of the `Ontobj()`. Instances of the `Ontobj()` class can be saved and loaded with pickle.

The code for initialization and training of an OntoVAE model as shown in the vignette is displayed below:

```
# initialize OntoVAE
pwo_model = OntoVAE(ontobj=pwo,             # the Ontobj we will use
                    dataset='PBMC_CD4T',    # which dataset from the Ontobj to use
                    top_thresh=1000,        # which trimmed version to use
                    bottom_thresh=30)       # which trimmed version to use
pwo_model.to(pwo_model.device)

# generate a directory where to store the best model
if not os.path.isdir(os.getcwd() + 'models'):
    os.mkdir(os.getcwd() + 'models')

# train the model
pwo_model.train_model(os.getcwd() + 'models/best_model.pt',
                      lr=1e-4,              # learning rate
                      kl_coeff=1e-4,        # KL loss weighting coefficient
                      batch_size=128,       # minibatch size
                      epochs=5,             # number of training epochs
                      run=None)             # Neptune run for logging can be passed here
```

An instance of the `Ontobj()` class needs to be provided and the user needs to specify which trimmed version and which matched dataset will be used. Here, we can also define

the number of neurons that should be used per ontology term. For the model training process, the user can tune the following parameters: learning rate, Kullback-Leibler (KL) loss weighting coefficient, minibatch size, and number of epochs. Additionally, the user can decide if losses of the run should be logged to Neptune, which is seamlessly integrated in OntoVAE, by passing a run to the run parameter.

The main functions that are available to the user to carry out analysis with a trained model are:

- `get_pathway_activities()`: This function retrieves the pathway activities, averaged by the number of neurons that was used per term.

- `get_reconstructed_values()`: This function retrieves the values of the reconstruction layer.

- `perturbation()`: This function takes as input a list of genes to be perturbed together with a list of their new input values, and retrieves either the pathway activities or the reconstructed values after the perturbation.

Additionally, functions were implemented for plotting and carrying out statistical tests.

3.3 Proof of concept: GTEx dataset

We hypothesize that the latent space and decoder part of OntoVAE are directly interpretable through their structure. Thus, we postulate that the activations, that can be retrieved for each neuron after passing samples through a pre-trained model, correspond to pathway or phenotype activities, given the used ontology. Furthermore, we assume that the results are somewhat reproducible for different runs of model training. To test the OntoVAE model, we used the bulk RNA-seq data from the Genotype Tissue Expression (GTEx) consortium, that measures gene expression in 31 different tissues.

3.3.1 Investigation of pathway activities

We trained an OntoVAE model with Gene Ontology (GO)-decoder, using pruning thresholds of 30 and 1,000, on the bulk RNA-seq samples from GTEx. We then extracted the pathway activities for each GO term, that are calculated as the average activation of the neurons representing this term, and checked if the results matched our biological knowledge, and if certain pathways displayed higher activities in the expected tissues. According to our expectations, *digestive system process* was especially active in *Small Intestine*, *Colon*, and *Stomach*, *glutamate receptor signaling pathway* was especially active in *Brain* (**Figure 3.4a**), *aortic valve morphogenesis* was especially active in *Heart*, and *axon ensheathment* was especially active in *Nerve* and *Brain* (**Figure 3.4b**).

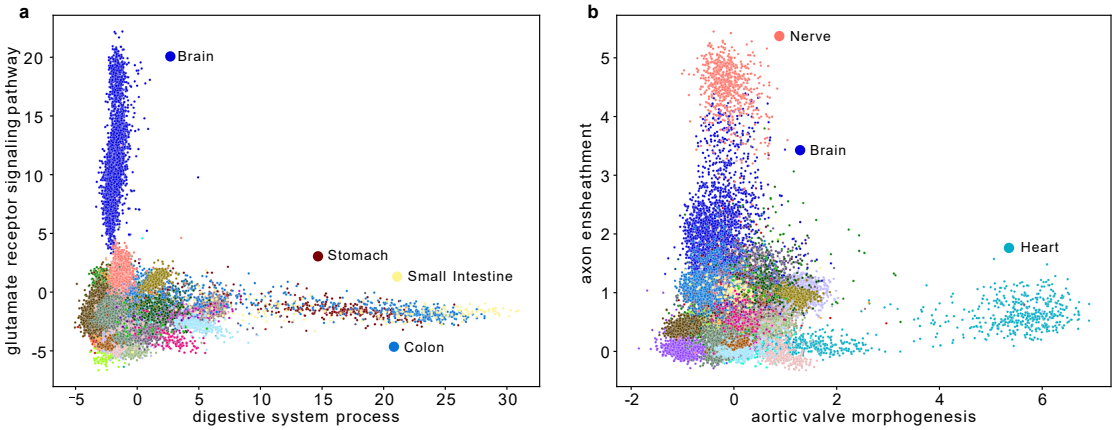

Figure 3.4: Scatter plot displaying the pathway activities for the examples *digestive system process* and *glutamate receptor signaling pathway* (**a**) and *aortic valve morphogenesis* and *axon ensheathment* (**b**). OntoVAE was trained with GO-decoder on bulk RNA-seq tissue samples from GTEx, the different tissues are displayed in different colors.

Next, we set out to investigate whether the pathway activities obtained from OntoVAE could classify the correct tissues with high accuracy. For this purpose, we trained a Naive Bayes classifier with 10-fold cross-validation on the pathway activities for each GO term. We performed this for each tissue separately in a 1-vs-all setting, and then

calculated the median area-under-the-curve (AUC) from the cross-validation. According to our expectations, the AUC is 0.5 for majority of the tissues, which means that the classification does not perform better than random, but a higher AUC is achieved for the seemingly correct tissues, e.g. for *Adipose Tissue* and *Breast* in *neutral lipid metabolic process*, for *Pancreas* and *Stomach* in *Digestion*, for *Liver* and *Small Intestine* in *drug metabolic process*, or for *Nerve* and *Brain* in myelination (**Figure 3.5a**). For some pathways, we also find some unexpected tissues with a high AUC, such as *Brain* for *muscle system process* and *gland development*, or *Blood* for *brain development*. However, when we look at the density of tissues with an AUC higher than 0.5 over the pathway activities, we see that in these cases, the accurate classification is due to a very low activity value of these tissues in these pathways (**Figure 3.5b**). Thus, a good classification accuracy of a pathway for a certain tissue does not necessarily correspond to a high activity of this pathway in the tissue. We also looked at the clustering of the top ten GO terms with AUC higher than 0.5 of each tissue, and saw that especially for *Brain*, there is a lot of overlap with other tissues, however, we speculate that this is due to low activity of *Brain* in many other pathways (**Figure 3.6**).

Since our model incorporates a hierarchical representation by design, we wanted to compare how different samples can take different trajectories through the interpretable decoder. We used the previously calculated median AUCs for quantitative assession and mapped them back to the ontology. We display the same example subnetwork of the GO graph for *Blood* and *Spleen* (**Figure 3.7**), with node colors corresponding to the median AUC for the respective term. Since *Blood* and *Spleen* are both composed of blood cells, the two tissues are somewhat related. The displayed subnetwork can roughly be divided into two branches. Interestingly, while both branches are lit up for *Blood*, only the right branch is lit up for *Spleen*. Upon closer inspection, we can see that the terms in the left branch are more related with the myeloid lineage, whereas the terms in the right branch are more related with the lymphoid lineage. Thus, our results can be explained by the fact that *Spleen* is a reservoir especially for lymphocytes and plays an important role in immune response. Another interesting observation is that different

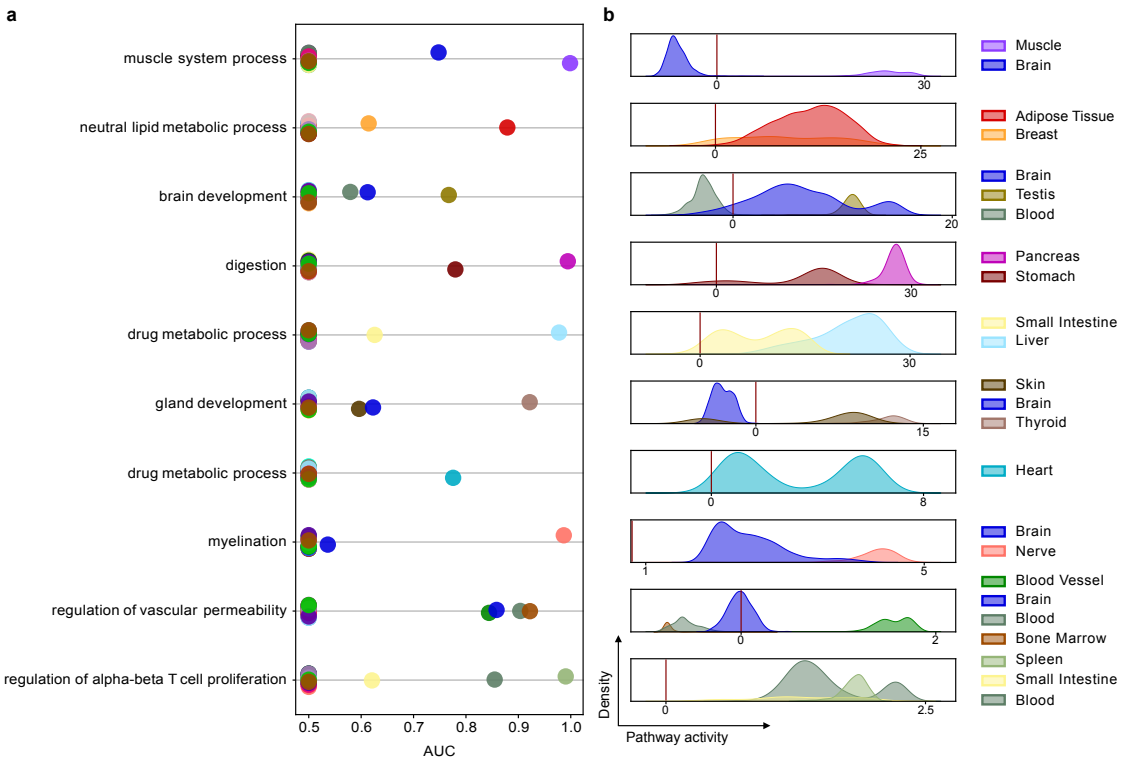

Figure 3.5: Pathway activities can classify correct tissues. Naive Bayes classifier was trained for each GO term for each tissue in a 1-vs-all setting with 10-fold cross-validation. (**a**) Median AUCs are displayed for ten example pathways. (**b**) For tissues with a median AUC higher than 0.5, density over the pathway activity is displayed.

depths of the ontology can provide different information. The node *B cell activation involved in immune response* shows a high AUC in both tissues, for example, whereas its parent terms have highly divergent AUCs.

Since the classification approach cannot distinguish between high and low pathway activities, we devised an alternative approach that allowed us to to investigate our results in a group-specific manner, and to discover terms that were most active in specific tissues. We cannot directly rank the pathway activities for a given tissue, since they are not directly comparable with each other due to internal biases at the nodes. Thus, we per-

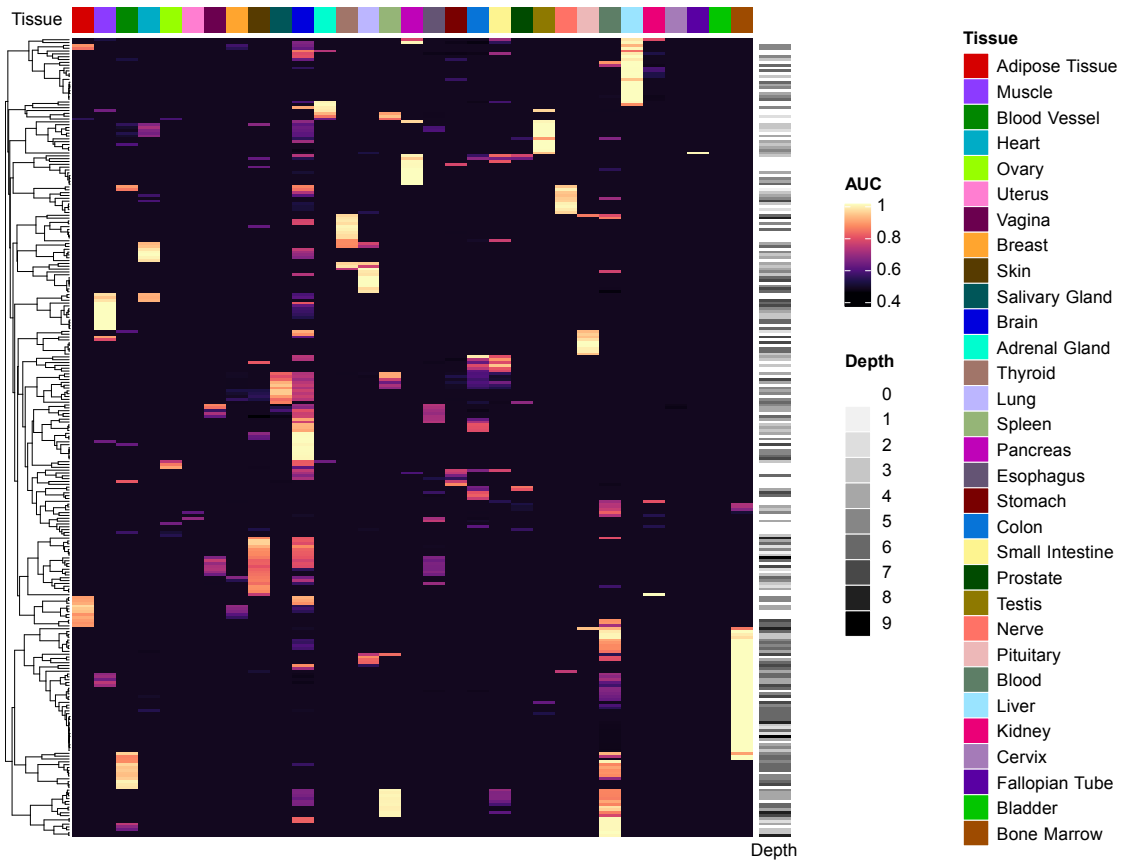

Figure 3.6: Heatmap of top classifying GO terms. Ten GO terms with the highest median AUC were selected for each GTEx tissue given the AUC was higher than 0.5. The terms are clustered and annotated with their depth level in the trimmed ontology.

formed separate comparisons between groups of samples at each node. More specifically, we computed Wilcoxon tests at each node for all possible two pairs of tissues, and called it a hit if a term was significantly more active in one tissue compared to another (p-value < 0.05). We than ranked all tissues for one term, with the tissue with most hits for this term receiving rank 1, the tissue with second most hits receiving rank 2 and so on. For a given tissue, we then sorted all the terms based on the rank the tissue had received for this term, the number of hits, and the median of the Wilcoxon test statistic. The top 100 terms for the tissue were then extracted and then clustered based on their Wang seman-

tic similarities. The cluster representative was either the common ancestor of all cluster members or the term with most annotated genes. The clusters for *Blood* and *Liver* are displayed in **Figure 3.8**. For *Blood*, we find many clusters that are in accordance with our biological intuition, such as *neutrophil degranulation*, *myeloid leukocyte migration*, *innate immune response*, *regulation of adaptive immune response*, *myeloid cell differentiation*, *positive regulation of T cell activation*, *negative regulation of leukocyte activation*, and *type I interferon signaling pathway*. For *Liver*, we also detect meaningful clusters, such as *regulation of lipid metabolic process*, *plasma lipoprotein particle remodeling*, *cellular oxidant detoxification*, *xenobiotic metabolic process*, *small molecule catabolic process*, *glucose homeostasis*, and *reponse to drug*.

3.3.2 Investigation of reproducibility

Although the examples we looked at agreed with our biological intuition, we wanted to make sure that the results generated by OntoVAE are actually reproducible. For the pathway activities to be truly meaningful, we would expect that, when we train a model twice with the same parameters, the results would still agree. In this context, we also set out to investigate the behavior of the model in case that the ontology was integrated in the encoder part of the model. We trained ten models for both cases with the same parameters, and also investigated the influence of another parameter, the number of neurons used per term. Here, we tried the values 1, 2, and 3, so that in total, 30 models were trained with a GO-decoder, and 30 models were trained with a GO-encoder. Training was still performed on bulk RNA-seq tissue samples from GTEx. The agreement of the models was evaluated by calculating the pearson correlation of the pathway activities for the same term between two separate models, e.g. $cor(termA_{modelA}, termA_{modelB})$, $cor(termB_{modelA}, termB_{modelB})$. We compared models that were trained with the same number of neurons per term (here, pearson correlations were calculated for all possible pairs of two models), and models that were trained with a different number of neurons per term (here, pearson correlations were only calculated between two models).

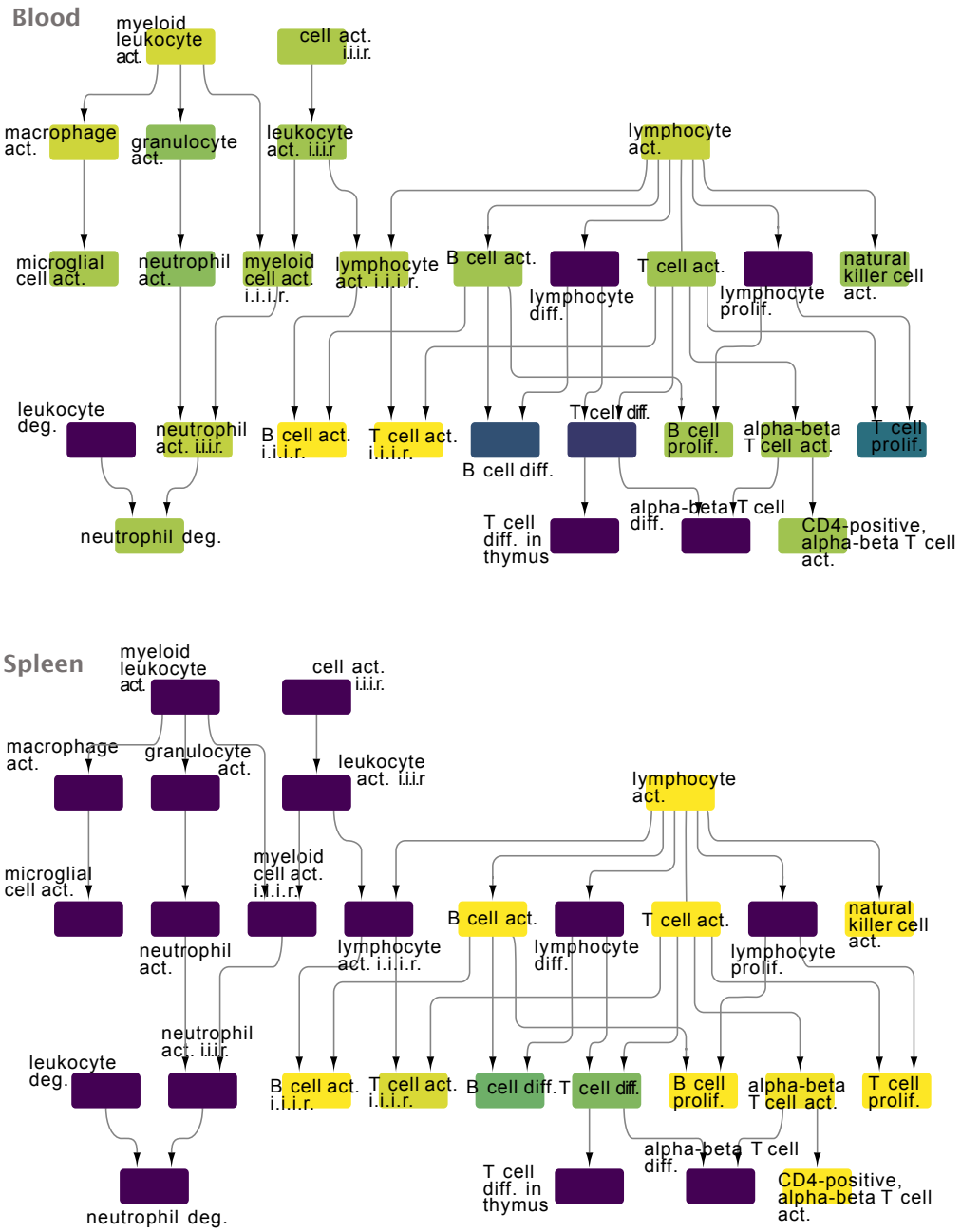

Figure 3.7: Subnetwork of the GO graph mapped on *Blood* (top) and *Spleen* (bottom). Nodes are colored by the median AUCs for the respective tissue. Abbreviations: act. - activation, diff. - differentiation, prolif. - proliferation, deg. - degranulation, i.i.i.r. - involved in immune response.

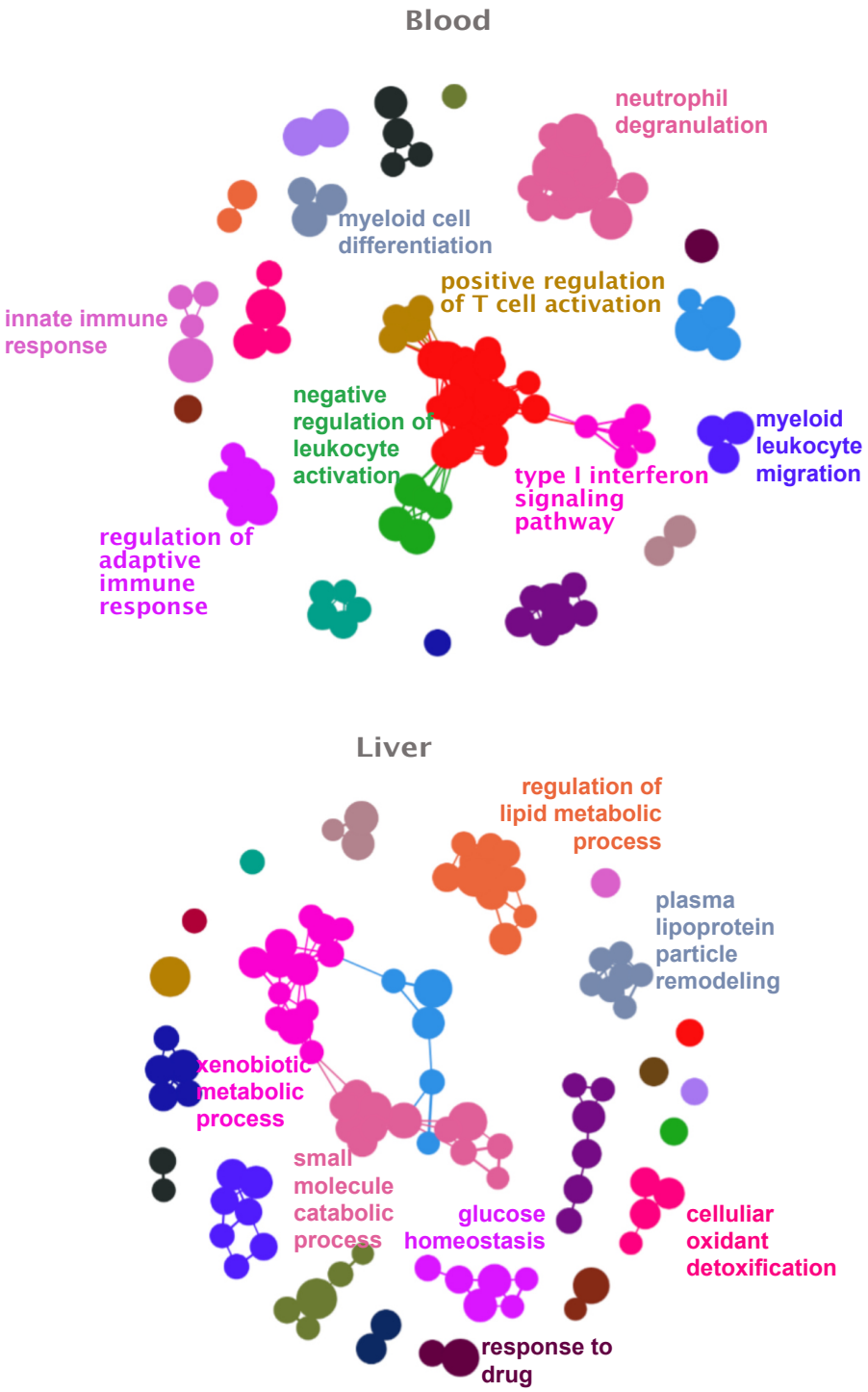

Figure 3.8: Network displaying the top GO term clusters in *Blood* (top) and *Liver* (bottom).

We can see that, when integrating the ontology in the decoder, models trained with the same parameters agree very well, with most of the pearson correlations being above 0.9, and even close to 1 between models that had been trained with the same number of neurons per term. On the other hand, when the ontology is integrated in the encoder, pearson correlations are worse, with half of the samples having values above 0.8, but a quarter of the samples having values below 0.2 (**Figure 3.9a**). We also compared the pearson correlations between GO-decoder and GO-encoder models, and found them to be 0.5 on average (**Figure 3.9b**).

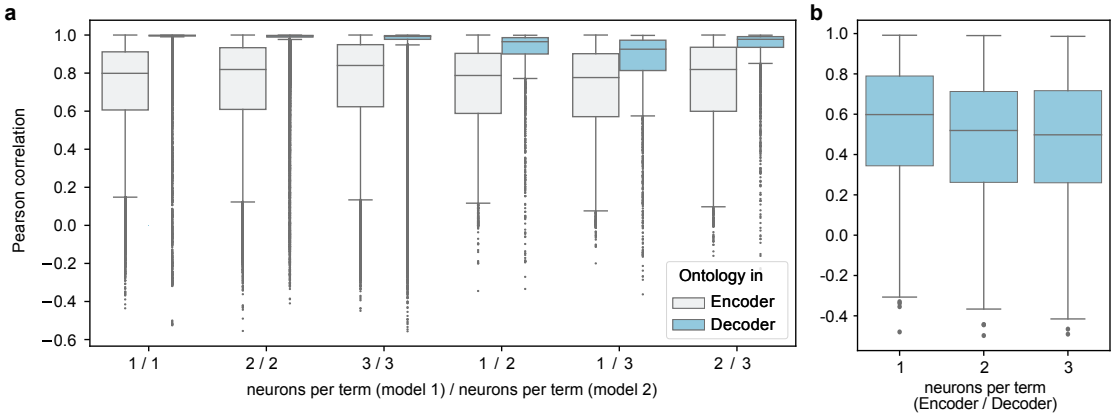

Figure 3.9: Boxplots displaying the reproducibility between models. (**a**) Pearson correlations of pathway activities between models that have been trained with the same parameters, with the ontology integrated in either the decoder or the encoder. Pairwise correlations between two models are shown. Models were trained either with 1, 2, or 3 neurons per term. (**b**) Pearson correlations between encoder and decoder ontology based models.

Next, we also wanted to investigate the influence of another important parameter on model reproducibility, the chosen trimming thresholds. For this, aside of the trimming thresholds of 1000 (top threshold) and 30 (bottom threshold), we also trained models with trimming thresholds of 1200 and 10, which is less strict, and trimming thresholds of 800 and 50, which is more strict. **Figure 3.10a** shows how many terms are left in each ontology depth level after the trimming with the different thresholds, whereby the

thresholds 1/1 refer to the untrimmed ontology. We can see that the distribution becomes bimodal after trimming, since the untrimmed ontology only has one root node, but the trimmed versions have multiple root nodes. Except for the root layer, the distribution is preservered, but the trimming is drastically reducing the number of terms. We then again calculated the pearson correlations between pathway activities for the same term for the terms that were still present in all trimmed ontologies, but between models trained with different thresholds. **Figure 3.10b** shows our results. The trimming thresholds 1000/30 and 1200/10 show a good agreement, especially for the GO-encoder model. However, both show a significantly worse agreement with the thresholds 800/50. We speculate that the thresholds of 800/50 are too strict, and too many terms are being removed, so that the ontology is not accurately preserved anymore. Thus, our choice of 1000/30 seems to represent a good compromise between reducing the complexity, and preserving the ontology. Since the pearson correlations were better for the GO-decoder model, we decided to make this the default configuration for OntoVAE, although our package also allows for ontology integration in the encoder.

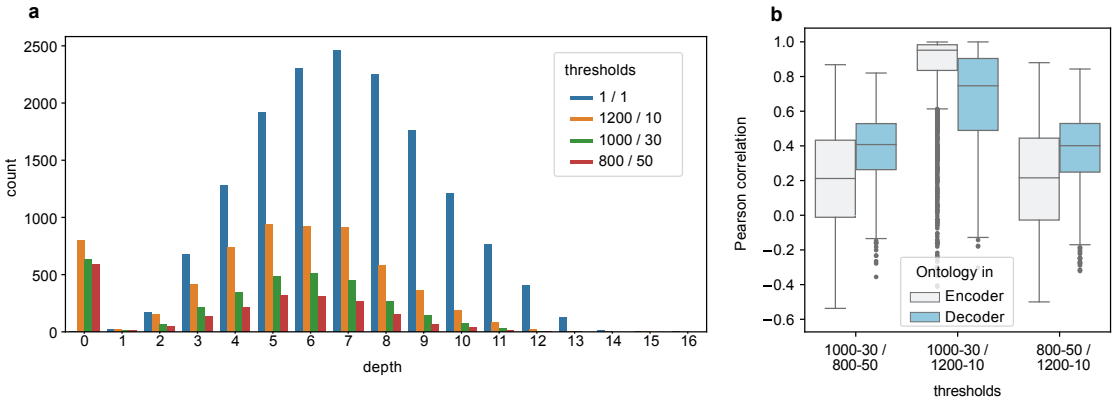

Figure 3.10: Influence of trimming thresholds on model reproducibility. (**a**) Number of GO terms in each ontology depth layer after applying different trimming thresholds. (**b**) Agreement between models with different trimming thresholds when ontology is integrated in encoder or decoder.

We also compared the validation loss of OntoVAE to the validation loss of a standard

VAE with fully connected encoder and fully connected decoder. According to our expectations, the validation loss of OntoVAE is higher, since we are trading off reconstruction accuracy for interpretability (**Figure 3.11**). With increasing number of neurons per term, the validation loss improves (**Figure 3.11a**), while changing the trimming thresholds does not have a major influence on validation loss (**Figure 3.11b**)

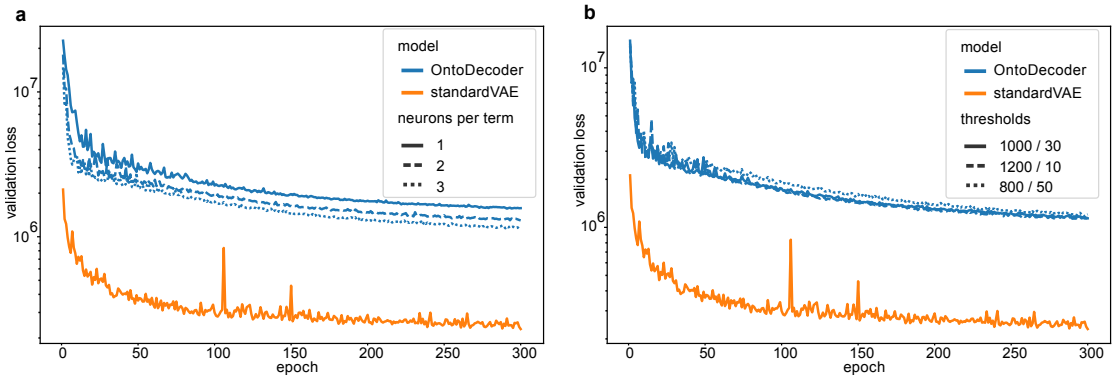

Figure 3.11: Comparison of validation loss between OntoVAE and vanilla VAE. Loss curves are shown for OntoVAE models that were trained with different numbers of neurons per term (**a**) or with different trimming thresholds (**b**).

As mentioned previously, the number of neurons per term was investigated to account for the fact that, due to the orientation of the ontology in the model, information is flowing from parent to children terms, and could therefore cause completely correlated children just because they share the same parent. To mitigate this, and allow more freedom to the model, we increased the number of neurons per term, and chose three as default, based on the results we see in **Figure 3.12**. The barplots are showing the number of pearson correlations for all pairwise term comparisons that fall above a certain threshold. This time, correlations were calculated within the same model, e.g. $cor(termA_{modelA}, termB_{modelA})$, $cor(termA_{modelA}, termC_{modelA})$ and so on. The number of highly correlated terms drastically decreases when using three neurons per term.

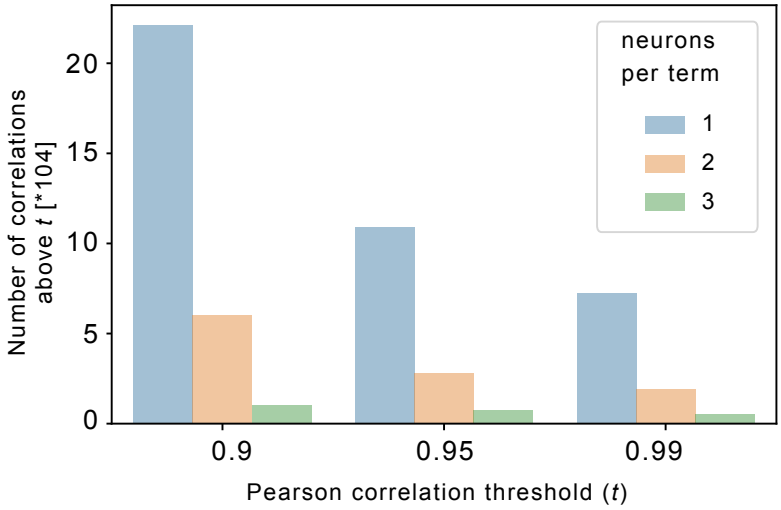

Figure 3.12: Barplots displaying amount of highly correlated terms for different numbers of neurons per term. Pearson correlations were computed between all pairs of terms, and number of terms with correlations higher than 0.9, 0.95, and 0.99 is displayed.

3.4 Chapter summary

We have developed OntoVAE, a novel interpretable VAE that can incorporate any biological ontology in its latent space and decoder part, thus making the nodes directly interpretable by assuming their activations correspond to pathway or phenotype activities. We have published the onto-vae python package on Pypi (`pip install onto-vae`) and GitHub under: `https://github.com/hdsu-bioquant/onto-vae` together with a vignette that demonstrates package use. The package is easy-to-use, a few simple functions suffice to preprocess an ontology and train an OntoVAE model. Furthermore, the package already implements easy Logging with Neptune, as well as functions for analysis such as retrieval of pathway activities. We have demonstrated the abilities of OntoVAE on Gene Ontology (GO) and bulk tissue RNA-seq samples from GTEx, using a pathway centric approach where we look at specific pathways and which tissues they are most active in, and a tissue centric approach where we construct GO networks for each tissue, highlighting the most important processes.

Chapter 4

COBRA: COvariate Biological Regulatory network Autoencoder

Disclosure: The results that are presented in this chapter are still unpublished.

With a rapidly growing number of single-cell datasets, deep learning methods are more and more becoming state-of-the-art for data processing and analysis, as they benefit from large amounts of data. When investigating the effects of cellular perturbations, such as disease or drug treatment, the cell type can be a strong confounder which masks the effect of interest. To address this, we expanded OntoVAE with an adversarial approach that encourages covariate disentanglement in the latent space. Thus, we dissect the contributions of different covariates while maintaining the interpretability in the decoder. We named the new tool COBRA (COvariate Biological Regulatory network Autoencoder).

4.1 Model architecture

COBRA combines the interpretable decoder of OntoVAE with an adversarial approach that leads to a splitting of covariates in the latent space, with the possibility of using either

a multi-layer (**Figure 4.1**) or a one-layer biological regulatory network (BRN) (**Figure 4.2**) as prior information. The adversarial approach implemented in COBRA relies on the previously published CPA tool (Lotfollahi, Klimovskaia Susmelj, et al. 2023). The mathematical details of this have already been described in the Introduction in **section 2.3.3**.

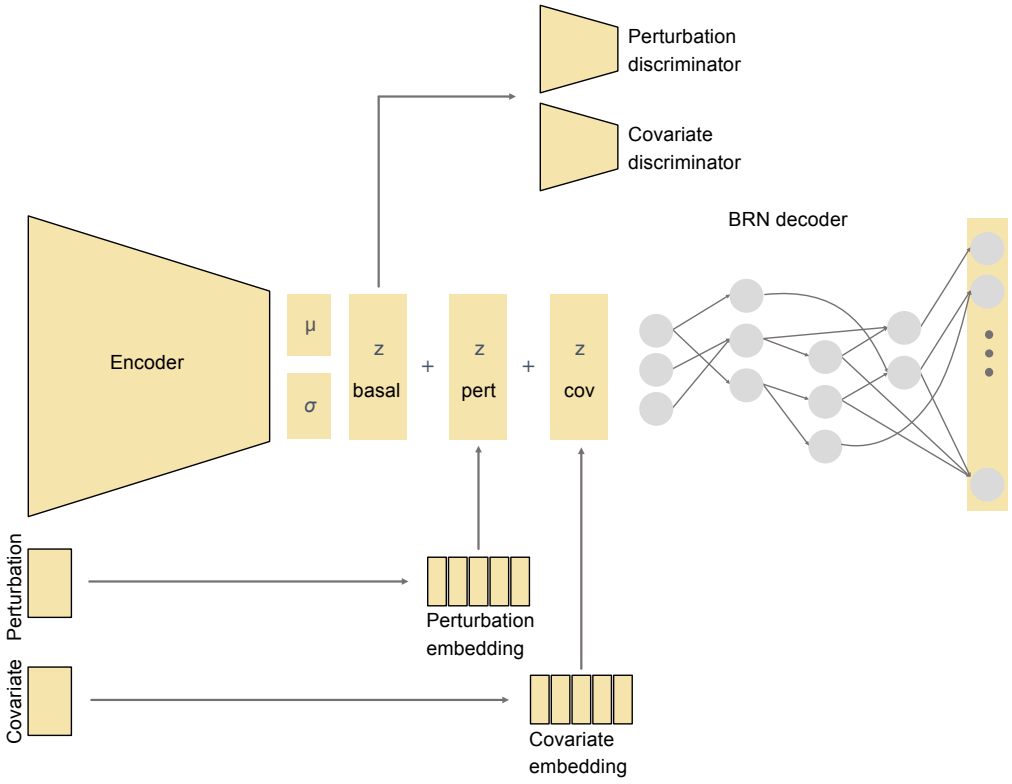

Figure 4.1: Overview of the multi-layer COBRA model. COBRA combines the interpretable decoder of OntoVAE with a splitting of covariates in the latent space. This is achieved through attaching discriminators for each covariate to the basal latent space embedding (z basal) and externally learning the embeddings for each covariate through torch.nn.Embedding layers. The separate embeddings are then added up, and the total embedding is fed into the interpretable decoder part.

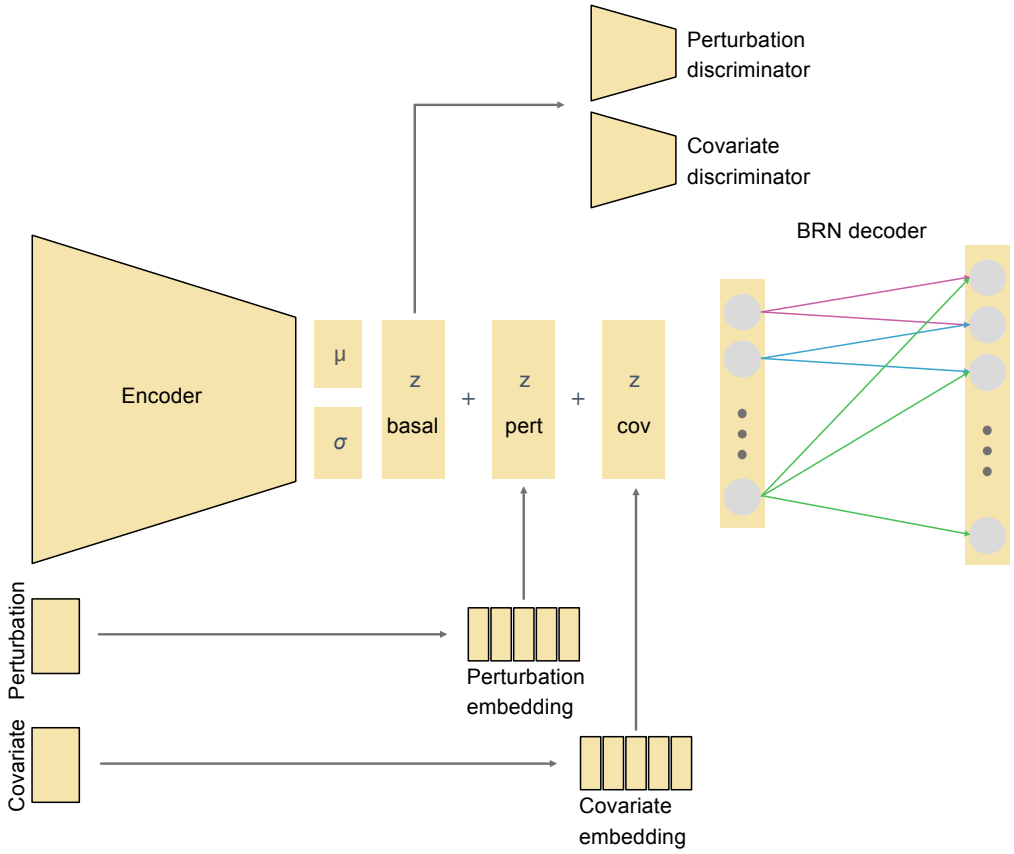

Figure 4.2: Overview of the single-layer COBRA model. As **Figure 4.1**, but interpretable decoder part consists of a single-layer BRN (bottom).

Unlike CPA, the goal of COBRA is more a qualitative description in terms of pathway activities than a quantitative prediction of gene expression levels. Additionally, it should be noted that so far COBRA only implements auxiliary classifiers for categorical covariates and does not model dose-response curves as in CPA. In brief, for each categorical covariate that we want to disentangle, an auxiliary classifier is attached to the learnt basal state in the latent space. The aim here is that the classifier is not able to distinguish between the different classes, hence leading to a mixing of the samples in the latent space. As in CPA, we call this the z basal embedding or z basal view. Note that the term 'view' is used to denote the contemplation of a certain combination of covariates. The covariate embeddings are then learnt separately using torch.nn.Embedding layers.

The advantage of using Embedding layers instead of one-hot encoding is that they also capture similarities between the samples (De Donno et al. 2023). The sum of the different latent space embeddings (z total) is then fed into the decoder during model training so that the original data can be reconstructed. During analysis, latent space embeddings from different views can be fed into the decoder to obtain pathway activities that only depend on a certain covariate.

4.2 Model implementation

COBRA is implemented in pytorch and will be made freely available through pip and github at `https://github.com/hdsu-bioquant/onto-vae/cobra-ai` under the GPLv3 license. The code for OntoVAE was also moved to this repository. Thus, there are two major contributions: first, the code for OntoVAE has been restructured and made more user-friendly and flexible. Second, the new COBRA model has been implemented that allows for additional disentanglement of covariates. We envision cobra-ai to become a toolbox with a set of models that share the interpretable decoder part. In the following, I will illustrate in more detail the key points of the implementation.

4.2.1 Reimplementation of OntoVAE

The code base for OntoVAE underwent some major and minor changes to make the tool more flexible, more user-friendly and more adequate to cope with larger and larger single-cell datasets. The major changes include:

- The `Ontobj` became more light-weight. It still stores the information about the ontology, including annotation, graph, and masks, but not the datasets since this was taking up too much memory for large datasets. The information is now stored as a json-file, so that the user needs to create a new instance of the `Ontobj`, and then feed a previously processed object with the `load()` function.
- As already stated, the datasets are not stored in the `Ontobj` anymore. The model is now interfacing with the `AnnData` class, pythons standard container class for

single-cell data, which stores the counts alongside with annotation information for samples and features.

The code below demonstrates how to initialize an OntoVAE model with the new implementation:

```
# import packages
import scanpy as sc
from cobra_ai.module.ontobj import *
from cobra_ai.module.utils import *
from cobra_ai.model.onto_vae import *

# load ontobj
ontobj = Ontobj()
ontobj.load('GO.ontobj')

# load anndata
adata = sc.read_h5ad('Kang_PBMC.h5ad')

# prepare anndata
adata = setup_anndata_ontovae(adata,
                              ontobj)

# create model
model = OntoVAE(adata)
```

- OntoVAE now also includes the possibility to use a one-layer BRN, for example if one wants to model connections between TFs and target genes. If the user wants to construct such an `Ontobj`, they just have to leave out the `obo` parameter in the initialization function and only provide a file with mappings from genes to gene sets.

```
ontobj = Ontobj()
ontobj.initialize_dag(gene_annot='/path/to/gene_term_mapping.txt')
```

Minor changes include:

- Similar to scVI, the user can provide batch information through an additional one-hot encoded neuron by specifying which column in adata.obs corresponds to the batch and passing it to the setup command.

```
# prepare anndata
adata = setup_anndata_ontovae(adata,
                              ontobj,
                              batch='batch')
```

- The user can now choose from many different parameters to have full control over the model. He can start the ontology in the latent space, or decide to move it to the first layer of the decoder. Furthermore, he can now use activation functions in the decoder, and really customize the model. Detailed explanations of the model creation parameters are given in **Table B.1**.

- Saving of model and train parameters together with the best model in the same folder. A trained model is now loaded as follows:

```
model = OntoVAE.load(adata, # data that was processed with setup function
                     modelpath) # path where the best model is stored
                                # together with the parameters
```

- The implementation of early stopping through setting parameters in the `train_model` function. Detailed explanations of the model training parameters are given in **Table B.3**.

- More things are happening under the hood, such as the default choice of trimming thresholds and moving the model to the device. Thus, it becomes even easier for the user to execute the code.

4.2.2 Implementation of COBRA

COBRA inherits from the OntoVAE class, so that many of the same principles apply. The major difference is that the user now specifies the covariates that should be disentangled in the latent space. Thus, the anndata needs to be prepared in a slightly different way, specifying the `cobra_keys`. The below code chunk demonstrates how to initialize a COBRA model.

```
# import packages
import scanpy as sc
from cobra_ai.module.ontobj import *
from cobra_ai.module.utils import *
from cobra_ai.model.cobra import *

# load ontobj
ontobj = Ontobj()
ontobj.load('GO.ontobj')

# load anndata
adata = sc.read_h5ad('Kang_PBMC.h5ad')

# prepare anndata
adata = setup_anndata_ontovae(adata,
                              ontobj,
                              cobra_keys = ['condition', 'celltype'])

# create model
model = COBRA(adata)
```

There is now an additional set of parameters the user can tune that define the structure of the auxiliary covariate classifiers. An overview is given in **Table B.2**. The main

difference in the implementation of COBRA compared to OntoVAE is the training loop. Due to the adversarial approach that COBRA was extended with, training is a two-step process. Again, see **section 2.3.3** for a more detailed explanation. In brief, first, samples of the minibatch are passed through the VAE and through the classifiers. The VAE loss is now integrated with the adversarial loss of the auxiliary classifiers, and backpropagation leads to a simultaneous update of the parameters of Encoder, Decoder, and covariate embedding layers. Second, the basal state of samples is passed through the auxiliary classifiers, and the adversarial loss is computed and then weighted by the gradient penalty, which is now commonly used during training of GANs (Gulrajani et al. 2017). Backpropagation then optimizes the parameters of the auxiliary classifiers. The new loss components that are introduced by this adversarial approach all have weights that can be tuned by the user. An overview of tunable training parameters that are new in COBRA is given in **Table B.4**.

Another novelty in COBRA is the way that model outputs are organized. The user can still run the main functions `to_latent()`, `get_pathway_activities()`, `get_reconstructed_values()`, and `perturbation()` to retrieve the latent space embedding, the pathway or TF activities, the reconstructed values, or any of the three after an *in silico* perturbation of the gene expression input values. While the output of OntoVAE is a matrix, COBRAs output is organized as a dictionary, with the different views as keys, and the corresponding matrices as values. This allows the user to compare the model output between different views, and thus, pinpoint covariate specific effects.

4.3 Proof of concept: Mouse interferon dataset

Disclosure: The results that are showed in this section have been generated by Dr. Carlos Ramirez under my supervision.

In principle, due to the flexible implementation of COBRA, any kind of BRN can be used in the interpretable part, and the user can decide whether the latent space already forms part of the BRN or whether the BRN is only accommodated in the decoder. However,

in contrast to the original OntoVAE model, the adversarial approach in COBRA makes model training less stable, thus we opted for a simpler structure for initial experiments, with the prospect of expanding again to a multi-layer network at some point. All analyses that were done with COBRA in this thesis use a model with the following structure: A non-linear encoder, a latent space that is a linear combination of the different views, a one-layer decoder followed by a ReLU activation function, still using three neurons per term. The terms we used for the different analyses were either TFs or Reactome pathways.

To showcase the utility of COBRA, we used a scRNA-seq dataset of mouse embryonic stem cells (ESCs) and mouse embryonic fibroblasts (MEFs) that had been treated with IFNβ for 0, 1, or 6 hours (Muckenhuber et al. 2023). We initialized COBRA with a one-layer decoder, and the collecTRI derived regulons with TF - target gene interactions as BRN (Müller-Dott et al. 2023). The model was trained with default parameters, passing the cell type and the stimulation time as COBRA covariates. Additionally, the VEGA model was trained on the same dataset for comparison (Seninge et al. 2021). UMAP was computed on the latent space embeddings of the different views of the COBRA model as well as on the VEGA model (**Figure 4.3**). For the basal view of COBRA, all cells are mixed, and no distinction can be made between cell type or stimulation time. For the cell type view, two clusters can be observed, each corresponding to one cell type. Within each cell type, the stimulation time is mixed. Analogously, for the stimulation time view, three clusters can be observed, with one cluster per treatment time, while the cell type is mixed within each cluster. In the total view, six individual clusters are formed, one cluster per cell type and per stimulation time. For VEGA, cells are clustering according to cell type and stimulation time, however, no real distinction can be made between 0h stimulation and the early stimulation time of 1h. This demonstrates that COBRA can better separate treatment effects, and isolate samples at an earlier timepoint.

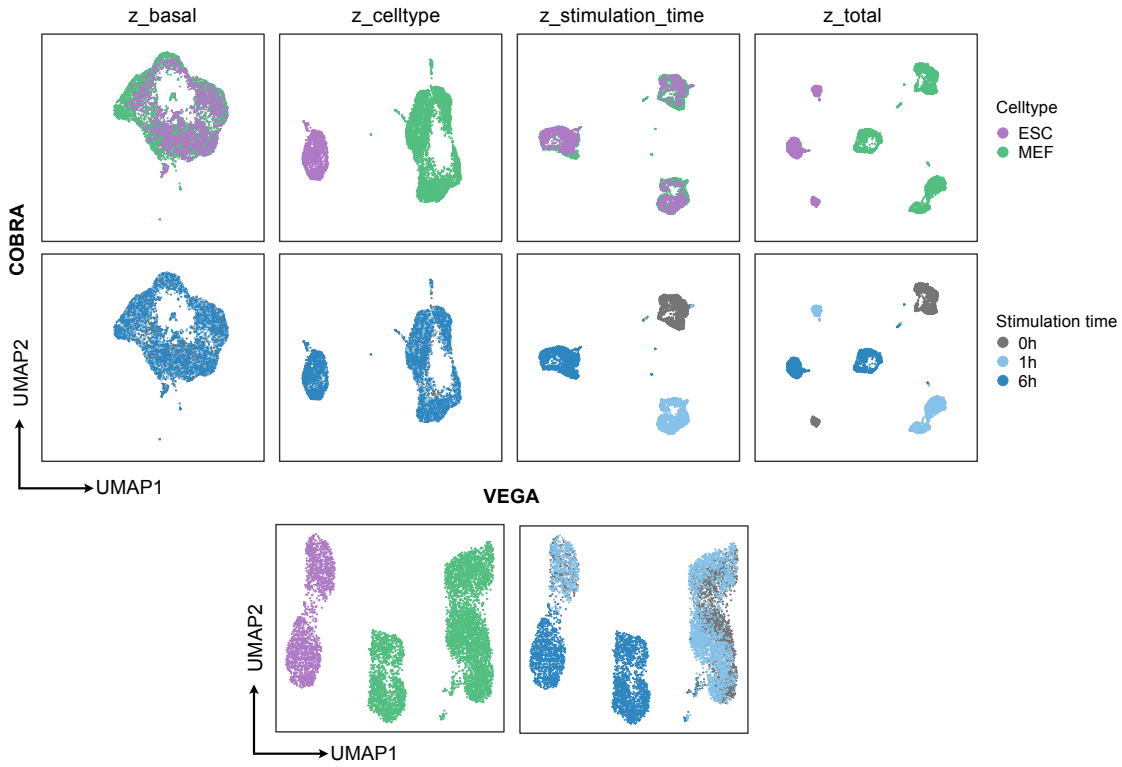

Figure 4.3: UMAP embedding of the latent spaces of different views. COBRA was run on the mouse interferon dataset using the cell type and stimulation time as covariates. The latent space embeddings of the different views are displayed as well as the embedding of the VEGA latent space (bottom). Each latent space representation is colored by cell type (left) and stimulation time (right).

We then set out to compare the TF activities in the different views of COBRA and VEGA. They were retrieved from COBRAs decoder and VEGAs latent space, respectively. We then unsupervisedly selected the top 50 TFs that had the highest variance in the stimulation time view of COBRA and plotted their activities, aggregated per cell type and stimulation time, in a heatmap (**Figure 4.4**). It immediately becomes obvious that for VEGA, the main clustering is by cell type, and within both cell types, the 0h and 1h samples cluster together, while for COBRA, in the cell type view, the samples

cluster by cell type, and in the stimulation time view, they cluster by treatment. With this unbiased way of preselecting TFs, many TFs were identified in this analysis that are known to play a role in interferon response, such as IRF7, IRF2, and IRF1, IRF3, IRF9, IRF4, IRF5, STAT1, and STAT2 (highlighted in **Figure 4.4**), that become especially active at the 6h stimulation timepoint. Both, COBRA and VEGA, are able to capture important TFs that become active upon IFN stimulation. However, COBRA is able to capture effects at an earlier timepoint since the model can separate the 0h and 1h treated samples from each other. The TFs IRF7, ETV7, and NANOG (highlighted in **Figure 4.4**) were selected to illustrate this in more detail (**Figure 4.5**). These three TFs display a high activity in stem cells as becomes apparent from COBRAs cell type view, as well as from VEGA. However, COBRAs stimulation time view additionally uncovers interesting dynamics for these TFs upon IFN treatment. For IRF7, the dynamics of both COBRA and VEGA agree, but COBRA additionally shows an effect in ESCs after 1h IFN stimulation. For ETV7, COBRA reveals that the activity first increases upon treatment (1h timepoint) and then drastically drops (6h timepoint). This connection between ETV7 and interferon response is in line with a recent work where the authors demonstrated that ETV7 acts as a repressor of a set of ISGs in breast cancer stem cells, and this could be partially reverted after treating with IFNβ (Pezzè et al. 2021). VEGA does not capture these effects as well. For Nanog, VEGA does not show any dynamic behaviour after IFN stimulation, however, COBRA reveals that the activity of Nanog is increasing as a consequence of treatment. Nanog is well known as one of the Yamanaka factors and plays an important role in the self-renewal and pluripotency of ESCs (Liu et al. 2008; W. Zhang et al. 2016). Interestingly, a recent study also demonstrated that type I IFNs significantly upregulate the Yamanaka factors and thus promote stemness in cancer (Musella et al. 2022). We speculate that VEGA is missing out on this due to the strong cell type specific effect which overshadows the responsiveness of Nanog to IFN treatment.

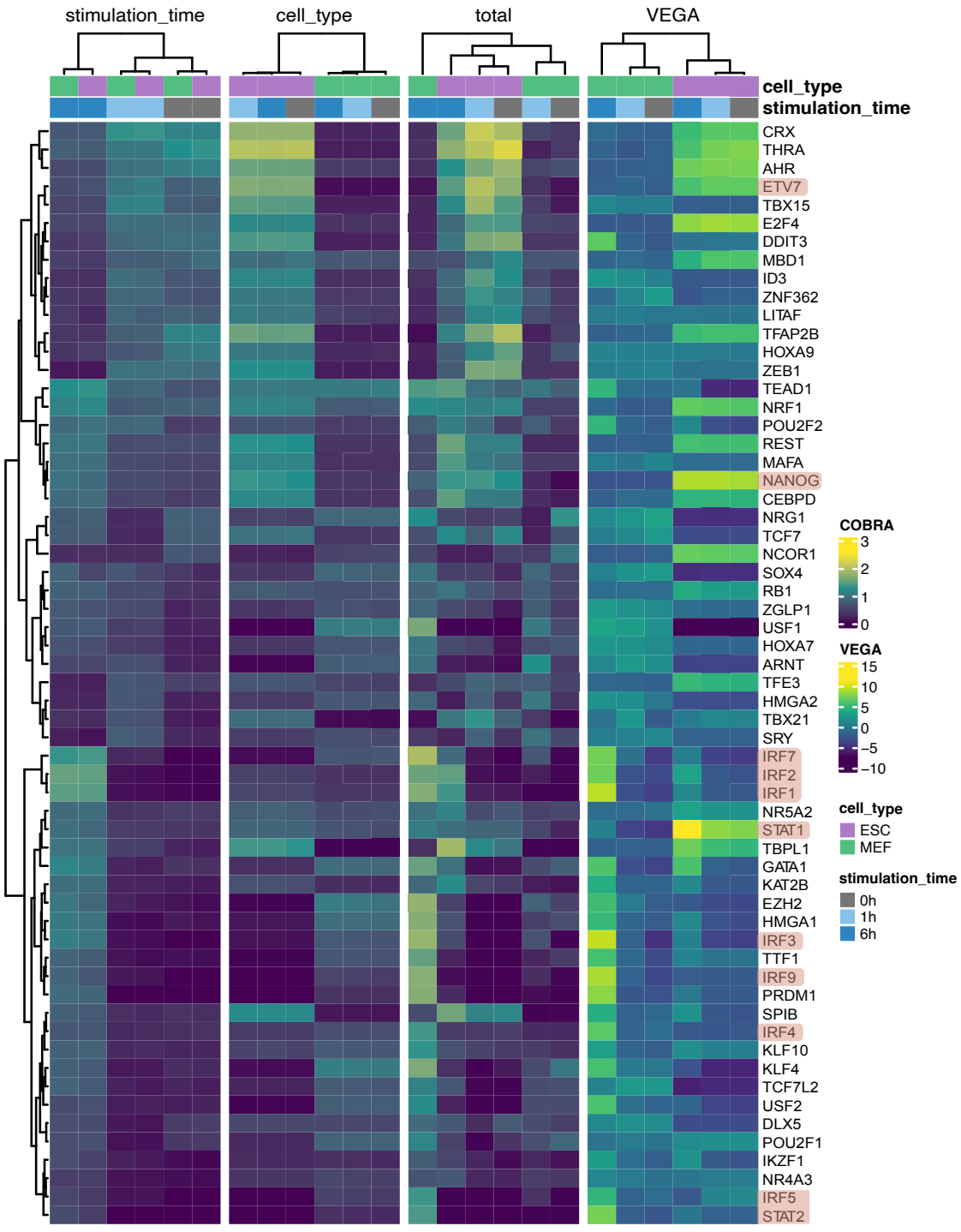

Figure 4.4: Heatmap displaying the top 50 variable TFs for the different views of COBRA and VEGA. Variance for TF activities was calculated on the stimulation time view of COBRA. Values for the single cells were aggregated per cell type and stimulation time. TFs highlighted in red are mentioned in the text.

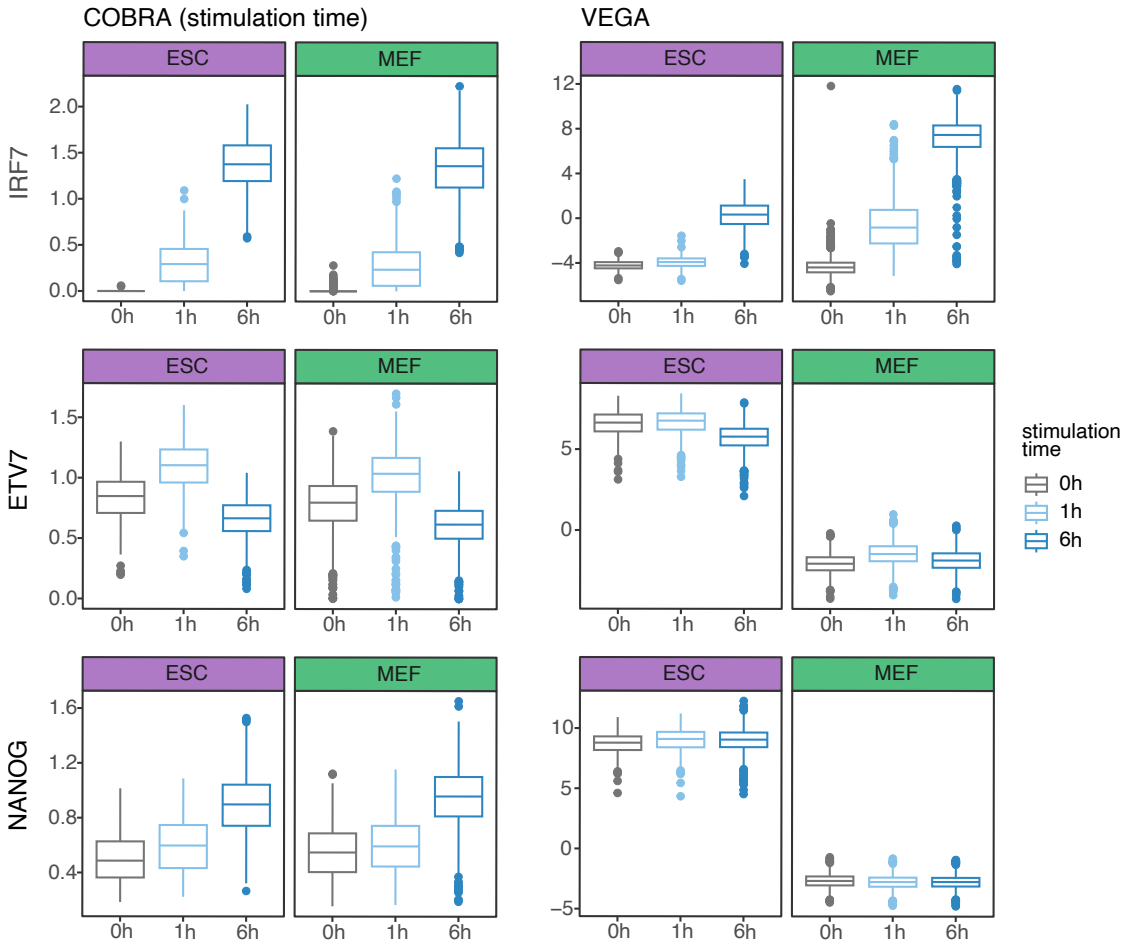

Figure 4.5: Comparison of COBRA stimulation time view with VEGA. Boxplots are displaying the TF activities of IRF7 (top), ETV7 (middle), and NANOG (bottom), for both VEGA (left) and the stimulation time view of COBRA (right).

Finally, we also trained COBRA on the dataset using the Reactome pathways as a prior. We then extracted the pathway activities from the stimulation view and the celltype view and performed a Random Forest analysis to determine which pathways contribute most to the variance observed in either view. Interestingly, we could observe an exclusive behavior, where majority of pathways either had a high feature importance in one or the other (**Figure 4.6**. For the stimulation time view, most important pathways were related

with viral mechanisms and interferon response, such as 'RIG1 MDA5 mediated induction of IFN alpha beta pathways', 'TRAF6 mediated IRF7 activation', and 'Regulation of IFNA signaling'. For the celltype view, most important pathways were related with the distinction between more and less differentiated and thus proliferating cells, such as 'Telomere maintenance', 'G0 and early G1', and 'G2M checkpoints'. Thus, COBRA can capture meaningful information using different biological priors.

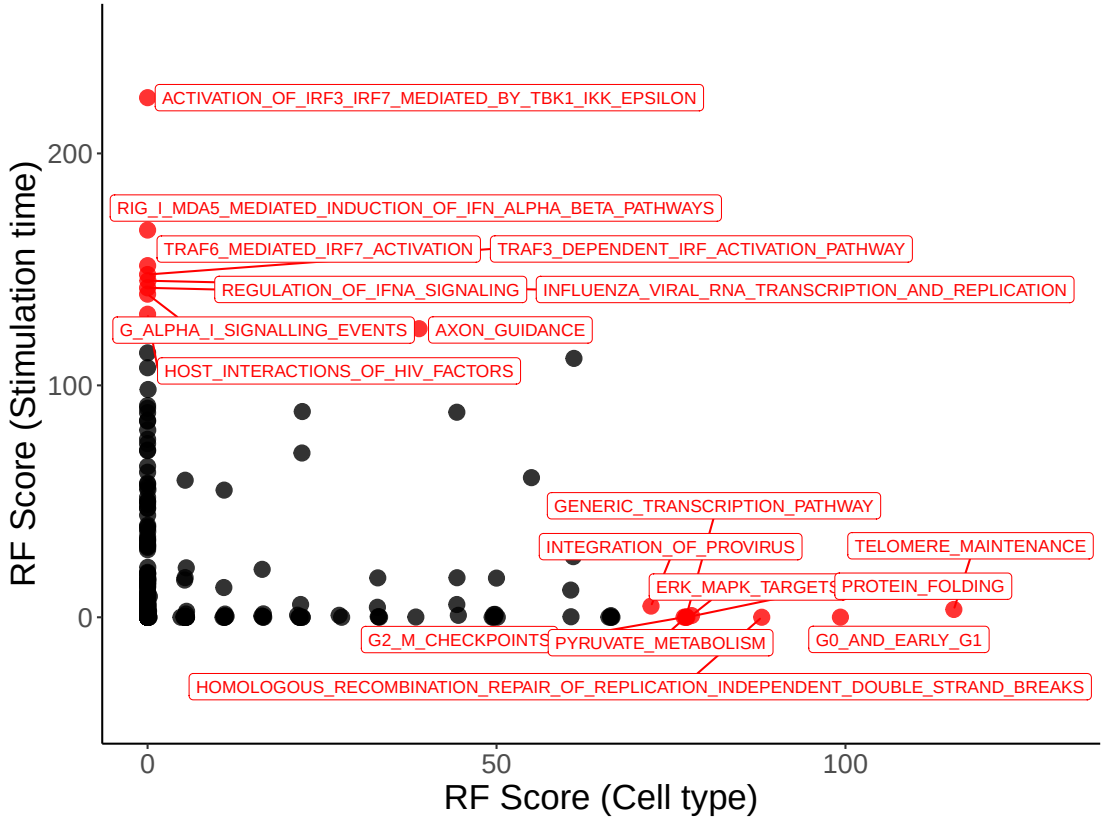

Figure 4.6: Comparison of Reactome pathway ranking between stimulation time and celltype view. Random Forest classification was performed for the different stimulation times and the different celltypes, using the pathway activities from the stimulation time view and from the celltype view, respectively, as predictors. Scatter plot shows feature importances of Reactome pathways for both views. Top pathways for either view are labelled in red.

4.4 Chapter summary

We developed COBRA, which extends OntoVAE with an adversarial approach that enables the disentanglement of covariates in the latent space, and subsequently in the interpretable part of the model. The code was further improved to make it more user-friendly and give the user more control over the model. Both models are now hosted in the same package that will be made available after publication: `https://github.com/hdsu-bioquant/cobra-ai`. We demonstrated the use of COBRA on a mouse interferon dataset using TF regulons and Reactome pathways as a biological prior. COBRA was able to identify TFs that are important in IFN signaling, and also revealed TF dynamics that are strongly overshadowed by the cell type confounding effect in comparable methods. COBRA also identified a distinct set of Reactome pathways for the stimulation time and for the celltype view.

Part II

Web Application Development

Chapter 5

OntoVAE Model Explorer: Dash Web Application to publish results

With an ever growing amount of research works, community efforts have been made to make research in general more transparent and more reproducible, one of them the FAIR guidelines, which stand for Findability, Accessibility, Interoperability, and Reuse of digital assets (Wilkinson et al. 2016). In line with this, to make our research more accessible and more transparent, we developed and deployed a web application, OntoVAE Model Explorer, that allows the user to browse all results from our publication (Doncevic and Herrmann 2023) that were obtained when training OntoVAE on GTEx data, which we used as a proof of concept. In our paper, we only showed selected pathways and tissue specific GO networks.

5.1 Web app usage

OntoVAE Model Explorer was developed with Dash and hosted on pythonanywhere, so that it can be accessed anywhere at any time under: `http://ontovaemodelexplorer.pythonanywhere.com`. The user can explore the pathway activities as computed by OntoVAE for all the GO terms by selecting a term from the drop-down menu (**Figure 5.1**). Scatter plots were generated with plotly, which allows the user to select or deselect certain tissues and to zoom in and out of the plot. Furthermore, the user can also explore the GO networks for all the tissues by selecting them from the drop-down menu (**Figure 5.2**). The networks were generated with python igraph and dash cytoscape. By hovering over the nodes in a network, the user can obtain the GO term and its cluster membership. The source code of the web application has been published on github and can be found under: `https://github.com/daria-dc/OntoVAE-Model-Explorer`.

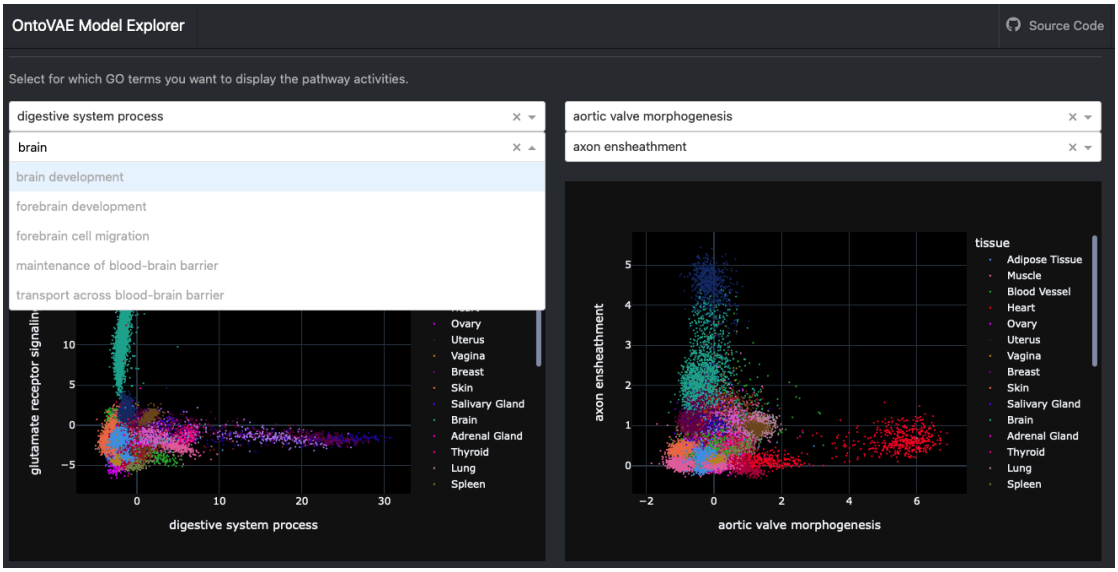

Figure 5.1: Web application allows browsing of pathway activities. The user can display four different GO terms at the same time, and check their pathway activities as computed by OntoVAE in the different tissues from the GTEx cohort. In the drop-down menu, the user can simply search for a pathway of interest, e.g. all pathways containing the sub-string 'brain'.

5.2 Chapter summary

We have demonstrated the abilities of OntoVAE on Gene Ontology (GO) and bulk tissue RNA-seq samples from GTEx, using a pathway centric approach where we look at specific pathways and which tissues they are most active in, and a tissue centric approach where we construct GO networks for each tissue, highlighting the most important processes. All of these results can be browsed under `http://ontovaemodelexplorer.pythonanywhere.com`, our Dash based web application that has been deployed on pythonanywhere, the source code of which is available at: `https://github.com/daria-dc/OntoVAE-Model-Explorer`.

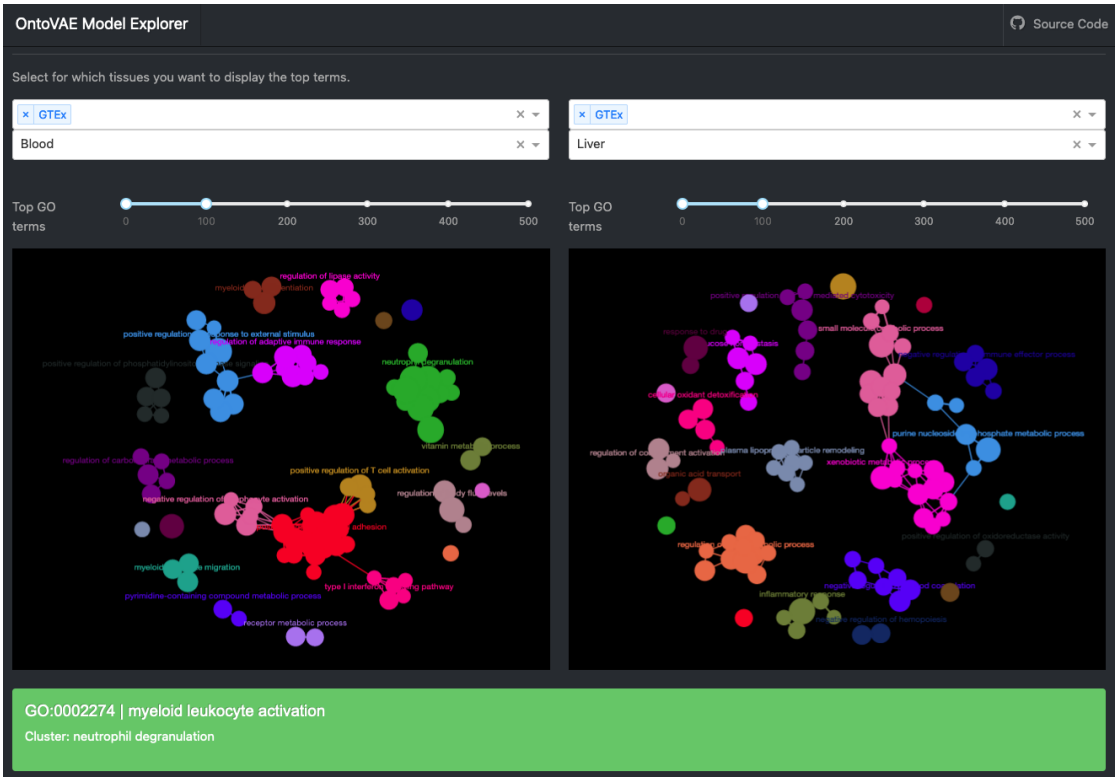

Figure 5.2: Web application allows browsing of GO networks. The user can look at two different networks at the same time, the tissue can be selected in the drop-down menu. In the graph, only the cluster representatives are labelled, but by hovering over the nodes, the user can obtain their information as it will show up in the green box below. The user can also change the size of the network by determining in the slider how many nodes/terms should be included.

Chapter 6

Interferon scRNAseq: Shiny Web Application to facilitate collaboration

Disclosure: The work presented in this chapter has been carried out in a collaboration with the group of Steeve Boulant and is still unpublished.

6.1 Workflow

In order to investigate IFN response in the gut, our collaborators grew intestinal organoids from Ileum and Colon, and treated them with IFNβ, IFNλ, or both, followed by scRNA-seq. See **Figure 6.1** for an experimental overview. For each condition, there were two replicates, resulting in 16 samples in total.

The data was pseudoaligned with kallisto and quantified with bustools (Melsted et al. 2021). Quality control and merging of replicates was performed with a standard preprocessing workflow using the Seurat package. The following filtering criteria were applied to remove low quality cells: minimum number of counts = 5000, maximum number of

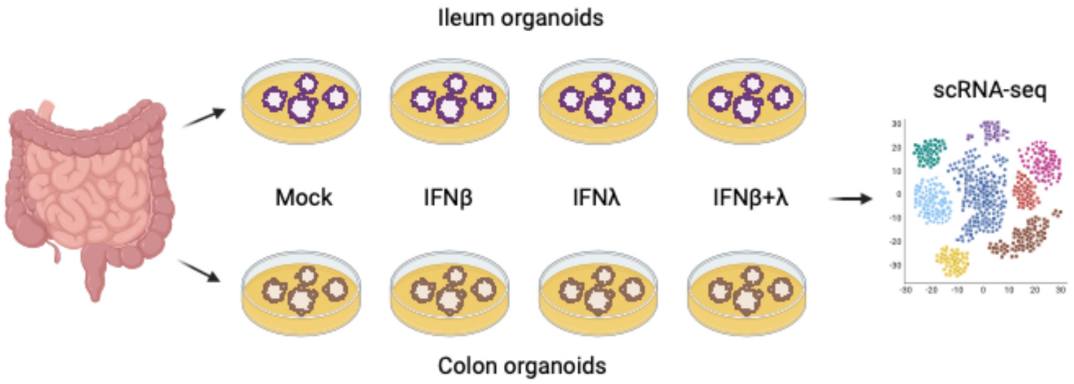

Figure 6.1: Schematic drawing of experimental workflow. Organoids were grown from ileum and colon, and treated with IFNβ, IFNλ, or a combination of both. The different samples then underwent scRNA-seq. Figure was created with biorender.com.

counts = 100,000, minimum number of expressed features = 500, maximum percentage of mitochondrial genes = 20. For the rest of the analyses presented here, Ileum and Colon were processed and treated separately, but the data from the different treatments was integrated using a standard integration workflow from Seurat. Cell type annotation was performed using label transfer based on a previously annotated dataset (Triana et al. 2021). Cell types identified in Ileum were Stem cells, Transit Amplifying (TA) cells, Cycling TAs, Type I Enterocytes, Type I and II Inmature Enterocytes, Enteroendocrine cells, and Goblet cells. Cell types identified in Colon were Stem cells, Transit Amplifying (TA) cells, Cycling TAs, Secretory TAs, Type I and II Enterocytes, and Type I and II Inmature Enterocytes.

Data was then further analyzed with standard analysis workflows, such as marker gene discovery to identify ISGs, followed by enrichment analysis against different databases such as GO and KEGG. Additionally, TF activities were computed using SCENIC, a tool that constructs the regulons in a data-driven way using random forest regression and then calculates TF activities per cell using this regulon information (Aibar et al. 2017). SCENIC also provides a possibility to binarize the TF activities, and simply

classify a TF as active or not active in a cell.

6.2 Web App

One difficulty of the organoid IFN dataset lies in the many possible comparisons that can be made. Since there are two tissues, with four conditions each, and 7-8 cell types per tissue, a vast amount of plots is produced during any kind of analysis, making it difficult to keep track of everything, to discuss results, and to isolate interesting findings. To mitigate this and facilitate our collaboration, I developed an interactive Shiny web application that allows for better organization and query of the results. So far, the web app is organized into four different tabs.

The first tab shows an overview of the data (screenshot in **Figure 6.2**) Hereby, the user can choose between Ileum and Colon. The tab is organized into two main panels: the top panel displays a UMAP visualization of the data, with the top row colored by cell type per default, and the bottom row colored by some quantitative variable, for example the number of counts, the number of features, or the mitochondrial percentage. The bottom panel then displays more clearly the quantification of this quantitative variable by boxplots or violin plots. The user could also restrict the view to certain cell types. For the bottom panel, he can also choose to look at the expression of a particular gene (screenshot in **Figure 6.3**).

The second tab includes all results from the find-markers analysis. All results that are organized in this tab are already pre-computed and just dynamically accessed through the web app. Mainly, differential gene expression analyses were performed between all treated and untreated conditions, separately for each cell type and each tissue, leading to a vast amount of gene marker lists. All of these lists were then also subjected to enrichment analysis with KEGG terms and GO pathways. The user can display either a heatmap showing the differentially expressed genes (screenshot in **Figure 6.4**), or the enrichment results for these genes (screenshot in **Figure 6.5**). In the command panel in the left, the user can select again at which tissue and cell type they want to look, and

whether they are interested in up- or downregulated genes.

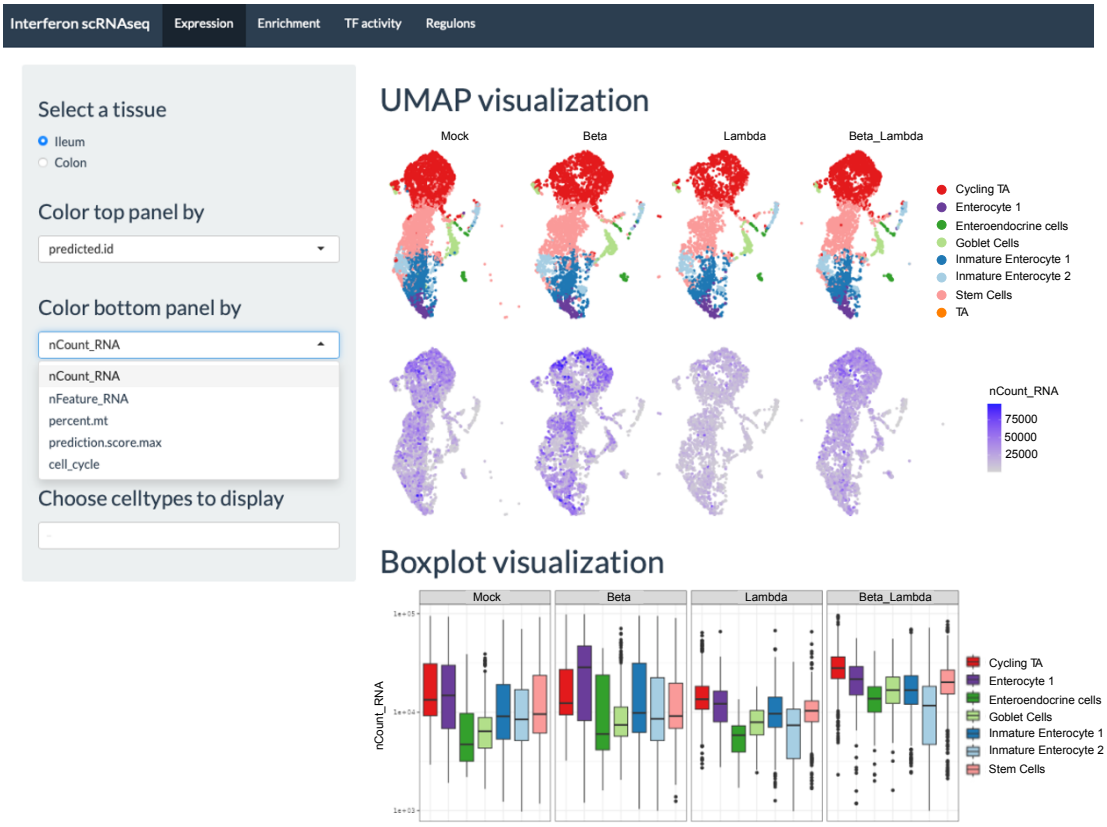

Figure 6.2: Expression tab of interferon web app. User can select between the available tissues. The top panel shows the UMAP visualization, colored by cell type (top) or by a quantitative variable (bottom). The bottom panel additionally display a quantification of the bottom variable.

Both, the third and the fourth tab, include the results from the SCENIC analysis. The third tab summarizes the TF activities. The top part of the tab allows the user to select any TF and have its activities displayed for the selected tissue (screenshot in **Figure 6.6**). The activities are compared between the four conditions in each cell type. The intensity of the color corresponds to the percentage of cells in that group that the TF is active in. The bottom part of the tab is a heatmap showing TFs that have high activity

in one group compared to the others, and thus helps the user in identifying important TFs (not shown). The user can perform this TF selection looking only at the cell type, or also considering the different treatments. He can also do this on a subset of the cell types.

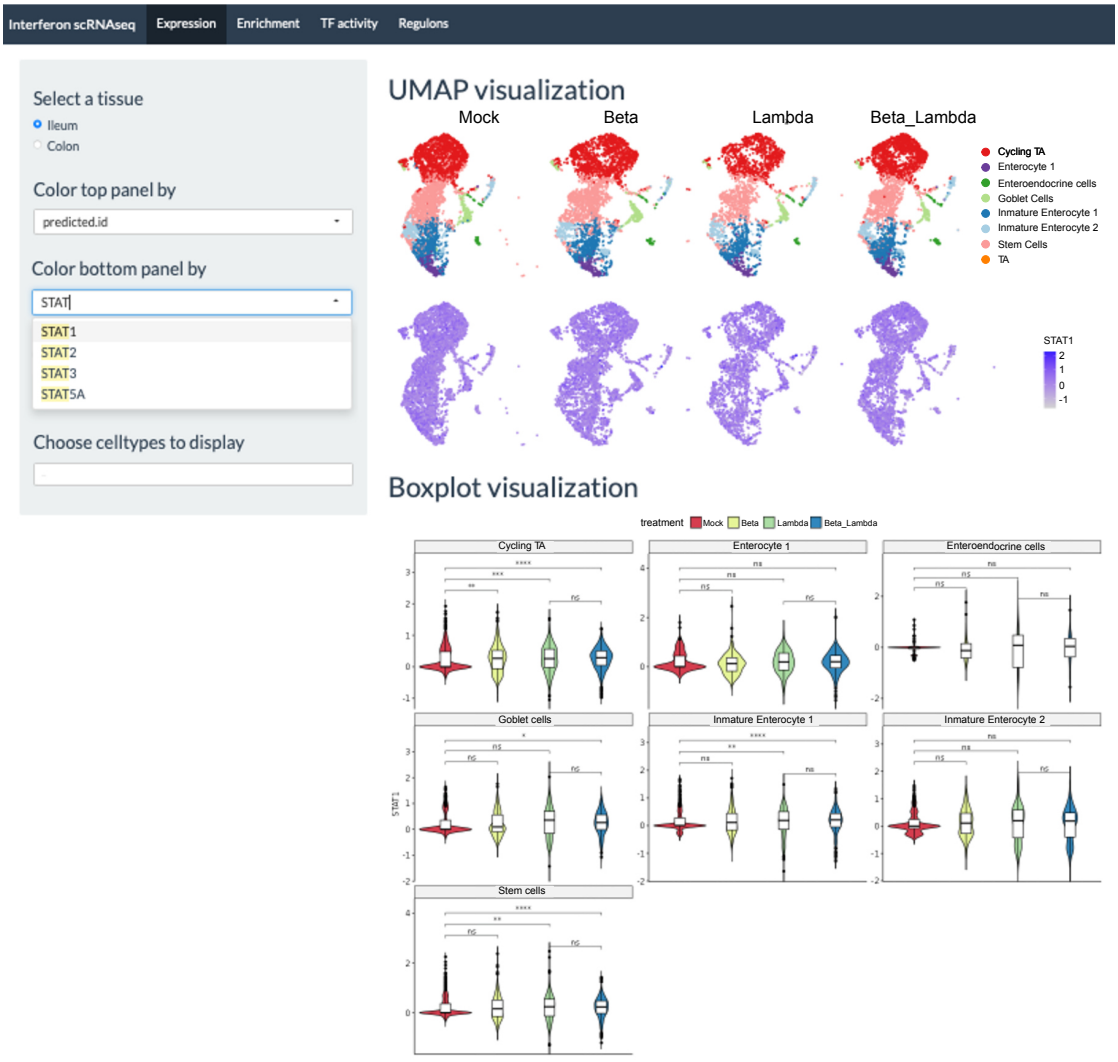

Figure 6.3: Gene expression in interferon web app. User can color the bottom panel with the expression of any gene from the dataset. The bottom panel then shows the quantification in violin plots per celltype and treatment.

Finally, the fourth tab intersects the SCENIC computed TF regulons with the results from the differential gene expression analysis (screenshot in **Figure 6.7**). The user can again select a tissue and a cell type and then obtains the differential genes between the different conditions, this time displaying the numbers of their overlap with regulons of different TFs. A Fisher's exact test was calculated for each overlap to determine its significance.

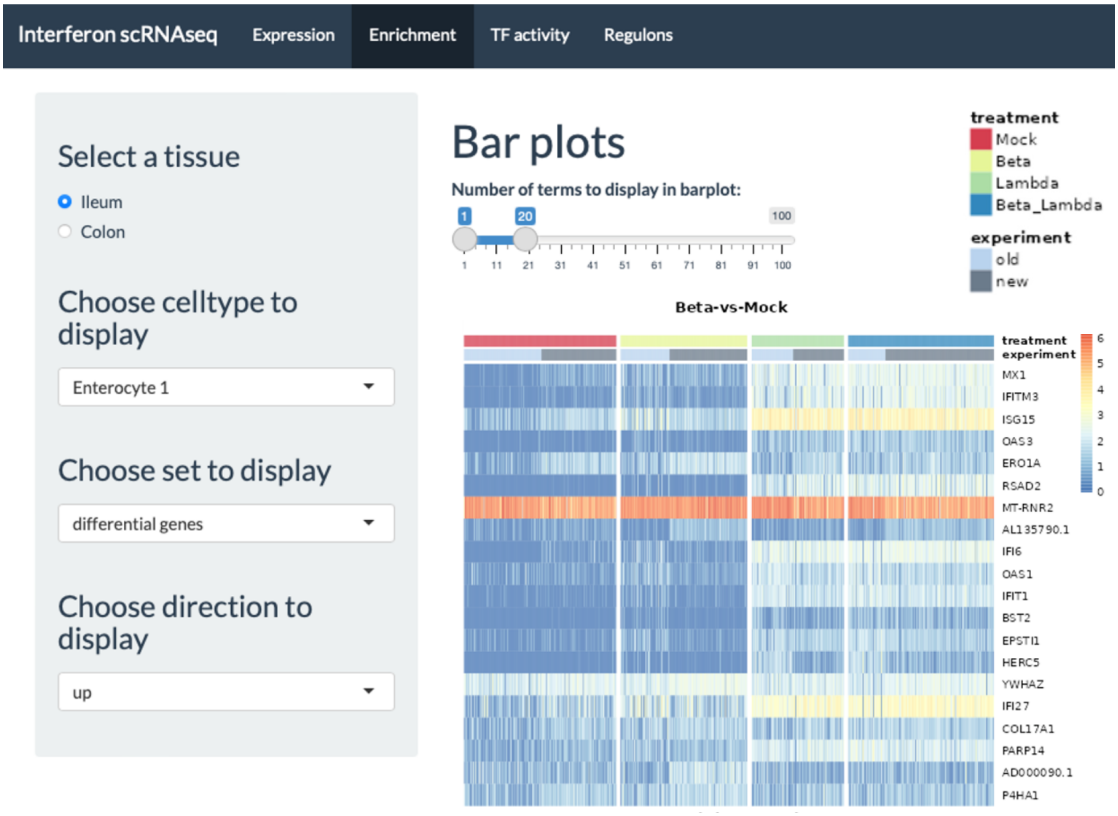

Figure 6.4: Web app displays heatmap of differentially expressed genes. User can select tissue and cell type, and if they want to look at up- or downregulated genes. For simplicity, only the heatmap for the comparison Beta vs Mock is shown, the app also displays the comparisons Lambda vs Mock, Beta_Lambda vs Mock, and Beta_Lambda vs Lambda.

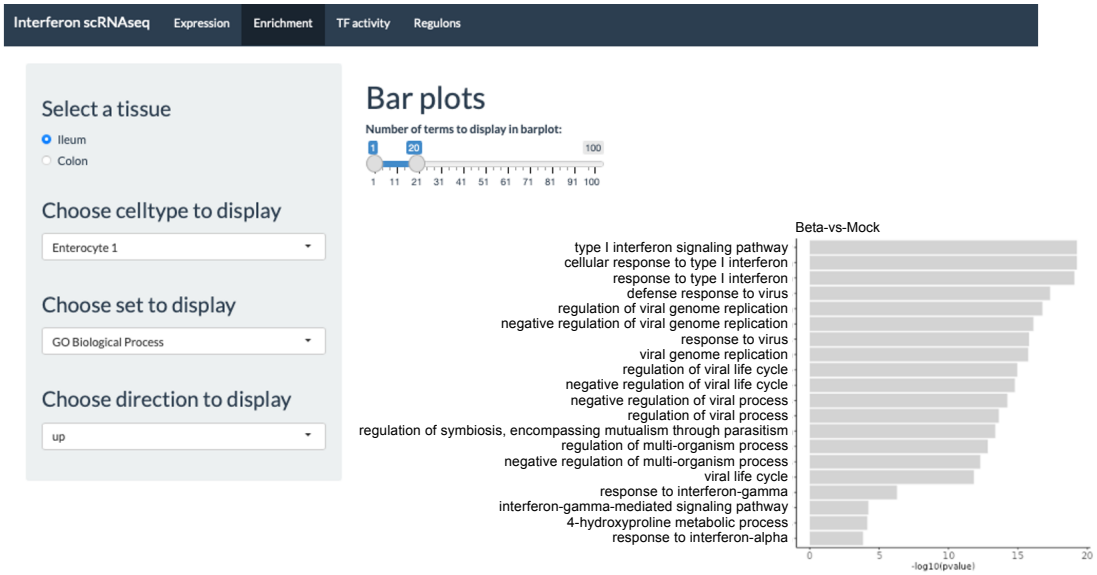

Figure 6.5: Web app displays enrichment results for differentially expressed genes. User can select tissue and cell type, and if they want to look at up- or downregulated genes. For simplicity, only the barplot for the comparison Beta vs Mock is shown, the app also displays the comparisons Lambda vs Mock, Beta_Lambda vs Mock, and Beta_Lambda vs Lambda.

6.3 Results

The web app allowed us to systematically browse through the results and pinpoint interesting findings. First, IFNβ and IFNΛ show a very distinct response in many ways, with the response of the double treatment resembling more the response of IFNΛ. This can be observed, for example, from the activity of the TF STAT1, which is known to be a core TF in type I and type III IFN response (**Figure 6.6**). The TF shows a high activation in Lambda and the double treatment but not so much in Beta. Another trend is observed in this analysis. The less differentiated cell types such as stem cells and cycling TAs seem to be less responsive to IFN treatment.

Another interesting finding consisted in the fact that, for some cell types, IFNβ seemed to induce a cell type specific response rather than an antiviral response. This is illustrated

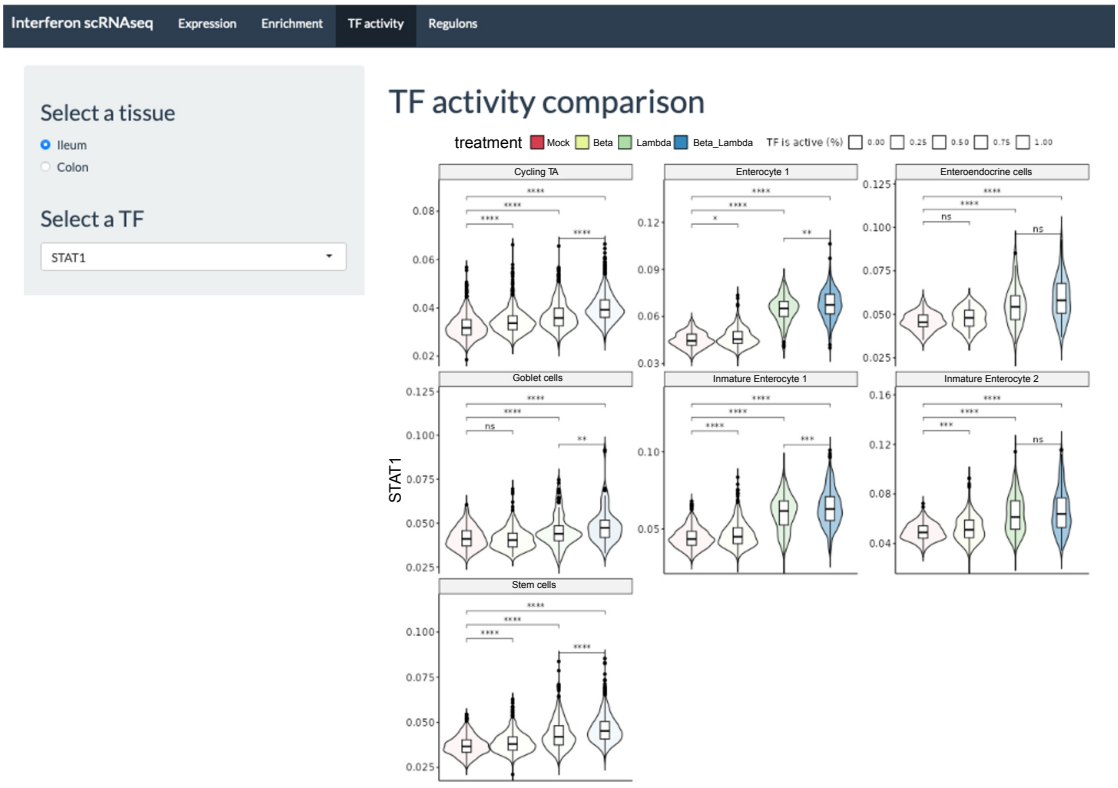

Figure 6.6: Web app displays TF activity. Violin plots are split by cell type, and compare the activity of the selected TF between the four conditions. Intensity of the color corresponds to the percentage of cells that the TF is active in.

for the Enteroendocrine cells (EECs) of the Ileum. In order to showcase this finding, we systematically collected results from the web app (**Figure 6.8**). First, we looked at the genes that were upregulated in Beta compared with Mock (**Figure 6.8a**), and found a handful of genes (*SCG3*, *SNAP25*, *TAGLN3*, *KCNB2*, *RGS4*, and *CALY*), none of which seem to be related to the classical antiviral IFN response. Interestingly, of those genes, *SNAP25*, *TAGLN3*, *KCNB2*, and *RGS4* were expressed in the Beta condition only. Enrichment analysis for the differential genes yielded terms related with potassium ion transport, and neurotransmitter signaling mechanisms (**Figure 6.8b**).

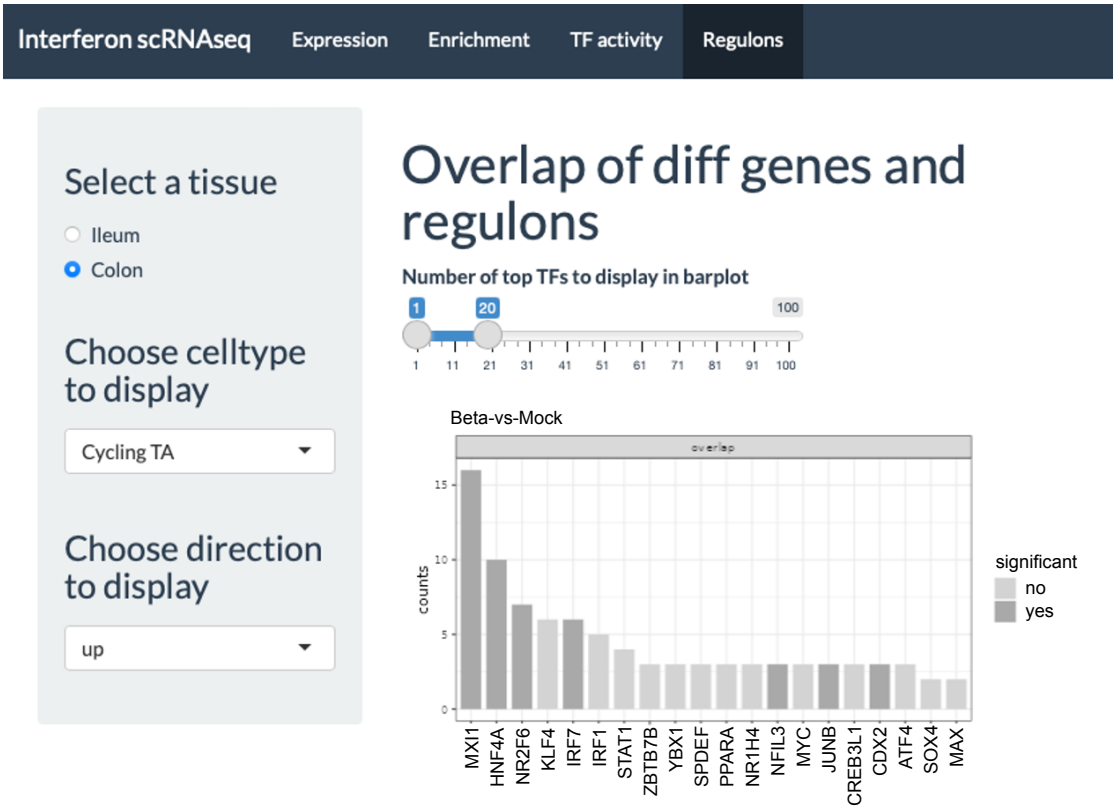

Figure 6.7: Web app displays intersection of regulons with differentially expressed genes. User can select tissue and cell type, and if they want to look at up- or downregulated genes. Bar plots then show the overlap of the gene list with the regulons of different TFs, sorted by overlap size and colored by significance. For simplicity, only the barplot for the comparison Beta vs Mock is shown, the app also displays the comparisons Lambda vs Mock, Beta_Lambda vs Mock, and Beta_Lambda vs Lambda.

EECs are hormone-secreting cells that functions as regulators of insulin secretion, intestinal motility, and food digestion (Worthington, Reimann, and Gribble 2018). Moreover, recent studies revealed that EECs establish a direct communication with peripheral neurons via synapses (Barton et al. 2023). This is in line with our findings, and also highlights a potential role for IFNβ in enhancing the gut-brain communication. We then

focused our analysis on identifying relevant TFs, and isolated RFX6, NEUROG3, and NEUROD1 from the SCENIC overlap analysis (**Figure 6.8c**). Further exploration of results showed that these TFs are indeed only active in EECs (**Figure 6.8d**), and become more strongly activated upon treatment with IFNβ, but not IFNλ (**Figure 6.8e**). One other interesting observation that can be made in this context is that IFNλ seems to counteract some of the effects of the IFNβ treatment. Since IFNλ treatment alone does not have any effect upon the selected TFs, but the double treatment does, this can solely be attributed to the effects of IFNβ. However, looking back at the differential genes, the double treatment behaves more like IFNλ. The potential synergistic mechanisms between the two treatments need to be further investigated.

6.4 Chapter summary

We set out to investigate the cell type specific IFN response in the gut. Thus, our collaborators grew organoids from the Ileum and the Colon, and collected scRNA-seq data after treatment with IFNβ, IFNλ, or both. One challenging aspect of the dataset consisted in the many possible comparisons that can be made, namely between Ileum and Colon, between different cell types within the same tissue, between the same cell type in different tissues, or between the different treatment conditions. To facilitate the organization of the results and the collaborative work, I developed a Shiny web application that was fed with the precomputed results of all analyses, such as marker gene discovery, enrichment analysis, and TF activity analysis. The web app allows interactive and fast exploration of the results and helped us in discovering some key findings: IFN response is indeed very cell type specific, and especially in some cell types, IFNβ seems to trigger different mechanisms than the classical antiviral ones. We highlighted this at the example of the EECs from Ileum.

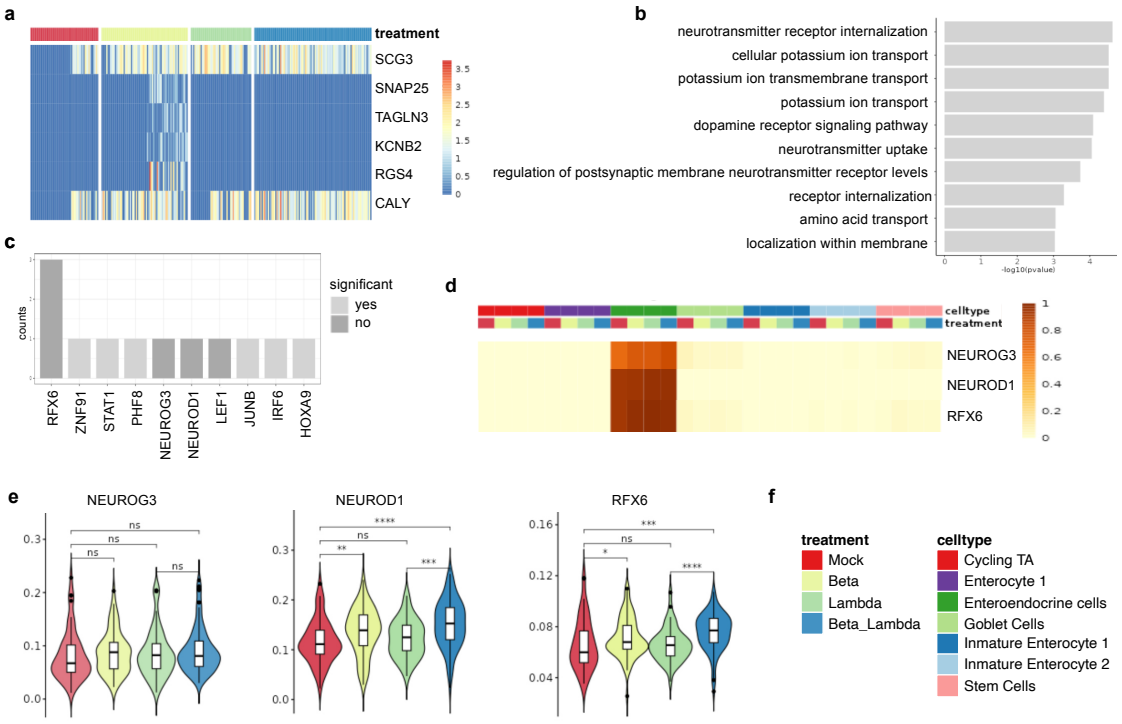

Figure 6.8: Enteroendocrine cells of Ileum respond to IFNβ in a cell type specific manner. **a** Heatmap of genes that are upregulated in Beta compared to Mock. **b** GO biological process enrichment of genes from **a**. **c** Overlap of genes from **a** with TF regulons from SCENIC. **d** Heatmap displaying the percentage of cells with the TF active. **e** TF activities of NEUROG3, NEUROD1, and RFX6. **f** Figure legend for treatment and celltype. All figures were collected from the web app. Except for **d**, all figures refer to enteroendocrine cells.

Part III

Perturbation modeling

Chapter 7

Predictive modeling with OntoVAE

Disclosure: The results that are presented in this section have been published in Doncevic and Herrmann (2023).

One important aspect in biomedical research is the investigation of how genetic perturbations change the behavior and transcriptional response of a cell. This can shed light on the functioning of the cell and on the downstream effects of different genes. Thus, gene perturbation experiments also have implications for drug design, since they can identify genes that are most likely to be successful in clinical trials (Nelson et al. 2015).

The state-of-the-art approach hereby is carrying out large CRISPR screens to assess the effects of perturbations of a wide range of genes. In pooled CRISPR screens, a variety of perturbations is induced into a pool of cells, and then they are exposed to different biological challenges such as viral infection or drug treatment. Subsequently, each perturbation can be quantified in the proliferated cells, thus allowing to evaluate if certain perturbations are more ore less favorable under certain conditions (Bock et al. 2022). However, this approach is very resource intensive since many genes can be tested, and the number of experiments increases with the number of cell lines, drugs,

and conditions to be tested. Additionally, since genes can also interact with each other in a synergistic manner, combinations of genes can also be tested, further scaling up the experiment (Roohani, Huang, and Leskovec 2022).

Thus, computational approaches have been designed that can predict perturbation response (Lotfollahi, Wolf, and Theis 2019; Roohani, Huang, and Leskovec 2024). We hypothesized that OntoVAE can also be applied in the context of predictive modeling. Hence, we developed a strategy where we manipulate the input values of genes prior to running samples through a trained model, and then compare the pathway activities before and after perturbation using paired Wilcoxon tests. Functions for performing gene perturbations have also been implemented in the onto-vae package and a use-case is demonstrated in the vignette. We distinguish between a gene-centric approach, where we perturb a single gene and assess the effects of this perturbation on all terms, and a term-centric approach, where we systematically perturb all genes one-by-one, and rank them based on their influence on a specific term. Both strategies will be discussed in more detail and with examples in the following sections. It should be noted that, in this chapter, we focus solely on the single-gene perturbation case, perturbing one gene at a time.

7.1 Gene-centric approach

To illustrate the gene-centric approach, we again used our OntoVAE model that had been trained with a GO-decoder on all bulk tissue RNA-seq samples from GTEx. As an example gene, we picked the *Duchenne muscular dystrophy* (*DMD*) gene, which encodes for the protein dystrophin. Dystrophin is located primarily in the muscle and functions as a linker that attaches the cytoskeleton to the extracellular matrix. Dystrophin is therefore crucial for proper muscle development and functioning. A mutation in the *DMD* gene leads to depletion of functional dystrophin protein and thus causes muscle weakness and muscle degradation. This disease phenotype is called 'Duchenne Muscular Dystrophy' (DMD) (Duan et al. 2021). We wanted to investigate whether OntoVAE can capture

the importance of *DMD* in muscle function, so we focused on the 881 muscle samples in the GTEx dataset, and performed an *in silico* knockout of the *DMD* gene in all muscle samples by setting the input value of *DMD* to zero prior to running them through the trained model. We then performed paired Wilcoxon tests comparing the muscle samples before and after the knockout at each node/term in latent space and decoder to find the GO terms that were most affected by the knockout (term-level analysis, **Figure 7.1** blue box). We also performed paired Wilcoxon tests for each gene in the reconstruction layer to identify the most affected genes and grouped them into GO terms using gene set overrepresentation analysis (gene-level analysis, **Figure 7.1** green box).

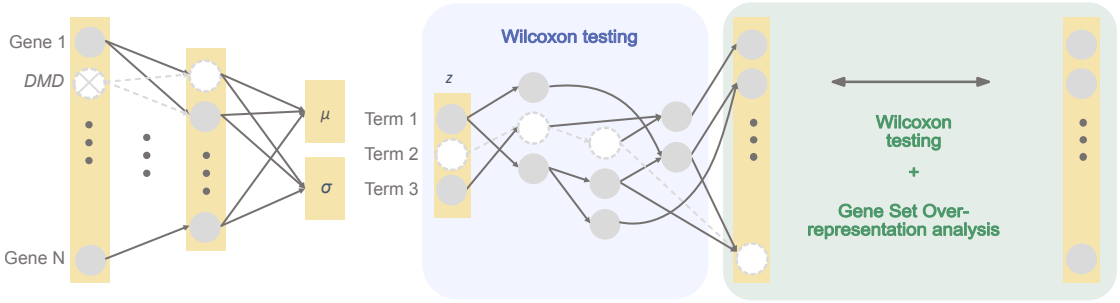

Figure 7.1: Schematic drawing of gene-centric gene perturbation approach. The input value of a gene of interest is perturbed, and then paired Wilcoxon tests pre- and post perturbation are performed at each node/term in latent space and decoder to detect the most significant terms (pathway-level analysis, blue box). Paired Wilcoxon tests can also be performed for each gene in the reconstruction layer to see which genes are affected. They can then be further grouped into terms using gene set overrepresentation analysis (gene-level analysis, green box).

From the term-level analysis, we identified many muscle process related terms, such as *muscle system process*, *positive regulation of cytoskeleton organization*, *muscle contraction*, and *positive regulation of actin filament bundle assembly* (**Figure 7.2a**). From the gene-level, we also obtained terms that are highly specific to muscle, such as *muscle sys-*

tem process, *muscle contraction*, *muscle organ development*, and *muscle filament sliding* (**Figure 7.2b**). Taken together, these results confirm that OntoVAE can predict the consequence of a gene knockout on a pathway level. In order to show that we did not only identify these muscle specific processes because we only considered muscle samples in our analysis, we also performed a synthetic knockout of two other genes: *SFSWAP*, which is a splicing factor, and *COX5A*, which is a member of the mitochondrial respiratory chain. From the term-level analysis (**Figure 7.3a**), we obtain terms related to RNA splicing and processing for *SFSWAP* and terms related to the respiratory electron transport chain and oxidative phosphorylation for *COX5A*. The same applies to the gene-level analysis (**Figure 7.3b**). Thus, we can conclude that the results produced by OntoVAE are not biased by the sample group.

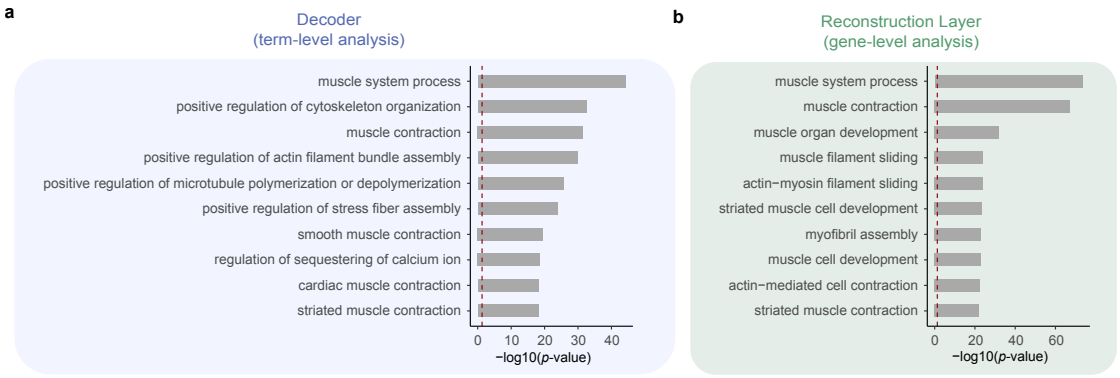

Figure 7.2: *DMD* knockout affects muscle specific processes. (**a**) Top ten GO terms obtained from decoder (term-level analysis). (**b**) Top ten GO terms obtained from reconstruction layer after grouping genes into GO terms (gene-level analysis).

7.2 Term-centric approach

To illustrate the term-centric approach, we employed two examples: the investigation of a disease-related perturbation by dissecting the outcome of Limb-girdle muscular dystrophy (LGMD), and the investigation of a treatment-related perturbation by predicting

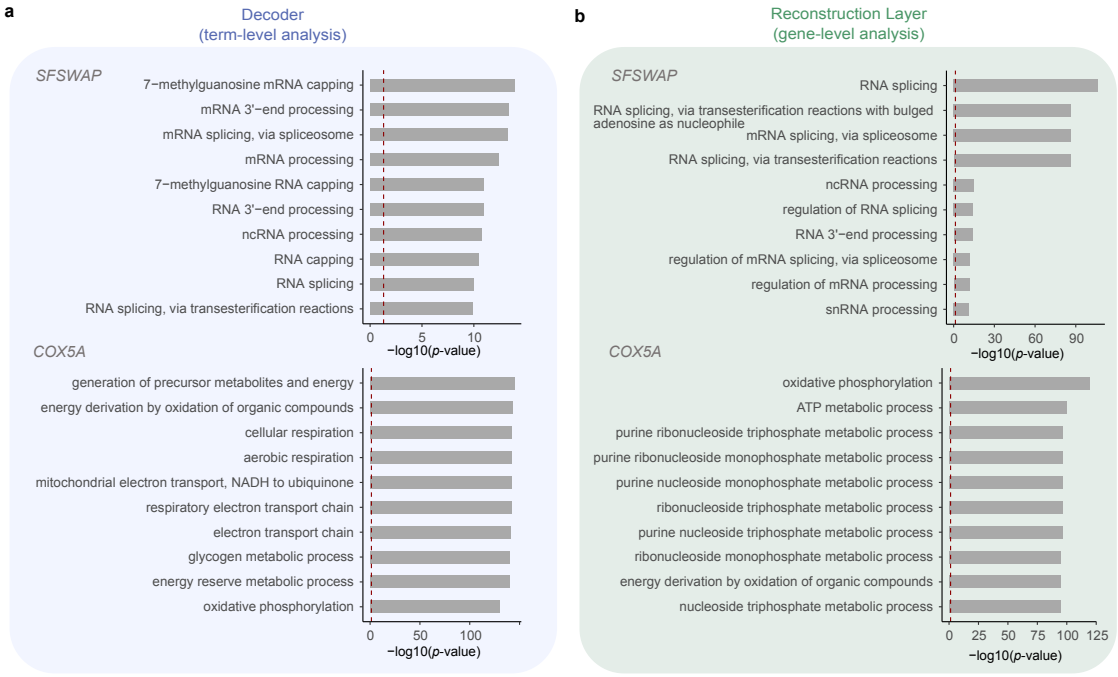

Figure 7.3: *SFSWAP* and *COX5A* knockout, respectively, affect gene function related terms. (**a**) Top ten GO terms obtained from decoder (term-level analysis) for *SFSWAP* (top) and *COX5A* (bottom). (**b**) Top ten GO terms obtained from reconstruction layer after grouping genes into GO terms (gene-level analysis) for *SFSWAP* (top) and *COX5A* (bottom).

interferon response. For both applications, the overall procedure was the same: a systematic one-by-one perturbation was performed for all genes $1..N$, followed by a paired Wilcoxon test pre- and post perturbation at the node of interest (LGMD and type I interferon signaling pathway, respectively), resulting in a ranked gene list where all genes were ranked according to their p-value (**Figure 7.4**). The two examples will be illustrated in more detail in the following sections.

7.2.1 Predicting the outcome of disease

To see if we can predict with OntoVAE how a disease context changes gene expression, we focused again on Muscular Dystrophy (MD), this time on Limb-girdle muscular dystrophy

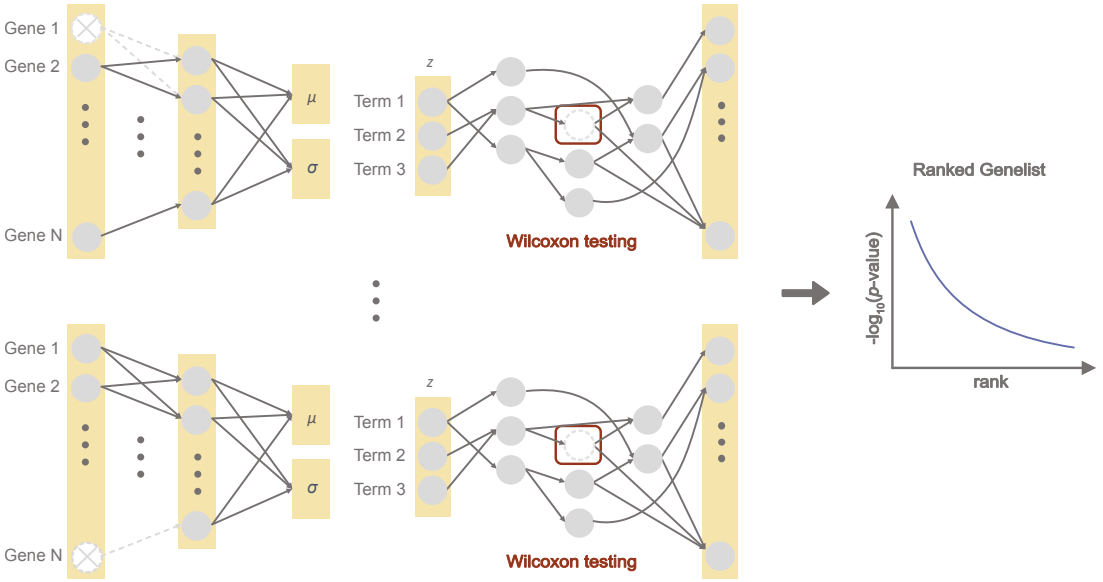

Figure 7.4: Schematic drawing of term-centric perturbation approach. Systematic one-by-one perturbation is performed for all genes $1..N$. Paired Wilcoxon test pre- and post perturbation is performed at the term of interest, and then all genes can be ranked according to their p-value.

(LGMD), which is a form of MD that primarily affects the muscles in arms and legs.

For this purpose, we incorporated the Human Phenotype Ontology (HPO) in the latent space and decoder of OntoVAE, as LGMD is a term in HPO. This time we used trimming thresholds of 10 and 1,000. We again trained the model on all bulk tissue RNA-seq samples from GTEx, and carried out the analysis on the 881 muscle samples. In all the samples, we performed a synthetic knockout for all genes that could be mapped to the ontology, 4,774 genes in total, by setting their input value to zero before passing the muscle samples through the trained model. We then specifically computed a paired Wilcoxon test for each knockout at the node corresponding to LGMD, and ranked all genes according to their p-value. As for HPO, many gene annotations originate from short nucleotide polymorphisms (SNPs) in that gene where an association to the disease had been demonstrated, the directionality of the relationship is not straightforward, meaning it is not always clear at the level of gene expression whether depletion of a gene will lead

to a higher or lower activation of the phenotype. Therefore, we looked at both directions and computed two paired Wilcoxon tests, one to identify genes significantly downregulating the LGMD node, and one to identify genes significantly upregulating the LGMD node. Thus, we also obtained two ranked genelists, predLGMD_dn and predLGMD_up, respectively. In order to validate our genelists, we downloaded an external dataset of LGMD. In this study, the authors have carried out bulk RNA-seq on muscle samples from 16 LGMD patients and 15 age-matched healthy individuals (Depuydt et al. 2022). We performed differential gene expression analysis (DGEA) between muscle samples of patients (n=42) and controls (n=33) using the R package *DESeq2* with a significance threshold of 0.1 for the adjusted p-value. Furthermore, we filtered for genes that could be mapped to HPO. Genes that were upregulated in patients were called LGMD_up, and genes downregulated in patients were called LGMD_dn. For validation of the ranked gene lists predLGMD_dn and predLGMD_up that resulted from the OntoVAE analysis approach, we performed a Geneset Enrichment Analysis (GSEA) in both ranked gene lists, using as gene sets the results from the DGEA, LGMD_up and LGMD_dn. For both, predLGMD_dn (**Figure 7.5a**) and predLGMD_up (**Figure 7.5b**), the gene set LGMD_dn displayed a significant enrichment, with p-values of 0.001 and 0.006, respectively. This confirms that OntoVAE can make meaningful predictions.

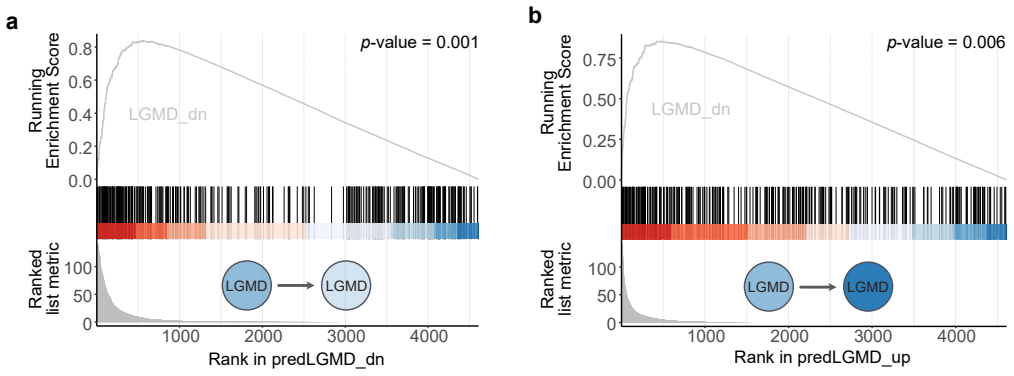

Figure 7.5: GSEA of LGMD DGEA gene sets in OntoVAE predicted ranked gene lists for LGMD. OntoVAE had been trained with HPO-decoder on GTEx samples. (**a**) GSEA in predLGMD_dn. (**b**) GSEA in predLGMD_up.

To closer investigate this, we fused the genes from the leading edges of both GSEA analyses, and checked their overlap with LGMD_dn, as well as with genes directly annotated to the LGMD node in HPO, finding an intersection of 147 and 10 genes, respectively (**Figure 7.6**).

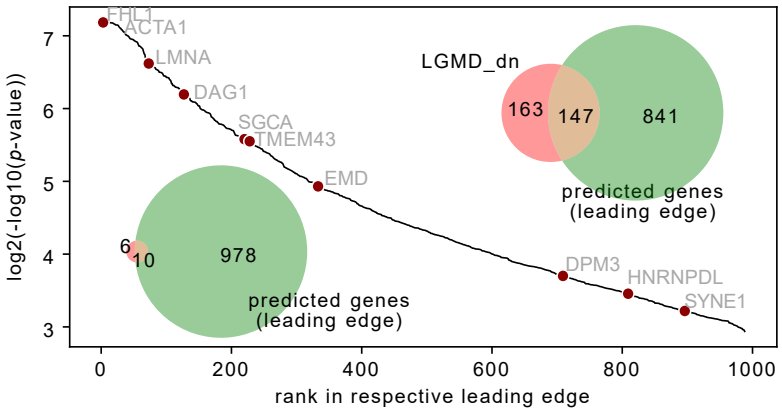

Figure 7.6: Overlap of GSEA results with LGMD_dn and LGMD HPO genes. Genes from both GSEA leading edges were fused (predicted genes; green circle) and their overlap with LGMD_dn (Venn diagram top right) and genes annotated to the LGMD node in HPO (Venn diagram bottom left) is shown. The hockey stick plot shows where the LGMD HPO genes fall in the ranked gene list.

We then set out to examine whether the detection of the 10 HPO genes was a direct product of their annotation to the LGMD term in HPO. For this purpose, we trained 10 new models, whereby in each model, we removed the link of one of the genes to the HPO LGMD term. We then reran the same analysis as before for the 10 new models, performing the systematic knockout followed by paired Wilcoxon testing to determine the new rank of the gene after its annotation had been removed. Of these 10 genes, seven had been found in the leading edge of genes downregulating LGMD activity. After link removal, they were still located in the leading edge (**Figure 7.7a**). Of the remaining three genes that had been found in the leading edge of genes upregulating LGMD activity,

one was still located in the leading edge (**Figure 7.7b**). Taken together, these results demonstrate that OntoVAE is capable of tolerating missing prior information to a certain degree, and can discover relationships that go beyond the available annotation.

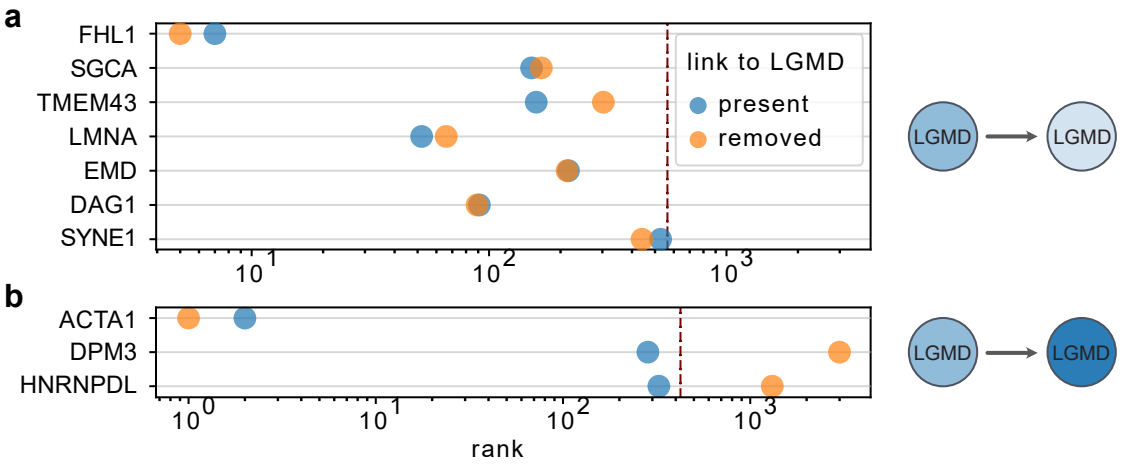

Figure 7.7: Rank of the 10 predicted genes that are also annotated to LGMD before and after link removal. The red dotted line represents the leading edge cutoff. (**a**) Genes that were found in the leading edge of genes downregulating LGMD activity. (**b**) Genes that were found in the leading edge of genes upregulating LGMD activity.

7.2.2 Predicting the outcome of treatment

As another application example, we wanted to study if OntoVAE could predict gene expression changes upon treatment. We chose the interferon (IFN) response for this, as it is a well studied process. The IFN response is triggered by viral infection, and is characterized by the release of a special type of cytokines, the IFNs, which induce a signal cascade that ultimately culminates in the transcription of interferon-stimulated genes (ISGs) (Schoggins 2019).There are three main classes of IFNs, type I, type II, and type III, which act on different receptors and induce different signaling cascades. In our application example, we are focusing on the type I IFNs.

To investigate if we could predict the type I IFN response, we again used Gene Ontology

(GO), where we focused on the term 'type I interferon signaling pathway'. This time, we chose a single-cell (sc) RNA-seq dataset of peripheral blood mononuclear cells (PBMC), where the authors had sequenced cells from Lupus patients that had been treated with IFN-β, a type I IFN, or control (Kang et al. 2018). UMAP of this dataset shows that the cells cluster by cell type (NK, Dendritic, CD4T, B, FCGR3A+Mono, CD14+Mono, and CD8T) and treatment (**Figure 7.8**). First, for each cell type separately, we performed Wilcoxon tests on the scRNA-seq data to identify genes that were upregulated in this cell type upon treatment, so that we could use these gene sets for validation of our analysis. We then trained the OntoVAE model only on the unstimulated cells, and then performed one-by-one *in silico* stimulation of all genes, followed by a paired Wilcoxon test at the node corresponding to 'type I interferon signaling pathway'. For this analysis, we restricted ourselves to CD4T cells as they represented the largest cell population in the dataset. As in the previous analysis, all genes were ranked by their p-value, and a GSEA was performed with CD4T_IFN-β_stim_up (genes upregulated in stimulated CD4T cells vs. control) to validate the ranked gene list (**Figure 7.9**). The enrichment of the validation gene set was significant, with a p-value of 0.001, confirming the ability of OntoVAE to predict treatment outcome without ever having seen the treated samples during training.

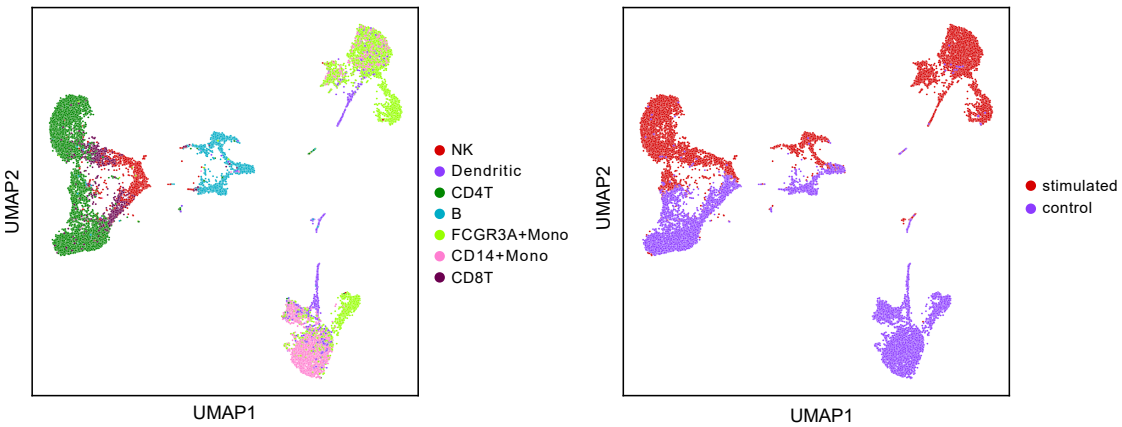

Figure 7.8: UMAP of the PBMC dataset. Cells are colored by cell type (left panel) or treatment (right panel).

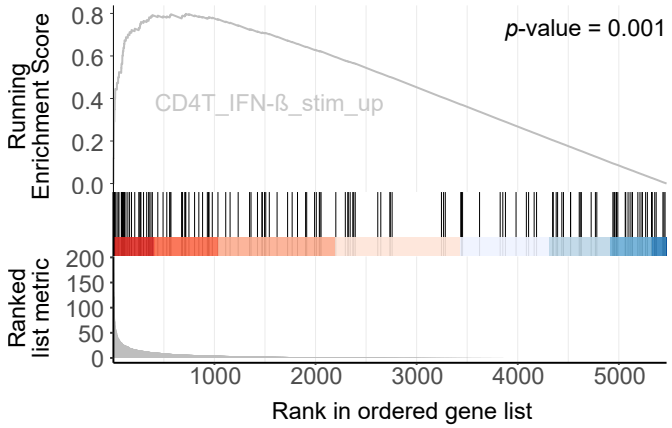

Figure 7.9: Validation of interferon treatment response prediction. GSEA was performed with CD4T_IFN-β_stim_up in the ranked gene list as predicted by OntoVAE.

Next, we wanted to see where our perturbed CD4T cells lie in the pathway activity space of OntoVAE compared to the control CD4T cells and the IFN stimulated CD4T cells (ground truth). For this, we extracted the pathway activities of all cells from the trained OntoVAE model, and then computed a UMAP on the pathway activities of control and ground truth CD4T cells. The pathway activities of the perturbed cells were projected onto that UMAP space. We can see that, when stimulating all 717 genes from the leading edge of the GSEA, perturbed cells are shifting from the control population to the ground truth population with increasing stimulation strength (**Figure 7.10a**), while they are remaining in place, when the bottom 717 genes are stimulated (**Figure 7.10b**).The stimulation was performed by adding values of 2, 4, 6, and 8 (**Figure 7.10** from left to right) to the non-zero cells for each gene. We also showed that a shift is induced when perturbing the top 5, 20, 50, or 100 genes of the leading edge (**Figure 7.10c,d,e,f** left panels), but more genes are needed when perturbing an equal number of random genes from the leading edge (**Figure 7.10c,d,e,f** right panels). Taken together, the genes predicted by OntoVAE indeed drive interferon response, and their ranking is meaningful as well.

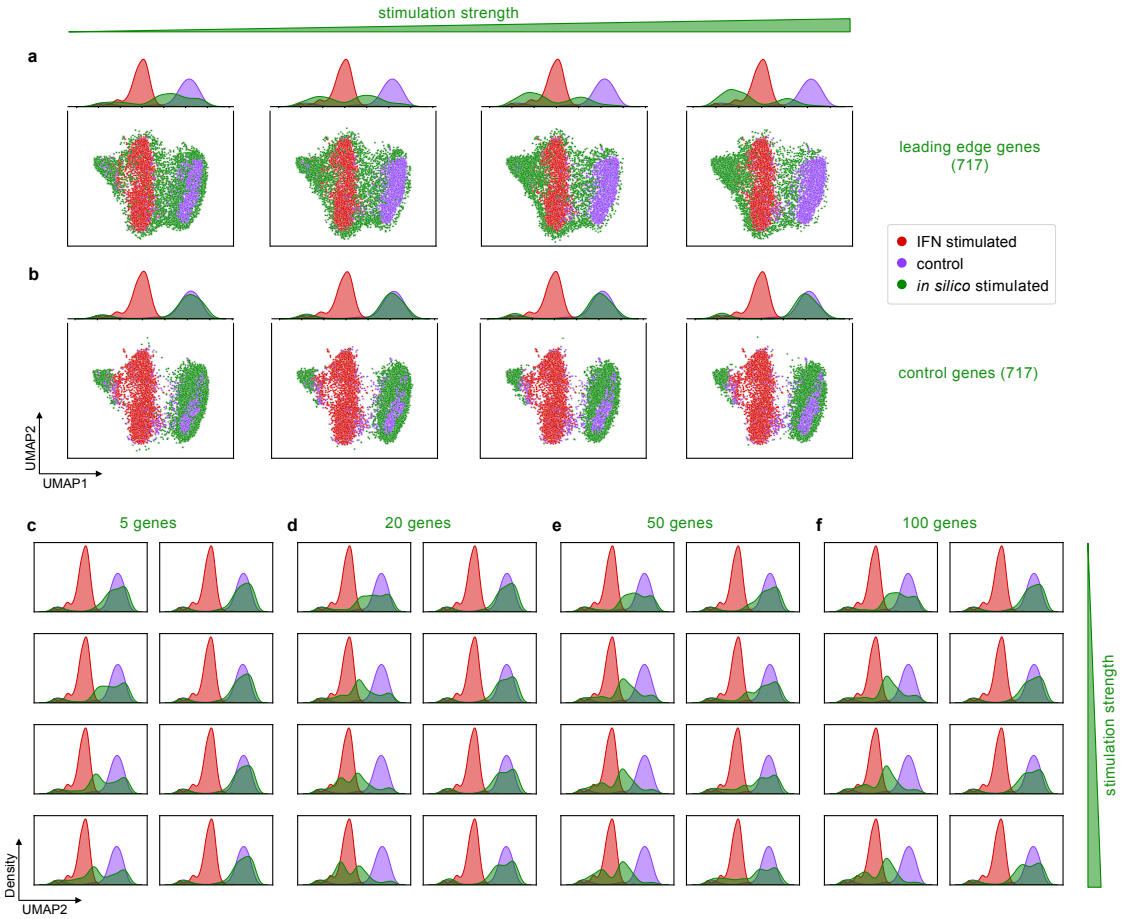

Figure 7.10: Perturbed cells approach ground truth cells in pathway activity space. UMAP was computed on OntoVAE pathway activities of control CD4T cells and IFN stimulated CD4T cells, and *in silico* stimulated CD4T cells were projected onto that UMAP space. (**a**) UMAP embedding of cells when stimulating the 717 genes from the leading edge of the GSEA. Density plots on top show the distribution of cells along UMAP2. The stimulation strength is increasing from left to right, with values of 2, 4, 6, and 8 for the stimulation. (**b**) The same as **a**, but for stimulation of the 717 bottom genes from the GSEA. (**c**,**d**,**e**,**f**) Density plots of cells over UMAP2 when 5, 20, 50, or 100 genes are stimulated, respectively. Left column always shows stimulation of top n genes, while right column shows stimulation of random n genes from the leading edge.

We then again had a closer look at the genes in the leading edge of the GSEA (**Figure 7.11**), our so-called 'predicted genes'. Their overlap with CD4T_IFN-β_stim_up amounts to 44 genes, which are additionally labelled in the hockey stick representation of the predicted genes in **Figure 7.11**. The color corresponds to their significance in CD4T_IFN-β_stim_up, a darker red indicating a higher significance. The small black triangles next to nine of the labels mean that these genes are either annotated directly to the GO term 'type I interferon signaling pathway' or to one of its descendant terms.

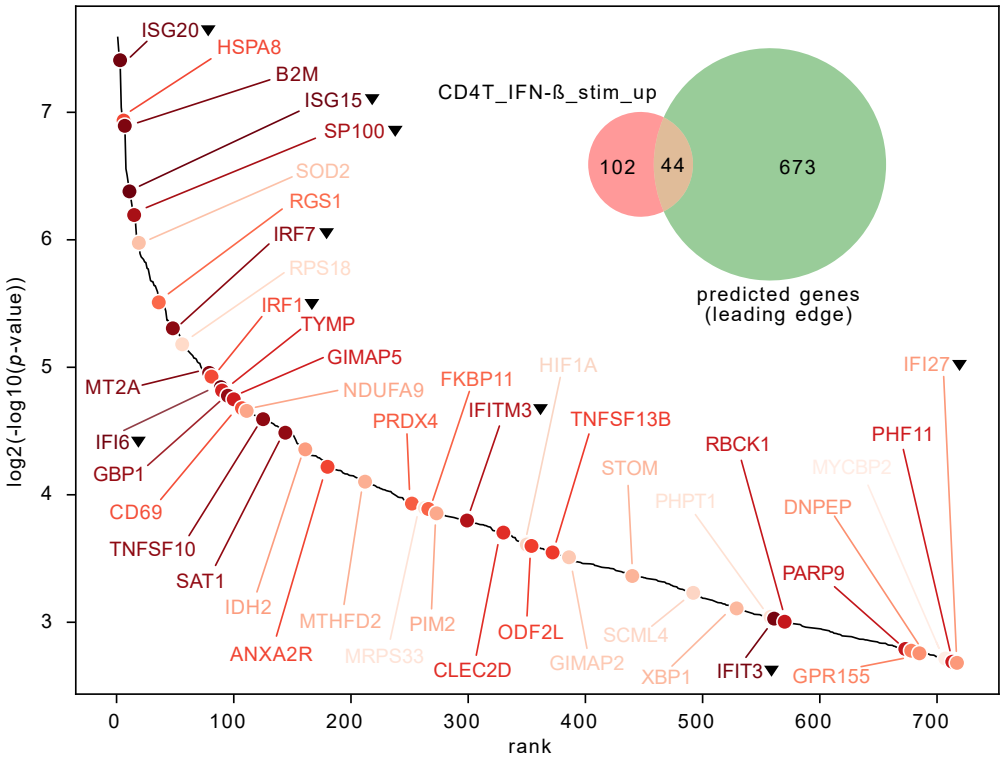

Figure 7.11: Overlap of predicted genes with actual differential genes. Predicted genes were extracted from the leading edge of the GSEA. Their overlap with CD4T_IFN-β_stim_up is shown in the Venn diagram. The 44 overlapping genes are labelled in the hockey stick plot, a darker red indicating a higher significance in CD4T_IFN-β_stim_up. Black triangles next to the labels indicate that a gene is annotated to the GO term 'type I interferon signaling pathway' or one of its descendant terms.

Thus, we can see that the majority of the 44 predicted genes is identified in the absence of direct annotation, highlighting the potential of OntoVAE to learn internal relationships in the data that go beyond the prior annotation. Next, we set out to further validate the remaining 673 predicted genes that were not overlapping with CD4T_IFN-β_stim_up. We first checked if there was any overlap with genes that were upregulated upon IFN stimulation in one of the other cell types from the PBMC dataset, and indeed found an especially high overlap with Dendritic cells (19 genes), FCGR3A+ Monocytes (33 genes), and CD14+ Monocytes (20 genes) (**Figure 7.12a**). We also performed a gene set overrepresentation analysis (ORA) with the C7 immunologic signature sets from MSigDB that are related to interferon, and found a significant enrichment of gene sets that were up in CD8T cells or a mature neuron cell line upon IFN treatment, and gene sets that were down in Dendritic cells after knockout of the IFN receptor (first, second, third and ninth set in **Figure 7.12b**). Taken together, our results confirm that OntoVAE can predict interferon response without having encountered treated cells in the training data set.

7.3 Comparison with other methods

Next, we wanted to compare OntoVAE with VEGA (Seninge et al. 2021) and expiMap (Lotfollahi, Rybakov, et al. 2023), two VAE methods that had been published previously and are similar to OntoVAE, as they both provide biological interpretability through their structure. In the VEGA model, a fully connected non-linear encoder is coupled to a sparse, one-layer linear decoder. The latent space can represent different biological entities, called gene module variables (GMV) by the authors, ranging from TFs to sets of gene pathways such as GO or Reactome. In principle, any kind of entity can be used that is annotated or linked to a set of genes. Thus, the decoder is sparse, since connections are only present e.g. between TFs and their target genes, or pathways and their annotated genes. In their manuscript, authors are demonstrating the use of VEGA to compute differential GMV activities between groups of samples, and how their model

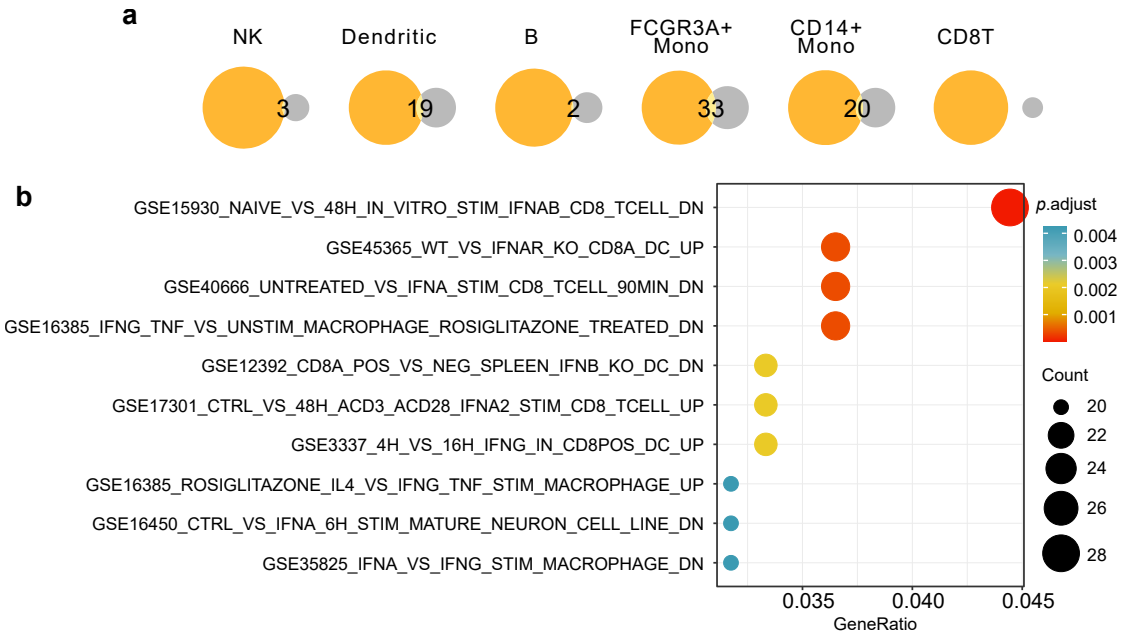

Figure 7.12: Follow-up analysis of predicted genes. (**a**) Overlap of 673 predicted genes that do not overlap with CD4T_IFN-β_stim_up (orange circles) with genes upregulated upon treatment in the other cell types of the PBMC dataset (grey circles). (**b**) Interferon related terms from MSigDB that are overrepresented in the 673 predicted genes.

can generalize on unseen training data. The expiMap model is similar to the VEGA model in that it also consists of a non-linear encoder, an interpretable latent space consisting of gene programs (GPs), and a one-layer linear decoder. In terms of applications, the authors of expiMap see it as an extension of their previous model scArches, a transfer learning strategy which allows for the mapping of query datasets onto large existing atlases (Lotfollahi et al. 2022). Through the interpretable latent space of expiMap, the mapping of query dataset onto reference atlas becomes more interpretable. Furthermore, the authors demonstrate the capacity of their tool to learn GPs *de novo* by adding additional nodes and allowing the model to learn the associated genes.

In order to compare these approaches to OntoVAE, we trained the VEGA and expiMap models on unstimulated PBMC data with Reactome pathways in their latent space.

For model training we followed the instructions given in `https://vega-documentation.readthedocs.io/en/latest/tutorials/vega_tutorial.html` (as of February 2023) for VEGA, and the instructions given in `https://scarches.readthedocs.io/en/latest/expimap_surgery_pipeline_basic.html` (as of February 2023) for expiMap. For better comparison, we also trained OntoVAE with Reactome pathways in the latent space, making it a single-layer model as well. To assess the quality of the produced latent spaces, we performed Leiden clustering and computed the adjusted rand index (ARI). The different methods yielded similar results, only expiMap performed slightly worse (**Figure 7.13a**). We assume that this is due to the higher KL loss weight in expiMap. Interestingly, the highest ARI value was obtained by OntoVAE coupled to a GO-decoder, although this model had a higher reconstruction error than the single-layer Reactome model (**Figure 7.13c**). For the VEGA and expiMap model, we then also performed the IFN response prediction analysis by stimulating all genes one-by-one followed by a paired Wilcoxon test at the node Reactome_Interferon_Alpha_Beta_Signaling for each gene, allowing us to create a ranked gene list. GSEA analysis with CD4T_IFN-β_stim_up reveals that VEGA and expiMap can predict the IFN response equally well as OntoVAE (**Figure 7.13b**, p-value 0.001).

7.4 Chapter summary

In this chapter, we showed how predictive modeling is a possible application of OntoVAE. We devised a gene-centric approach, where we perturbed one gene and analysed the influence of this perturbation on all the ontology terms. As an example, we performed an *in silico* knockout of *DMD*, a gene that is essential for muscular stability, and showed that this knockout affected muscle related terms. We also devised a term-centric approach, where we systematically perturbed all genes one-by-one, and analysed the outcome of the perturbation on a specific node, ranking the genes based on their influence on this node. One example that we used was the disease LGMD together with HPO as a prior, where we validated our findings in an external dataset. Another example was IFN response,

where we had a dataset with treated and untreated samples, trained the model only on the untreated samples, and used the treated ones for validation purposes. All in all, we demonstrated that OntoVAE is a useful tool, and might be applied in large *in silico* screens to preselect candidates that could be further validated experimentally.

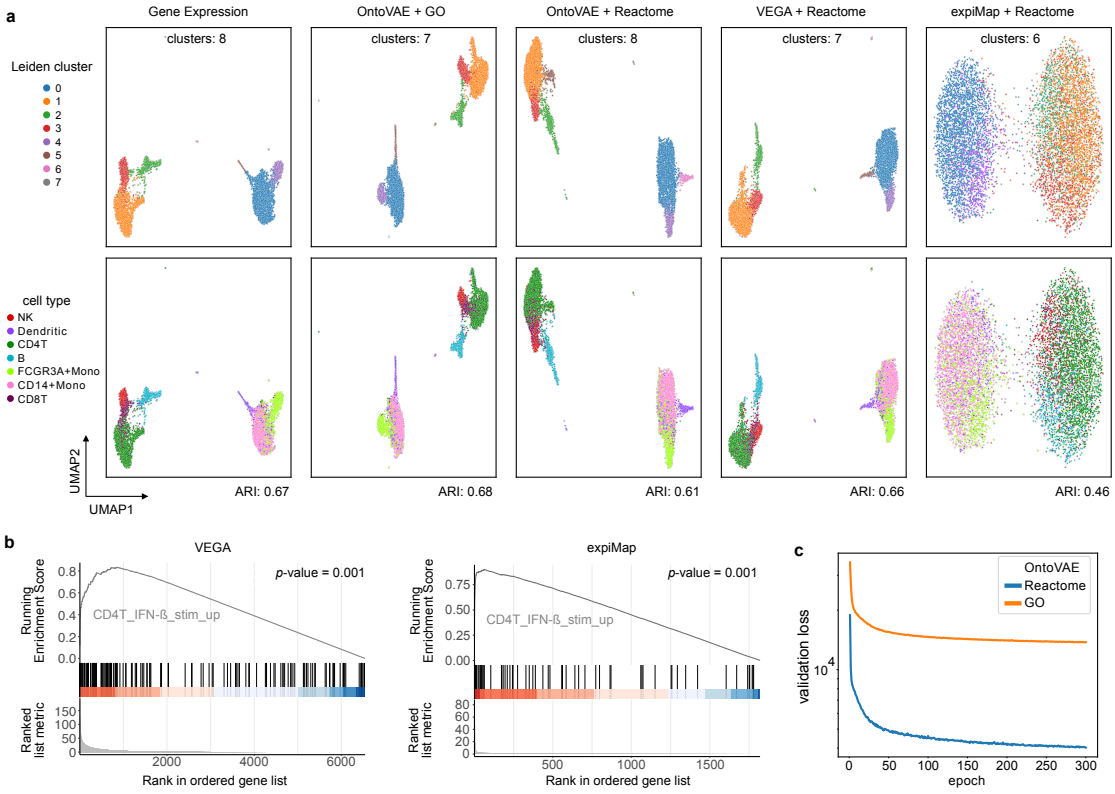

Figure 7.13: Comparison of OntoVAE with VEGA and expiMap. (**a**) Leiden clustering of unstimulated PBMC cells on gene expression data, the latent space of OntoVAE + GO, OntoVAE + Reactome, VEGA + Reactome, and expiMap + Reactome. (**b**) GSEA analysis as in **Figure 7.9**, this time for the VEGA and expiMap models. (**c**) Comparison of validation loss curves of OntoVAE + GO and OntoVAE + Reactome.

Chapter 8

Mechanistic dissection with COBRA

8.1 Development of the adrenal medulla

We applied COBRA on an scRNA-seq dataset of the developing adrenal medulla, which has been published in Jansky *et al* (2021) and is shown in **Figure 8.1**. The adrenal medulla is the inner part of the adrenal gland which is located on top of the kidney. In response to the sympathetic nervous system, the adrenal medulla mainly produces and secretes the steroid hormones epinephrine and norepinephrine into the circulation. The adrenal medulla is also of scientific interest because it can give rise to neuroblastoma, a pediatric cancer affecting the developing sympathetic nervous system. For their study, Jansky *et al* collected adrenal medulla cells from different timepoints post-conception. The UMAP of the dataset captures the lineage trajectories quite well (**Figure 8.1a**). Schwann cell precursors (SCPs) differentiate over intermediate states into two separate lineages: neuroblasts and chromaffin cells.

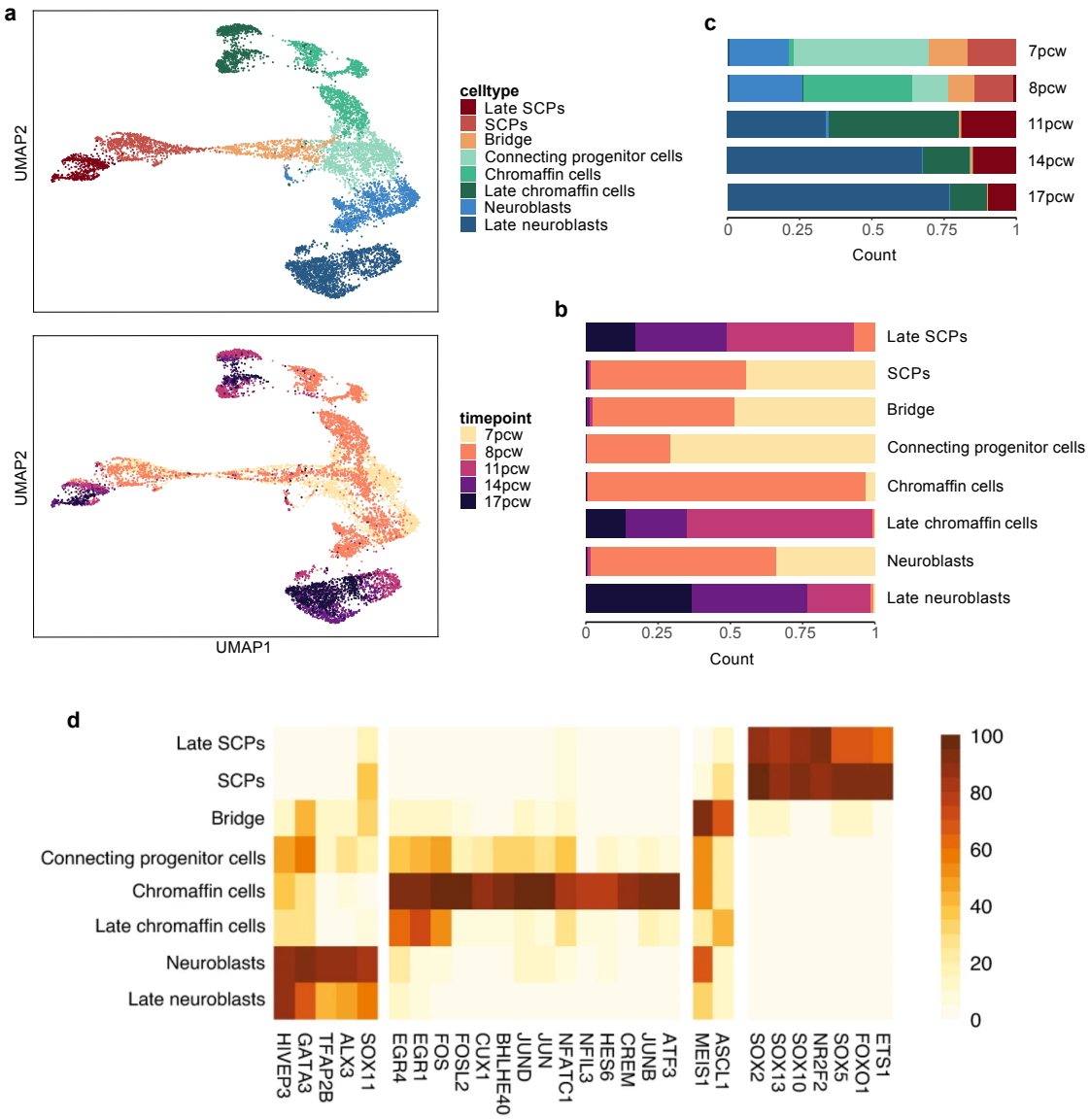

Figure 8.1: The developing adrenal medulla. UMAP representation of the data (**a**), cells are colored by their cell type (top) or the timepoint measured in post-conception weeks (pcw) (bottom). Barplots show the composition of the dataset aggregated by celltype (**b**) and timepoint (**c**). A heatmap displays the activities of TFs that are specific for the distinct celltypes (**d**), adapted from Jansky *et al* (2021).

The abundance of different cell types is strongly correlated with the developmental timepoint as measured in post-conception weeks (pcw), with a larger amount of Late SPCs, Late neuroblasts, and Late chromaffin cells in the later timepoints 11pcw, 14pcw, and 17pcw (**Figure 8.1b,c**). In their work, Jansky *et al* (2021) moreover identified TFs that are specific for the different cell types (**Figure 8.1d**).

Within our analysis, we asked two main questions. First, we wanted to see if COBRA would allow us to decouple the cell type effect from the developmental timepoint effect in the dataset to identify TFs and pathways that drive differentiation in general. Second, we wanted to see how COBRA performs in an *out-of-distribution* (ood) setting when applied on a previously unseen cell condition. Therefore, we designed the study as follows: We used the dataset specific SCENIC regulons that were generated by Qian-Wu Liao as part of his master thesis as well as the Reactome pathways as prior. The interpretable term layer was placed in the decoder of COBRA, followed by a ReLU activation function. We then defined the celltype and the timepoint as covariates for training COBRA. For both priors, we trained three models, whereby one model was trained on all cells (we call this full-model from now on), one model was trained on all cells but Late neuroblasts (we call this nb-model from now on), and one model was trained on all cells but Late chromaffin cells (we call this chrom-model from now on). Visualizations and follow-up analyses were then performed on all cells together.

8.1.1 COBRA disentangles celltype and timepoint effects

COBRA successfully disentangled the effects of celltype and timepoint in the latent space for both, the nb-model (**Figure 8.2**) and the chrom-model (**Figure 8.3**), using either the SCENIC TF prior or the Reactome pathways prior. While in the z basal view, cells are mixed, in the z timepoint view they are clustering according to timepoint, and in the z celltype view according to celltype. Since during model training, either Late neuroblasts (nb-model) or Late chromaffin cells (chrom-model) were never seen, for their projection onto the celltype embedding, they were encoded as neuroblasts and chromaffin cells, respectively. Interestingly, it can be observed that in the celltype view, cells of the

previously unseen celltype already form subclusters. This highlights the potential of COBRA to identify them as separate celltypes.

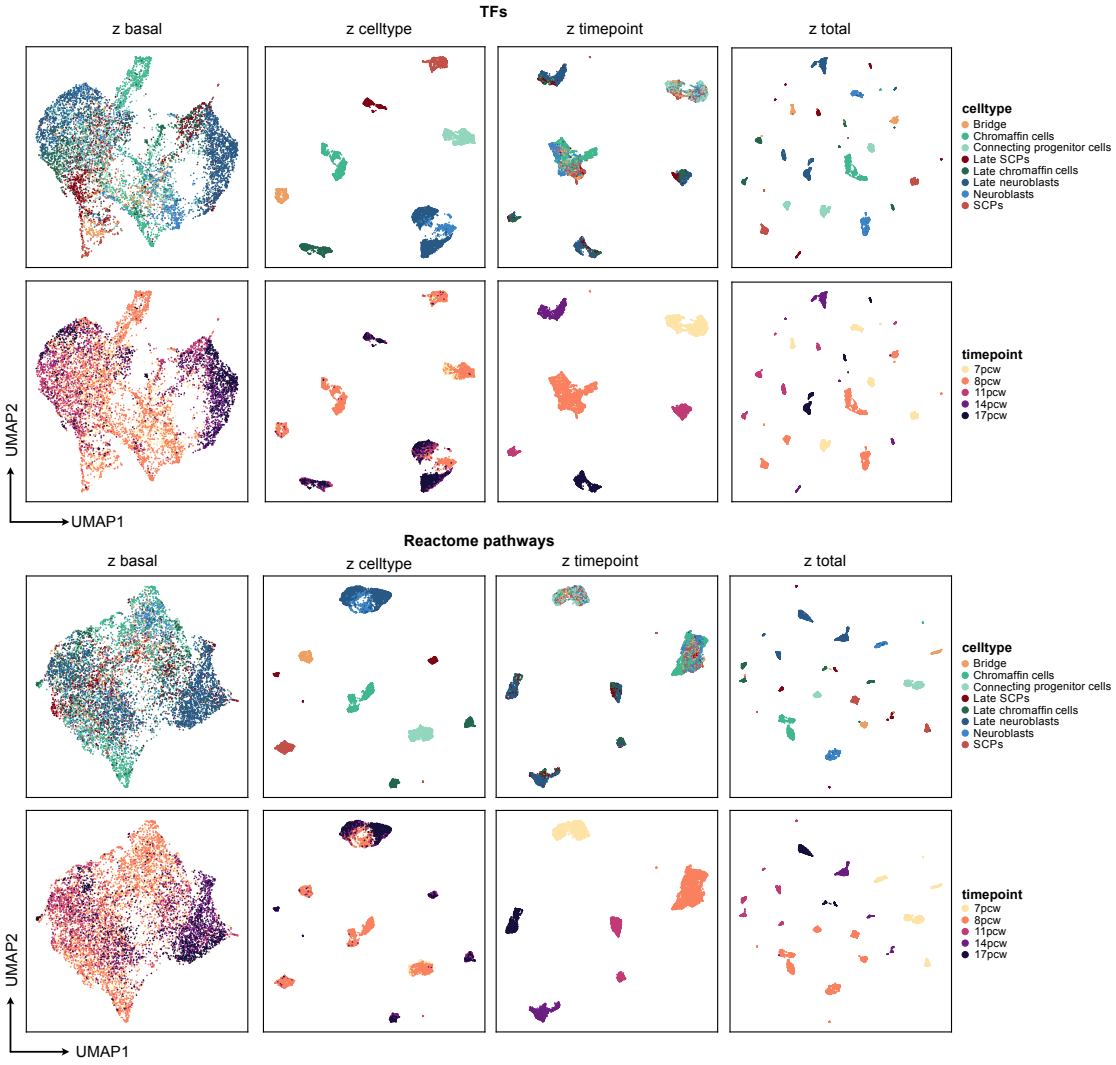

Figure 8.2: COBRA nb-model segregates the adrenal medulla dataset by celltype and timepoint. Latent space embeddings of the four different views z_basal, z_timepoint, z_celltype, and z_total are displayed (from left to right). Cells are colored by celltype (top) or timepoint (bottom). Model was trained on all cells but Late neuroblasts. Top panel: SCENIC TFs, bottom panel: Reactome pathways.

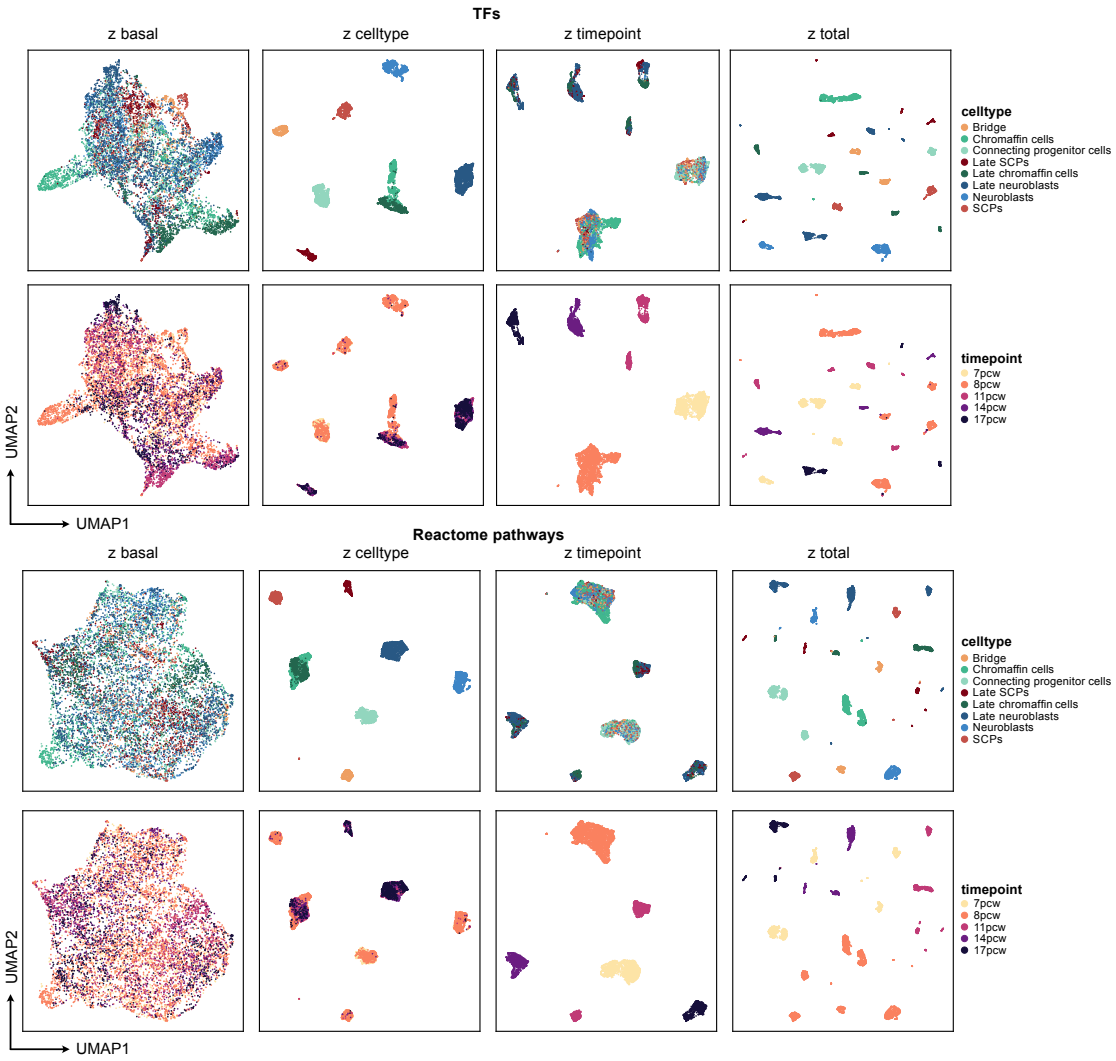

Figure 8.3: COBRA chrom-model segregates the adrenal medulla dataset by celltype and timepoint. Like **Figure 8.2**. Model was trained on all cells but Late chromaffin cells. Top panel: SCENIC TFs, bottom panel: Reactome pathways.

8.1.2 COBRA identifies TFs and pathways that drive differentiation

We then set out to identify TFs and pathways that play a role in development and differentiation of the adrenal medulla independently of cell type. For this, we extracted the activities from the timepoint view of the three models, the full-model, the nb-model

and the chrom-model. For each term, we then calculated the correlation between its activity and the timepoint as measured in pcw. The activities of the top 10 correlated terms are displayed in heatmaps for the three models (**Figure 8.4a** for TFs, **Figure 8.5a** for Reactome). We wanted to systematically investigate how well the nb-model and the chrom-model agree with the full-model, so we compared the correlations with each other, and found a significant agreement between the correlations both for the nb-model (**Figure 8.4b** for TFs, **Figure 8.4b** for Reactome) and for the chrom-model (**Figure 8.4c** for TFs, **Figure 8.4c** for Reactome). However, it is noted that the agreement between full-model and *ood*-model is better for the TF prior than for the Reactome prior. For both priors, a subset of terms displayed opposite correlations in the two models, but we showed this this was due to their lower absolute correlation values (**Figure 8.4d** for TFs, **Figure 8.5d** for Reactome). Here as well, this effect is more pronounced for the TF prior than for the Reactome prior. Taken together, we can assume that TFs and pathways that are indeed important for differentiation and development and have a strong correlation with timepoint, are also accurately captured by COBRA, even in the setting of an *ood*-model where the information of one Late celltype is completely absent during training. We speculate that the better agreement of an *ood*-model with the full-model when using the SCENIC prior is due to the fact that the SCENIC TF regulons were computed directly from the data and, hence, describe the data better than the more generic Reactome pathways. This highlights the importance of the prior in a model like COBRA.

Of the TFs we identified, CEBPA, SHOX2, TEAD4, REST, MITF, and GATA6 correlated positively with development and, thus, were especially active in the later timepoints 11pcw, 14pcw, and 17pcw. On the other hand, ZNF491, HOXD8, POU4F1, and ZNF263 correlated negatively with development, being more active in the earlier timepoints 7pcw and 8pcw. We conducted literature research to see what is known about these TFs in the context of development and differentiation. We found that CEBPA regulates myeloid cell differentiation (Pundhir et al. 2018), REST is a master regulator in neuronal differentiation (Hwang and Zukin 2018), SHOX2 is essential during heart development

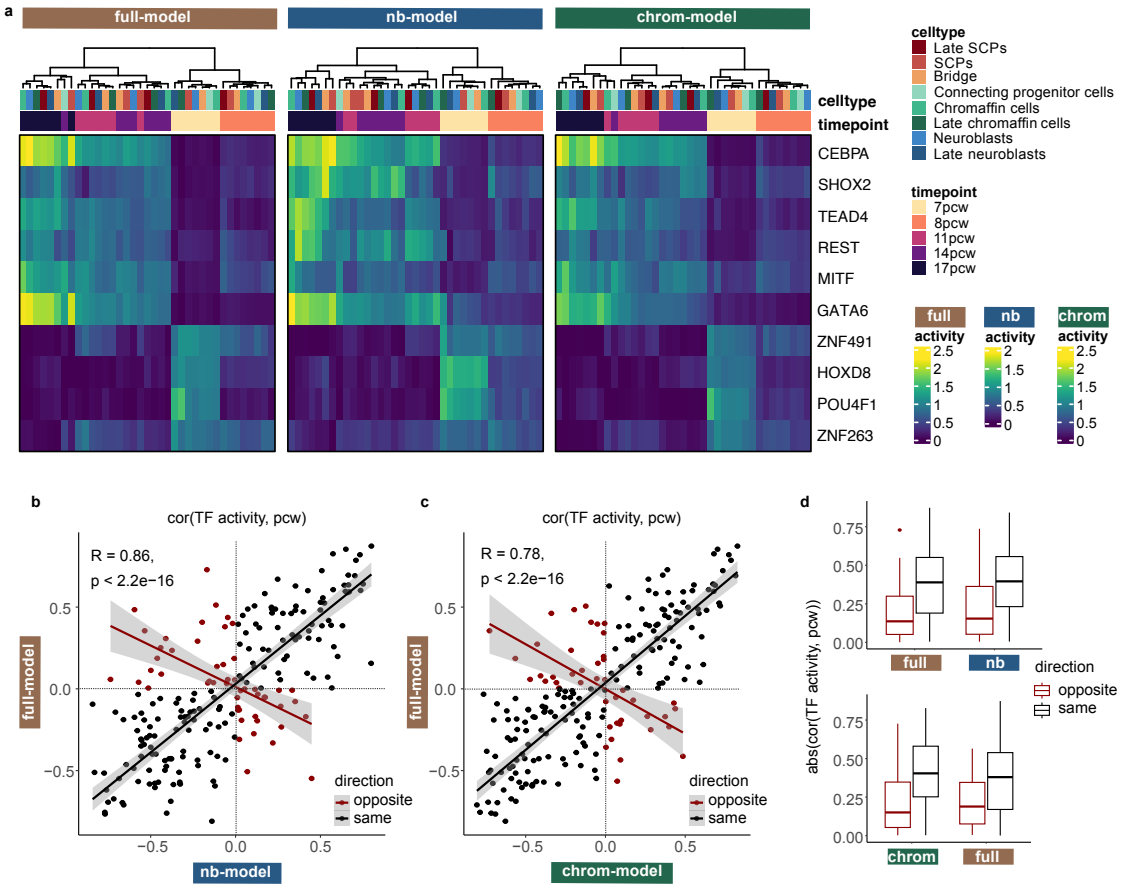

Figure 8.4: COBRA identifies TFs associated with timepoint. A heatmap shows the activities of TFs that have the highest correlation with timepoint in the nb-model (left), and the chrom-model (right) (**a**). A scatter plot displays the correlation between the correlation vectors of the two models for every TF (**b**). TFs that display the same direction of correlation in both models are colored in black, while TFs that display opposite directions of correlation are colored in darkred. R correlation coefficient and p-value are given for TFs of same directionality. A boxplot is comparing the absolute correlation values between TFs of opposite and same directionality (**c**).

(Espinoza-Lewis et al. 2009), TEAD4 plays an important role in trophectoderm differentiation (Stamatiadis et al. 2022), MITF drives melanocyte differentiation (Lee, Lim,

and Lim 2024), GATA6 leads to differentiation (Wamaitha et al. 2015), and POU4F1 is crucial for the differentiation of sensory neurons (Hudson et al. 2008). Furthermore, it is known that HOX genes influence stem cell differentiation during development (Bhatlekar, Fields, and Boman 2018). Hence, the identified TFs play a role in developmental processes, but to what extent they are involved in the development of the adrenal medulla remains to be investigated.

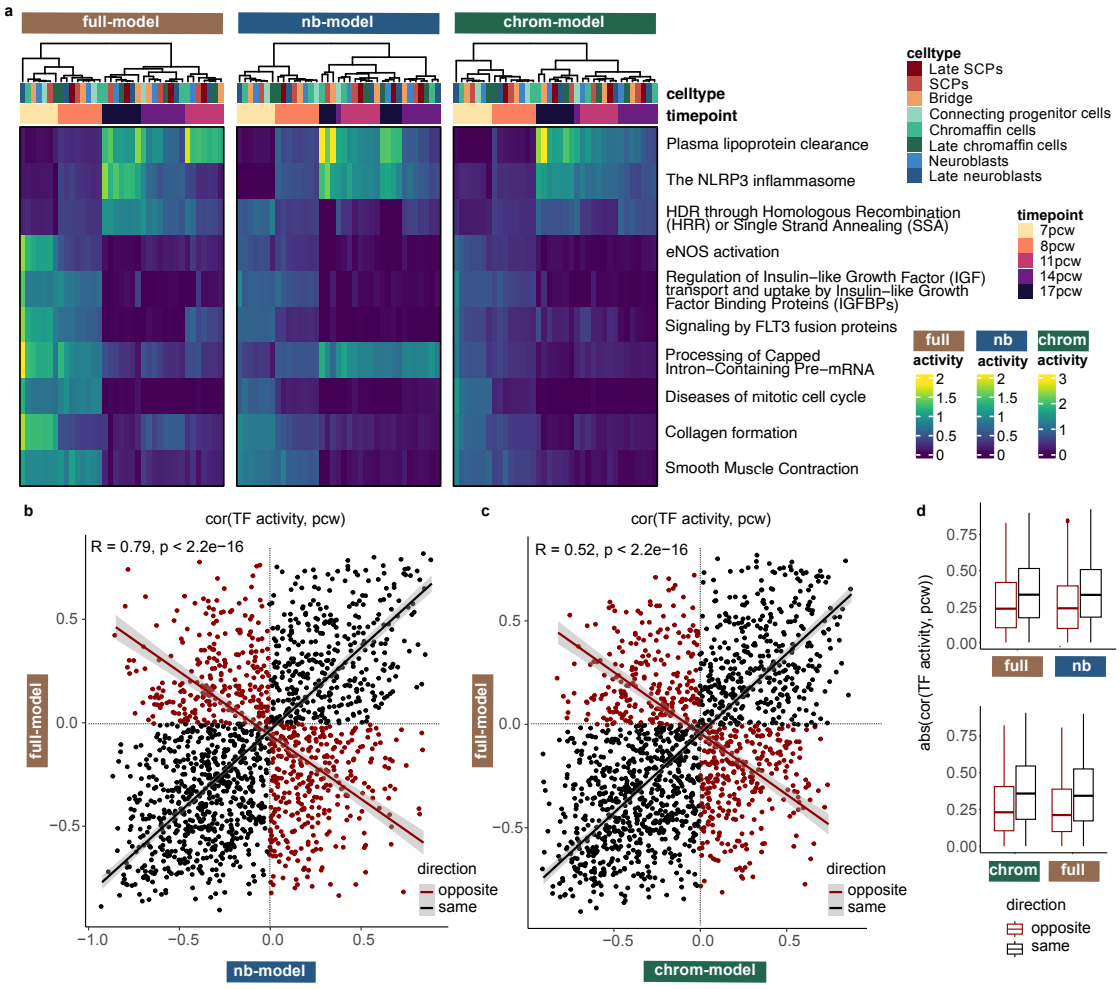

Figure 8.5: COBRA identifies pathways associated with timepoint. Like **Figure 8.4**, but with Reactome pathways as prior.

In terms of pathways, we find for example 'eNOS activation' to be negatively correlated

with timepoint. Nitric oxide (NO) has been shown to preserve pluripotency or induce differentiation dependent on the dose (Caballano-Infantes et al. 2022). Among the top 10 terms, we also find pathways that are oppositely correlated with timepoint in the nb-model compared to the full-model, such as 'Processing of Capped Intron-Containing Pre-mRNA', but speculate that this is due to the less adequate prior.

8.1.3 COBRA identifies celltype specific TFs and pathways

Next, we wanted to see whether COBRA could also identify celltype specific TFs and pathways. For an unbiased selection of terms, we calculated the variance of the activities in the celltype view. The top 50 terms that were selected this way are shown in the heatmap in **Figure 8.6** for TFs, and **Figure 8.7** for Reactome. Many of the TFs we find were also identified in the analysis of Jansky *et al* (2021), such as SOX2, ETS1, SOX10, and SOX13 in SCPs and Late SCPs, GATA3, ALX3, TFAP2B, and HIVEP3 in Neuroblasts and Late neuroblasts, MEIS1 and ASCL1 in Bridge, and EGR1, FOS, and EGR4 in Chromaffin and Late chromaffin cells.

Regarding the pathways, in the SCPs, which are more stem-like, we find terms such as 'TGF-beta receptor signaling in EMT (epithelial to mesenchymal transition)'. In Neuroblasts, we find pathways auch as 'Activation of AMPK downstream of NMDARs', 'Other semaphorin interactions', and 'Olfactory Signaling pathway', which are related with neuronal processes. In the Chromaffin cells, we find 'Metabolism of amine-derived hormones', such as epinephrine and norepinephrine, and 'Presynaptic depolarization and calcium channel opening'. Without a benchmark to compare against, more literature research is needed to determine the relevance of our identified pathways.

8.1.4 COBRA can predict late-timepoint celltypes

Finally, we wanted to investigate COBRAs ability to predict TF and pathway activities for the *ood* celltype, which was Late neuroblasts in the nb-model and Late chromaffin cells in the chrom-model. For this analysis, we exploited the fact that Late neuroblasts can

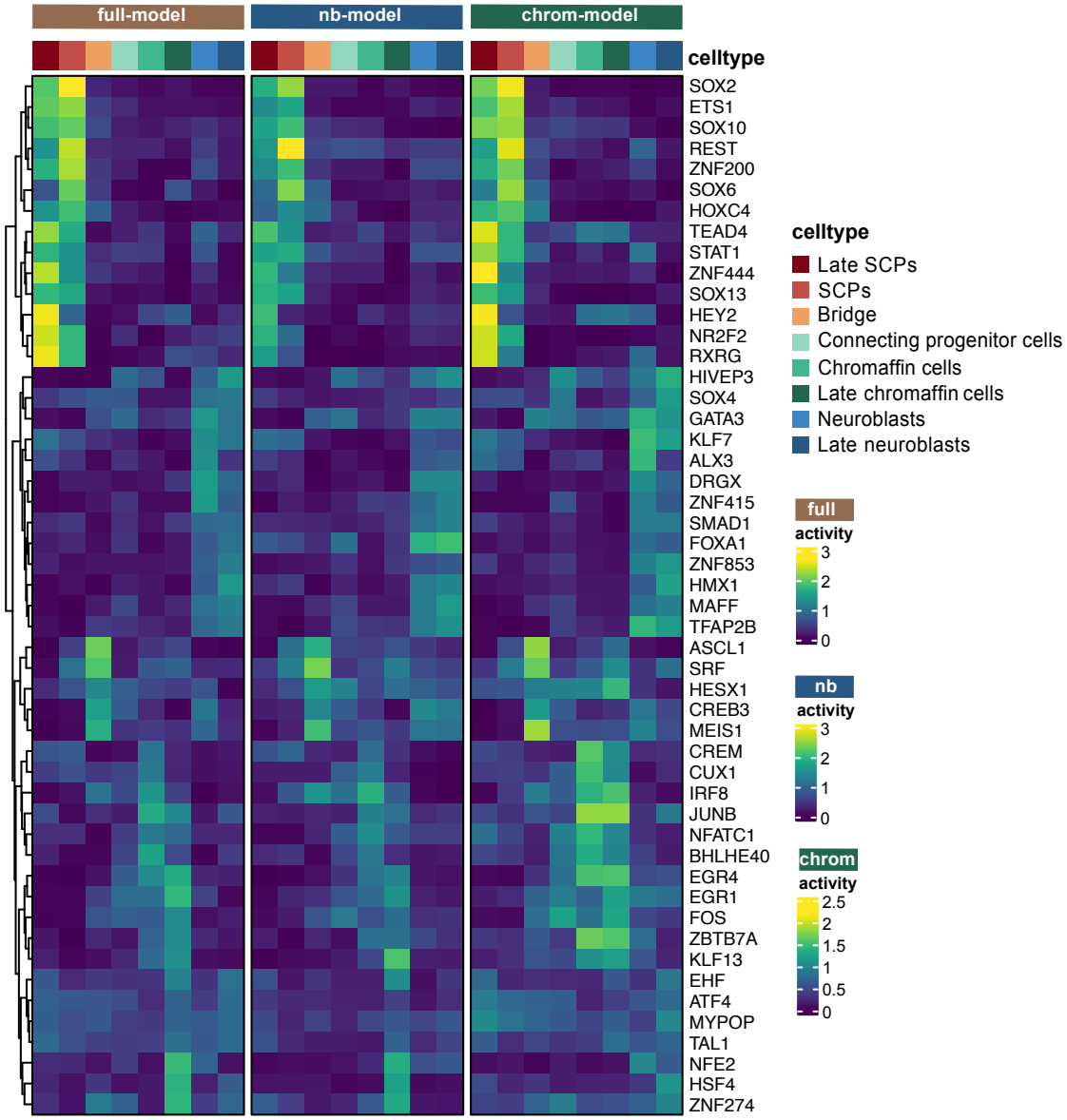

Figure 8.6: COBRA identifies celltype specific TFs. A heatmap shows the activities of TFs that have the highest variance in the celltype view for the full-model (left), the nb-model (left), and the chrom-model (right).

also be considered as neuroblasts from a later timepoint, and Late chromaffin cells can also be considered as Chromaffin cells from a later timepoint, respectively. Indeed, the different timepoints exhibit distinct cell type compositions, with the Late celltypes being

more prevalent at later timepoints. (**Figure 8.1c**). Thus, since the *ood* models could not learn a separate embedding for the missing celltype, when passing data through the model, we encoded the Late neuroblasts as neuroblasts, and the Late chromaffin cells as chromaffin cells, respectively. Thus, their learnt cell type embeddings were the same. But, since the Late cell types predominate at later timepoints, the timepoint embeddings differ. Hence, the z total view allows a separation of the populations, which is also observed in the UMAPs (**Figure 8.2,8.3**). We then compared the activities in the z total view between the full-model, the nb-model and the chrom-model for a set of selected TFs: FOS, EGR1, and EGR4, which are predominant in (Late) chromaffin cells, and ALX3, TFAP2B, GATA3, and HIVEP3, which are predominant in (Late) neuroblasts (**Figure 8.8**), and a set of selected pathways: 'Transcriptional Regulationby MECP2', 'Postsynaptic acetylcholine receptors', 'Olfactory Signaling Pathway', and 'Phase 0 - rapid depolarisation', which are predominant in (Late) neuroblasts and 'Circadian Clock', 'Tryptophan catabolism', and 'Antimicrobial peptides', which are predominant in (Late) chromaffin cells (**Figure 8.9**). In terms of TFs, we observe that, for EGR1 and EGR4, the trend between chromaffin cells and Late chromaffin cells is the same in the chrom-model, which never encountered the Late cells during training, as in the full-model. Only for FOS, the chrom-model captures the opposite trend. For ALX3, TFAP2B, GATA3, and HIVEP3, the trend between neuroblasts and Late neuroblasts is the same in all three models, although the nb-model has never seen Late neuroblast samples during training. For the displayed pathways as well, the same trends are captured between cells and their late timepoint equivalents in all three models, except for the pathway 'Nicotinate metabolism', where the nb-model shows the opposite compared to the full-model. These results confirm that, through the decoupling of celltype and timepoint, COBRA is able to predict *ood* Late celltypes since it can still accurately learn the timepoint embedding from the remaining celltypes.

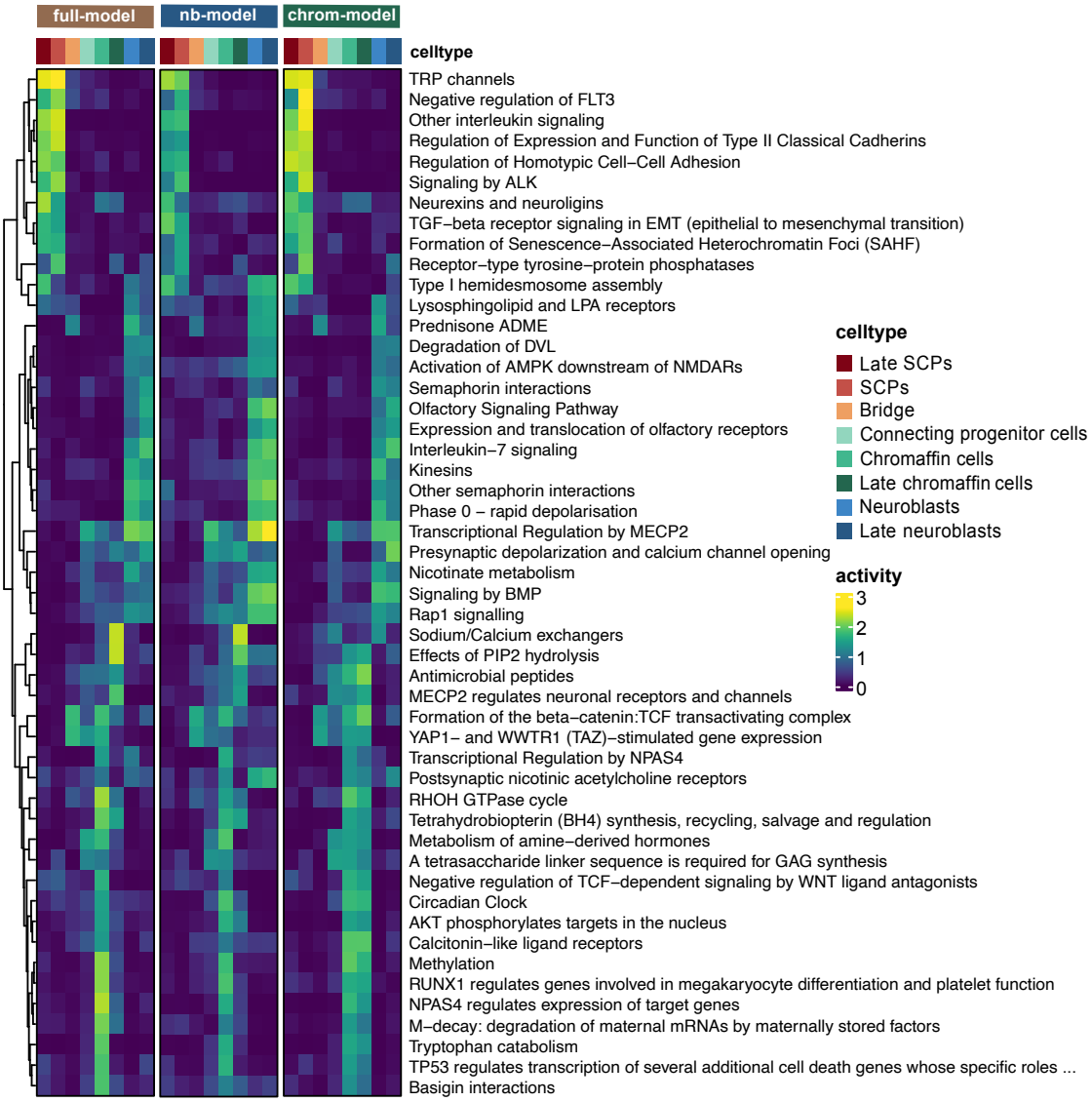

Figure 8.7: COBRA identifies celltype specific pathways. Like **Figure 8.6**, but for Reactome pathways.

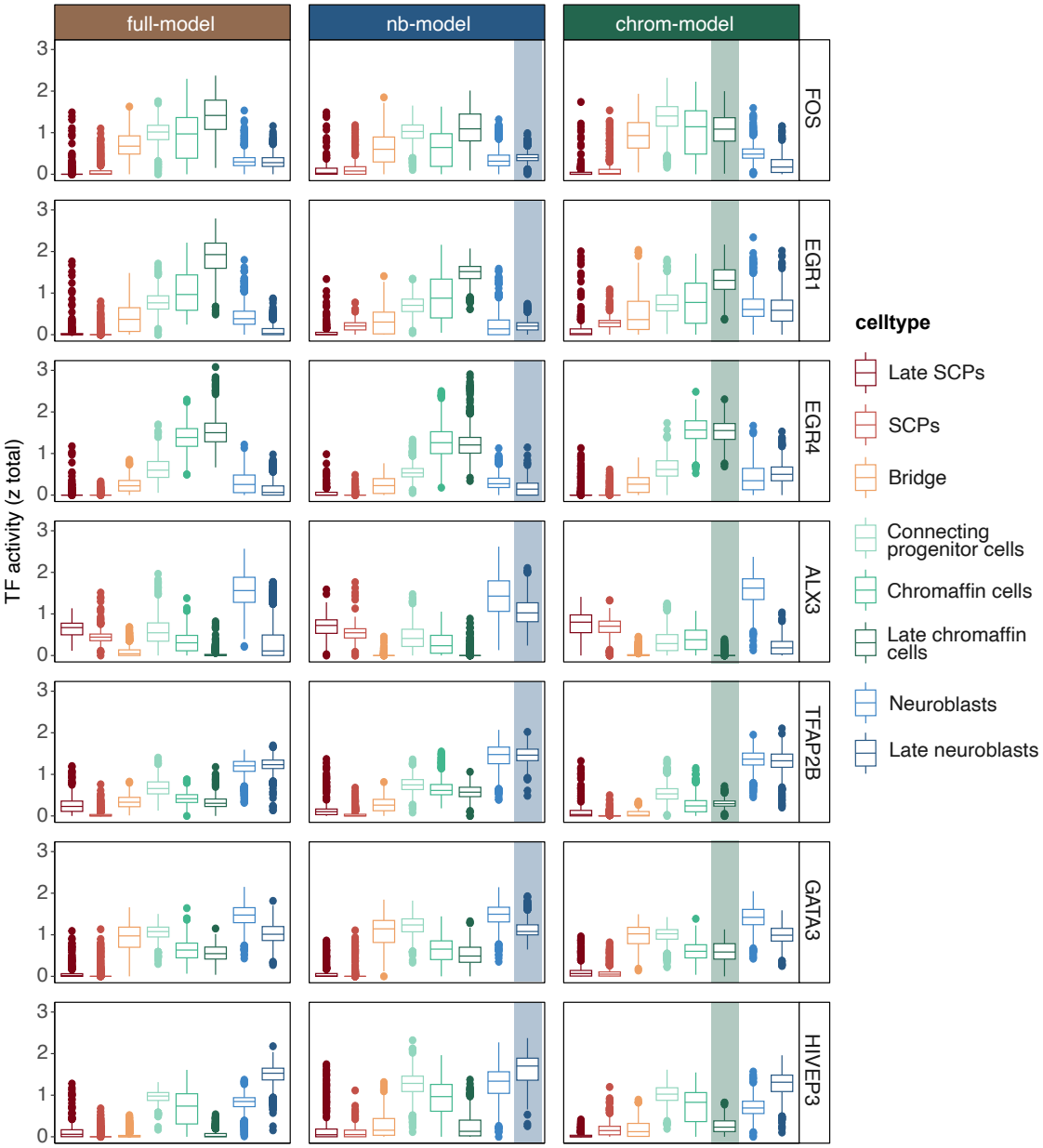

Figure 8.8: COBRA can predict late-timepoint celltypes. Boxplots show activities of selected TFs by celltype in the z total view for the full-model (left), the nb-model (middle), and the chrom-model (right). The *ood* celltype is highlighted in the respective model.

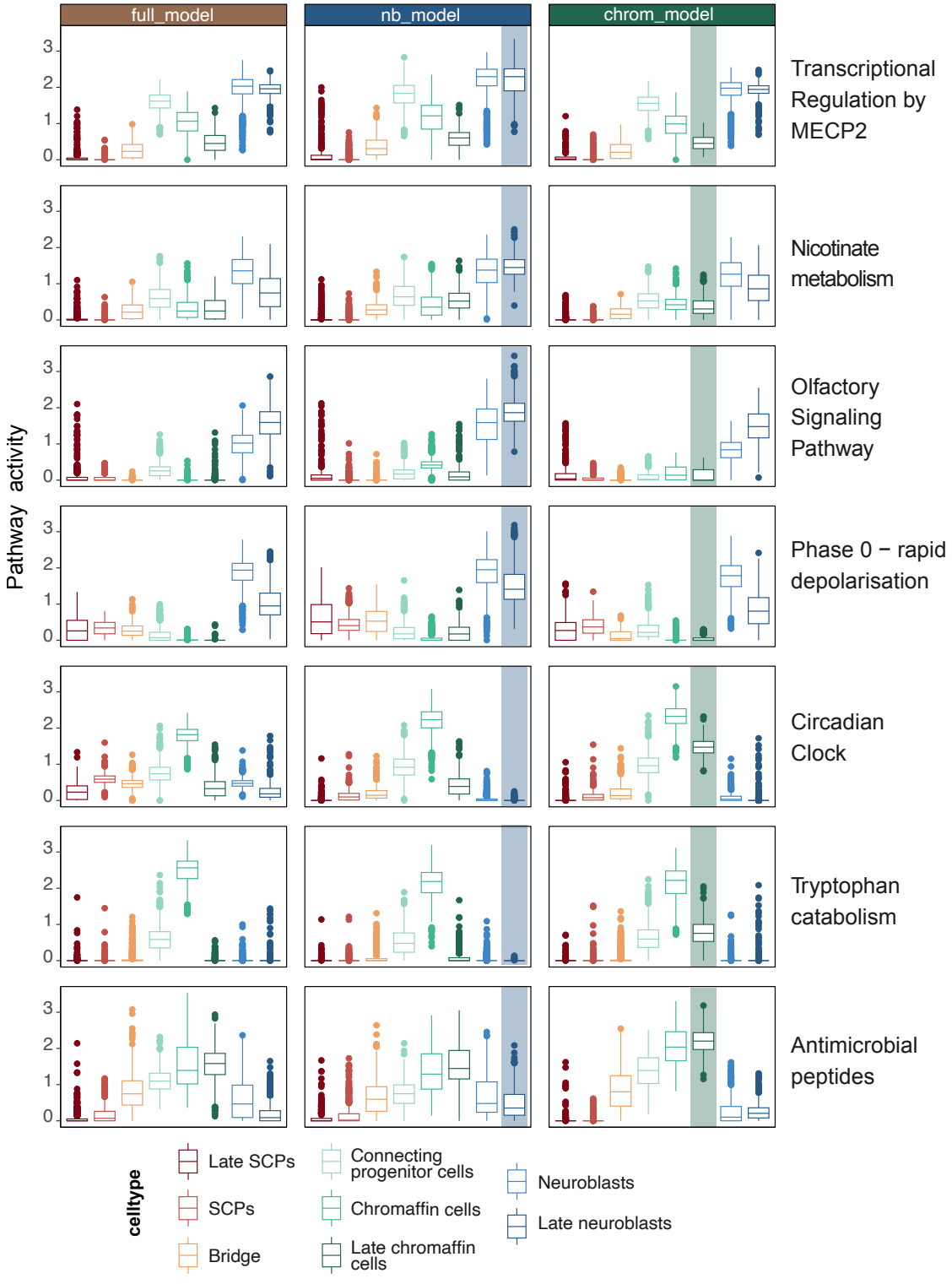

Figure 8.9: COBRA can predict late-timepoint celltypes. Like **Figure 8.8**, but for Reactome pathways.

8.2 Limitations: Classification of schizophrenia

We applied COBRA on a single-cell RNA-seq dataset of schizophrenia (SCZ) that was generated by the PsychENCODE consortium and comprises 468,727 nuclei isolated from 140 individuals across two cohorts (Ruzicka et al. 2024). The dataset consists of different neuronal celltypes, excitatory and inhibitory, as well as Astrocytes (Ast), Oligodendrocytes (Oli), Oligodendrocyte precursor cells (OPC), and small fractions of Microglia (Mic), Endocytes (Endo), and Pericytes. Furthermore, the dataset consists of two cohorts, McLean and MtSinai. Throughout this section, model training was always performed on the larger McLean cohort, while analysis and plotting was performed on the smaller MtSinai cohort. A UMAP representation of the MtSinai cohort of the dataset as well as celltype proportions are given in **Figure 8.10**. The clustering is strongly driven by the celltype covariate, with the phenotype covariate mixed in each celltype cluster.

We wanted to see whether COBRA could separate the celltype and phenotype effects from each other, and thus identify pathways that are important for the distinction of SCZ versus control. Thus, we trained COBRA on the McLean cohort, passing the celltype and phenotype as covariates and using Reactome pathways in the decoder as a prior, and then applied the trained model on the MtSinai cohort. A UMAP representation of the different latent space views is displayed in **Figure 8.11**. As expected, in the z basal view, the cells are mixed, while in the covariate embedding views they cluster according to celltype or phenotype, respectively. However, since the phenotype was already mixed, we wondered whether COBRA actually captured a biologically meaningful separation between SCZ and control, or whether this was due to the fact that it was passed the labels during training. To investigate this closer, we performed two permutation analyses, shuffling either the phenotype or the celltype labels before training. We observed that, when the phenotype labels were shuffled, the latent space embeddings for z basal and z phenotype looked the same as previously, with a good mixing in the basal view, and a separation of SCZ and control in the phenotype view (**Figure 8.12a**). On the other

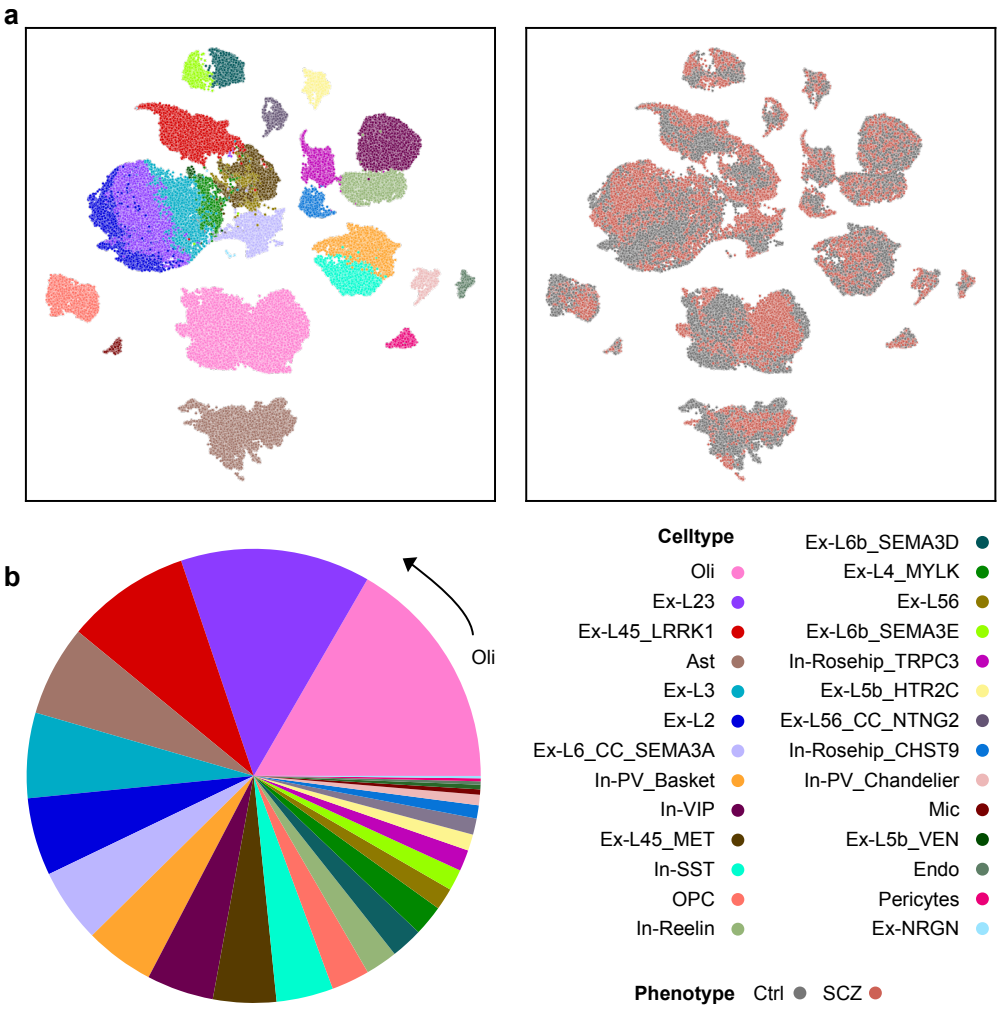

Figure 8.10: PsychENCODE schizophrenia single-cell RNA-seq dataset. **a** UMAP representation of the dataset, with cells colored by celltype (left) or phenotype (right). **b** Pie chart showing the proportions of different celltypes. Oli- Oligodendrocyte, Ast - Astrocyte, OPC - Oligodendrocyte precursor, Mic - Microglia, Endo - Endocytes.

hand, when the celltype labels were shuffled, the celltypes separated from each other in the celltype view, but were not mixed in the basal view (**Figure 8.12b**). From these analyses, we conclude that, the model will always learn the correct clustering in the covariate embedding views, since it is passed the different labels. However, if

data is mislabelled, the adequate learning of the basal view is affected, since the model optimizes the encoding of z basal together with the covariate embeddings. Or, in other words, the model cannot subtract out the covariate effects from the different classes if the data is mislabelled, thus it cannot achieve a mixing of cells in the basal view in that scenario. This means that we cannot derive from this experiment whether the separation between SCZ and control is meaningful, since the cells were already mixed with respect to that covariate before applying COBRA. Therefore, we decided to further investigate the pathways that were best separating SCZ from control in the phenotype view. As mentioned previously, due to the adversarial approach taken by COBRA, training is less stable and results are less reproducible than for the original OntoVAE model. Thus, we trained the normal model five times, the phenotype shuffled model twice, and created five random rankings of Reactome terms. For the trained models, we trained Naive Bayes classifiers on the phenotype view activities, and ranked the terms by their AUC as done previously. We then compared the top x terms between different model combinations, either by directly computing the intersection (overlap) of terms, or by computing the mean jaccard similarity (MJS), so that also similar terms are taken into account (**Figure 8.12c**).

The formula to compute the jaccard similarity between two sets of terms is given in **equation** (8.1).

$$J(A,B) = \frac{A \cap B}{A \cup B} \tag{8.1}$$

The jaccard similarity was first calculated for all pairwise combinations of top x ranked terms between two models A and B to obtain a matrix with jaccard similarities:

$$J_{x,x} = \begin{pmatrix} J(A_1,B_1) & J(A_1,B_2) & ... & J(A_1,B_x) \\ J(A_2,B_1) & J(A_2,B_2) & ... & J(A_2,B_x) \\ ... & ... & ... & ... \\ J(A_x,B_1) & J(A_x,B_2) & ... & J(A_x,B_x) \end{pmatrix} \tag{8.2}$$

The MJS in dependence of x was then calculated the following way:

$$MJS = \frac{1}{x}\sum_{i=1}^{x} max_{j=1}^{x} J(A_i,B_j) \tag{8.3}$$

To create a baseline (What overlap would we expect when comparing two random sets of x terms?), we calculated the overlap and the MJS between all possible combinations of our previously defined random term rankings. We then calculated these measures also between the normal models, as well as between normal and phenotype shuffled models. We observed that, for the comparisons normal-vs-normal and normal-vs-shuffled, the overlap and the MJS are higher than for the comparison random-vs-random, indicating that the identified terms are not random. However, measures are around the same for normal-vs-normal and normal-vs-shuffled, which means that COBRA captures some other effect than the biology between SCZ and control and results are probably based on a model bias here.

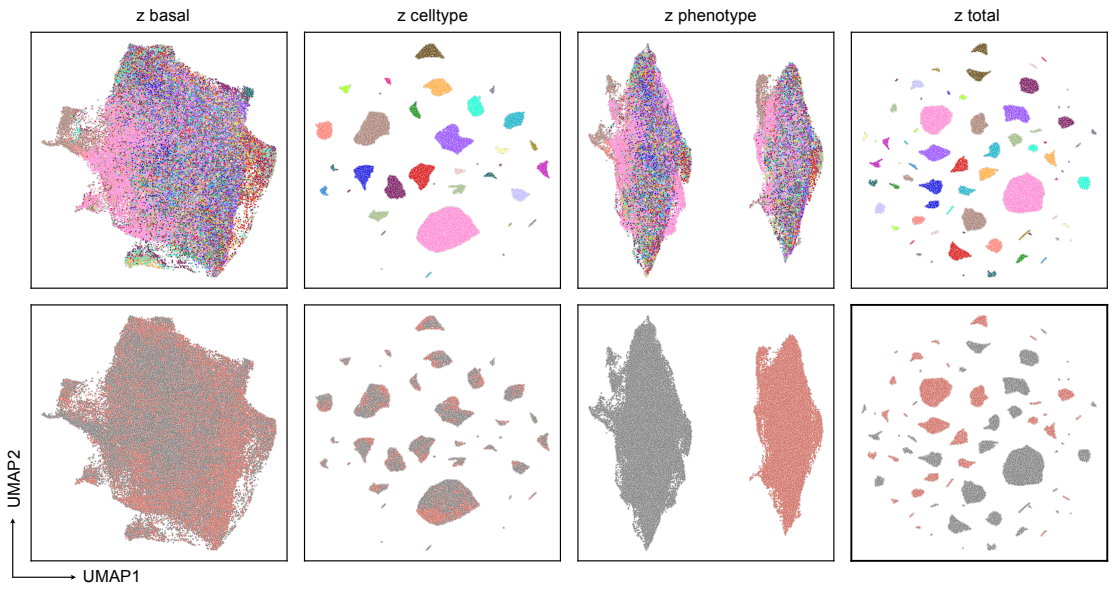

Figure 8.11: Different COBRA views of the PsychENCODE schizophrenia single-cell RNA-seq dataset. Trained model was applied on the MtSinai cohort, displayed are the UMAPs on the latent space embeddings for the different views basal, celltype, phenotype, and total. Cells were colored by celltype (top), and phenotype (bottom). Color legend can be found in **Figure 8.10**.

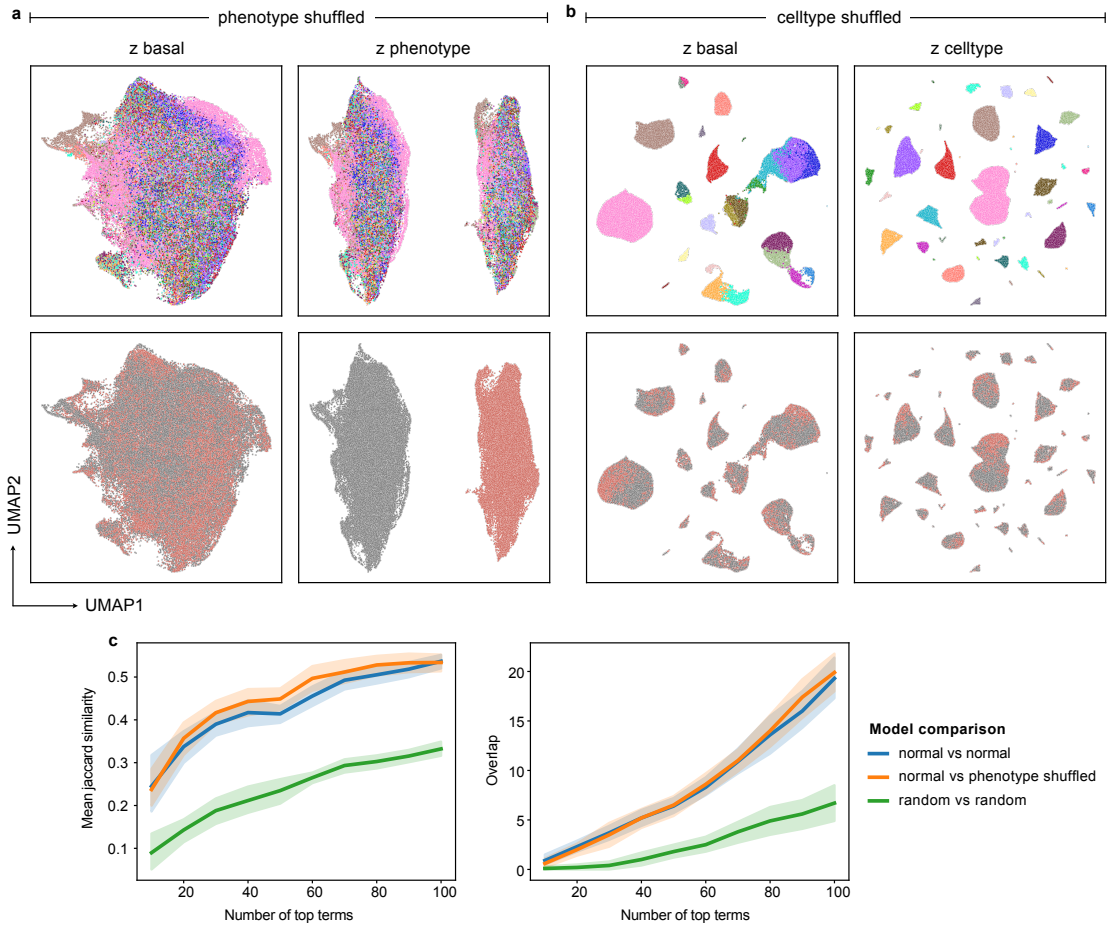

Figure 8.12: Shuffling of phenotype and celltype labels. COBRA was trained on the dataset after shuffling the phenotype labels (**a**), or the celltype labels (**b**), and UMAP plots for different latent space views are shown. Color legend can be found in **Figure 8.10**. **c** Line plots are showing the mean jaccard similarity or the total overlap when comparing the top phenotype associated terms between multiple runs of the same model for different comparisons.

We speculated about possible reasons why COBRA fails on the PsychEncode dataset. First, indeed, if the effect of a covariate is as small as is seen between SCZ and control, the adversarial approach might simply fail as the cells are already completely mixed to

begin with. Second, in a dataset, where a covariate effect is not global, but only restricted to a subset of the dataset, COBRA would still learn global separate embeddings for each covariate. For example, it might be that not all cell types are affected equally by the SCZ condition. Indeed, in the original publication, authors described the major changes to occur in the excitatory neuron populations (Ruzicka et al. 2024). However, COBRA cannot distinguish these kind of effects, thus it is also not suited to reveal celltype specific responses.

8.3 Chapter summary

We applied our tool COBRA on two datasets: an scRNA-seq dataset of the developing adrenal medulla (Jansky et al. 2021) and an scRNA-seq dataset of post-mortem brains of schizophrenia patients and healthy controls (Ruzicka et al. 2024). We wanted to demonstrate COBRAs ability in decoupling the effects of different covariates and how we can exploit that to get mechanistic insights.

For the adrenal medulla dataset, we used two different priors: SCENIC TF regulons that were computed directly from the data, and generic Reactome pathways. We trained two models each, leaving out either Late neuroblasts or Late chromaffin cells during training. We showed how COBRA can be used to decouple the celltype effect from the timepoint effect, and how this fact could be used to predict the Late unseen celltype. However, we also noted that model performance was better using a data-driven prior, thus highlighting also the importance of choosing an appropriate prior.

For the schizophrenia dataset, we used Reactome pathways as a prior, and trained COBRA using the celltype and the phenotype as covariates. Although we observed a split in phenotype, we performed additional experiments where we shuffled phenotype or celltype labels. These experiments demonstrated that the adversarial approach is compromised when a covariate is already well mixed from the beginning. Hence, we could not extract any meaningful biological mechanisms when applying COBRA on this dataset.

Part IV

Final Remarks

Chapter 9

Discussion and conclusion

9.1 Development and limitations of OntoVAE

OntoVAE has been implemented in pytorch, which recently became the most popular DL library in python, and published as a python package that is available under: `https://github.com/hdsu-bioquant/onto-vae`, and can be installed via `pip install onto-vae`. OntoVAE is an interpretable VAE framework that can incorporate any kind of biological ontology into its latent space and decoder part, so that the activations of the interpretable neurons directly correspond to pathway activities. Since a biological ontology is a directed acyclic graph (DAG), one major difficulty during development was the modeling of skip connections, connections that are present between non-neighboring layers. This was achieved trough a step-wise concatenation process during the for loop in the training, and binary masks at each step indicating presence or absence of a connection. The package is composed of two main classes: the `Ontobj()` class which implements functions for processing and trimming the ontology and generating the masks, and the `OntoVAE()` model class, which accommodates the pytorch implementation for building and training the DL model. The `OntoVAE()` model class itself is composed of modular building blocks, mainly an encoder and a decoder which incorporates all the logic with the ontology. This modular implementation makes OntoVAE a flexible tool. Another

advantage of OntoVAE is the user friendliness. The user can perform the whole workflow by means of a few simple commands as illustrated in the package vignette. Besides, useful functions for carrying out analysis on a trained model are implemented to facilitate retrieval and plotting of pathway activities, performance of *in silico* perturbations, and statistical testing.

We demonstrated in this work that the pathway activities retrieved by OntoVAE are meaningful and reproducible. However, as with any model that relies on prior information, one limitation of OntoVAE is that the prior will heavily influence the results, since only relationships and trajectories through the graph can be explored that are already known. Some of the related one-layer models do not hard-code the gene annotations, but impose a penalty on term on connections that are not present in the prior, allowing to 'learn' the prior during training. However, with OntoVAE, this would be hard to realize, since connections could in principle be possible between any nodes from any two layers. Additionally, the use of the prior restricts the model to make use only of genes that can be annotated to the prior, which *a priori* excludes non-coding genes from any kind of analysis that can be made. Other limitations of the model arise due to the nature of DL models. For once, there is no way to model different connections. For this reason we limited OntoVAE to encode only is-a connections. However, this might not be an accurate enough description of the underlying biology. Moreover, it is difficult to interpret the weights that are learned in the decoder. The weights in the reconstruction layer, the ones that connect every term to their associated genes, are biologically different from the weights that connect parent and children terms. One intuitive explanation would be that the decoder weights reflect the strength of the connection, but since there is just one set of weights per trained model, no comparison between different sample groups can be made on this level. Finally, due to internal model biases, comparisons can only be made between groups at one particular neuron/term, but not by ranking all neurons for one particular group or sample.

9.2 Perturbation prediction with OntoVAE

One possible application of OntoVAE is the prediction of perturbations. This can be done in a gene-centric manner (Which terms are mostly influenced by the gene?) or a term-centric manner (Which genes mostly influence this term?). For this kind of analysis, the input values for certain genes are manipulated, e.g. set to zero, and then the pathway activities at the terms of interest that are obtained before and after the *in silico* manipulation are compared. Evaluation of the statistical significance is based on a paired Wilcoxon test. In this work, we demonstrated how OntoVAE can predict the outcome of a gene knockout, and also how it can predict the influence of a disease or drug treatment.

One problematic of this analysis is the sensitivity of the paired Wilcoxon test, which usually produces many significant hits, even after multiple testing correction. This might include a high amount of false positives. Additionally, especially for nodes with generally low activities, results may vary due to the stochasticity of latent space sampling. Furthermore, the distinction between causal and indirect relationships is not possible, and the directionality of the influence sometimes hard to interpret. One other aspect is the comparison with more simple, one-layer models. In our benchmark, there was no clear advantage of using a more complex model over a more simple one in terms of predictive power. Nevertheless, we believe that the additional benefit of OntoVAE lies in the possibility to trace different activation trajectories through the hierarchy of the graph, and to see at what point different groups diverge.

9.3 Development and limitations of COBRA

We further developed the OntoVAE tool and made it more flexible and user-friendly, and hosted the code in the same GitHub repository as the new COBRA model: `https://github.com/hdsu-bioquant/cobra-ai` which will be made available following publication. The COBRA model extends OntoVAE with an adversarial approach that encour-

ages a decoupling of specified covariates in the latent space. The purpose of this is to capture more subtle effects at the level of pathway activities that might be overshadowed by a strong confounder such as celltype. In this work, we showed how COBRA could extract IFN treatment related effects that were not visible when the celltype was not decoupled. Since COBRA still uses an interpretable decoder framework that is based on a prior, the same limitations apply that were previously discussed for OntoVAE. Additionally, due to the adversarial approach implemented in COBRA, model training is less stable, meaning that obtained results vary more between different training runs of the same model than for the original OntoVAE model.

Other limitations became apparent when applying COBRA on different datasets. First, we noted a strong dependence on the used prior. In the adrenal medulla dataset, reproducibility metrics were better when a dataset specific prior (the SCENIC TF regulons) was used compared to the general Reactome pathways. Especially for single-cell data, genesets from Reactome and GO might not be adequate anymore since they were mostly computed on bulk data. However, this means that another method would have to be applied first, to compute the prior from the data. We can also imagine different approaches to generate an adequate prior from the dataset at hand or from a related dataset. One could for example trim the ontology to only keep branches that are important in distinguishing groups in the given dataset, or create a custom ontology in a bottom-up approach by computing gene-gene relationships from the data and organizing them in a hierarchical way. In this work, we only used COBRA with a one-layer interpretable decoder structure, since model reproducibility became worse with a multi-layer ontology (results for this are not shown in this work). However, in the future, we would like to expand COBRA to our multi-layer model again, and we speculate that more adequate priors might also generate more reproducible results.

One major limitation of COBRA became apparent when we applied the tool to study a scRNA-seq dataset of schizophrenia. In this dataset, cells clustered according to their celltype, with no visible separation between schizophrenia and control. On a transcrip-

tomic level, the effects of schizophrenia are very subtle and thus hard to isolate. Thus, we wanted to see whether COBRA could capture something meaningful in this scenario. However, we noted that, while COBRA segregated schizophrenic and control cells in the phenotype view, it also did so in a shuffled scenario where the label should not carry any meaning anymore. On the other hand, when we shuffled the celltype variable, we still obtained a segregation in the celltype view, but not a mixing of cells in the basal view. Thus, we concluded that COBRA cannot accurately remove the covariate effects from the basal view if data is mislabelled. However, since data was already mixed in terms of phenotype, there was no way to tell if the model learnt a meaningful separation or not. We then looked at top pathways, and found the same proportions of overlap between models that were trained with correct labels and models that were trained with wrong labels. This demonstrated that COBRA will always learn a separation in the covariate embeddings because it is given the labels, and that it learned some bias rather than an actual biological effect for the schizophrenia dataset. We speculate that the adversarial approach is compromised when cells are already mixed well from the beginning. Furthermore, we believe it is a problem if one of the studied effects is not global but limited only to a subset of cells. For example, if the schizophrenic condition only affected a subset of the celltypes, COBRA would still learn a global schizophrenic effect and a global celltype effect, and then make linear combinations of both for each celltype in the latent space. Thus, by design, it is not possible to decouple an effect only from part of the cells.

9.4 IFN response in the gut

We analyzed scRNA-seq data from Ileum and Colon organoids that had been treated with IFNβ, IFNλ, or both, to study the IFN response in the human gut. Due to the composition of the dataset, there are many possible research questions and directions in which to take the project. One focus could be the comparison between the two different tissues Ileum and Colon, but also between the different celltypes that the tissues are composed of. Here, one could investigate either the similarities and differences between

celltypes of the same tissue, or between the same celltype of the two different tissues. Another possible focus could be the comparison between the different treatment conditions, since IFNβ is a type I IFN, and IFNλ a type III IFN. Moreover, their potential synergistic action could be investigated in the double treatment. To better organize and visualize the vast amount of results that were generated, we developed a collaborative Shiny web application that allows to browse the results in a systematic way, and to look at ISGs, enrichment analyses and TF activities in the two tissues and the different celltypes. We found multiple lines of interesting findings, among others the fact that IFN response is indeed very cell type specific, and also that, especially for some cell types, IFNβ response triggers other than the classical antiviral mechanisms. We highlighted this at the example of the EECs from Ileum, hormone secreting cells that are important for the regulation of digestive processes. In this context, IFNβ seemed to amplify the communication between the EECs and peripheral neurons. However, one should be aware that the numbers of EECs in the dataset were rather low, thus, further experiments are needed to confirm this.

Another interesting discovery consisted in the potential synergistic mechanism of IFNβ and IFNλ. This also opens up a possible application for COBRA, where we could try to see whether we can predict the double treatment from the single treatments. Here, we imagine an approach where we modify samples in the latent space, by adding up the learnt covariate embeddings from both categories. We call that 'latent space engineering'. We could also look at intermediate states by looking at different mixing proportions of the two latent space embeddings.

9.5 Final remarks and overall perspective

The main work of this thesis consisted in the development of two interpretable VAE models, OntoVAE and COBRA. While OntoVAE allows for direct interpretability of its decoder which is structured like a biological network, COBRA extends this approach with an additional decoupling of covariates in the latent space. Thus, both models

address the need of the scientific community for trustworthy models that can be applied in critical domains such as personalized medicine. In the field of omics, the VAE has been one of the most popular models so far due to its ability to capture the essence of high-dimensional data, with many applications ranging from the impute of missing values to batch correction and data integration. However, the field of deep learning in the life sciences is rapidly evolving, and since the breakthrough of ChatGPT, the transformer model class, which is state-of-the-art for NLP tasks, has now also become the focus of attention of life scientists.

Transformers exploit a mechanism called self-attention which helps them to discover relevant connections in the data independent of the proximity of the data points. Recently, the first transformer based models for single-cell data were published, among them scGPT (Cui et al. 2024) and scFoundation (Hao et al. 2024). Both models are so-called foundation models, meaning that they are pretained on a very large corpus of data, and can then be fine-tuned on specific tasks with less data available (Boiarsky et al. 2023). The application of foundation models has been demonstrated on various tasks, such as cell type annotation, perturbation prediction, data integration and gene interaction analysis. In terms of interpretability, transformer based methods natively provide an additional layer of interpretability due to their attention mechanism which captures gene-gene interactions, and scGPT for example was also applied in the context of learning gene regulatory networks. A recent work benchmarked pretrained foundation models against transformer models without pretraining and simple models like logistic regression (Boiarsky et al. 2023). The authors came to the conclusion that more research is needed to determine the value of foundation models and find an adequate area of application, since many tasks can also be addressed with simpler models. Thus, we believe that, as method development in the field progresses, both, VAE and transformer based models, will still be used and further developed. We also envision overlaps, as in principle, we could also combine the heart of our models, which is the interpretable decoder part, with a transformer based encoder.

References

10 Aibar, Sara, Carmen Bravo González-Blas, Thomas Moerman, Vân Anh Huynh-Thu, Hana Imrichova, Gert Hulselmans, Florian Rambow, et al. 2017. "SCENIC: Single-Cell Regulatory Network Inference and Clustering." *Nature Methods* 14 (11): 1083–86. https://doi.org/10.1038/nmeth.4463.

Angermueller, Christof, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. 2016. "Deep Learning for Computational Biology." *Molecular Systems Biology* 12 (7): 878. https://doi.org/https://doi.org/10.15252/msb.20156651.

Ashburner, Michael, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, et al. 2000. "Gene Ontology: Tool for the Unification of Biology." *Nature Genetics* 25 (1): 25–29. https://doi.org/10.1038/75556.

Bach, Alexander AND Montavon, Sebastian AND Binder. 2015. "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation." *PLOS ONE* 10 (7): 1–46. https://doi.org/10.1371/journal.pone.0130140.

Barton, Joshua R, Annie K Londregan, Tyler D Alexander, Ariana A Entezari, Manuel Covarrubias, and Scott A Waldman. 2023. "Enteroendocrine Cell Regulation of the Gut-Brain Axis." *Front Neurosci* 17: 1272955. https://doi.org/10.3389/fnins.2023.1272955.

Bassett, Anne S., and Eva W. C. Chow. 2008. "Schizophrenia and 22q11.2 Deletion Syndrome." *Current Psychiatry Reports* 10 (2): 148–57. https://doi.org/10.1007/s11920-008-0026-1.

Bhatlekar, Seema, Jeremy Z Fields, and Bruce M Boman. 2018. "Role of HOX Genes in Stem Cell Differentiation and Cancer." *Stem Cells Int.* 2018 (July): 1–15.

Birnbaum, Rebecca, and Daniel R. Weinberger. 2017. "Genetic Insights into the Neurodevelopmental Origins of Schizophrenia." *Nature Reviews Neuroscience* 18 (12): 727–40. https://doi.org/10.1038/nrn.2017.125.

Blei, David M., Alp Kucukelbir, and Jon D. McAuliffe. 2017. "Variational Inference: A Review for Statisticians." *Journal of the American Statistical Association* 112 (518): 859–77. https://doi.org/10.1080/01621459.2017.1285773.

Bock, Christoph, Paul Datlinger, Florence Chardon, Matthew A. Coelho, Matthew B. Dong, Keith A. Lawson, Tian Lu, et al. 2022. "High-Content CRISPR Screening." *Nature Reviews Methods Primers* 2 (1): 8. https://doi.org/10.1038/s43586-021-00093-4.

Boiarsky, Rebecca, Nalini Singh, Alejandro Buendia, Gad Getz, and David Sontag. 2023. "A Deep Dive into Single-Cell RNA Sequencing Foundation Models." *bioRxiv*.

Bourgeais, Victoria, Farida Zehraoui, Mohamed Ben Hamdoune, and Blaise Hanczar. 2021. "Deep GONet: Self-Explainable Deep Neural Network Based on Gene Ontology for Phenotype Prediction from Gene Expression Data." *BMC Bioinformatics* 22 (10): 455. https://doi.org/10.1186/s12859-021-04370-7.

Bourgeais, Victoria, Farida Zehraoui, and Blaise Hanczar. 2022. "GraphGONet: a self-explaining neural network encapsulating the Gene Ontology graph for phenotype prediction on gene expression." *Bioinformatics* 38 (9): 2504–11. https://doi.org/10.1093/bioinformatics/btac147.

Bourlard, H., and Y. Kamp. 1988. "Auto-Association by Multilayer Perceptrons and Singular Value Decomposition." *Biological Cybernetics* 59 (4): 291–94. https://

doi.org/10.1007/BF00332918.

Brown, Alan S, and Elena J Derkits. 2010. "Prenatal Infection and Schizophrenia: A Review of Epidemiologic and Translational Studies." *Am J Psychiatry* 167 (3): 261–80. https://doi.org/10.1176/appi.ajp.2009.09030361.

Caballano-Infantes, Estefania, Gladys Margot Cahuana, Francisco Javier Bedoya, Carmen Salguero-Aranda, and Juan R Tejedo. 2022. "The Role of Nitric Oxide in Stem Cell Biology." *Antioxidants (Basel)* 11 (3). https://doi.org/10.3390/antiox11030497.

Chai, Junyi, Hao Zeng, Anming Li, and Eric W. T. Ngai. 2021. "Deep Learning in Computer Vision: A Critical Review of Emerging Techniques and Application Scenarios." *Machine Learning with Applications* 6: 100134. https://doi.org/https://doi.org/10.1016/j.mlwa.2021.100134.

Choi, Yongin, Ruoxin Li, and Gerald Quon. 2023. "siVAE: Interpretable Deep Generative Models for Single-Cell Transcriptomes." *Genome Biology* 24 (1): 29. https://doi.org/10.1186/s13059-023-02850-y.

Cui, Haotian, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. 2024. "scGPT: Toward Building a Foundation Model for Single-Cell Multi-Omics Using Generative AI." *Nature Methods*. https://doi.org/10.1038/s41592-024-02201-0.

De Donno, Carlo, Soroor Hediyeh-Zadeh, Amir Ali Moinfar, Marco Wagenstetter, Luke Zappia, Mohammad Lotfollahi, and Fabian J. Theis. 2023. "Population-Level Integration of Single-Cell Datasets Enables Multi-Scale Analysis Across Samples." *Nature Methods* 20 (11): 1683–92. https://doi.org/10.1038/s41592-023-02035-2.

Depuydt, Christophe E., Veerle Goosens, Rekin's Janky, Ann D'Hondt, Jan L. De Bleecker, Nathalie Noppe, Stefaan Derveaux, Dietmar R. Thal, and Kristl G. Claeys. 2022. "Unraveling the Molecular Basis of the Dystrophic Process in Limb-Girdle Muscular Dystrophy LGMD-R12 by Differential Gene Expression Profiles in Diseased and

Healthy Muscles." *Cells* 11 (9). https://doi.org/10.3390/cells11091508.

Doncevic, Daria, and Carl Herrmann. 2023. "Biologically Informed Variational Autoencoders Allow Predictive Modeling of Genetic and Drug-Induced Perturbations." Edited by Pier Luigi Martelli. *Bioinformatics* 39 (6). https://doi.org/10.1093/bioinformatics/btad387.

Duan, Dongsheng, Nathalie Goemans, Shin'ichi Takeda, Eugenio Mercuri, and Annemieke Aartsma-Rus. 2021. "Duchenne Muscular Dystrophy." *Nature Reviews Disease Primers* 7 (1): 13. https://doi.org/10.1038/s41572-021-00248-3.

Eraslan, Gökcen, Lukas M. Simon, Maria Mircea, Nikola S. Mueller, and Fabian J. Theis. 2019. "Single-Cell RNA-Seq Denoising Using a Deep Count Autoencoder." *Nature Communications* 10 (1): 390. https://doi.org/10.1038/s41467-018-07931-2.

Espinoza-Lewis, Ramón A, Ling Yu, Fenglei He, Hongbing Liu, Ruhang Tang, Jiangli Shi, Xiaoxiao Sun, et al. 2009. "Shox2 Is Essential for the Differentiation of Cardiac Pacemaker Cells by Repressing Nkx2-5." *Dev Biol* 327 (2): 376–85. https://doi.org/10.1016/j.ydbio.2008.12.028.

Fortelny, Nikolaus, and Christoph Bock. 2020. "Knowledge-Primed Neural Networks Enable Biologically Interpretable Deep Learning on Single-Cell Sequencing Data." *Genome Biology* 21 (1): 190. https://doi.org/10.1186/s13059-020-02100-5.

Goetz, Laura H, and Nicholas J Schork. 2018. "Personalized Medicine: Motivation, Challenges, and Progress." *Fertil Steril* 109 (6): 952–63. https://doi.org/10.1016/j.fertnstert.2018.05.006.

Goodfellow, Ian J, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. "Generative Adversarial Networks."

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Gulrajani, Ishaan, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. 2017. "Improved Training of Wasserstein GANs."

Hao, Minsheng, Jing Gong, Xin Zeng, Chiming Liu, Yucheng Guo, Xingyi Cheng, Taifeng Wang, Jianzhu Ma, Xuegong Zhang, and Le Song. 2024. "Large-Scale Foundation Model on Single-Cell Transcriptomics." *Nature Methods*. https://doi.org/10.1038/s41592-024-02305-7.

Hasin, Yehudit, Marcus Seldin, and Aldons Lusis. 2017. "Multi-Omics Approaches to Disease." *Genome Biology* 18 (1). https://doi.org/10.1186/s13059-017-1215-1.

Hinton, Geoffrey E, and Richard Zemel. 1993. "Autoencoders, Minimum Description Length and Helmholtz Free Energy." In *Advances in Neural Information Processing Systems*, edited by J. Cowan, G. Tesauro, and J. Alspector. Vol. 6. Morgan-Kaufmann. https://proceedings.neurips.cc/paper/1993/file/9e3cfc48eccf81a0d57663e129aef3
pdf.

Hudson, C D, A E Sayan, G. Melino, R A Knight, D S Latchman, and V. Budhram-Mahadeo. 2008. "Brn-3a/POU4F1 Interacts with and Differentially Affects P73-Mediated Transcription." *Cell Death & Differentiation* 15 (8): 1266–78. https://doi.org/10.1038/cdd.2008.45.

Hwang, Jee-Yeon, and R Suzanne Zukin. 2018. "REST, a Master Transcriptional Regulator in Neurodegenerative Disease." *Curr Opin Neurobiol* 48 (February): 193–200. https://doi.org/10.1016/j.conb.2017.12.008.

Insel, Thomas R. 2010. "Rethinking Schizophrenia." *Nature* 468 (7321): 187–93. https://doi.org/10.1038/nature09552.

Jaenisch, Rudolf, and Adrian Bird. 2003. "Epigenetic Regulation of Gene Expression: How the Genome Integrates Intrinsic and Environmental Signals." *Nature Genetics* 33 (3): 245–54. https://doi.org/10.1038/ng1089.

Jansky, Selina, Ashwini Kumar Sharma, Verena Körber, Andrés Quintero, Umut H.

Toprak, Elisa M. Wecht, Moritz Gartlgruber, et al. 2021. "Single-Cell Transcriptomic Analyses Provide Insights into the Developmental Origins of Neuroblastoma." *Nature Genetics* 53 (5): 683–93. https://doi.org/10.1038/s41588-021-00806-1.

Ji, Yuge, Mohammad Lotfollahi, F. Alexander Wolf, and Fabian J. Theis. 2021. "Machine Learning for Perturbational Single-Cell Omics." *Cell Systems* 12 (6): 522–37. https://doi.org/10.1016/j.cels.2021.05.016.

Jovic, Dragomirka, Xue Liang, Hua Zeng, Lin Lin, Fengping Xu, and Yonglun Luo. 2022. "Single-Cell RNA Sequencing Technologies and Applications: A Brief Overview." *Clin Transl Med* 12 (3): e694. https://doi.org/10.1002/ctm2.694.

Kang, Hyun Min, Meena Subramaniam, Sasha Targ, Michelle Nguyen, Lenka Maliskova, Elizabeth McCarthy, Eunice Wan, et al. 2018. "Multiplexed Droplet Single-Cell RNA-Sequencing Using Natural Genetic Variation." *Nature Biotechnology* 36 (1): 89–94. https://doi.org/10.1038/nbt.4042.

Kingma, Diederik P., and Max Welling. 2019. "An Introduction to Variational Autoencoders." *Foundations and Trends® in Machine Learning* 12 (4): 307–92. https://doi.org/10.1561/2200000056.

Kingma, Diederik P, and Max Welling. 2013. "Auto-Encoding Variational Bayes." https://arxiv.org/abs/1312.6114.

Kunugi, H., S. Nanko, and R. M. Murray. 2001. "Obstetric Complications and Schizophrenia: Prenatal Underdevelopment and Subsequent Neurodevelopmental Impairment." *The British Journal of Psychiatry* 178 (S40): s25–29. https://doi.org/10.1192/bjp.178.40.s25.

Le Page, C, P Génin, M G Baines, and J Hiscott. 2000. "Interferon Activation and Innate Immunity." *Rev Immunogenet* 2 (3): 374–86.

Lee, Aram, Jihyun Lim, and Jong-Seok Lim. 2024. "Emerging Roles of MITF as a Crucial Regulator of Immunity." *Experimental & Molecular Medicine* 56 (2): 311–18.

https://doi.org/10.1038/s12276-024-01175-5.

Lewis, D A, and J A Lieberman. 2000. "Catching up on Schizophrenia: Natural History and Neurobiology." *Neuron* 28 (2): 325–34. https://doi.org/10.1016/s0896-6273(00)00111-2.

Li, Peng, Gretchen L Snyder, and Kimberly E Vanover. 2016. "Dopamine Targeting Drugs for the Treatment of Schizophrenia: Past, Present and Future." *Curr Top Med Chem* 16 (29): 3385–3403. https://doi.org/10.2174/1568026616666160608084834.

Li, Xuhong, Haoyi Xiong, Xingjian Li, Xuanyu Wu, Xiao Zhang, Ji Liu, Jiang Bian, and Dejing Dou. 2022. "Interpretable Deep Learning: Interpretation, Interpretability, Trustworthiness, and Beyond." *Knowledge and Information Systems* 64 (12): 3197–3234. https://doi.org/10.1007/s10115-022-01756-8.

Lightbody, Gaye, Valeriia Haberland, Fiona Browne, Laura Taggart, Huiru Zheng, Eileen Parkes, and Jaine K Blayney. 2019. "Review of applications of high-throughput sequencing in personalized medicine: barriers and facilitators of future progress in research and clinical application." *Briefings in Bioinformatics* 20 (5): 1795–1811. https://doi.org/10.1093/bib/bby051.

Linardatos, Pantelis, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. 2020. "Explainable AI: A Review of Machine Learning Interpretability Methods." *Entropy (Basel)* 23 (1). https://doi.org/10.3390/e23010018.

Liu, Xiaosong, Jinyan Huang, Taotao Chen, Ying Wang, Shunmei Xin, Jian Li, Gang Pei, and Jiuhong Kang. 2008. "Yamanaka Factors Critically Regulate the Developmental Signaling Network in Mouse Embryonic Stem Cells." *Cell Research* 18 (12): 1177–89. https://doi.org/10.1038/cr.2008.309.

Lopez, Romain, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. 2018. "Deep Generative Modeling for Single-Cell Transcriptomics." *Nat Methods* 15 (12): 1053–58. https://doi.org/10.1038/s41592-018-0229-2.

Lotfollahi, Mohammad, Anna Klimovskaia Susmelj, Carlo De Donno, Leon Hetzel, Yuge Ji, Ignacio L Ibarra, Sanjay R Srivatsan, et al. 2023. "Predicting Cellular Responses to Complex Perturbations in High-throughput Screens." *Molecular Systems Biology* 19 (6): e11517. https://doi.org/https://doi.org/10.15252/msb.202211517.

Lotfollahi, Mohammad, Mohsen Naghipourfar, Malte D. Luecken, Matin Khajavi, Maren Büttner, Marco Wagenstetter, Žiga Avsec, et al. 2022. "Mapping Single-Cell Data to Reference Atlases by Transfer Learning." *Nature Biotechnology* 40 (1): 121–30. https://doi.org/10.1038/s41587-021-01001-7.

Lotfollahi, Mohammad, Sergei Rybakov, Karin Hrovatin, Soroor Hediyeh-zadeh, Carlos Talavera-López, Alexander V. Misharin, and Fabian J. Theis. 2023. "Biologically Informed Deep Learning to Query Gene Programs in Single-Cell Atlases." *Nature Cell Biology* 25 (2): 337–50. https://doi.org/10.1038/s41556-022-01072-x.

Lotfollahi, Mohammad, F Alexander Wolf, and Fabian J Theis. 2019. "scGen Predicts Single-Cell Perturbation Responses." *Nat Methods* 16 (8): 715–21. https://doi.org/10.1038/s41592-019-0494-8.

Lu, Chenyue, Di Jin, Nathan Palmer, Kathe Fox, Isaac S. Kohane, Jordan W. Smoller, and Kun-Hsing Yu. 2022. "Large-Scale Real-World Data Analysis Identifies Comorbidity Patterns in Schizophrenia." *Translational Psychiatry* 12 (1): 154. https://doi.org/10.1038/s41398-022-01916-y.

Luecken, Malte D., M. Büttner, K. Chaichoompu, A. Danese, M. Interlandi, M. F. Mueller, D. C. Strobl, et al. 2022. "Benchmarking Atlas-Level Data Integration in Single-Cell Genomics." *Nature Methods* 19 (1): 41–50. https://doi.org/10.1038/s41592-021-01336-8.

Lundberg, Scott, and Su-In Lee. 2017. "A Unified Approach to Interpreting Model Predictions." https://arxiv.org/abs/1705.07874.

Ma, Jianzhu, Michael Ku Yu, Samson Fong, Keiichiro Ono, Eric Sage, Barry Dem-

chak, Roded Sharan, and Trey Ideker. 2018. "Using Deep Learning to Model the Hierarchical Structure and Function of a Cell." *Nature Methods* 15 (4): 290–98. https://doi.org/10.1038/nmeth.4627.

Mangurian, Christina, John W Newcomer, Chelsea Modlin, and Dean Schillinger. 2016. "Diabetes and Cardiovascular Care Among People with Severe Mental Illness: A Literature Review." *J Gen Intern Med* 31 (9): 1083–91. https://doi.org/10.1007/s11606-016-3712-4.

Melsted, Páll, A. Sina Booeshaghi, Lauren Liu, Fan Gao, Lambda Lu, Kyung Hoi (Joseph) Min, Eduardo da Veiga Beltrame, Kristján Eldjárn Hjörleifsson, Jase Gehring, and Lior Pachter. 2021. "Modular, Efficient and Constant-Memory Single-Cell RNA-Seq Preprocessing." *Nature Biotechnology* 39 (7): 813–18. https://doi.org/10.1038/s41587-021-00870-2.

Michelucci, Umberto. 2022. "An Introduction to Autoencoders." arXiv. https://doi.org/10.48550/ARXIV.2201.03898.

Muckenhuber, Markus, Isabelle Seufert, Katharina Müller-Ott, Jan-Philipp Mallm, Lara C Klett, Caroline Knotz, Jana Hechler, Nick Kepper, Fabian Erdel, and Karsten Rippe. 2023. "Epigenetic Signals That Direct Cell Type-Specific Interferon Beta Response in Mouse Cells." *Life Sci Alliance* 6 (4). https://doi.org/10.26508/lsa.202201823.

Müller-Dott, Sophia, Eirini Tsirvouli, Miguel Vazquez, Ricardo O Ramirez Flores, Pau Badia-i-Mompel, Robin Fallegger, Dénes Türei, Astrid Lægreid, and Julio Saez-Rodriguez. 2023. "Expanding the Coverage of Regulons from High-Confidence Prior Knowledge for Accurate Estimation of Transcription Factor Activities." *Nucleic Acids Research* 51 (20): 10934–49. https://doi.org/10.1093/nar/gkad841.

Musella, Martina, Andrea Guarracino, Nicoletta Manduca, Claudia Galassi, Eliana Ruggiero, Alessia Potenza, Ester Maccafeo, et al. 2022. "Type i IFNs Promote Cancer Cell Stemness by Triggering the Epigenetic Regulator KDM1B." *Nature Immunology*

23 (9): 1379–92. https://doi.org/10.1038/s41590-022-01290-3.

Nelson, Matthew R, Hannah Tipney, Jeffery L Painter, Judong Shen, Paola Nicoletti, Yufeng Shen, Aris Floratos, et al. 2015. "The Support of Human Genetic Evidence for Approved Drug Indications." *Nature Genetics* 47 (8): 856–60. https://doi.org/10.1038/ng.3314.

Nishioka, Masaki, Miki Bundo, Kiyoto Kasai, and Kazuya Iwamoto. 2012. "DNA Methylation in Schizophrenia: Progress and Challenges of Epigenetic Studies." *Genome Med* 4 (12): 96. https://doi.org/10.1186/gm397.

Odaibo, Stephen G. 2019. "Tutorial: Deriving the Standard Variational Autoencoder (VAE) Loss Function." *CoRR* abs/1907.08956. http://arxiv.org/abs/1907.08956.

Os, Jim van, Gunter Kenis, and Bart P. F. Rutten. 2010. "The Environment and Schizophrenia." *Nature* 468 (7321): 203–12. https://doi.org/10.1038/nature09563.

Pezzè, Laura, Erna Marija Meškytė, Mattia Forcato, Stefano Pontalti, Kalina Aleksandra Badowska, Dario Rizzotto, Ira-Ida Skvortsova, Silvio Bicciato, and Yari Ciribilli. 2021. "ETV7 Regulates Breast Cancer Stem-Like Cell Features by Repressing IFN-Response Genes." *Cell Death & Disease* 12 (8): 742. https://doi.org/10.1038/s41419-021-04005-y.

Picchioni, Marco M, and Robin M Murray. 2007. "Schizophrenia." *BMJ* 335 (7610): 91–95. https://doi.org/10.1136/bmj.39227.616447.BE.

Prince, Martin, Vikram Patel, Shekhar Saxena, Mario Maj, Joanna Maselko, Michael R Phillips, and Atif Rahman. 2007. "No Health Without Mental Health." *Lancet* 370 (9590): 859–77. https://doi.org/10.1016/S0140-6736(07)61238-0.

Pundhir, Sachin, Felicia Kathrine Bratt Lauridsen, Mikkel Bruhn Schuster, Janus Schou Jakobsen, Ying Ge, Erwin Marten Schoof, Nicolas Rapin, Johannes Waage, Marie Sigurd Hasemann, and Bo Torben Porse. 2018. "Enhancer and Transcription Factor Dynamics During Myeloid Differentiation Reveal an Early Differentiation Block in

Cebpa Null Progenitors." *Cell Rep* 23 (9): 2744–57. https://doi.org/10.1016/j.celrep.2018.05.012.

Qiu, Peng. 2020. "Embracing the Dropouts in Single-Cell RNA-Seq Analysis." *Nature Communications* 11 (1): 1169. https://doi.org/10.1038/s41467-020-14976-9.

Regev, Aviv, Sarah A Teichmann, Eric S Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, et al. 2017. "The Human Cell Atlas." *Elife* 6 (December). https://doi.org/10.7554/eLife.27041.

Reuter, Jason A, Damek V Spacek, and Michael P Snyder. 2015. "High-Throughput Sequencing Technologies." *Mol. Cell* 58 (4): 586–97.

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. 2016. ""Why Should i Trust You?": Explaining the Predictions of Any Classifier." https://arxiv.org/abs/1602.04938.

Ripke, Stephan, Benjamin M. Neale, Aiden Corvin, James T. R. Walters, Kai-How Farh, Peter A. Holmans, Phil Lee, et al. 2014. "Biological Insights from 108 Schizophrenia-Associated Genetic Loci." *Nature* 511 (7510): 421–27. https://doi.org/10.1038/nature13595.

Roohani, Yusuf, Kexin Huang, and Jure Leskovec. 2022. "GEARS: Predicting Transcriptional Outcomes of Novel Multi-Gene Perturbations." *bioRxiv*. https://doi.org/10.1101/2022.07.12.499735.

———. 2024. "Predicting Transcriptional Outcomes of Novel Multigene Perturbations with GEARS." *Nature Biotechnology* 42 (6): 927–35. https://doi.org/10.1038/s41587-023-01905-6.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. 1986. "Learning Internal Representations by Error Propagation." In.

Ruzicka, W. Brad, Shahin Mohammadi, John F. Fullard, Jose Davila-Velderrain, Sivan

Subburaju, Daniel Reed Tso, Makayla Hourihan, et al. 2024. "Single-Cell Multi-Cohort Dissection of the Schizophrenia Transcriptome." *Science* 384 (6698): eadg5136. https://doi.org/10.1126/science.adg5136.

Rybakov, Sergei, Mohammad Lotfollahi, Fabian J. Theis, and F. Alexander Wolf. 2020. "Learning Interpretable Latent Autoencoder Representations with Annotations of Feature Sets." *bioRxiv*. https://doi.org/10.1101/2020.12.02.401182.

Schoggins, John W. 2019. "Interferon-Stimulated Genes: What Do They All Do?" *Annual Review of Virology* 6 (1): 567–84. https://doi.org/10.1146/annurev-virology-092818-015756.

Schultz, Stephen H, Stephen W North, and Cleveland G Shields. 2007. "Schizophrenia: A Review." *Am Fam Physician* 75 (12): 1821–29.

Seninge, Lucas, Ioannis Anastopoulos, Hongxu Ding, and Joshua Stuart. 2021. "VEGA Is an Interpretable Generative Model for Inferring Biological Network Activity in Single-Cell Transcriptomics." *Nature Communications* 12 (1): 5684. https://doi.org/10.1038/s41467-021-26017-0.

Shrikumar, Avanti, Peyton Greenside, and Anshul Kundaje. 2019. "Learning Important Features Through Propagating Activation Differences." https://arxiv.org/abs/1704.02685.

Stamatiadis, P, G Cosemans, A Boel, B Menten, P De Sutter, D Stoop, S M Chuva de Sousa Lopes, F Lluis, P Coucke, and B Heindryckx. 2022. "TEAD4 Regulates Trophectoderm Differentiation Upstream of CDX2 in a GATA3-Independent Manner in the Human Preimplantation Embryo." *Hum Reprod* 37 (8): 1760–73. https://doi.org/10.1093/humrep/deac138.

Stuart, Tim, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M Mauck 3rd, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. 2019. "Comprehensive Integration of Single-Cell Data." *Cell* 177 (7): 1888–1902.e21.

Svensson, Valentine, Adam Gayoso, Nir Yosef, and Lior Pachter. 2020. "Interpretable factor models of single-cell RNA-seq via variational autoencoders." *Bioinformatics* 36 (11): 3418–21. https://doi.org/10.1093/bioinformatics/btaa169.

Triana, Sergio, Megan L Stanifer, Camila Metz-Zumaran, Mohammed Shahraz, Markus Mukenhirn, Carmon Kee, Clara Serger, et al. 2021. "Single-Cell Transcriptomics Reveals Immune Response of Intestinal Cell Types to Viral Infection." *Mol Syst Biol* 17 (7): e9833. https://doi.org/10.15252/msb.20209833.

Wamaitha, Sissy E, Ignacio del Valle, Lily T Y Cho, Yingying Wei, Norah M E Fogarty, Paul Blakeley, Richard I Sherwood, Hongkai Ji, and Kathy K Niakan. 2015. "Gata6 Potently Initiates Reprograming of Pluripotent and Differentiated Cells to Extraembryonic Endoderm Stem Cells." *Genes Dev.* 29 (12): 1239–55.

Wang, Jingshu, Divyansh Agarwal, Mo Huang, Gang Hu, Zilu Zhou, Chengzhong Ye, and Nancy R. Zhang. 2019. "Data Denoising with Transfer Learning in Single-Cell Transcriptomics." *Nature Methods* 16 (9): 875–78. https://doi.org/10.1038/s41592-019-0537-1.

Wang, Zhong, Mark Gerstein, and Michael Snyder. 2009. "RNA-Seq: A Revolutionary Tool for Transcriptomics." *Nat Rev Genet* 10 (1): 57–63. https://doi.org/10.1038/nrg2484.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, et al. 2016. "The FAIR Guiding Principles for Scientific Data Management and Stewardship." *Scientific Data* 3 (1): 160018. https://doi.org/10.1038/sdata.2016.18.

Worthington, J J, F. Reimann, and F M Gribble. 2018. "Enteroendocrine Cells-Sensory Sentinels of the Intestinal Environment and Orchestrators of Mucosal Immunity." *Mucosal Immunology* 11 (1): 3–20. https://doi.org/10.1038/mi.2017.73.

Xu, Chenling, Romain Lopez, Edouard Mehlman, Jeffrey Regier, Michael I Jordan, and

Nir Yosef. 2021. “Probabilistic Harmonization and Annotation of Single-cell Transcriptomics Data with Deep Generative Models.” *Molecular Systems Biology* 17 (1): e9620. https://doi.org/https://doi.org/10.15252/msb.20209620.

Yang, Karren Dai, Anastasiya Belyaeva, Saradha Venkatachalapathy, Karthik Damodaran, Abigail Katcoff, Adityanarayanan Radhakrishnan, G. V. Shivashankar, and Caroline Uhler. 2021. “Multi-Domain Translation Between Single-Cell Imaging and Sequencing Data Using Autoencoders.” *Nature Communications* 12 (1): 31. https://doi.org/10.1038/s41467-020-20249-2.

Zhang, Wei, Yi Sui, Jun Ni, and Tao Yang. 2016. “Insights into the Nanog Gene: A Propeller for Stemness in Primitive Stem Cells.” *Int J Biol Sci* 12 (11): 1372–81. https://doi.org/10.7150/ijbs.16349.

Zhang, Xiaoyu, Jingqing Zhang, Kai Sun, Xian Yang, Chengliang Dai, and Yike Guo. 2019. “Integrated Multi-Omics Analysis Using Variational Autoencoders: Application to Pan-Cancer Classification.”

Appendix A

Derivation of Kullback-Leibler loss term

According to Bayes rule, it applies for a set of observed variables x and a set of latent variables z:

$$p(z/x) = \frac{p(x/z)p(z)}{p(x)} \tag{A.1}$$

where $p(z/x)$ is the *posterior*, $p(x/z)$ is the *likelihood*, $p(z)$ is the *prior*, and $p(x)$ is the *marginal* or *evidence*. The evidence is calculated by marginalizing out the latent variables z from the joint density.

$$p(x) = \int p(x,z)d_z \tag{A.2}$$

Since there is no closed-form solution for the evidence integral and computation time scales exponentially, for most models it is intractable. Thus, we cannot compute the posterior directly, but have to find an approximation $q(z) \approx p(z/x)$ which is as good as possible. For this, the Kullback-Leibler (KL) divergence has to be minimized, which is defined as the distance between two distributions and is given by:

$$\begin{aligned} KL(q(z)//p(z/x)) &= \mathbb{E}_{z\ q(z)}[log\frac{q(z)}{p(z/x)}] \\ &= \int_{z_0} ... \int_{z_{D-1}} log\frac{q(z)}{p(z/x)} d_{z_0} ... d_{z_{D-1}} \end{aligned} \tag{A.3}$$

Again, in **equation** (A.3), the posterior $p(z/x)$ cannot be computed directly, thus, we have to rearrange the terms as follows:

$$\begin{aligned} KL(q(z)//p(z/x)) &= \int_z q(z) log(\frac{q(z)p(x)}{p(z,x)}) d_z \\ &= \int_z q(z) log(\frac{q(z)}{p(z,x)}) d_z + \int_z q(z) log(p(x)) d_z \\ &= \mathbb{E}_{z\ q(z)}[log(\frac{q(z)}{p(z,x)})] + \mathbb{E}_{z\ q(z)}[logp(x)] \\ &= -\mathbb{E}_{z\ q(z)}[log(\frac{p(z,x)}{q(z)})] + logp(x) \end{aligned} \tag{A.4}$$

The first term of the final rearranged **equation** (A.4) is also called the *evidence lower bound* (ELBO).

$$ELBO\mathcal{L}(q) = \mathbb{E}_{z\ q(z)}[log(\frac{p(z,x)}{q(z)})] \tag{A.5}$$

Thus, we can simplify **equation** (A.4) as follows:

$$KL = -\mathcal{L}(q) + logp(x) \tag{A.6}$$

Since the KL divergence is a distance measure, it applies $KL \geq 0$. The value of a probability lies between 0 and 1, thus it also applies: $logp(x) \leq 0$, and $\mathcal{L}(q) \leq 0$. From this, it follows that $\mathcal{L}(q) \leq logp(x)$ and this is why $\mathcal{L}(q)$ is called the *evidence lower bound*. Thus, from **equation** (A.6) we can see that, in order to minimize the KL divergence, the ELBO needs to be maximized. This approach is taken by the VAE and by other variational inference methods.

Appendix B

Model initialization and training parameters

Table B.1: Tunable parameters for OntoVAE model initialization in cobra-ai.

Parameter	Type	Default	Structure	Description
adata	AnnData	-	-	Anndata that was prepared with setup_anndata_ontovae function
use_batch_norm_enc	bool	True	Encoder	whether to use batch norm
use_layer_norm_enc	bool	False	Encoder	whether to use layer norm
use_activation_enc	bool	True	Encoder	whether to use activation functions
activation_fn_enc	nn.Module	nn.ReLU	Encoder	which activation function to use
bias_enc	bool	True	Encoder	whether to learn bias for linear layers
hidden_layers_enc	int	3	Encoder	how many hidden layers to use
inject_covariates_enc	bool	False	Encoder	whether covariate information should be injected into each layer
drop_enc	float	0.2	Encoder	dropout rate
z_drop	float	0.5	latent space	dropout rate
root_layer_latent	bool	False	latent space	if ontology should start in latent space
latent_dim	int	128	latent space	latent space dimension
neuronnum	int	3	Decoder	number of neurons per term
use_batch_norm_dec	bool	True	Decoder	whether to use batch norm
use_layer_norm_dec	bool	False	Decoder	whether to use layer norm
use_activation_dec	bool	True	Decoder	whether to use activation functions
activation_fn_dec	nn.Module	nn.ReLU	Decoder	which activation function to use
rec_activation	nn.Module	None	Decoder	which activation function to use on reconstruction layer
bias_dec	bool	True	Decoder	whether to learn bias for linear layers
inject_covariates_dec	bool	False	Decoder	whether covariate information should be injected into each layer
drop_dec	float	0	Decoder	dropout rate

Table B.2: Additional tunable parameters for COBRA model initialization.

Parameter	Type	Default	Structure	Description
hidden_layers_class	int	2	Classifier	number of hidden layers
neurons_per_class_layer	int	64	Classifier	number of neurons per layer
use_batch_norm_class	bool	True	Classifier	whether to use batch norm
use_layer_norm_class	bool	False	Classifier	whether to use layer norm
use_activation_class	bool	True	Classifier	whether to use activation functions
activation_fn_class	nn.Module	nn.ReLU	Classifier	which activation function to use
bias_class	bool	True	Classifier	whether to learn bias for linear layers
inject_covariates_class	bool	False	Classifier	whether covariate information should be injected into each layer
drop_class	float	0.2	Classifier	dropout rate
average_neurons	bool	False	Classifier	if neuronnum averaging is to be performed before input in classifier

Table B.3: Tunable parameters for OntoVAE model training in cobra-ai.

Parameter	**Type**	**Default**	**Description**
modelpath	str	-	where to save trained model
save	bool	True	whether to save trained model
train_size	float	0.9	fraction of samples to use for training
seed	int	42	seed for the train/val split
lr	float	1e-4	learning rate
kl_coeff	float	1e-4	KL weighting coefficient
batch_size	int	128	minibatch size
optimizer	Optimizer	AdamW	optimizer
pos_weights	bool	True	whether to make decoder weights positive
epochs	int	300	number of training epochs
early_stopping	bool	True	whether to use early stopping
patience	int	10	after how many epochs early stopping applies
run	Neptune.run	None	run for logging with Neptune

Table B.4: Additional tunable parameters for COBRA model training.

Parameter	**Type**	**Default**	**Description**
lr_vae	float	1e-4	VAE learning rate
lr_adv	float	1e-3	adversarial learning rate
adv_coeff	float	1e3	adversarial loss weighting coefficient
pen_coeff	float	2.0	gradient penalty weighting coefficient
adv_step	int	1	after how many epochs adversarial training starts