Université Paris Cité

École doctorale EDITE - ED130
Institut Pasteur, Machine Learning for Integrative Genomics (ML4IG)
ENS ULM, Department of Applied Mathematics, Centre Science des Données (CSD)

Transformers on single-cell RNA-sequencing data as large cell models

Par Jérémie KALFON

Thèse de doctorat d'Intelligence Artificielle

Dirigée par Laura CANTINI
Et par Gabriel Peyré

Présentée et soutenue publiquement le 15 Décembre 2025

Devant un jury composé de :

AAAA CANTINI, HDR	CNRS, France	Rapporteur
Prénom NOM, DR	Université de Ville	Rapportrice
Prénom NOM, CR-HDR	Université Paris Cité	Examinatrice
Prénom NOM, DR	Université de Y	Examinateur
Prénom NOM, MCF	Université Paris Cité	Examinatrice
Prénom NOM, PU-PH	Université Paris Cité	Membre invité
Prénom NOM, PhD	Entreprise X	Membre invité
Laura CANTINI, HDR	CNRS	Directrice de thèse
Gabriel Peyré, HDR	CNRS	Directeur de thèse

Résumé

Titre : Modèles d'Intelligence Artificielle sur la génétique à cellule unique comme modèle de la cellule.

Mots clefs : scRNA-seq ; Foundation-model ; Interprétation ; Intégration de données ;

Abstract

Title : Single Cell Foundation Models as Large Cell Models.

Keywords : sc-RNA-seq ; Foundation-model ; Transformers ; Virtual Cell ; Network Inference ; AI ;

Abstract : Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Essentially, all models are wrong,
but some are useful

George E. P. Box - 1976

A Ph.D. is not a scprint,
it is a marathon

Someone

Remerciements

Je tiens à remercier chaleureusement mes collègues du laboratoire ML4IG de l'institut Pasteur ainsi que ceux du centre de science des données de l'ENS et tout particulièrement Jules Samaran, Remi Trimbour et Geert Huizing, pour leur accueil et leur aide durant cette thèse.

Je veux aussi bien sûr remercier Laura Cantini et Gabriel Peyré, mes co-encadrants, pour leur soutien et leur disponibilité durant ces années.

Je remercie ma compagne, Juliette Hirsch, pour avoir toujours été à mes côtés ainsi que pour ses conseils, ses questions et son écoute constante.

Je remercie Patrice, Johanna, Marie-Astrid et Emmanuel Kalfon, ainsi que Monique Obineau, Genevieve et Henri Spanjersberg, sans qui je n'en serais pas ici aujourd'hui et qui m'ont conduit, par leur confiance, à faire ce travail.

Je remercie Alex Wolf, Sergei Ribakov, Brice Rafestin et beaucoup d'autres pour leur aide dans le développement informatique des méthodes que je présente ici.

Je remercie les membres de l'institut Pasteur, du département de biologie computationnelle, le hub de bio-informatique, du département de Mathématiques appliquées de l'ENS, ainsi que ceux du supercalculateur Jean Zay sans qui ces méthodes n'auraient pu être entraînées.

Je remercie mes amis, Baptiste Tesson, Oscar Simon, Pier Michele Kaubari, Emile D'allens, Louis Cauquelin, Suzanne Lazarus, Anne-lise Aupetit, Quentin Van Straaten.

Je remercie aussi les membres de Nucleate, Whitelab Genomics, Blossom, dot Omics, Biographica et d'autres pour leur intérêt et soutien durant cette thèse.

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations

AI LLMs ML NN GNs CGTs ASOs

Preamble

This Ph.D. started relatively late in my career. I'd like to spend some of the introduction mentioning the reasons that pushed me to do a Ph.D. now.

0.1 My background for this thesis

Many of the opportunities I had coming out of school have been very exciting. Initially, I decided to create a company called PiPle with a friend, Paul Best, who is now a Post-doctoral Researcher at the University of Vienna in Machine Learning for bio-acoustics.

Funily enough, it was completely unrelated to biology. We worked on creating novel means of communication. We had—and still have—big ideas about how to improve utterly inadequate messaging apps, emails, and similar tools using machine learning and innovative designs. Doing this, we learnt a lot about managing complex projects, selling ideas, building large codebases, teamwork, and designing interfaces.

However, we did not gain enough traction from this, and we felt that after a year of hard work, the road ahead was paved with too many sacrifices.

This is when I passed on Ph.D. opportunities a second time to work at the Broad Institute instead. Having visited the labs, Boston, and Kendall Square, I knew that this was the kind of experience I wanted to have, and Ph.D.s seemed long and cumbersome. At Broad, I worked on many very-high-impact research projects, and I felt I was part of something bigger than myself. I published as the first author and even started my own research projects, which would inform the thesis I am presenting here.

While I still understood that a Ph.D. was the best place to undergo such projects, I was uncertain about the specifics. I also understood the length, harshness, and sometimes arbitrary nature of U.S. Ph.D. programs. I also wanted to continue working on team-based projects and wanted to experience the start-up environment.

Along with some other personal decisions, it led me to return to France and work as the team lead of the computational biology group at Whitelab Genomics in Paris.

In Whitelab, I learned how to build a team and how to manage people. I learned a lot about what it means to grow companies from 10 to 50 people. I also learned about the biotech industry and how to build and sell such products.

Whitelab had a good mix of expertise in computational biology, machine learning, structural biology, and business development. While starting the first project there, I significantly enhanced the potential of foundation models for the biotech industry.

From DNA language models to cell foundation models and knowledge-graph-based models, it became clear that they would be the path forward to aggregate the sparse and disparate information across many fields of biology and medicine.

I was not looking for any other positions and intended to stay at least a few years to assess how we had grown during that time.

However, I was already in contact with Laura Cantini, with whom I had previously discussed Ph.D. projects. At some point, Laura came back to me with this Ph.D. proposal. I spent the good part of a month in a challenging position. Thinking about which decision would not become a regret in the future.

There was no perfect time to do this, but it felt like it was now or never. I was also very impressed by the level of various Ph.D. students in the labs of Laura and Gabriel. Seeing people 4 years younger than me, already having such a high level of expertise and knowledge, was very humbling. Finally, the Ph.D. topic and group were really on point with what I wanted to do. I knew I could do it in way less than 3 years, and Laura was ok with it. But mostly, my work/life environment was welcoming, surrounded by family, friends, and activities. I knew what I wanted to work on and what I wanted to learn.

0.2 Introduction

In this Thesis, we will focus on models of the Cell.

In the mid-17th century, Robert Hooke made a groundbreaking discovery while observing a piece of cork through his microscope. He observed structures that he named “cells" and, as a result, marked the beginning of cellular biology. Cells have since been identified as life’s fundamental structural and functional units, and biologists have endeavored to map the diverse cell types that comprise multicellular organisms. Additionally, they have sought to understand the transient cell states that occur during development, disease progression, and tissue regeneration.

The objectives of cellular biologists are to understand and control, with the dream of engineering life from plants to animals and even generating entirely new synthetic life.

But what for ?

0.2.1 The promises of cellular biology

Before developing a drug for a disease, one must understand the disease and identify a potential target gene or set of target genes. It refers to the genes in specific cell types that need to be reactivated, deactivated, or modified to address the disease’s underlying

mechanism.

But drugs don't have to be small molecules. CAR-T cell therapies have revolutionized blood cancer treatment by modifying a patient's own immune cells to fight the cancer. Similar approaches could be developed for many other conditions. Here, the drug becomes a cell.

But diseases are not the nails one could hit with such a mighty hammer. Indeed, life is everywhere, and engineering has already helped us make better crops, create synthetic meat, and design fungi that remove pollution. When people talk about nanorobots, I urge you to think about engineered cells.

Richard Feynman famously said, "What I cannot create, I do not understand." Therefore, the Modeling of the cell stands as a key milestone in cellular biology, and indeed, one cannot succeed in the aforementioned promises without a correct cellular blueprint.

Therefore, hundreds of companies, from tech to bio, and dozens of institutes are pursuing efforts to create such virtual cellular models. Achieving even limited predictive accuracy would have significant impacts on cellular biology.

We will investigate the primary data modality used to create cell models and how we can train them using modern machine learning methods based on neural networks. We will explore how these models can be useful today and how we might make them better in the future. For that, we will need to understand some essential facts about the cell and how biologists think about it.

0.2.2 GRN and the cell

The cell is the fundamental unit of life and is composed of various components, including proteins, nucleic acids, lipids, and carbohydrates. Each of these components plays a crucial role in the cell's structure and function. Proteins are responsible for most cellular processes, while nucleic acids (DNA and RNA) carry genetic information. Lipids form cell membranes, and carbohydrates serve as energy sources and structural components.

RNA biology is a critical aspect of cellular function, encompassing processes such as transcription, translation, and regulation. Transcription is the process by which DNA is copied into RNA, which then serves as a template for protein synthesis during translation. Regulation of these processes is essential for maintaining cellular homeostasis and responding to environmental changes. This regulation can occur at multiple levels, including transcriptional control, RNA processing, and post-translational modifications.

The RNA hypothesis posits that RNA molecules were the first self-replicating entities, leading to the evolution of life as we know it. This hypothesis suggests that early life forms relied on RNA for both genetic information storage and catalytic functions, paving the way for the development of DNA and proteins, showing how RNA might be one of the most central components of the cell.

Indeed, many different types of RNA exist, each with distinct functions. Messenger RNA

(mRNA) carries genetic information from DNA to ribosomes for protein synthesis, while transfer RNA (tRNA) and ribosomal RNA (rRNA) allow translation of mRNAs into proteins. Other types of RNA, such as small interfering RNA (siRNA) and microRNA (miRNA), are involved in gene regulation and silencing. Long-non-coding RNAs (lncRNAs) also play crucial roles in regulating gene expression and chromatin structure. Unfortunately, many of these RNA types are still poorly understood, and their functions are an active area of research.

Gene expression is the process by which information from a gene is used to synthesize a functional gene product, typically a protein. This process involves several key steps, including transcription, where DNA is transcribed into mRNA, and translation, where mRNA is translated into a protein. Transcription factors (TFs) are proteins that bind to specific DNA sequences to regulate the transcription of genes. They play a crucial role in determining which genes are expressed in a cell at any given time, influencing cellular function and identity.

They also interact with other proteins, such as cohesin, which helps maintain chromatin and the specific 3D structure of the DNA. Chromatin is the complex of DNA and proteins that forms chromosomes within the nucleus of eukaryotic cells. The organization of chromatin is essential for regulating gene expression, as it determines the accessibility of DNA to transcription machinery.

In this context, biologists have relied on the concept of gene regulatory networks (GRNs) to simplify the complex interactions within the cell. GRNs are networks of molecular interactions that govern gene expression levels in a cell. They consist of genes, transcription factors, and other regulatory elements that interact to control the timing and level of gene expression. Although very coarse and likely wrong in many ways, these modeled interactions allow researchers to gain insights into how cells might respond to various stimuli, differentiate into specific cell types, and maintain homeostasis.

Gene networks (GNs) are a more general concept that encompasses not only gene regulatory networks but also other types of interactions, such as protein-protein interactions and metabolic pathways. While GRNs focus specifically on the regulation of gene expression, GNs provide a broader view of the cellular processes and interactions that contribute to the overall function of the cell.

0.2.3 Single-cell genomics

Genomics is the study of an organism's complete set of DNA, including all of its genes. It involves sequencing and analyzing the entire genome to understand its structure, function, and evolution. The development of high-throughput sequencing technologies has revolutionized genomics, allowing researchers to sequence entire genomes quickly and cost-effectively.

Initially, DNA sequencing was performed using Sanger sequencing, the first method developed for this purpose. It involves selectively incorporating chain-terminating dideoxynucleotides during DNA replication, allowing researchers to determine the sequence of

nucleotides in a DNA molecule. This method was labor-intensive and time-consuming, but it laid the foundation for modern sequencing techniques.

Nowadays, we use next-generation sequencing (NGS) technologies, which allow for massively parallel sequencing of millions of DNA fragments—also called reads—simultaneously. This has significantly reduced the time and cost required for genome sequencing, enabling large-scale genomic studies and personalized medicine approaches.

Reads, these small chunks of DNA, often likened to tiny puzzle pieces, are multiplied and sequenced in parallel. The resulting sequences are then aligned (or mapped) to a reference genome, which serves as a template for assembling the reads into a complete genome sequence. The average number of overlapping reads that cover a specific region of the genome is referred to as sequencing depth or coverage. Higher sequencing depth generally leads to more accurate and reliable results, as it reduces the likelihood of errors and increases the confidence in variant detection.

Large-scale sequencing efforts such as the 1 million genomes project, the Human Genome Project, and the 1000 Genomes Project have provided valuable insights into human genetic variation and disease susceptibility. Genetic sequencing now allows us to define the genetic basis of many diseases, identify which drug might work for specific patients, and establish follow-ups for high-risk patients. It is driving more and more clinical decisions and is becoming a standard part of patient care.

But sequencing also allowed many new applications, such as the study of gene expression by sequencing instead the mRNAs in tissues, a process known as RNA sequencing (RNA-seq). The sequencing of DNA states such as methylation (BS-seq), open chromatin (ATAC-seq), and chromatin immunoprecipitation (ChIP-seq) provides a view of how the genome is being read at a point in time. From DNA and its mutations to its state in different contexts and how these lead to changes in RNAs and their states, we begin to develop a holistic view of various cellular mechanisms.

However, these methods only provide a view of the average state of the tissue, not of the individual cells.

In 2014, the first single-cell RNA sequencing (scRNA-seq) methods were developed, allowing researchers to analyze gene expression at the single-cell level. This was a breakthrough in genomics, as it enabled the study of cellular heterogeneity and the identification of rare cell populations that may be missed in bulk RNA-seq analyses.

Since then, single-cell sequencing technologies have rapidly advanced, with new methods being developed to sequence the other omics modalities at the single-cell level. Studies conducted on tens of thousands of cells in the 2010s are now done on millions of cells.

Moreover, the development of spatial transcriptomics and imaging techniques has allowed researchers to study the spatial organization of gene expression within tissues, providing a more comprehensive view of cellular function in its native context, along with cell imaging. Protein measurements are also being developed, unlocking an additional layer of information.

Current applications have been primarily focused on the understanding of diseases and

drug effects within tissues. The technique allowed the identification of hundreds of new cell types and states, and improved the study of cellular development and differentiation. It had a substantial impact on cancer, neurological diseases, and immunology.

Finally, together with the development of perturbation techniques such as CRISPR screens, we can now go beyond observations and start to understand the causal relationships between genes and their functions. Indeed, CRISPR screens allow researchers to systematically deactivate genes in individual cells and observe the resulting changes in gene expression, providing insights into gene function and regulatory networks.

However, single-cell sequencing also comes with its own set of challenges. One of the main issues is the sparsity and noise underlying most of the current single-cell sequencing methods. Secondly, some strong "batch-effect" biases in the data generation process can make it challenging to perform analysis across datasets.

Worse, the dataset themselves have issues. We currently miss many tissues and cell types that are rare and hard to sequence. We have yet to perform such analysis on species other than humans and mice, and specific cell states related to aging, fetal growth, disease, and treatment are missing. It is hoped that machine learning and AI might allow us to solve these issues and infer the missing data computationally.

0.2.4 Current single cell tasks

While these models could be very performant, reliable, and reproducible, benchmarks that align with real use-cases from the user's perspective remain scarce. In single-cell RNA-seq data, a typical pipeline is as follows :

1. It all starts with preprocessing the raw sequencing data : detecting/imputing the cell index, aligning the sequencing reads with a reference genome's gene locations, detecting low-quality cells, reads, doublet events, cell death events, and more. These choices will impact the output dataset as biases.

2. The data might be normalized to correct for sequencing depth and other gene-level biases, and clusters of cells would be defined based on their expression profile similarity. Finally, differential-expression analysis is performed, whereby clusters of cells are compared to identify genes that are differentially expressed between them. This might also be done after the other steps.

3. The dataset might be aligned to a reference atlas in cases where this exists. This is done by using batch-correction methods, often built around nearest neighbor mapping, matrix factorization, or Neural Networks (NN) called Variational Auto-Encoders (VAEs), which are the current state of the art in the domain.

4. Cluster-level and dataset-level labels might be inferred, such as cell-type, tissue, disease, and age. These often come from prior knowledge, manual annotation based on differential expression features, automated tools, or alignment with other prelabeled datasets. Thanks to these correlations between gene expression, labels might be defined and guide further research. Clustering tools and dimensionality reduction techniques used for visualizing

high-dimensional datasets as point clouds on 2D flat surfaces often employ nearest-neighbor-based methods.

5. In the case where specific clusters contain a low amount of cells, denoising or zero imputation methods can be used, but they haven't shown usefulness in practice since they are based on using cluster-level information, which is itself already used for most other tasks. Nearest-neighbors-based smoothing is the state of the art in this domain.

6. In the case where users would have access to a dataset of a similar tissue from other modalities like sc-ATAC-seq, sc-BS-seq, or protein measurements, imaging data, and more. Such multimodal alignment methods exist, often reusing prior knowledge and paired datasets where these two modalities are measured in the same cells. Thanks to these alignments, one can define more complex relationships between how DNA readings and related DNA mutations impact the expression of genes, and how these affect the downstream expression of proteins—bridging the map between the genotype (DNA) and the phenotype (cell features, patient diseases, etc). State-of-the-art tools there combine VAEs and heuristics.

7. If the dataset was measured in a non-static context, one can infer "cellular trajectories", meaning how cells seem to go from one state to another based on single measurements of many cells, such as in differentiation or during perturbations. Many tools exist for such trajectory inference, but it is an open question that often requires specific RNA-sequencing methods. State-of-the-art methods usually employ techniques like optimal transport (OT).

8. If the dataset contains spatial information : how cells were positioned in a tissue, one can infer "cell-cell interactions", meaning how cells influence each other based on their proximity and expression profiles. This often requires specialized imaging techniques, where cell morphology is also assessed. Cell-cell interaction is still in its infancy, but methods there frequently use foundation models and heuristics based on cell type analysis and proximity.

9. If the dataset includes perturbation experiments, one can predict how cells respond to specific perturbations, such as drug treatments or genetic modifications. This can help identify key regulatory pathways and potential therapeutic targets.

0.2.5 AIVC : the virtual cell models

A virtual cell model has been the dream of computational and systems biologists for decades. Initially, these models were based on simplified representations of cellular processes, often focusing on specific pathways or interactions. The models examined chemical reaction parameters involving proteins, RNAs, and DNA. However, in addition to computational challenges, these models were not able to generate realistic predictions of cellular behavior.

Nowadays, the dominant idea is that Artificial Intelligence, or AI, would allow us to solve some of these problems. But first, what is AI, and what is the difference between ML, Data science, and informatics ?

0.2.6 AI and neural networks

Data Science encompasses the gathering, management, and analysis of data in information systems. Machine Learning happens when one uses statistical methods to generate predictions from data. But these methods can be more complex, and while they can be seen in the framework of statistics, they also have an underpinning in other domains, like neuroscience (with neural networks) and applied mathematics with Algebra, Topology, Analysis, and most importantly, Optimization. These methods often allow powerful modeling of the data statistics.

Artificial Intelligence is a broad term that has had multiple meanings in society and culture. But it mainly refers to applications of machine learning methods to human and animal-related tasks, such as understanding images, videos, speech, and text, as well as robot manipulation. This creates many bridges and links between data science, machine learning, statistics, informatics, and AI.

Recently, ML has made great strides in many areas, primarily due to a significant increase in data generation, along with improvements in optimization methods and neural networks.

These tools convert concepts and objects into high-dimensional vectors, called embeddings.

These embedding vectors are then processed through layers of mathematical operations, often involving matrix multiplications and non-linear functions. The layers are designed to learn hierarchical representations of the input data, capturing increasingly complex features as the data passes through the network.

Large Language Models (LLMs) have become ubiquitous nowadays, and many in the field are trying to generalize them to world models. These models, trained on physics simulations and videos, aim to predict the next state of complex physical phenomena, such as what happens when someone throws a ball or the physics of water flowing down a river.

They have already shown promise in powering humanoid robots. Google DeepMind recently released a model for predicting the weather better and faster than the best "physics-based" models.

But why and how are these models so powerful? While this remains an active area of research, essential theories have been presented.

Contrary to the commonly accepted machine learning dogma, deep learning researchers showed, in the 2010s, that increasing the number of parameters did not necessarily lead to overfitting, i.e., the model learning by heart to reproduce the data. Instead, thanks to some light regularization methods, such as dropout and weight decay, which involve randomly removing some neurons during training and penalizing large weights, the models were able to learn more complex patterns in the data.

Thanks also to GPUs and interconnects allowing large parallel matrix operations in a distributed manner, such models could be scaled to orders of magnitude more parameters, thereby demonstrating the power of weak learning methods.

Finally, advanced first-order optimization methods, such as Adam, together with back-propagation, allowed even such a large number of parameters to be trained efficiently.

0.2.7 Optimization and loss landscapes

Indeed, stochastic gradient descent (SGD) methods like Adam not only minimize the loss function but also help escape local minima and saddle points.

To understand this, we need to understand the loss landscape. Imagine a 3D landscape where the height represents the loss value, and the two other "surface" dimensions are the model's parameters. The goal of the model is to find the lowest point in this landscape, which corresponds to the best set of parameters. However, this landscape is often very complex, with many local minima and saddle regions. The model wanders blindly and can only sense its immediate surroundings. If it falls into a local minimum, it should get stuck.

In very high dimensions, however, when the model has millions of parameters and SGD, by being stochastic, alters the loss landscape each time, the model can escape these local minima and saddle points.

Behind this unexpected behavior is the unsettling theory of emergence. Or how small objects can combine and interact in ways that would be unexpected and difficult to predict based on their individual properties. This theory tries to explain phenomena in dunes, snowflakes, ant colonies, and life itself, which might explain how large neural networks achieve such complex behaviors.

0.2.8 LLMs Bio-Foundation models

For text, images, videos, and audio, LLMs are everywhere now. In other domains, we often call similar models, trained on all the available data, foundation models.

This epitomizes a switch from small, simple neural architectures trained per dataset to larger neural architectures, called transformers, trained across the entirety of the available data. These models are called foundation models. The stated goal is for them to generalize better to their training data and task, and perform better than state-of-the-art small models.

After this initial training phase using a very generic unsupervised task, often referred to as pre-training, these models can be further fine-tuned to specific downstream objectives. Fine-tuning involves training the model on a smaller dataset with a goal specific to the task we want it to learn. In other domains, this has allowed the model to adapt its learned representations, enabling it to quickly gain higher accuracy than would be achieved without pretraining. For example, going from predicting the next word in a sentence to classifying the sentiment of a text, or even to being a chatbot. For this reason, essential innovations in fine-tuning methods involve creating losses that best align the model with the task.

These foundation models are still in their infancy. We want to understand how they work, if they can be useful to current research, and how we can improve on their training,

architecture, and generalization capabilities for the modality at hand.

0.2.9 Current single cell foundation models

The first practical example of a single-cell (RNA-seq) foundation model could have been scBERT, released in 2021. However, it was only used and benchmarked for cell type classification and pretrained on 1 million single cells. The first real foundational model, Geneformer, was released a year later. There, the author displayed the model's ability to perform various single-cell tasks, such as cell type classification, gene regulatory network inference, and perturbation prediction. Geneformer was trained on a much larger dataset of 33 million single-cells.

However, Geneformer, like scBERT, was really a complete reuse of the BERT model architecture. Furthermore, while showcasing interesting behaviors, it did not outperform simpler models on classifcal benchmarks of single-cell data.

In 2023, a year after Geneformer, a few additional foundation models were released from scGPT, showcasing a take on the GPT architecture and presenting various losses for fine-tuning the model. It was the first example of fine-tuning in a single cell and a more in-depth benchmark of the model across four different abilities : cell type prediction, gene network inference, perturbation prediction, and batch correction. However, it did not outperform state-of-the-art (SOTA) methods.

At the same time, Universal Cell Embedding demonstrated how one could train these models across multiple species to achieve state-of-the-art cross-species cell embeddings —vectors that represent the cell according to the model. It also showed a new kind of loss function to generate embeddings of the cells.

Finally, scFoundation, despite being closed-source, showcased a truly novel architecture specifically built for single-cell data and a novel training method based on the noise-to-sequencing-depth relationship of single-cell data.

Thesis Objectives

At the start of the project, two things were certain : I wanted to understand how single-cell foundation models worked and whether they indeed worked. I also wanted to see if I could improve gene regulatory network (GRN) inference from single-cell RNA sequencing (scRNA-seq) data. Knowing that I might see a possible interplay between the two.

To start, it is important to reflect on what I wanted to achieve at the start of the project, without knowing what I know now.

The second part will introduce the different chapters of the thesis and how they relate to my Ph.D. objectives and novel questions that arose along the project. This being a Thesis by Articles, each of the three chapters relates to a specific publication.

0.2.10 Personal Objectives during the thesis

A main mistake during one's Ph.D. is to not see the time passing by. My goal for this Ph.D. is to be as product-first as I was at Whitelab. Delivering results quickly & improving until it is publishable.

This mistake, thinking "Well, I have 3 years…" is what makes people go overboard, finish stressed, and unprepared for what is next. Thus, I plan to do mine in 2 years. And I will prepare everything around this idea. I will also start to prepare for what is next from the start.

To do that best, one needs to take the opportunity of the Ph.D. to make connections with other labs (industry or academic). Moreover, a good advice I have been given is to *know what you want to do and what you don't want to do*. Know what you are here for. Learn to say no. And I learned to say no in the last 4 years. My goal is to work on large models & large datasets, mostly in transcriptomics, and always to go back to first principles and biology.

I also know I want to make something useful, create something that can be a stepping stone for others. Something that has an impact on the community. I know that to do that, you have to go the extra mile in terms of development and be honest with yourself about any shortcomings.

Finally, I have been fortunate to become often addicted to my work. I like working hard and challenges. But for this to happen, I need to keep enjoying what I am doing. I also wish to have no regrets about this decision. Thus, my final goal is to enjoy it as much as I can.

0.2.11 Initial Ph.D. objectives

This Ph.D. aimed to develop new approaches using deep learning, possibly by using innovative graph neural network architectures on large scRNAseq datasets, to assess their predictability in high-quality benchmarks and package them as an open-source Python library.

Graph Neural Networks (GNN)s : are a class of deep learning models designed to operate on graph-structured data. They are specifically tailored to handle and process data represented as graphs, where edges connect the elements (nodes or vertices).

Traditional neural networks are primarily designed for processing grid-like data, like images, or sequential data, such as text. However, GNNs extend this capability to graph-structured data by incorporating graph-related operations and architectures.

Objectives. We wished to improve GRN predictions from scRNAseq data. Our approach is :

- To use larger neural network models that scale linearly with the dataset size, taking advantage of the tens of millions of data points now becoming available.
- To use novel GNN layers that could reduce the "search space" of the model by constraining the set of possible topologies it is learning.
- To improve the pretraining and fine-tuning of these models to the predictive task they have to perform, and the constraints of the system they are predicting.
- To formulate better layers that correspond to the sparse interactions between genes and our current knowledge about their functions.
- To create formal and rational benchmarks that best capture the ability of a GRN methodology.
- To assess predictions and any usefulness or lack of it by having biologists test hypotheses using the model.

Impact of the project. This Ph.D. project would thereby contribute to methodological breakthroughs by providing new tools and methods to use neural networks on unstructured data, like scRNAseq. To improve the state of the art in GRNs prediction from scRNAseq.

The proposed methodologies will impact computational (bioinformatics, machine learning) and biomedical fields. These new GNN layers could solve challenges faced by fields similar to those utilizing scRNAseq profiles, including environmental research, industrial biotechnology, and biofuel studies.

The initial objectives were separated into three possible work packages :

- Review of current tools and creation of a set of benchmarks
- GNN model/layers to better predict TF-gene relationships
- Collaboration to test the model's prediction on novel data

0.2.12 Revised objectives

The first objective was thus to review existing methods for inferring GRNs from multi-omics data, as well as the existing foundation models and benchmarks used to evaluate the performance of these methods.

What I quickly learnt is that GNNs were not the best approach for this problem. Indeed, we do not have good ground truths for GRNs, so we cannot start from a known graph. Moreover, GNNs tend not to scale well, and benchmarks show them as almost invariably performing worse than transformer-based models.

Thus, we quickly decided to stick with transformers, which can be seen as GNN working on fully-connected graphs. The main issue with transformers is their quadratic complexity in relation to the number of input tokens. Thus, we worked on making them scale sub-quadratically with the number of input genes and cells. This is one of our contributions on this side of the thesis.

Moreover, while we aimed to benchmark current methods, we also sought to create a new benchmark better suited to the single-cell field and the current data available.

Finally, while we managed to get some collaborations going, I realized the need for these foundation models to be made more accessible, which also became one of my objectives. It is represented not only in the effort to release easy-to-use open source models but also in various side contributions and outreach efforts.

Transformer-based models can be seen as GNNs, and examples have already been presented of them generating networks and GRNs directly from non-graph data, which is a feat that regular GNNs cannot achieve.

However, when we benchmarked them, we realized there were many shortcomings with these foundation models, which led us to create our own. Improving these models to scale better, adapt more effectively to the data, and generate better representations of genes, cells, and their networks thus became the main objective of this Ph.D.

0.3 Chapters overview & main contributions

This thesis is structured as follows :

— In Chapter 1, we present scPRINT, a single-cell foundation model for transcriptomics with a focus on gene network inference. We examine other foundation models and state-of-the-art tools in the field, creating a benchmark to compare them, first on GRN inference and then on other single-cell tasks. We show that current foundation models are performing better than random predictors, but worse than SOTA tools in most of our benchmarks. We show that our contributions in scPRINT make it a superior foundation model than the existing ones across all our benchmarks. We showcase the need for improved architectures and tasks that reflect the potential offered by foundation models, as well as reproducible benchmarks of the impact of different

design choices.
— In Chapter 2, we present Xpressor, a transformer-based compression mechanism, and fine-tuning approach to build foundation models across scales. We show how to plug a model that talks about proteins and genes in a model that talks about cells, and how this makes both models better.
— In Chapter 3, we present scPRINT-2 a next-generation single-cell foundation model whose design decisions were driven by an in-depth additive study of each model's components. We also present many contributions in architecture, training, datasets, losses, and fine-tuning. We showcase them in novel tasks and approaches.

0.3.1 Other contributions

This thesis presented other contributions not in the form of publications :

— The first ones are six open source Python packages named :

scPRINT : where the model and its v2 are available, together with training scripts and notebooks to use the model, functions to download and preprocess data, and more.

scDataloader : a package to load efficiently thousands of large single-cell datasets, with preprocessing, filtering, and loading options. It also allows a first-of-its-kind efficient weighted random sampling over billions of elements.

Bengrn : a package to benchmark GRN inference methods on single-cell data, using multiple types of metrics and ground truth networks.

GRNNdata : a package to work with gene regulatory networks and single-cell data jointly, using the Anndata format.

Xpressor : a package to reproduce the second paper's experiments and create an Xpressor model from scratch.

Simpler Flash : Initially a package to facilitate the use of flash attention before it became part of the pytorch implementation itself. It now includes multiple types of efficient attention mechanisms, such as softpick-flash and our flash criss-cross attention mechanism.

Hierarchical Classifier : a package to implement hierarchical classification for single-cell data.
— Another contribution, as previously mentioned, is around accessibility. Not only did I release model weights and inference code, but I also provided easy-to-use inference tools, pre-training methods and datasets, training traces, and documentation. Moreover, tutorials were implemented in Google Colab, and versions of the models got released on the Chan-Zuckerberg's model hub and superbio.ai's platform.
— Finally, I made a Docker implementation of scPRINT, scGPT, and Geneformer for their benchmarking on the open problems platform and participated in the improvement and creation of two benchmarks on the platform.

— In addition to publication, I wrote a blog post with Lamin.ai on training on large datasets. I also wrote with multiple x-plainers on the X, LinkedIn, Bluesky, and threads platform, as well as my personal website, to share some of our findings with a possibly wider audience. Similarly, I wrote vulgarisation articles for the Pasteur Institute's and CNRS's websites and created one of the most viewed videos on the Pasteur Institute's YouTube and Instagram accounts, presenting my work. I also got highlighted on Whitelab's blog posts and released four YouTube videos of diverse presentations of my work.

— Other outreach efforts were done through conference presentations and invited talks with :

    A participation in three international ML conferences

    Over 25 invited talks and five poster presentations.

— Finally, I also contributed to the European ecosystem of start-ups, translating works from Academia to Industry. I became part of a worldwide organization called Nucleate to help Master students, PhDs, and Post-docs translate their research into start-ups. I also worked as a consultant for four start-ups : Whitelab Genomics, Biographica, Blossom, and dot-omics, assisting them in developing strategies for foundation models in single-cell RNA-seq and DNA sequencing.

scPRINT : pre-training on 50 million cells allows robust gene network predictions

1.1 Summary

A cell is governed by the interaction of myriads of macromolecules. Inferring such a network of interactions has remained an elusive milestone in cellular biology. Building on recent advances in large foundation models and their ability to learn without supervision, we present scPRINT, a large cell model for the inference of gene networks pre-trained on more than 50 million cells from the cellxgene database. Using innovative pretraining tasks and model architecture, scPRINT pushes large transformer models towards more interpretability and usability when uncovering the complex biology of the cell. Based on our atlas-level benchmarks, scPRINT demonstrates superior performance in gene network inference to the state of the art, as well as competitive zero-shot abilities in denoising, batch effect correction, and cell label prediction. On an atlas of benign prostatic hyperplasia, scPRINT highlights the profound connections between ion exchange, senescence, and chronic inflammation.

1.2 Introduction

Understanding the cellular mechanism is considered a milestone in biology, allowing us to predict cell behavior and the impact of drugs and gene knock-outs[**lahnemannElevenGrandChallenges2**
roohaniPredictingTranscriptionalOutcomes2024, **gonzalezCombinatorialPredictionTherapeuti**
haradaDistinctCoreRegulatory2022a, **haradaLeukemiaCoreTranscriptional2023**]. A cell is regulated by a complex interplay of myriads of macromolecules that define its state. We can simplify these interactions via a gene network[**badia-i-mompelGeneRegulatoryNetwork2023**] (GN). Many approaches have been developed to infer these networks, focusing on transcription factor (TF)-to-gene links using single-cell omics data modalities like scRNAseq and scATACseq[**littmanSCINGInferenceRobust2023**, **aibarSCENICSinglecellRegulatory2017**, **bravogonzalez-blasSCENICSinglecellMultiomic2023**, **suInferringGeneRegulatory2024**,

ganInferringGeneRegulatory2024, **raharinirinaInferringGeneRegulatory2021**, **chenGraphAttentio**
wangDictysDynamicGene2023, **zhangInferenceCellTypespecific2023**, **wangInductiveInferenceGen**
kamimotoDissectingCellIdentity2023, **shuModelingGeneRegulatory2021**]. This gene network subset regulating the cell gene expression levels is often called a gene regulatory network (GRN). However, many other gene products than TFs impact RNA abundances in the cell, like RNA-RNA and protein-TF interactions[**boijaTranscriptionFactorsActivate2018**, **gretafriarItTakesThree2023**, **oksuzTranscriptionFactorsInteract2023**, **talukdarTranscriptionalCoa**
statelloGeneRegulationLong2021]. Most GRN inference methods do not scale to the number of genes and cells present in single-cell RNA datasets, and they need many cells, thus impairing their ability to reconstruct cell-state-specific networks. Other methods consider datasets where differentiating cells can be ordered temporally to predict more causal GRNs. While this approach is interesting, temporal ordering is often hard to predict[**wangDictysDynamicGene2023**, **wangDecipheringDriverRegulators2023**].

Benchmarks like BeeLine[**pratapaBenchmarkingAlgorithmsGene2020**] and MCalla et al.[**mccallaIdentifyingStrengthsWeaknesses2023**] have shown that despite the existence of many methods, GN inference remains a challenging problem. Indeed, it is underconstrained and has limited prior knowledge. New foundational models trained on tens of millions of measurements could help solve these difficulties. Transformers like BERT[**vaswaniAttentionAllYou2023**, **devlinBERTPretrainingDeep2019**] have gained traction in computational biology and have held promise to learn a model of the cell that would translate across many tasks of cellular biology, such as cell type annotation, batch-effect correction, perturbation prediction, and gene network inference[**theodorisTransferLearningEnables2**
Among them, scGPT[**cuiScGPTBuildingFoundation2024**] got much attention, proposing a novel encoding of genes and their expression, a new pretraining methodology similar to autoregressive pretraining in language models, and the possibility of extracting GRN from its model (see 1.5. methods).

Inspired by these efforts, we propose scPRINT (single-cell PRe-trained Inference of Networks with Transformers), a foundation model designed for gene network inference. scPRINT brings inductive biases and pretraining strategies better suited to GN inference while answering issues in current models (see 5. Supplementary Table S1). scPrint outputs cell type-specific genome-wide gene networks but also generates predictions on many related tasks, such as cell annotations, batch effect correction, and denoising, without fine-tuning.

We extensively benchmark scPRINT on challenging gene network inference tasks, from literature-based networks to cell type-specific ones generated via orthogonal sequencing methods. We show that scPRINT outperforms the state of the art on most of these atlas-level benchmarks. In addition, our model focused on GN inference, is also competitive on a compendium of tasks like denoising, cell type prediction, and embedding with batch effect correction. This suggests that by learning a cell model, scPRINT gains zero-shot abilities in many tasks of cellular biology. We use scPRINT to analyze an atlas of normal and senescent prostate tissues where we identify rare cell populations with early markers of the tumor microenvironment in B-cells. In fibroblasts, we study gene networks and recover known hubs such as PAGE4, linking the senescence of fibroblasts to changes in the ECM and downstream inflammation. We find key interconnected pathways of the oxidative stress response and extracellular matrix building via metal and ion exchange in the gene network

of BPH-associated fibroblasts. We also show that healthy and disease-related cells exhibit different network patterns, demonstrating that scPRINT can help identify novel pathways and targets while considering them in their specific cellular and molecular contexts.

scPrint[**kalfonCantinilabScPRINT2025**] (`https://github.com/cantinilab/scPRINT`) is a fast and open-source tool that can be readily integrated into the bioinformatics pipeline. We make public the code and model weights, but also the pretraining strategies, datasets, and our own dataloader for use with vast training sets like the cellxgene databas[**programCZCELLxGENEDiscover2023**]. We also release a Gene Network benchmarking suite : *BenGRN*[**kalfonJkobjectBenGRNAwesome2025**] and *GrnnData*[**kalfonCantinilabGRnnData2025**].

1.3 Results

1.3.1 scPRINT : a scRNAseq foundation model for gene network inference

We propose scPRINT (Figure 1.1A), a state-of-the-art bidirectional transformer designed for cell-specific gene network inference at the scale of the genome. scPRINT is trained with a custom weighted-random-sampling method[**jeremiekalfonTrainingFoundationModels**] over 50 million cells from the cellxgene[**programCZCELLxGENEDiscover2023**] database from multiple species, diseases, and ethnicities, representing around 80 billion tokens (see 1.5. Methods). We train scPRINT at various scales (from 2M to 100M parameters) and very efficiently by using flashattention2[**daoFlashAttention2FasterAttention2023**], e.g., only requiring an A40 GPU for 48 hours to train our medium model, significantly reducing the barrier to entry for any computational biology lab (see 5. Supplementary Table S2).

To push scPRINT to learn meaningful gene networks (GN) and their underlying cell model, we design a unique set of pretraining tasks, as well as expression encoding and decoding schemes (Figure 1.1B).

scPRINT's pretraining is composed of three tasks which loss are added and optimized together : a denoising task, a bottleneck learning task, and a label prediction task. The objective is to let scPRINT learn to represent meaningful gene connections while also endowing it with a breadth of zero-shot prediction abilities.

Indeed, similarly to ADImpute[**leoteRegulatoryNetworkbasedImputation2022**, **eraslanSinglecell**
we expect a good gene network to help denoise an expression profile by leveraging a sparse and reliable set of known gene-gene interactions.

We implement this denoising task as the upsampling of transcript counts per cell (see 1.5. Methods). While most other methods have been using masking as a pretraining task, our method is related to the downsampling and masking task of scFoundation[**haoLargescaleFoundationModel2**
We show that this strategy performs better than masked language modeling and gives scPRINT the ability to upsample any expression profile.

In addition, we expect that a cell model tasked to compress expression profiles into embeddings can learn the regularities of modules and communities of gene networks. Therefore, the bottleneck learning task drives scPRINT to generate an embedding and a cell expression profile from its embedding only. The embedding is generated by scPRINT and is used again, this time without the cell expression values, to regenerate the true profile (see 1.5. Methods).

Finally, the cell's gene network should represent the cell state and its different phenotypic facets. Effectively, scPRINT generates not just one embedding per cell but multiple. A hierarchical classifier is then applied to each distinct cell embedding to predict its associated class, such as cell type, disease, sex, organism, ethnicity, and sequencing platform. The embeddings thus become disentangled, each representing a specific facet of the cell state[**piranDisentanglementSinglecellData2024**]. This last training task pushes the large cell model and its gene network to represent the cell state.

Thanks to the cellxgene database requirement for complete annotations and our innovative hierarchical classifier, we have added label prediction as part of the pretraining of scPRINT. While the assumption is that in other modalities, the scarcity and noisiness of such labels make it infeasible, we show that this approach is a net positive in our case (see 5. Supplementary Table S3, 1.5. Methods). Indeed, it helps us disentangle the various cell embeddings and performs zero-shot predictions on unseen datasets. These disentangled embeddings are opening a future possibility to perform counterfactual generation : mixing embeddings representing different facets of cell states, e.g., fibroblast + cancer + pancreas tissue + female, to generate novel unseen expression profiles.

scPRINT converts the gene expression of a cell to an embedding by summing three representations or tokens : its id, expression, and genomic location (Figure 1.1A, see 1.5. Methods). scPRINT encodes the gene IDs using protein embeddings. This gene representation is made using the ESM2[**rivesBiologicalStructureFunction2021**] amino-acid embedding of its most common protein product (see 5. Supplementary Figure S1). First proposed in UCE[**rosenUniversalCellEmbeddings2023**], the model learns to leverage representations that can potentially apply to unseen genes and species, using the structural and evolutionary conservation of the sequence encoded by ESM2. While drastically reducing the number of weights used in the model compared to scGPT and Geneformer (see 1.5. Methods), this representation also contains some priors needed to infer protein-protein[**huImprovingProteinproteinInteraction2024**] interactions (Figure 1.1A).

The gene's expression is tokenized via a multi-layer perceptron (MLP) using log-normalized counts. This MLP lets the model learn a metric behind gene expression, whereas scGPT and Geneformer apply a specific prior for the encoding of their gene expression (see 1.5. Methods).

Finally, we help the model know that genes with similar locations tend to be regulated by identical DNA regions, using the positional encoding of their location in the genome (see 1.5. Methods).

These three embeddings are summed and then concatenated across all the genes expressed in a cell together with additional placeholder cell embeddings to form the transformer model's input.

scPRINT is pretrained using 2,200 randomly selected expressed genes in a cell profile. If a cell doesn't have enough expressed genes, the list is padded with randomly selected unexpressed genes. A context of 2200 genes, while not genome-wide, captures all the expressed genes in more than 80% of the cell profiles in the cellxgene database. We also show that scPRINT can make predictions on much larger sequences of genes at inference time without using attention approximation methods[**choromanskiRethinkingAttentionPerformers2022**].

Using unexpressed genes, combined with the denoising task, let scPRINT discriminate the true zeros from dropouts in scRNAseq47. The expression decoder of scPRINT further helps model this statistic of the data. It is a zero-inflated negative binomial graphical model inspired by previous literature in single-cell RNAseq modeling48. Here, the loss (also used for bottleneck learning) is thus the log-likelihood of the gene expression given the distribution parameters.

As shown in Figure 1.1C, at inference time, scPRINT can generate multiple outputs across any scRNA-seq-like cellular profile of various mammalian species without fine-tuning. Figure 1.1D shows scPRINT's prediction at the scale of an atlas of 2M randomly sampled cells from cellxgene. From its pre-training, scPRINT performs denoising, label prediction, and cell embedding without fine-tuning. However, a critical emergent output of scPRINT is its cell-specific gene networks. Following a similar approach to ESM2, we generate cell-level gene networks via the bidirectional transformer's input-wise weighted matrices, called attention matrices -or heads-. They represent general gene-gene connections and can be subsetted to TF-gene connections (i.e., GRNs). Remarkably, we made this approach scalable enough to compute attention heads-based gene networks for 1 to 10,000 cells, at the genome scale, with commodity hardware and in a few minutes. These networks both showcase the ability of scPRINT to model cellular biology and help make it a more explainable tool for the community, showing the network assumptions made during inference. The attention heads are either all aggregated by averaging or can be selected to better reflect connections of interest (Figure 1.1C). This is done using the average of the heads most correlated with literature or perturbation-based ground truth networks. Finally, while we do not assess scPRINT's ability to model inhibition due to the scarcity of such annotations, we leave open the possibility of using our head selection technique for such a task.

Similarly to what has already been done in ESM2 and the Large Language Model literature[**abnarQuantifyingAttentionFlow2020**, **clarkWhatDoesBERT2019a**, **bibalAttentionExpl**
we deeply investigate the meaning of attention matrices in the context of cellular biology, an aspect under-studied in the literature of foundation models applied to genomics.

In the following sections, we benchmark scPRINT on gene network inference against scGPT, DeepSEM[**shuModelingGeneRegulatory2021**], GENIE3[**huynh-thuInferringRegulatoryNet**
and Geneformer v2[**chenQuantizedMultitaskLearning2024**], the updated version of Geneformer. scGPT and Geneformer v2 are highly cited and published transformer models for single-cell RNAseq, mentioning the inference of gene interactions[**cuiScGPTBuildingFoundation2024**].
DeepSEM is an autoencoder model jointly learning its weights and a gene network matrix. GENIE3 generates networks via regression by finding the set of genes that best predict another gene's expression. It is one of the top-performing and most used methods for GRN inference (see 1.5. Methods). However, it suffers from very long run times and high memory requirements (see 5. Supplementary Table S4).

1.3.2 scPRINT recovers biological features in its gene networks

We benchmark scPRINT against the state-of-the-art based on whether their recovered networks contain meaningful biological knowledge. We consider two main benchmarking methodologies, one using a simulated expression profile from a well-established biological network. Because simulated data does represent real cell expression data (see 1.5. Methods), our second and main approach focuses on biological features of a network inferred from real cell expression profiles. Indeed, we assume that a meaningful gene network should have some of its hub nodes being TFs. TFs should be more connected to their known target, on average. We should recover known gene-gene connections and expect enrichment of cell type-specific marker genes in the network.

We compare each gene network inference method's ability to recover a known network from 1000 simulated single-cell RNAseq expression profiles generated by the Sergio ODE model[**dibaeiniaSERGIOSingleCellExpression2020**] from the ground truth network Regnetwork[**liuRegNetworkIntegratedDatabase2015**] (see 1.5. Methods). Only scPRINT was able to recover meaningful connections (see 5. Supplementary Table S5). One explanation is that through its training, scPRINT has learned the common gene connections that also exist in the RegNetwork ground truth.

On gene network inference from real expression data, we noticed that depending on cell type and datasets, the different tools could vary greatly in the similarity of their GNs to the Omnipath[**tureiOmniPathGuidelinesGateway2016**] ground truth. Because of this, we focused our benchmark on three randomly selected test datasets of kidney, retina, and colon tissues comprising 26 cell types[**marshallHighresolutionSlideseqV2Spatial2022**, **wangSinglecellMultiomeHuman2022b**, **kongLandscapeImmuneDysregulation2023**] (see 1.5. Methods, per dataset results in 5. Supplementary Figure S2). Of note is that we could not determine if these datasets were used during the training of scGPT or Geneformer.

We build one network per cell type, using the same 1024 cells and their 5000 most differentially expressed genes for all benchmarked methods. We evaluate the quality of the networks based on their overlap with Omnipath. We also compute the network enrichment for cell type markers, TFs, and ENCODE TF targets using the prerank[**subramanianGeneSetEnrichment2005a**] algorithm (Figure 1.2A).

Although the scGPT code mentions GRN inference only using perturb-seq data, we reapply the same method without the perturbation-baseline comparison. This is to make it comparable with other benchmarked methods and because most of our datasets are not perturbation-based. Similar to what is presented in its paper, we use the mean of the attention matrices across cells and the four attention heads of the last layer of the human pre-trained model. We retain this method across our benchmarks for scGPT (see 1.5. Methods). We apply a similar strategy for Geneformer (see 1.5. Methods).

For scPRINT, we generate three network versions : one simply called scPRINT, based on the average of all heads in the model. scPRINT (omnipath's heads), based on the average of heads selected with our abovementioned head selection method inspired by ESM2, and scPRINT (genome), which is like the scPRINT network but uses our method to generate genome-wide networks (see 1.5. Methods) instead of using the 5000 most differentially

expressed genes. Indeed, in transformer models, the choice of attention heads is important. Although transformers can learn the causal structure of their input, it has been shown that some attention heads, especially in larger networks, can become unused, containing predominantly random connections[**nichaniHowTransformersLearn2024**]. Some work has been done at pruning these heads[**shimLayerwisePruningTransformer2021**] or forcing a head selection mechanism during inference and training[**fedusSwitchTransformersScaling2022**]. For scPRINT (omnipath's heads), we select heads based on a linear classifier's prediction of the best set of heads to predict a subset of Omnipath (see 1.5. Methods). Similarly to the scPRINT network, these heads are then averaged to generate the scPRINT (omnipath's heads) gene network. To perform this selection, we split the omnipath dataset into train/test and select heads, using 50% of the ground truth and only the first cell type of each dataset. We then use the same combination of heads across all other cell types. This shows that our selection process builds consistent networks across cell types and parts of the ground truth. This innovative approach contrasts with previous ones like scGPT's and GENIE3 by using part of an available ground truth to select heads.

First, we look at how much information from Omnipath is contained in the inferred networks. Omnipath contains around 90,000 curated gene-gene connections, mainly from the literature. These connections are cell type agnostic, and most are TF - gene. On this benchmark, we evaluate the networks based on Area Under the Precision-Recall Curve (AUPRC) and Early Precision Rratio (EPR), two metrics often used in GRN benchmarks[**pratapaBenchmarkingAlgorithmsG**
(see 1.5. Methods), where we define our task as a binary classification of connections on all gene-gene pairs. Due to the row-wise normalization of networks generated by all methods, and because Omnipath has many sources with only a few targets (see 5. Supplementary Figure 1.2), we here use the transpose of our inferred networks when making comparisons with Omnipath (see 1.5. Methods). In Figure 1.2B, we can see that scPRINT (omnipath's heads) outperforms all methods on average across all cell types. While scPRINT (omnipath's heads) uses some ground truth information to select its head, we see that scPRINT still outperforms scGPT and Geneformer v2 on the EPR metric, showing that its top predicted edges more closely match the ground truth.

AUPRC results are very low overall because we do not expect most Omnipath connections to be present in the cell type's gene network, as many connections in Omnipath might only be true in some cellular contexts. Moreover, we do not expect most connections in our generated network to exist in Omnipath as it only contains a small fraction of all real gene-gene connections. Although overall AUPRC values are small, we can see that both scGPT and scPRINT outperform the other methods in the number of connections recovered. Indeed, on average, scGPT and scPRINT respectively recover 42% and 67% more connections than GENIE3.

However, GENIE3 is often used by biasing the method to only predict TF-gene connections (see 1.5. Methods). This type of network, usually called a gene regulatory network (GRN), is most often used, given the importance of TFs in regulating gene expression. To compare the other methods to this GRN version of GENIE3, we also use a GRN version of their networks by subsetting them to TF-gene connections only. In this context, all the methods significantly improve their predictions without altering their relative performances (Figure 1.2B). This is unsurprising, considering that Omnipath is strongly biased towards

TF-gene interactions.

Interestingly, we have seen that smaller scPRINT models containing fewer heads perform better when taking the average of their heads. In contrast, head selection is often more advantageous in larger models with more heads (see 5. Supplementary Table S6). As presented at the beginning of the results section, it might be that as models become larger and less regularized, some heads tend to become unused and contain mostly noise. As a consequence, a head selection is advantageous in larger models.

We also expect biologically meaningful gene networks to have their central nodes enriched for TFs. In addition, because these networks are cell type-specific, we expect their central nodes to be enriched for some marker genes of their associated cell types (see 1.5. Methods). In this regard, both scGPT and scPRINT achieve very similar and strong network enrichment for TFs compared to GENIE3, DeepSEM, and Geneformer v2, whose networks are not enriched for TFs (Figure 1.2C).

Moreover, amongst the 178 cell types we have marker gene sets for in pangaloDB[**franzenPanglaoDBWeb**
all methods find some enrichment, especially GENIE3 and scGPT (see 1.5. Methods). We notice that selecting heads based on Omnipath significantly improves scPRINT's network enrichment for cell-type markers. Of note, our goal is not to annotate cell types from the gene network but mainly to showcase the network's cell type specificity.

Finally, we also examine how much the connections of each TF are enriched for that TF's target. Here, scPRINT overperforms all other methods (Figure 1.2D). In the scPRINT networks, 20% of the Transcription Factors for which we have data on ENCODE have connections significantly enriched for their ENCODE-validated gene targets[**liberzonMolecularSignaturesDatabas**
Interestingly, only our large cell model achieved a great performance, and scGPT did not display any enrichment across the 26 cell types assessed. While we acknowledge that ENCODE is used in the Omnipath database, we cannot expect Omnipath to represent the ENCODE targets. Indeed, it combines and processes 57 additional data sources to build its consensus network.

scPRINT (genome) has been added despite its performance not being comparable to other methods. Indeed, comparing its overlap with Omnipath is unfair as it includes many more genes and connections, many of which will have almost no data on this ground truth. While scPRINT (genome) showcases our ability to generate genome-wide networks, it also shows strong performances in TF enrichment and ENCODE TF-target enrichments. This highlights that even at such a large scale, networks generated by scPRINT are enriched in biological knowledge gained solely from its pre-training tasks.

Overall, we have shown that scPRINT generates, in one forward pass, cell type-specific gene networks that are biologically meaningful. We will now examine them using cell type-specific ground truths extracted from orthogonal experiments.

1.3.3 scPRINT outperforms the state of the art on cell type-specific ground truths

Although we have shown that our networks represent meaningful biology, the Omnipath ground truth is literature-based and not cell type-specific. Here, we use two different modalities, perturb-seq[**dixitPerturbseqDissectingMolecular2016**], and ChIP-seq[**parkChIPSeqAdvantages**] as ground truths to compare predicted gene networks against.

In the MCalla et al.[**mccallaIdentifyingStrengthsWeaknesses2023**] ground truth, ChIP-sequencing and perturb-seq are intersected to get at the small subset of possibly direct connections between TFs and genes for both human and mouse embryonic stem cells (ESC) (Figure 1.3A, see 1.5. Methods). We have seen that these ground truth networks show a different pattern than literature-based networks (see 5. Supplementary Figure S3). Some TFs regulate only a few genes, whereas others are highly connected.

To generate our networks, we use as input one human and two mouse ESC scRNA-seq datasets from MCalla et al. with the addition of another human dataset from Yan et al.[**yanSinglecellRNASeqProfiling2013**] Networks are generated over the same 1024 cells, and the 5000 most variable genes for all methods. For scPRINT, three networks have been generated : one averaging all the attention heads (scPRINT), one averaging heads selected based on how well they predicted Omnipath ground truth data : scPRINT (omnipath's heads), and one averaging heads selected from one of the MCalla ground truths : scPRINT (Han et al.'s heads). For more details, see the results section 2 : scPRINT recovers biological features in its gene networks. Of note, due to the small amount of genes assessed in the ground truth, we do not add the genome-wide network version here. Moreover, only the TF version of GENIE3 and the TF-gene subsets of the other method's networks are used since the ground truth only contains TF-gene connections.

Contrary to Omnipath, some elements in these biological networks are highly connected, whereas many others display no connections. This imbalance means that a method predicting only the highly connected TFs will perform well on the MCalla et al. benchmark. As a consequence, we are not transposing the attention matrix as done in the previous section.

Based on both AUPRC and EPR, scPRINT outperforms all other methods on this benchmark (Figure 1.3B). This means, for example, that when training GENIE3 to only predict a gene's expression based on TF expressions, it is not selecting the right TFs amongst the set of a few dozen assessed in MCalla et al.

scGPT, Geneformer v2—and, in a few cases, scPRINT—can have values worse than random guessing. Thus, their predictions are often specific to some TFs but not necessarily the right ones (Figure 1.3B).

It also appeared that selecting heads based on Omnipath, although helping slightly in one instance, is not a net benefit for this dataset (see 5. Supplementary Table S9). This makes sense since MCalla et al. itself does not overlap much with Omnipath (see 5. Supplementary Table S7). However, selecting heads based on the ground truth itself, only using 50% of the connections available, shows substantial improvement. These same heads also show reliable behavior when using them on the second dataset of the same species.

This shows that scPRINT can better decipher direct from indirect TF-gene connections than scGPT, DeepSEM, Geneformer v2, and GENIE3, although more tests would likely be needed.

However, the results also highlight that the high imbalance (i.e., TFs being not connected or highly connected) combined with the dataset size (i.e., only a few dozen TFs assessed) and the low number of cells make the results in MCalla et al. very variable. Some of this might be true biology or explained by ChIP-seq, which can be very noisy depending on the quality of its antibodies[**kidderChIPSeqTechnicalConsiderations2011**].

To answer this issue, we selected another dataset : genome-wide perturb-seq (gwps)[**replogleMappingInfo**
Here, we measured the effect on transcription of knocking out all expressed genes in the K562 cell line. We transformed it into a network using a cutoff of 0.05 on the significance level of each gene's differential expression before and after the KO of each other gene. Although this does not tell us which connections are direct or indirect, we now have a much broader set of connections over thousands of genes and better statistics to assess our gene network inference methods.

GENIE3 performs best, directly followed by scPRINT. Interestingly, Geneformer v2 shows poor performance (Figure 1.3C). Perturbation experiments are known to correlate somewhat to expression correlation, and this might explain GENIE3's strong performance. However, when using our head selection mechanism, scPRINT (gwps' heads) outperforms GENIE3. Again in this dataset, selecting heads based on Omnipath does not help ; the small overlap between the gwps network and the Omnipath ground truth network seems likely to be the culprit (see 5. Supplementary Table S7). These overlaps show that the three ground truth networks are very different and that a different set of heads predicts each type of ground truth. We also assess the networks on the TF-gene only subset of the gwps ground truth. Here, we see a large drop in performances for most methods, except GENIE3 (see 5. Supplementary Figure S4).

Finally, we have seen that on both MCalla and gwps, scPRINT also predicts networks that agree with the Omnipath ground truth and are again enriched for cell type markers and TFs (see 5. Supplementary Table S8, S9).

Since GNs can be seen as approximations of a cell model, we expect that when a tool has good internal cell models, it should generate meaningful results on tasks such as denoising, cell type prediction, embedding and batch effect correction, perturbation prediction, trajectory inference, and more. We will now focus on three tasks orthogonal to GN inference to compare the ability of scPRINT to the state-of-the-art.

1.3.4 scPRINT is competitive on tasks orthogonal to GN inference

To test the quality of the cell model learned by scPRINT, we now consider denoising, cell type prediction, and batch effect correction as a representative set of classic scRNAseq and cellular biology benchmarks.

Similarly to our pretraining task, we simulate lower transcript count profiles and then ask scPRINT and two other state-of-the-art methods, MAGIC[**dijkRecoveringGeneInteractions2018**]

and KNNsmoothing2[**wagnerKnearestNeighborSmoothing2018**], to recreate the true expression profile. We use Spearman correlation to the original gene expression profile as our metric. In Figure 1.4A, we show the increase in correlation after denoising the downsampled profile on 3 test set datasets, composed of ciliary body, colon, and retina tissues[**wangSinglecellMultiomeHuman2022b**, **vanzylCellAtlasHuman2022**, **burclaffProximaltol** randomly selected from cellxgene (see 1.5. Methods).

ScPRINT is competitive with both SOTA methods, while contrary to MAGIC and KNNSmoothing2, it operates independently over each cell in the test set (see 1.5. Methods). We have also seen a 10% variability in denoising ability across the different datasets used (see 5. Supplementary Table S10). This was similar across all tools and possibly related to the number of genes expressed in each dataset.

However, these test cases mostly contain very similar cell states, whereas denoising is helpful in cases with rare cell types or transitory cell states that have low cell counts by default. We show that since scPRINT does not aggregate profiles over neighboring cells, it outperforms MAGIC and KNNsmoothing2 in rare cell states subsets of the datasets (respectively : pericytes microfold cells of epithelium of small intestine and microglial cells) with around 10 to 200 cells (Figure 1.4A, 5. Supplementary Figure S5). Computing MAGIC and KNNsmoothing2 over only this rare cell population gives even lower performances for MAGIC and creates an error for KNNsmoothing2 (see 5. Supplementary Table S10). These results suggest that a good cell model reliably using learned gene-gene interactions can help denoise an expression profile.

For cell type classification, we expect scPRINT to be able to find sets of genes that can predict a cell type across multiple batches and under the high dropout rate of single-cell RNAseq. To evaluate cell type classification, we use the multi-batch benchmark pancreas dataset of openproblems, its metrics, preprocessing, and hyperparameter choices (see 1.5. Methods)[**lueckenBenchmarkingAtlaslevelData2022**, **OpenproblemsbioOpenproblemsv22024**].

scPRINT is a zero-shot predictor of cell labels. Indeed, it does not need to train on the dataset itself to make its predictions, unlike other methods that often need to use more than 70% of the test dataset for training. scPRINT also makes predictions over more than 200 cell type labels, while other methods often only predict a few cell types. Conversely, the other classifier methods, like Logistic Regression or XGBoost, and previous foundation models are trained or fine-tuned on the test dataset, thus giving a strong advantage over scPRINT. We, therefore, also compare scPRINT to the marker-based classifier CellTypist[**condeCrosstissueImmuneCell2022**] and its pancreas marker database (see 1.5. Methods). A method that also does not use the labels of the test dataset.

scPRINT reaches 62% classification accuracy, largely outperforming CellTypist (Figure 1.4B, 5. Supplementary Figure S6). Interestingly, with the macro F1 score, which considers each cell type group equally regardless of its size, scPRINT achieves similar results to the state-of-the-art[**OpenproblemsbioOpenproblemsv22024**] methods : Logistic Regression and XGBoost. This is probably because scPRINT is not influenced by the number of cells in each category.

In addition, we have noticed that scPRINT is challenged by some specific pancreatic cell types in this dataset. Indeed, scPRINT often switches the assignment of A, B, D, and E cells.

Thus, when using the coarser "endocrine pancreatic cell" label to define these cell types, we see a big improvement in the accuracy and macro-F1 score of scPRINT, even outperforming state-of-the-art methods.

Here, we have shown the accuracy of scPRINT independently of cell neighborhood. However, like gene marker-based methods, scPRINT can annotate cell types in novel datasets. In this context, its predictions could be smoothed and improved using majority voting over predefined cell clusters.

Finally, scPRINT predictions are given as probability vector overall cell type labels. They can be used to display the top K labels and learn about the model's uncertainty.

Thanks to its disentangled embeddings, scPRINT can generate cell representations that partially remove batch effects from cell profiles. On the human pancreas and lung datasets of open problems[**sikkemaIntegratedCellAtlas2023**], we see that, based on the scIB metrics, scPRINT shows convincing batch effects removal ability, while not on par with the SOTA methods scGEN and scVI (Figure 1.4C, 5. Supplementary Figure S7). Concerning foundation models, scPRINT and scFoundation show strong zero-shot performances compared to Geneformer v2 and scGPT. Except for Geneformer v2, scGPT, and scFoundation, we did not rerun previous algorithms for this benchmark and show their performances from the openproblems portal (open-problems-v2.3.6, march 2024). However, we also ran the Geneformer v2 and scGPT foundation models on the openproblems benchmark and showed that without fine tuning on this specific dataset, they are not able to meaningfully correct for batch effect (see 1.5. Methods).

Moreover, scPRINT is one of the few methods that do not train on the test dataset and do not use already annotated batch labels. When only looking at methods that do not use batch labels as prior information, e.g., SAUCIE[**amodioExploringSinglecellData2019**], LIGER[**liuJointlyDefiningCell2020**], scPRINT is the top performer. We have also noticed that the scPRINT cell embeddings preserve biological information competitively to state-of-the-art methods (Figure 1.4D, 5. Supplementary Figure S8). This also exemplifies that a reliable cell model can perform well at disentangling the different facets of a cell expression profile and its underlying batch effect.

Overall, we have seen that scPRINT can achieve zero-shot performances on par with many famous single-cell RNAseq tools on multiple important tasks of single-cell biology, showing that our architecture and foundational pre-training tasks are a powerful new foundation for large cell models.

1.3.5 scPRINT highlights the role of ion exchange and fibrosis in the ECM of Benign Prostatic Hyperplasia

To showcase the ability of scPRINT, we focus on premalignant neoplasms from an atlas of two studies of human prostate tissues[**josephSingleCellAnalysis2021**]. The data contains both normals and pre-cancerous lesions, also called benign prostatic hyperplasia (BPH), across sequencers and age groups. Starting from post-alignment raw counts, scPRINT generates a consistent and batch-corrected embedding of the datasets (Figure 1.5A, 5.

Supplementary Figure S9). scPRINT also annotates the cell type, sequencer, sex, ethnicity, and disease type of each cell with an accuracy of 0.71, 0.99, 0.99, 0.95, and 0.85, respectively.

We then focus on a switched memory B-cell cluster composed of a group of cells labeled as benign prostatic hyperplasia and another as normal (Figure 1.5A). B-cells are known to be dominant in prostate cancer and are often switched memory B-cells[**saudiImmuneActivatedCellsAre2023**] First, we show that they differentially express many known B-cell markers (see 5. Supplementary Figure S10). In addition, when comparing the BPH to the normals B-cells, we recover that the top 10 BPH B-cells differentially expressed genes contain many known cancer markers, B cell markers, and a specific B-cell associated prostate cancer markers : BAG5[**Bcl2AssociatedAthanogene**] (highlighted in Figure 1.5B, Table S11). Moreover, many other genes have evidence in other cancers, like CLIC4, known to be involved in the maintenance of the tumor microenvironment (TME) in breast cancer[**HostCLIC4Expression**].

However, the number of healthy cells, especially normal memory B-cells, in this dataset is small : only 26. By performing denoising, we can recover genes that might have been missed during differential expression analysis of such a low cell count. Increasing the counts of all the genes by a factor of ten and re-doing differential expression analysis highlights some new genes whose differential expression scores are even higher than those previously cited.

Interestingly, amongst them, TSEN54, EHMT2, and IL10RB are known to impact the function of B-cells in malignancies (see 5. Supplementary Table S11). Other genes have evidence in immunity and cancer, like TAP1, which is known to be highly expressed in immune organs and is an immunomodulation gene known to play many roles in various cancers[**zhuTAP1PotentialImmunerelated2023**], while some genes have, of yet unknown significance, like LIP, whose paralog LIPA is a known cancer target[**TargetingLIPAIndependent**] (Figure 1.5B).

This demonstrates how scPRINT can embed, align, and annotate diverse datasets in a meaningful way so that one can then analyze specific and rare cell clusters to recover both known and new biology.

Finally, for the second part of the analysis, we move to another cell type of interest : fibroblasts. Fibroblasts are known to be involved in cancer[**CancerassociatedFibroblastsBasic**], also called cancer-associated fibroblasts (CAFs), of which many subtypes exist, with different roles in tumor progression and invasion[**FibroblastHeterogeneityProstate**]. In our dataset, we can see a large cluster of cells labeled as “fibroblast of connective tissue of glandular part of prostate”, of which 500 are coming from normal tissues, and 600 are coming from hyperplasia and are possible precursors of CAFs (Figure 1.6A). Interestingly, 40% of the cells annotated as BPH-associated fibroblasts are coming from healthy tissue, according to the authors of the dataset. However, it is known that more than 50% of adult males over the age of 50 will have BPH[**EpidemiologyClinicalBenign**]. Thus, one possibility is that some of the fibroblasts of these healthy tissues already present patterns of gene activation similar to those of pre-cancerous ones.

We generate a gene network of the BPH and normal fibroblasts using the 4000 most variable genes and taking the average over all heads in the network (Figure 1.6A). Looking at the top 15 hubs, using degree centrality, we can see S100A6 as the top element in normal fibro-

blasts. This gene is known to be a fibroblast and epithelial cell marker that regulates, among other things, cell cycle and differentiation[**kuznickiCalcyclinMarkerHuman1992**, **S100A6MolecularFun** We also see MIF, IGFBP7, and other genes involved in immune signaling and growth[**WikiPathways2024Next** **ROLEBIOMARKERMACROPHAGE**, **IGFBP7PromotesEndothelial**].

However, some of these genes are not in common with the BPH fibroblasts ones. Over the set of 2881 common nodes between the two networks, the genes HSPA1A, MT2A, SPOCK3, ATP6V0C, DEFA1, EIF4A1, and CD99 are considered differential hubs (i.e., more central) in the BPH fibroblasts compared to normal ones (see Figure 1.6A, Table S12).

Another definition of centrality, eigenvector centrality, recovers 55% of the genes already identified as hubs, plus some new ones. As an example, Prostate Associated Gene 4 (PAGE4), which is part of the GAGE family of genes, is expressed in a variety of tumors and reproductive tissues, especially BPH, where it is related to oxidative stress response and fixation (i.e., anti-invasion)[**liProstateAssociatedGene42022**, **josephSingleCellAnalysis2021**, **lvPAGE4PromotesPr** Interestingly, although the networks share 75% of their genes, they only share 50% of their edges when considering the top 20 edges per gene. It shows that over the same set of genes, scPRINT discovers distinct gene networks across biological contexts. Taking as an example the differential hub PAGE4 (Figure 1.6A), we see that it is connected to many of the top 15 hub nodes in the BPH network, such as MT2A, HSPA1A, SPOCK3, and CD99. This shows a master node sub-network linking metal and ion exchange, oxidative stress response, and inflammation[**diasDownregulationMetallothionein2A2022**, **luoMechanismPrognosticMarker2023**, **defreitasCirculating70KDa2022**, **CD99CrossroadsPhysiolog** Some genes are also part of the IL24 signaling inflammatory pathway (EIF4A1 ;COL6A2 ;HLA-C ;HSPE1), and the secretory senescence phenotype (H2AZ1 ;UBE2S ;UBE2C ;IGFBP7)[**milacicReactomePath** hallmarks of fibrosis and malignancies[**mausIronAccumulationDrives2023**, **qianEstablishmentCancera** The PAGE4 network in normal fibroblasts, while having some elements in common, like metal transport, is much less connected (seen by the strength of the edges in Figure 1.6A). It also contains a different set of genes, which are less related to senescence, inflammation, and ion exchange (see 5. Supplementary Figure S11).

Furthermore, we can use these networks, defined over only a few cells, to perform community detection. Taking community 4, containing 92 genes and defined with the Louvain algorithm on the BPH-associated fibroblasts GN, we see two hub nodes : SPOCK3 and HERC3 (Figure 1.6B). Interestingly, not much is known about those genes except that HERC3 has been linked to inflammation and the extracellular matrix (ECM) via metallopeptidase and the NCOA1 gene[**liAccumulationNCOA1Dependent2023**]. SPOCK3, moreover, is known to be related to prostate malignancies and collagen in the ECM[**luoMechanismPrognosticMarker2023**]. Gene set enrichment tells us that the genes in this subnetwork are primarily related to calcium, sodium, iron, and metal transport, validating the evidence around HERC3 and SPOCK3 (Figure 1.6B). In normal fibroblast, however, taking the community most associated with metal transport (community 4, see details in 5. Supplementary Figure S12 and Methods) shows RNASEK, SELENOM, and an unknown ubiquitin ligase, paralog of ITCH. While RNASEK is related to RNA degradation, its expression has been linked to a lower risk of prostate cancer[**kladi-skandaliExpressionalProfilingClinical2018**]. SELENOM is of unknown function, but some SEL proteins have been related to cell adhesion[**SelenoproteinDeficiencyAlters**]
.

Through its networks, scPRINT highlighted the role of ion exchange and fibrosis in the ECM in Benign Prostatic Hyperplasia. While some of the same genes would have been found from differential expression analysis, these results show us how gene networks can be used to describe the intersection of genes and their molecular functions. Putting genes into the context of their connections, one can validate known functions or relate them to new ones. From such contextualization, a picture starts to emerge, whereby through specific genes, glandular fibroblasts in senescence enter a wound-healing state. This fibrosis is caused by the export of more metal and ions to generate ECM and change its acidity levels. This might cause a loss in tissue flexibility and potentially create oxidative stress[**EffectPHExtracellular**]. In our networks, these pathways seem connected to inflammation. Chronic inflammation and wound healing states are hallmarks of BPH and a predisposition to future malignancies[**colottaCancerrelatedInflammationSeventh2009**, **hanahanHallmarksCancerNext2011**].

1.4 Discussion

We can simplify the complex macromolecular interactions governing a cell through what is often referred to as a gene network. However, creating such a network in a meaningful way remains a challenging task.

We have created and benchmarked scPRINT, a novel single-cell RNA sequencing foundational model trained on more than 50 million single-cell profiles across tissues, diseases, and species contexts. scPRINT uses three foundational pre-training tasks, as well as new encoding and decoding mechanisms specifically designed for gene expression data. Although it has not been directly trained for it, scPRINT generates gene networks. These networks can be used to better understand the model predictions and help make more informed decisions about the significance and role of a potential target. Finally, we present a mechanism to best select heads containing the known biology of these networks. This approach also helps users fine-tune the type of network they are interested in. Given the discrepancy amongst ground truth networks, we advise users to consider using all-head averaging and to only revert to head selection when some high-confidence interactions are available. Indeed, general collections like Omnipath did not improve performance in most of our tests.

We show that we outperform other foundation models on most of our benchmarks while using a similar model size. We believe that our inductive biases and training procedures helped scPRINT achieve such a performance. Moreover, while GENIE3 is still a competitive tool, we outperformed it on many of our benchmarks, showing that pushing training to millions of cells and large parameter sizes will be an essential direction for further work on gene network inference.

In addition, contrary to any other method assessed, our large cell model can also achieve zero-shot performances on par with many famous single-cell RNAseq tools on multiple important cell biology tasks. While some specialized tools might be better suited to some use cases, scPRINT's versatility and speed make it a worthwhile alternative in many instances. Indeed, users can directly use scPRINT in their bioinformatics workflows with commodity

hardware (1 CPU, 1 GPU with 10GB of memory and 16GB of memory).

Finally, we put scPRINT to the test on a challenging atlas of normal and senescent prostate tissues showing benign prostatic hyperplasia. We identify rare cell populations with early markers of TME in B-cells. In fibroblasts, we study gene networks and recover known hubs such as PAGE4, thereby linking the senescence of fibroblasts to changes in the ECM and downstream inflammation. We find key interconnected pathways of the oxidative stress response and extracellular matrix building via metal and ion exchange in the gene network of BPH-associated fibroblasts. We also show that healthy and disease-related cells exhibit different network patterns, demonstrating that scPRINT can help identify novel pathways and targets while considering them in their specific cellular and molecular contexts.

An assumption in natural language processing is that fewer inductive biases make for better models. Our work shows that adding good inductive biases and rethinking architectures will likely be important directions for AI models in biology.

A challenging aspect of GN inference is that no perfect ground truths exist, and many GN methods are, unfortunately, benchmarked on ODE-generated mock-up expression data. In contrast, ChIP-seq, perturb-seq, and literature-based ground truths remain scarce and ambiguous. With BenGRN and GRnnData, our suite of tools for benchmarking Gene Networks inferred from single-cell RNA sequencing, we present an extensive set of real-world ground truths representative of the diversity of networks we can assess. However, improvement in performance and benchmarking will need to come from innovative experimental approaches that can produce causal, genome-wide, and cell-type-specific networks containing the many different types of connections and regulations that exist, from PPI, RNA-DNA, RNA-protein, to inhibition, activation, cooperation, and more.

We acknowledge that work remains to be done, from the transformer's ability to generate graphs to their explainability and the breadth of tasks they can undertake. Questions still remain regarding the pre-training tasks and how to integrate additional data modalities into foundational models.

Transcription is much more complex than what gene networks currently represent. In the future, we expect such large cell models to work in tandem with new sequencing techniques measuring information such as time, space, protein amounts, DNA configuration, and non-coding RNA species to solve the gap in our understanding and our ability to model cell biology.

1.5 Methods

we propose scPRINT, a foundation model designed for gene network inference. ScPRINT brings novel inductive biases and pretraining strategies better suited to GN inference while answering issues in current models. scPrint outputs cell type-specific genome-wide gene networks but also generates predictions on many related tasks, such as cell annotations, batch effect correction, and denoising, without fine-tuning.

1.5.1 Architecture

The model architecture is composed of :

- An encoder that takes the raw data and embeds it in a high-dimensional space used by the transformer.
- A bidirectional multi-head transformer
- A decoder to transform the expression embeddings into expression values
- A decoder that transforms the cell embeddings into cell-specific label prediction over a range of classes.

Expression encoder

In scPRINT, each gene in a cell is converted to an embedding : It corresponds to the sum of 3 different elements :

1. An embedding representing the gene itself (see 5. Supplementary Table S2 for model embedding size). ESM2 embedding of each gene's most common protein product was used to represent that gene. While imperfect in some ways, this inductive bias allows the model to learn representations that potentially apply to even unseen genes from unseen species or integrate specific genetic mutations into its representation. First implemented in UCE, this provides the model information related to the gene product's structure, ontology, and similarity to other genes. This also speeds up the training greatly, particularly for small models. We show that this is a great gene representation, but that model performance can be increased by refining gene embeddings further during training. However, we elect not to do so to maintain the model's versatility in working on unseen genes.

We encode the genes' embeddings using ESM2. The mapping process happens the following way :

- A gene name is mapped to its canonical protein name using Ensembl.
- We recover the protein sequence of the protein using Ensembl
- We use the protein sequence to generate an embedding using ESM2 by averaging all the amino-acid output embeddings, as done in the ESM2 paper.

With the embedding function provided in our code, one can easily do this with any species in Ensembl.

scPRINT can effectively be retrained with any set of gene embeddings, which can be frozen during training or used only for initialization (tried, for example, in our ablation studies, Table S3).

2. An embedding of the gene location in the genome. This has also been proposed in UCE and helps the model understand that genes with similar locations tend to be regulated by similar regulatory regions, a relationship well-known in cellular biology.

We encode the genes' locations using positional encoding. Every gene less than 10,000 bp from the next is said to be in the same location ; otherwise, we increment location by 1. We do this for all genes in the Ensembl database per species.

We then embed these locations by applying the Positional Encoding (PE) algorithm of Vaswani et al. .

3. An embedding of the gene expression in the cell. For this, we embed the gene's expression using an MLP. While GeneFormer devised a ranking strategy based on a gene expression compared to a baseline expression, scGPT instead used binning of log normalized counts. On our end, we haven't found that this approach was the simplest, nor was it performing better than only using the log-transformed counts. We thus directly take the log-transformed counts

$$\mathbf{e_{i,j}} = MLP(log_2(x_{i,j} + 1)),\ x_{i,j} \in \mathfrak{R}\ ,\ \mathbf{e_{i,j}} \in \mathfrak{R}^{\,d}, \quad (1)$$

where $\exp r_{i,j}$ is the embedding of the expression, $x_{i,j}$ is the expression value of the gene j in the cell i, and the MLP is a two-layer neural network, where each layer is composed of

$$Dropout(ReLU(LayerNorm(Linear(\mathbf{e_{i,j}})))) \ , \quad (2)$$

where the Dropout rate is fixed at 0.1, and the dimensions are specified as $1 \rightarrow d$ for the first layer of the MLP and $d \rightarrow d$ for the second layer, with d representing the model dimension.

Of Note : Geneformer used positional encoding to encode gene expression, a function often used to encode the position of words in a text. Similarly to gene name token, scGPT learned an embedding for different ranges of expression values, binning them to remove sampling noise.

Both approaches apply a specific prior for the metric that defines expression. Geneformer defines expression amount as ranking based on how each gene is expressed in the cell compared to its average across all cells. Unregarding the batch effect issues, this is an assumption that expression values are not meaningful and only the ranking of the relative abundance is meaningful information. Meanwhile, scGPT has the bias that an expression of 1, 2, or 3 are the same and that an expression 1, and 5 are different by some amount learned by the model.

By using an MLP with two layers, we effectively let the model learn the metric of transcription expression. Moreover, again, we decrease the number of parameters used compared to scGPT while being able to make predictions on count values unseen during training, such as those of bulk or pseudo-bulk RNAseq.

Finally, when encoding a cell expression profile, only a subset of 2200 genes is used during pretraining. If less than 2200 genes are expressed, we randomly choose 2200 expressed genes and pad them with randomly sampled unexpressed genes (meaning with an expression value of 0). This approach allows the model to see different patches of the same cell profile during training. We chose 2200 genes as 2/3rds of the cells in cellxgene had less than this number of genes expressed, striking a balance between computation and gene usage.

We decided to add unexpressed genes because, combined with our denoising methodo-

logy, this lets the model figure out that some genes are true 0s during training. In contrast, others are only caused by dropout and a function of the transcript counts. This causes scPRINT to model dropout as a function of read depth (i.e., total transcript count).

Moreover, this completes the minibatch by token matrix without padding and fully utilizes the GPU during the attention computation.

Of note, some models have been able to reach context lengths of 20,000 genes using the performer architecture. Performer is an often-cited method and part of the literature on attention approximation. However, most state-of-the-art transformer models do not use attention approximation as they are known to lead to worse performance.

Moreover, in cellxgene, more than 80% of the cells have less than 2200 genes being measured. This means that most of the memory and compute power is likely lost on tokens that are almost always zeros due to dropout.

The full set of embeddings of cell i sent to the transformer is the matrix X_i where

$$X_i = [\mathbf{g_0} + \mathbf{e_{i,0}} + \mathbf{l_0}\ ,\ \mathbf{g_1} + \mathbf{e_{i,1}} + \mathbf{l_1}\ ,\ \dots\ , \mathbf{e_{i,t}}\ , \mathbf{p_{default}},\ \mathbf{p_{celltype}}\ , \mathbf{p_{disease}}\ ,\ \dots], \quad (3)$$

where $\mathbf{g_j}$ is the gene j encoding, $\mathbf{e_{i,j}}$ is the encoding of the expression of gene j in cell i, $\mathbf{l_j}$ is the gene j location encoding, and $\mathbf{p_A}$ is a learnt embedding for the class A.

The total count information is stored separately and encoded similarly to the expression,

$$\mathbf{e_{t,i}} = MLP(log_2(1 + t_i)), \text{ where } t_i = \sum_j x_{i,j}, \quad (4)$$

with $x_{i,j}$ the expression value of gene j in cell i, and the MLP is a two-layer neural network similar to the previous one.

The full cell total count (t) lets scPRINT model its denoising based on this required total count parameter.

The placeholder tokens (total count, default cell embedding, cell type, disease, sex, ethnicity, assay, organism) are learned embeddings that stay the same across all inputs. They only act as placeholders for the model to fill in during the forward process. At the transformer's output, they will have been modified to contain the embeddings requested. At least two are used, one containing the default cell embedding and another the profile's total depth. More tokens can be used, one for each predicted cell label.

Model

The model is a bidirectional autoencoder similar to BERT with n layers, h attention heads, and a dimension of d. It uses the flashattention2 methodology implemented in Triton to compute its attention matrix. It uses the pre-normalization technique, with a sped-up layer norm implemented in Triton's tutorial. It uses a stochastic depth with increasing dropout probability.

It has a 2-layer MLP with a 4x width increase in its hidden layer and a GELU activation function.

Expression decoder

scPRINT uses a novel expression decoder for foundation models, which outputs the parameters of a zero-inflated negative binomial (*ZiNB*) function for each gene *i* in cell *j*. The *ZiNB* distribution is defined as

$$X \sim ZiNB(\mu,\ \theta,\ \pi), \quad (5)$$

where the parameters $\mu,\ \theta,\ \pi$ are obtained from a multi-layer perceptron (MLP) applied to the expression embeddings outputted by the transformer model at its last layer (e), which are the :

$$\mu,\ \theta,\ \pi\ =\ MLP(\mathbf{e}) \quad (6)$$

The MLP is a two-layer neural network with dimensions [*d, d*, 3]

Based on the work of Jiang et al., zero inflation is the best distribution when considering a broad range of transcriptomic measurements, where some have enough dropouts, and a zero inflation term is needed to model it. In our case, and similarly to scVI, we define our *ZiNB* as

$$ZiNB(x \mid \mu,\ \theta,\ \pi)\ = \pi\delta_0(x)\ +\ (1\ -\pi)NB(x \mid \mu,\ \theta)\ , \quad (7)$$

where $\delta_0(x)$ is a point mass at zero, and $NB(x \mid \mu, \theta)$ is the negative binomial distribution with mean μ and dispersion θ.

With these parameters, the negative binomial distribution is represented in the following way

$$NB(x \mid \mu,\ \theta) = \frac{\Gamma(x+\theta)}{x!\Gamma(\theta)}\left(\frac{\mu}{\mu+\theta}\right)^x\left(\frac{\theta}{\mu+\theta}\right)^\theta , \quad (8)$$

where μ is the mean and θ the overdispersion parameter, representing the inverse of the dispersion. From Hibe et al., we know that this is a parameter change from the most used probability mass function (PMF) given by

$$P(X = x) = \binom{x\ +\ r-1}{x}(1-p)^r p^x \quad (9)$$

where r is the number of successes, p is the probability of success, and k is the number of failures.

One can interpret such a negative binomial distribution as a Poisson distribution with an additional overdispersion term that makes the variance not tied to the mean. In scPRINT, we use the zero-inflated Poisson for count downsampling as we can't easily infer the gene overdispersion parameter from each cell profile. By removing this zero-inflated Poisson from the gene expression profile, we keep the potential overdispersion in the profile (see the Negative Binomial to Poisson relationship section in Methods).

Compared to scVI, where the overdispersion parameter θ is learned for each gene, we make scPRINT output it together with $\mu,\ \pi$ (see 5. Supplementary Figure S13)

Effectively, the model learns that the dispersion might change depending on the gene, the sequencer, the cell type, and the sequencing depth.

Class decoder

scPRINT also outputs a variety of class embeddings, such as default cell embedding, cell type embedding, disease embedding, etc., by filling the different placeholder tokens given as input (see the Expression encoder section in the Methods).

Effectively, for each class, we have the model learn to produce a new disentangled embedding (e.g., cell type, disease, tissue, age). This means the model uses an MLP to transform each token where A is a class. For each, we jointly train a classifier :

$$\widehat{\mathbf{c}_\mathbf{A}} = \sigma(MLP_A(\widehat{\mathbf{e}_A})), \quad (10)$$

where :

- $\widehat{\mathbf{c}_A}$ represents the logits for a class A of a dimension d_A whose size corresponds to the number of labels.
- σ denotes the Sigmoid activation function.
- MLP_A stands for the Multi-Layer Perceptron trained to predict the logits of the class A.
- $\widehat{\mathbf{e}}_A$ is the output embedding for the class A of dimension d.

However, some classes, like cell type, have up to 800 labels. Fortunately, cellxgene classes follow an ontology, a robust structure that defines relationships among the labels. We reduce the size of the output labels by training the model only on the leaf labels in the ontology hierarchy (i.e., the most precise available). For cell types, this represents around 400 different labels (see 5. Supplementary Table S13).

Thus, when a label is not very specific for a cell type (e.g., neuron), the model will predict the best leaf label (e.g., dopaminergic neuron). This way, we can generate meaningful training signals from even very coarse labels (see The classification task section in methods for more information and definition of the loss). We only apply this hierarchical classifier to the cell type, disease, and assay labels.

In the following section, we show how we train such classifiers. During the classifiers' training, we sum up their loss without applying any scaling between the different classes.

1.5.2 Ablation study

We perform an ablation study of multiple of our additions in scPRINT for its medium size version. Removing positional encoding, replacing log-normalization with a total-normalization, replacing denoising with masking, using the cell-gene product method of scGPT vs our own encoder-decoder approach to learn a cell embedding, using 2 vs 4 heads per attention blocks, not using weighted random sampling, not freezing the gene ID embeddings, and using mean-squared-error instead of the ZINB loss. For each, we re-train scPRINT entirely on the same dataset and validate its test performance with our automated benchmark platform. We provide the results in Table S3.

1.5.3 Pretraining

The three tasks of the multi-task pretraining are the denoising task, the classification task, and the bottleneck learning task. While the denoising loss enhances the model's ability to find meaningful gene-gene connections, the other two try to make the model and its underlying networks more robust and cell-type-specific. All three losses are summed without rescaling.

Optimization method

The optimization is done with fused ADAMW, with a weight decay of 0.01. We noticed a total inability to learn when using base ADAM, which has a similar weight decay. This can be explained by a known inequivalence issue in ADAM.

We use the stochastic weight averaging method during training with a learning rate of 0.03.

During pre-training, the hyperparameters are set to dropout of 0.1, a learning rate (LR) of 1e-4, the precision is set to 16-mixed with residuals in fp32. We clip gradients to 100 and train in many sub-epochs of 7000 training batches and 2000 validation batches with a warmup duration of 500 steps.

Across epochs, we use a linear LR decrease of 0.6 with a patience of 1 and stop training after three consecutive increases in validation loss (patience : 3). In the final layer of the class decoders, we initialize values to a normal distribution around 1 for weights, 0 for biases, and -0.12 for biases.

Our batch size is 64, and we use a pre-norm strategy for the transformer with a linearly increasing stochastic depth dropout rate of 0.02 per layer. We use a noise parameter of 60%. We split the cells in the datasets into 98% train and 2% validation and reserve at minimum 2% of separated datasets for testing.

Finally, we use weighted random sampling on our training data based on the different class values we have to predict. We use a factor of 50, meaning the rarest elements will, on average, be sampled only 50 times less than the most common ones. The sampling factor used for each group is then $\frac{50}{count+50}$, instead of $\frac{1}{count}$ where count is the number of cells in each group.

The classification task

We perform label prediction during pretraining for different classes, currently : cell type, disease, sequencer, ethnicity, sex, and organism. Due to issues in the ontologies, we have omitted tissue type and age classes.

Due to the hierarchical structure of the prediction, we also created a hierarchical loss. Here, we compute the loss regularly when the label is a leaf label. Otherwise, we replace all

associated leaf labels to the given label by the log-sum-exp, such that for a cell label, the loss is :

$$Loss_{classification} = CE(\sigma(\bar{\mathbf{c}}, \mathbf{c}), \quad (11)$$

with :

$$\bar{\mathbf{c}} = \begin{cases} \widehat{\mathbf{c}} & if \ \{i \mid c_i = 1\} \subseteq T \\ LSE\,(\widehat{\mathbf{c}}_d) \,||\, \widehat{\mathbf{c}}_{\sim d} & else \end{cases} \quad (12)$$

where :

- $\widehat{\mathbf{c}}$ is the predicted vector with dimension equal to the number of leaf labels
- **T** being the set of label indices marking the labels that are leaf labels.
- $\widehat{\mathbf{c}}_d = \{\widehat{c_i}, \ \forall\, i \in T\}$ all the values in vector $\widehat{\mathbf{c}}$ whose indices are in T. Same for **c**.
- $\widehat{\mathbf{c}}_{\sim d} = \{\widehat{c_i}, \ \forall\, i \notin T\}$ all the values in vector $\widehat{\mathbf{c}}$ whose indices are not in T. Same for **c**.
- LSE is the log-sum-exp operation

The CE (cross-entropy) is defined as :

$$CE(\mathbf{p}, \mathbf{q}) = -\sum_u q_u log(p_u) \ . \ (13)$$

And the LSE (log-sum-exp) is defined as

$$LSE(X) = log(\sum_{p \in X} e^p). \ (14)$$

This loss allows the classifier to learn even in cases where the labels can be of varying coarseness without the coarseness of some labels impacting the ability of the model to predict the true fine-grained labels (see 5. Supplementary Figure S14)

The loss is hierarchical for the classes : cell type, disease, sequencer, ethnicity ; the labels follow a hierarchy defined by (Cell Ontology, MONDO, EFO, HANCESTRO), respectively.

We do not compute the loss for cells where a class has an unknown label. We perform these classification tasks in one pass, using the embeddings generated directly from the downsampled expression profile.

The denoising task

Similarly to ADImpute, we expect a good gene network to help denoise an expression profile by leveraging a sparse and reliable set of known gene-gene interactions. In addition, we expect a good cell model to help embed and reconstruct an expression profile by leveraging the regularities of modules and communities within its network.

We view denoising similarly to upsampling, and inversely, we view adding noise as downsampling a cell profile.

Noise is similar to downsampling because of the distribution we are working with. Note that contrary to vision tasks (e.g. diffusion models), where additive Gaussian noise is added, in the context of expression data, where the distribution is often seen as a Poisson, NB,

or ZINB, the data is already noisy, and the more counts are sampled, the less noise. No information is similar to not sampling data.

We downsample an expression profile using a zero-inflated Poisson model of the data. With this formulation, on average, half of the counts to be dropped are dropped by randomly removing a number of reads per gene, given by sampling from a Poisson whose lambda parameter is proportional to the number of counts in that gene. The remaining half of the counts to be dropped are dropped by randomly setting some genes to 0, i.e. a complete dropout of that gene. It is to be noted that with this definition of downsampling, the exact average amount of counts dropped for both parts depends slightly on the dropout *r*. During our pretraining, *r* is set to 0.6, meaning, on average, 60% of the transcript counts are dropped per cell.

Let $\mathbf{x_i}$ be the gene expression vector of cell i with dimensions n_{genes} ; we create a downsampled *version* by doing

$$\widehat{\mathbf{x}}_\mathbf{i} = max((\mathbf{x_i} - \mathbf{p_i}) \cdot \pi_\mathbf{i}, 0), \quad (15)$$

with :

- $\mathbf{p_i} \sim Poisson(\mathbf{x_i} \times r \times 0.55)$ a vector of size n_{genes} where the poisson is samples for each element $\mathbf{x_i}$ of x
- $\pi_\mathbf{i} = I(u \geq r \times 0.55)$ a vector of size n_{genes} , the binary mask vector indicating non-dropout genes.
- $\mathbf{u_i} \sim Uniform(0, 1)$, a vector of size n_{genes}. of random values drawn from a uniform distribution.
- $\cdot$ denotes the element-wise multiplication.
- *r* being the dropout amount. We scale it by a tuning hyperparameter of 0.55 instead of 0.5 for numerical reasons.

The goal of the model is then using $\widehat{\mathbf{x}}_\mathbf{i}$ as an input to output the parameters $\mu_\mathbf{i}$, $\theta_\mathbf{i}$, $\pi_\mathbf{i}$ of a *ZINB* distribution of the true profile $\mathbf{x_i}$, all vectors of size n_{genes}. The contribution of cell i to the loss is then computed as the negative log-likelihood of the count data given the distribution parameters being generated by the model

$$Loss_{denoising} = Loss_{ZINB} = -\frac{1}{n_{gene}m} \sum_{i=0,j=0}^{n_{gene},m} log(L(x_{i,j}|\ \mu_{i,j}\ ,\ \theta_{i,j}\ ,\ \pi_{i,j})), \quad (16)$$

where n_{gene} is the size of the expression profile $\mathbf{x_i}$, m is the size of the minibatch and

$$L\ (x|\mu, \theta, \pi) = \begin{cases} \frac{\pi}{\pi - \theta \bullet (\log(\theta) - log(\theta+\mu))} & if\ x = 0 \\ \frac{(\frac{\mu}{\theta+\mu})^x \cdot \Gamma(x+\theta) \bullet \sigma(-\pi)}{exp(\pi) \bullet \left(\frac{\mu}{\theta+\mu}\right)^\theta \bullet \Gamma(\theta) \bullet \Gamma(x+1)} & if\ x\ > 0 \end{cases} \quad (17)$$

with σ the sigmoid function.

We show that models trained with such a framework perform better than regular MSE-trained models (see 5. Supplementary Table S3), for which one only outputs one value instead of three, directly representing the data's log-transformed count. In this case, the loss is the mean squared error between the predicted and true count values.

scPRINT effectively lets the user choose between the three formulations : *ZINB* with a *ZINB* loss, NB with an NB loss, and direct log-transformed count reconstruction with an *MSE* loss.

However, we have noted that the *NB* and *ZINB* loss still have some notable issues. They can easily overflow, especially when working with lower precision systems (like fp16, bf16, etc). These losses are also proportional to the total expression count, meaning cells with higher expression will have a higher loss on average. It also appears that the log-likelihood cannot go below ~1.1 loss on average and plateaus quickly. This makes evaluation of the loss less practical when comparing models. Finally, this minimal loss also depends on the total number of zeros in the true expression dataset, as the zero-inflation part of the loss converges smoothly to 0.

The bottleneck learning task

Bottleneck learning is a method that drives the model to generate a cell expression profile only from its embedding. Cell-embedding which can be passed again to that same model without the gene expression information, such that from the cell-embedding only, scPRINT can re-generate the cell's expression profile. The model thus finds the best compression of the cell's expression according to the information-theoretic theorem by Tishbi et al. .

While many transformer models and Geneformer directly use the average of gene embeddings to generate a cell embedding, this will likely squash the expression information. scGPT used another methodology (called MVC) to generate an embedding vector such that

$$x_{i,j} = \mathbf{e_i} \odot \mathbf{g_j} \quad , (18)$$

where $x_{i,j}$ is the expression of gene j in cell i, and $\odot$ is the dot product. For each gene embedding $\mathbf{g_j}$, the embedding only contains information about the gene name, not gene expression. Regular MSE on each $x_{i,j}$ is then used as the training loss.

This pushes the cell embedding $\mathbf{e_i}$ to contain all the expression information of the cell i.

This is less computationally intensive to train than our bottleneck learning method. However, we have noticed poorer reconstruction through this methodology than ours (see 5. Supplementary Table S3).

In our case, we consider that our model scPRINT can act as two parts of an autoencoder. The encoding part is when we give scPRINT the expression profile of a cell and retrieve a set of disentangled cell embeddings (see the Class decoder section of the methods). The decoder part is when we provide scPRINT only the gene labels without their corresponding expression values and the disentangled cell embedding in place of the empty placeholder embeddings (see 5. SupplementaryFigure S15).

This means the encoder is considered as

$$\mathbf{e_{A,i}} = scPRINT([\mathbf{g_O} + \mathbf{e_{0,i}} + \mathbf{l_0},\ \mathbf{g_1} + \mathbf{e_{1,i}} + \mathbf{l_1},\ \ldots,\ \mathbf{p_A}]), (19)$$

where $\mathbf{e}_{A,i}$ is the output embedding of the placeholder embedding token A for the cell i (in our case, we use multiple (default, totalcount, cell_type, disease, sex, organism, ethnicity,

sequencer). Then the decoder is defined as

$$\mu_\mathbf{i}, \theta_\mathbf{i}, \pi_\mathbf{i} = scPRINT([\mathbf{g_0} + \mathbf{l_0},\ \mathbf{g_1} + \mathbf{l_1},\ ...],\ \mathbf{e_{0,i}},\ \mathbf{e_{1,i}},\ ...,\ \mathbf{e_{t,i}}), \quad (20)$$

With $\mu_\mathbf{i}$, $\theta_\mathbf{i}$, $\pi_\mathbf{i}$ vectors of size n_{genes}. Finally, the loss is given by the ZINB loss :

$$Loss_{bottleneck} = \sum_{i=0}^{m} Loss_{ZINB}(\mathbf{x_i}|\ \mu_\mathbf{i}\ ,\ \theta_\mathbf{i}\ ,\ \pi_\mathbf{i})\ , \quad (21)$$

where $\mathbf{x_i}$ is the cell i expression profile and m the minibatch size.

Implementing a set of disentangled embeddings is not straightforward. In our case, we push the embeddings to be as different from one another as possible with a contrastive loss defined as

$$Loss_{constrastive} = \frac{1}{m^2} \sum_{i=1}^{m} \sum_{i'}^{m} 1 - cos(\mathbf{e_i}, \mathbf{e_{i'}}), \quad (22)$$

where $\mathbf{e_i}$ and $\mathbf{e_{i'}}$ are the cell embeddings, m is the minibatch size, and *cos* denotes the cosine similarity. This pushes each embedding to represent the correct information using the classifiers. However, more is needed to remove all the batch effects or entirely prevent information leakage across embeddings.

Finally, we have also used the classifier output logits as cell embeddings. This works particularly well for cell type, disease, or sequencer classes containing many labels. It has been shown that classifier logit outputs behave similarly to embeddings and, in our case, offer an even better removal of the batch effects (see 5. SupplementaryFigure S7).

For the bottleneck loss, we directly reconstruct expression using the cell embeddings generated from the noisy, downsampled expression profile of the denoising process, doing the entire process in one single pass. We sum all the losses without scaling them :

$$Loss = Loss_{contrastive} + Loss_{bottleneck} + Loss_{denoising} + Loss_{class} \quad (23)$$

1.5.4 scDataloader

Parallel to this work, we worked with Lamin.ai to develop a dataloader for large cell atlases, described and benchmarked in Rybakov et al.. One key advantage of this dataloader is its ability to perform weighted random sampling on hundreds of millions of cells without being a bottleneck during pretraining. scDataloader samples cells amongst the 800+ datasets of cellxgene's mid-2023 release, using the cell labels to inform how rare the specific combination of labels is.

From this, the dataloader produces a cell sampling weight, rescaled with a hyperparameter. The dataloader will sample, with replacement, more consistently rare cell types than more common ones.

We have produced an additional wrapper package around the laminDB "mapped-dataset" called scDataloader. scDataloader works with lamin.ai but can also interface with scVI and AnnData formats to enable downloading, preprocessing, and QC of large single-cell databases and datasets. It is very flexible and can represent expression data in the formats used by scPRINT, scGPT, and Geneformer. It also implements a lightning datamodule scheme

and command line interfaces for quick setup (see 5. SupplementaryFigure S16).

Overall, we preprocess each of the 1200 datasets in cellxgene by only keeping primary cells from either humans or mice and dropping all the spatial omics datasets. Spatial omics are not true single-cell assays, and we decided for now not to include them. We also drop any cells with less than 200 expressed genes. Finally, we drop any resulting dataset smaller than 100 cells, with less than 10,000 genes, or from which more than 95% of the cells have been removed. This results in a new database of 54,084,961 cells and 548 datasets.

We believe that the weighted random sampling strategy allowed our pre-training to be much faster by creating more diverse minibatches.

1.5.5 Extracting meta-cell gene networks from attention matrices in scPRINT

Transformers compute multiple attention matrices per layer, called attention heads. This is done by splitting the generated $\boldsymbol{K}$, $\boldsymbol{Q}$, and $\boldsymbol{V}$ embedding into m sub-embeddings, thus defining m attention heads. Each attention head computes the attention matrix via the equation :

$$softmax(\frac{\mathrm{Q}\mathrm{K}^T}{\sqrt{d_k}}). \quad (24)$$

However, we would want to aggregate those over multiple cells from a similar cell state to increase the signal obtained from only one cell. We are doing so by averaging the Keys and Queries embeddings over the set of cells U passed to the model :

$$softmax(\frac{mean_U(\mathrm{Q})mean_U(\mathrm{K})^T}{\sqrt{d_k}}). \quad (25)$$

By doing this, the attention matrix behaves as if each query vector for cell i was "looking" across the key vectors of all the cells in U.

The resulting object is a row-wise normalized n^*n matrix, where n is the size of the input context (i.e. the number of genes passed to the model). However, we also include the possibility to generate large matrices and gene networks, referred to as genome-wide gene networks. We take the average over different sets of expressed genes for each cell in the set U. This allows us to compute a genome-wide attention matrix while only doing forward passes on smaller subsets of the genome per cell.

1.5.6 Heads selection

With scPRINT, we present a method to select heads based on some available ground truth data. This is inspired by the ESM2 paper and uses a somewhat similar method. Using all the available attention matrices from all of the model's heads, we use a linear classifier RidgeClassifier from scikit-learn (with an L2 penalty set to 1, a positivity constraint on the coefficients, and without an intercept) to classify the ground truth's edges based on a combination of each head. The classifier converts the target values into {-1, 1} equals to

{no connections, connections} and then treats the problem as a regression task with mean squared error.

Instead of taking the classifier's output, we use the average of the subset of heads associated with a non-zero coefficient in the classifier, without weighting them. Thus, the classifier only serves as a means to select the heads with relevant information in predicting a ground truth of interest and decreases the possibility of overfitting (see Figure 1C).

1.5.7 Normalization and network interpretation

In scPRINT and scGPT, the attention matrix is normalized via the softmax function over the query (i.e., row) dimensions. This means that all row elements sum up to 1 or that the same mass flows from each network component. This rescaling is essential as it corrects that some row element scales can be much higher than others in the attention matrix. Similarly, in regularized models like GENIE3, only a small set of genes are connected for each gene in the matrix, meaning all genes have directed edges toward a small subset of genes. Thus, our interpretation is that the row elements are the targets in our network, each connected to a small subset of genes. The column elements are thus the regulators and can regulate many / most genes in the network.

For biological ground truths like MCalla et al. and gwps, which fit this assumption of highly connected regulators and sparsely regulated targets, we directly compare them to the inferred network. Tables S12 and S13 show that this performs better than taking the opposite view by transposing the inferred networks.

This assumption is challenged for Omnipath, which has most of its elements connected to a sparse set of other elements (see 5. SupplementaryFigure S3). Due to the sparsity of connections for regulators (i.e., sources) in the ground truth network and the large number of regulators (8000+), the methods are challenged and perform much better when taking the transpose of their network and matching the regulators to the sources and sources to regulators.

1.5.8 Simulated datasets, BoolODE and Sergio

BoolODE is a method to generate count data via a stochastic differential equation applied over a user-defined Boolean network. It was used and developed as part of the BEELINE benchmark algorithm, which was created as an improvement over the GeneNetWeaver algorithm. However, this model is still very simple compared to cell biology. Due to its computational complexity, it can only model up to a couple hundred gene relationships over a few dozen genes.

Sergio, a slightly more recent ODE model marks an improvement over BoolODE on the size of the networks it can simulate (up to a thousand genes) and its similarity to scRNAseq data.

Indeed, Sergio's simulated data is not similar to real expression data. This means that the

biases that Transformer models learn should not help them predict Sergio's data. Correlation and regression-based methods do not have biases. They are therefore expected and have traditionally shown better performance on these benchmarks.

We generated the Sergio ground truth network and simulated single cell expression by using the notebook : `https://github.com/g-torr/SERGIO/blob/v2/minimal_example.ipynb` from the repository : `https://github.com/g-torr/SERGIO` which present some debugs and improvements to the initial repository : `https://github.com/PayamDiba/SERGIO`. Indeed only this fork of the initial Sergio repository led us to successfully generate a network.

We used RegNetwork as input and simulated 1000 cells from its 3546 connections over 813 genes with default parameters from the notebook.

1.5.9 BenGRN and gene network metrics

We use the packages benGRN and GRnnData released with this manuscript to work With Gene networks and perform our benchmarks.

Our three main metrics are EPR, AUPRC, and enrichment. They all take advantage of the fact that the predictions are generated as scores over edges between nodes :

- We have computed the Early Precision Ratio (EPR) as the diagnostic odds ratio : (TP x TN) / (FP x FN) at the cutoff of the scores giving K positive predictions, where K is the number of positive elements in the ground truth.
  In this context, 1 is a random prediction, and inf is a perfect prediction ; values below one mean that inverting the predictor would provide better results.
- Area Under the Precision-Recall Curve (AUPRC) is the area (computed with the composite trapezoidal rule) under the curve defined by the precision ($PR = TP / (TP + FP)$) and recall ($RE = TP / (TP + FN)$) where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives. This curve is obtained through a range of cutoffs going from 0 predicted positives to all predicted positives. Here, we compute a version of the AUPRC where the floor of the area is not given by the Precision=0 line but by the line of the prevalence of the positive class. Moreover, we do not interpolate the curve between the last recall value and the perfect recall : 1. We do this to properly compare AUPRC values across benchmarks and models. Random precision values are given in the supplementary data.
- Enrichment is computed using the prerank methodology, where, given an ordered set of genes, it is computed by :
  - 1. Summing all scores of edges of the matrix row-wise. (Target - Hub) Or
  - 2. Summing all scores of edges of the matrix column-wise. (Regulators - Hub) Or
  - 3. Computing the eigenvector centrality of nodes in the graph using NetworkX's implementation. Prerank's background comprises all the genes in the set (centrality).

Of note, we did not design an automated method for cell-type enrichment. Instead, the assessment of whether or not a network is enriched for the correct cell type is done manually, identifying cell type names in the top 10 cell types listed in the enrichment results of the network.

1.5.10 Other evaluation metrics

All evaluation metrics from the section "scPRINT is competitive on tasks orthogonal to GN inference" of the results come from the openproblems benchmark and are standards in the field.

scIB's batch correction score is an average of the avgBatch score and the avgBio score, which are themselves averaged over many scores. Details of each value are available in our package's notebooks.

- scIB avgBio is a combination of label-based and label-free metrics using for example : the Adjusted Rand Index (ARI) and the Normalized Mutual Information (NMI) on clusters computed from the K-Nearest Neighbor graph. Other scores are used, some using the conservation of trajectories and of the cell cycle variance, and some on the rare cell population conservation, overlap of highly variable genes (see scIB), and more.
- scIB avgBatch is a similar combination of label-based and label-free metrics, using, for example, the average connectivity across clusters of different batches : ASW, the graph integration local inverse Simpson's Index : graph iLISI, the k-nearest-neighbor Batch Effect Test (kBET), and more.

Finally, we also use two metrics in our classification task :

- Macro-F1 : also called macro-average, is the average of the F1 score across each class in a multi-class task. Where the F1 score is : $2 \times \frac{PR*RE}{PR+RE}$.
- Accuracy : the accuracy is computed as $\frac{TP + TN}{TP+TN +FN+FP}$

1.5.11 Denoising Benchmarks

To validate the denoising ability of scPRINT, MAGIC, and KNNsmoothing2, our test function, available in the scPRINT package, uses a representative subset of 10,000 cells of each dataset to generate the denoised expression over the 5000 most variable genes in this dataset.

Before that, counts are removed from the dataset following the same procedure as done for scPRINT's pretraining (see The denoising task section of the methods).

For each cell, we compare the denoised and un-denoised profiles to the true profile (e.g. before denoising). We compute the Spearman's correlation over the genes initially expressed

in the cell, taking the average across all cells. We do not use the unexpressed genes as we are working with a dataset with high dropout and expect that a good denoiser will set genes that are 0 in the profile with some value. We notice that this improves the score of all denoising methods and makes more sense given the data.

For the rare cell population test, we keep everything similar but compute only the Spearman correlation over a rare cell population in the dataset.

We run KNNsmoothing2 with default parameters and a K of 10. We run MAGIC using the Scanpy implementation with default parameters and the approximate solver for computational speed. When computing KNNsmoothing2 or MAGIC over a small set of cells we use a K of 5 for the nearest neighbors.

1.5.12 Fine-tuning

Contrary to most other foundation models for scRNAseq, we do not finetune scPRINT at any moment in our benchmark and all results are provided for the pre-trained model only.

While we haven't assessed fine-tuning we believe this is an important feature of foundation models and release various scPRINT models so that they can be re-trained, fine-tuned, and modified by the community for novel tasks or to improve its performance on the tasks we have presented.

1.5.13 State-of-the-art methods used in benchmarking

All methods presented here generate networks from their input data. Given gene-level expression data, they will generate gene-networks. Without additional information, no method can distinguish the type of molecular interactions that underpin their predicted network edges.

Gene network inference with an ensemble of trees (GENIE3)

Developed originally for bulk transcriptional data, *GENIE3* computes the regulatory network for each gene independently. It uses a random forest, a weak learner ensemble method, to predict the expression profile of each target gene from profiles of all the other genes. The weight of an interaction comes from the feature importance value of an input gene in the predictor for a target gene's expression pattern. Aggregating these weighted interactions over all the genes yields the regulatory network. This method was the top performer in the DREAM4 in silico network challenge (multifactorial subchallenge).

GENIE3 can be seen as a generalization of correlation-based methods for inferring gene networks. Instead of looking at genes that correlate most with another gene, GENIE3 finds how to combine a set of correlated genes to get an even better correlation. We run GENIE3 on raw counts as it is said from both the BEELINE benchmark and the R package vignette that GENIE3 can be run on either log normalized or raw count data and that while it will

change the results, there are no preferred methods. This is something we have also noted in our trials.

We use all default parameters and choose 100 trees for computational feasibility reasons. We compute the networks on the same set of cells and genes as the other methods.

We also use a TF-gene only version of the method where the regression is performed only using the expressed transcription factors instead of all expressed genes as input. This is the most used version of *GENIE3* and is much faster.

DeepSEM

DeepSEM is an autoencoder model made for gene network inference. It learns to decompose a set of cells as a set of embedding and an adjacency matrix (i.e., a gene network). The formula of the VAE then becomes : $\mathbf{X} = f_1((\mathbf{I} - \mathbf{W}^\top)^{-1}\mathbf{Z})$, for the decoder and $\mathbf{Z} = (\mathbf{I} - \mathbf{W}^\top)^{-1} f_2(\mathbf{X})$ for the encoder, where $\mathbf{X}$ is the expression data, $\mathbf{Z}$ is the embedding dimension, $\mathbf{W}$ is the adjacency matrix, $\mathbf{I}$ the identity, and f_1, f_2 are MLPs.

We preprocess the anndata by normalizing gene expression to 10,000 genes, applying a logp1 transformation, and then computing the z-score per gene, as explained in the associated research paper.
We use DeepSEM with default parameters and on the same set of cells and genes as the other methods. We use the DeepSEM-provided functions for loading and parsing Anndatas.

Single-cell generative pretraining transformer (scGPT)

scGPT is a transformer-based model of roughly 100M parameters, pre-trained with a generative process similar to Language models. scGPT proposes to build similarity networks based on the output gene embeddings of the model but also based on its attention matrices. It computes networks as the difference between the rank-normalized version of the average attention matrix in a baseline expression profile vs a perturbed one in perturb-seq data. The attention matrix is the average of attention matrices over the heads of the last layer and over the cells given to the model.

We run scGPT following the examples given in their “Tutorial_Attention_GRN.ipynb” notebook.

We use the “scGPT_human/best_model.pt” from the list of available models with default parameters. All runs are in our fork : “https ://github.com/jkobject/scGPT” in the “mytests/” folder. Similarly, we take the mean over cells and over the heads of the last layer. We compute softmax similarly to the attention computation but without applying the rescaling factor $\sqrt{d_k}$. We finally drop the first element corresponding to the cell embedding token.

We extract cell embeddings from scGPT by directly using the cell embedding token of the model without fine-tuning it on a batch correction task. This is done in order to compare it to scPRINT which is itself not fine-tuned. We compute the networks on the same set of cells and genes as the other methods.

Geneformer

Geneformer is a BERT model. Gene expression data is transformed into a sentence or genes ordered by their scaled expression. It is trained with mask language modeling and contains somewhere around 80M parameters. We use the new versions of 2024 Geneformer models trained on 100M cells (2x more than scPRINT). We follow the preprocessing and inference scripts used in the geneformer huggingface repository and notebooks : `https://huggingface.co/ctheodoris/Geneformer/tree/main`. Our inference script updates to extract gene networks from Geneformer are available in our scPRINT repository : `https://github.com/cantinilab/scPRINT/tree/dev/tools`.

We extract gene networks from Geneformer using the mean of all attention heads per cell. Since Geneformer only uses expressed genes in a cell, we have to map the attention matrices back to the full network size before computing its average over cells, taking into account the NaN values. We compute the networks on the same set of cells and genes as the other methods.

We extract a cell embedding from Geneformer using the cell embedding from the "gf-12L-95M-i4096_MTLCellClassifier_CELLxGENE_240522" model that has been fine-tuned on predicting the cell labels of cellxgene datasets.

scFoundation

scFoundation is a foundation model for single-cell RNAseq based on the xtrimogene architecture. It was built by the Biomap company. It is able to work on the full genome sequence of transcripts for each cell by considering the high number of zeros and embedding them separately. The tool is aimed at performing a range of tasks, such as denoising, embedding, and predicting perturbation response. It has been trained with a mixed masking and denoising pre-training. However, we could not compare scFoundation to scPRINT and MAGIC on the denoising benchmark, as scFoundation's denoising only happens at the level of the cell embedding at inference time.

We could not validate scFoundation on our Gene network inference benchmark as extracting a network from the attention matrices was much more complex due to the xtrimogene architecture. scFoundation mentions the generation of gene modules using clustering of its output gene embeddings. It also mentions the interference of gene networks. However, it is achieved using RcisTarget, a prior gene network based on motif analysis. This approach is not comparable to the gene networks generated by scPrint, Geneformer, and scGPT. Indeed, RcisTarget could be applied to every model we have benchmarked and would prevent us from doing an unbiased benchmark. Neither our approach nor Hao et al.'s could extract gene networks directly from scFoundation. It is being left to further investigations.

For batch effect correction, we use scFoundation with default parameters and follow the steps for cell embedding in the "model/README.md" file in their GitHub repository : https://github.com/biomap-research/scFoundation. However, we give scFoundation single cell profiles of the 5000 most variable genes in each dataset. This is because we could not

run scFoundation on genome-wide expression profiles with our GPU. We then apply a PCA to the output embedding to reduce the dimensionality from 3224 to 512. This is because the initial dimension was too high for scIB to compute a score from on our machine (40CPU Intel Xeon, 32GB RAM + 64GB SWAP, GPU NVIDIA A4500 with 20GB of memory).

Marker-based cell type prediction with CellTypist

To showcase the novel ability of scPRINT to perform zero-shot prediction of cell type labels, we use the CellTypist method, which similarly performs de-novo prediction of cell type labels given its precomputed databases of cell type markers.

CellTypist works by mapping cell gene expression to genes known to be specifically expressed in combination in a cell type. Thus, it predicts cell type from these marker genes.

We use it with default parameters on the normalized and log-transformed counts over the full set of genes in the dataset. We use the 'Adult_Human_PancreaticIslet' database, which contains markers for 14 cell types and overlaps with only four of the cell types in the dataset.

We decided to still use it as is to showcase the marker-based method's inability to recover the full set of cells and the tradeoff between the number of cell types and accuracy.

Fortunately, these four cell types (A, B, D, PP) represent 70% of the dataset. With its current database, CellTypist can only reach a maximum accuracy of 70%. Even when taking this into account, CellTypist only overperforms scPRINT on the accuracy metric and by roughly 9 points.

Classification benchmark and associated methods

Our classification benchmark is run using following the openproblems benchmark. It uses the same input, output data, and metric. It also similarly splits the train-test by batch and preprocesses the expression matrix to what is presented in the open problem benchmarks.

For this task, methods can access the full set of genes by default. scPRINT will use its random sampling of genes approach with a context of 4000 genes. Classifiers like logistic regression and xgboost were run according to the openproblem process, using the 25 principal components of the count normalized, logp1 transformed expression data. CellTypist was run on the normalized and logp1-transformed cell expression profile.

1.5.14 Ground truth preparation

McCalla et al.

For the MCalla et al. dataset, we downloaded the data from the supplementary datasets of their paper . After undoing the logp1 transform, we re-generate the true count expression

matrix from the normalized one by dividing the expression of each cell by the smallest value in its expression profile. This fully recovered the true counts, all values being integers. For the additional human dataset we used, we downloaded it from the gene expression atlas database.

We used the intersection (gold standard) ground truth dataset for both human and mouse, converting this list of sources to target genes into a directed binary network.

Omnipath

We generate the Omnipath network using all the interactions from the Omnipath Python package, excluding small molecules, lncRNAs, and any element without a unique HGNC symbol. We then transform it into a directed binary network of source to target. These interactions are extracted from the literature and represent mainly TF to gene connections as well as many protein-protein interaction connections and a small number of other connections known from the literature like RNA-RNA interactions, protein-RNA interactions, and more. All interactions are mapped back to their gene IDs, generating a gene-gene network encompassing the various interactions the genes and their molecular products can have.

Gene networks from genome-wide perturb-seq

We created a gene network from the genome-wide perturb-seq dataset using the supplementary matrix containing the results of differential expression in the dataset. This matrix represents the multiple hypothesis testing corrected p-values of a differential expression test of cells with KO of gene A compared to the baseline cell expression. This is available for all 8000+ expressed genes in the K562 cell line. We used a cutoff of 0.05 on these values to define the directed binary connection between genes.

This effectively gives a gene x gene-directed binary graph that tells if a statistically significant connection exists from the source $gene_A$ to the target $gene_B$ according to genome-wide perturb-seq.

For all ground truths, download, preprocessing, and extraction of the network and expression data are available in the BenGRN package.

1.5.15 Details on the Benign Prostatic Hyperplasia analysis

We download our dataset from cellxgene under the reference : *574e9f9e-f8b4-41ef-bf19-89a9964fd9c7*.

We preprocess the dataset using scDataloader's preprocessing function. We generate embedding and classification using 3000 expressed genes in each cell. Similarly to pretraining, we take 3000 randomly expressed genes ; if less than 3000 are expressed, we complete with randomly selected unexpressed genes. We display embeddings generated using the cell type classifier logits (see section The classification task in methods)

We use the Scanpy toolkit to generate our Umap plots directly from the embeddings, as well as our differential expression results and our clusters. We define the clusters using the Louvain algorithm with 10 k-nearest-neighbors and a resolution of 1. We perform denoising on 5000 genes per cell selected similarly to the embedding and classification part. We use the 4000 most variable genes in each cell type to generate our gene networks in the BPH and normal fibroblasts.

On the gene networks, we perform gene set enrichment with the Enrichr method on the GO_MF_2023 gene sets. For community detection, we use the Louvain algorithm with parameter 1.5. We perform analysis only on the communities with between 200 and 20 genes. (4 and 5 in the BPH-associated fibroblasts, 3 and 4 in the normal fibroblasts)

All analysis and results are available in the *cancer_usecase_1* and *cancer_usecase_2* notebooks.

1.5.16 Negative Binomial to Poisson relationship

As explained in The denoising task and Expression decoder section of the methods, in our model, we have used the ZINB as our loss, an extension of the NB distribution to zero-inflated data.

Moreover, we have also used the zero-inflated Poisson mechanism to downsample the cell expression profiles. These are consistent because we can view the Poisson distribution as a NB without overdispersion. The relationship between *NB* and *Poisson* is given by making the dispersion term go to 0 and the inverse dispersion term $\theta \longrightarrow \infty$. Doing so, the term $\frac{\theta}{\theta+\mu}$ approaches 1. Thus, the PMF simplifies to :

$P(X = x) \approx \frac{\Gamma(x+\theta)}{x!\Gamma(\theta)} 1^{\theta} (\frac{\mu}{\theta+\mu})^{x}$ (26)

For large θ, we use Stirling's approximation of the Gamma function : $\Gamma(\theta) \approx \sqrt{2\pi\theta}(\frac{\theta}{e})^{\theta}$

we get :

$\Gamma(x+\theta) \approx \sqrt{2\pi(x+\theta)}(\frac{x+\theta}{e})^{x+\theta}$ (27)

$\Gamma(\theta) \approx \sqrt{2\pi\theta}(\frac{\theta}{e})^{\theta}$ (28)

Simplifying the ratio of the Gamma functions :

$$\frac{\sqrt{2\pi(x+\theta)}(\frac{x+\theta}{e})^{x+\theta}}{\sqrt{2\pi\theta}(\frac{\theta}{e})^{\theta}} = \sqrt{\frac{x+\theta}{\theta}}(\frac{x+\theta}{\theta})^{\theta}(\frac{x+\theta}{e})^{x} . \quad (29)$$

For large θ, $\frac{x+\theta}{\theta} \sim 1$, so : $\sqrt{\frac{x+\theta}{\theta}} \approx 1$

$$(\frac{x+\theta}{\theta})^{\theta} \approx 1$$

Thus, the expression simplifies to :

$P(X = x) \approx \frac{1}{x!}(\frac{\mu}{\theta+\mu})^x(\frac{\theta + x}{\theta})^x$ (30)

Finally, $(\frac{x + \theta}{\theta + \mu})^x \approx 1$ for large θ, so :

$\lim_{\theta\to\infty} P(X = x) = \frac{\mu^x}{x!}e^{-\mu}$ (31)

This is the PMF of the Poisson distribution with mean μ.

1.5.17 Data availability

The model weights are publicly available on Hugging Face. Pre-training logs to assess the model's training are available on Weights and Biases. The full pre-training dataset is publicly available on CellxGene under its census data release version : LTS 2023-12-15, accessible at `https://cellxgene.cziscience.com/`. All other datasets used in this work can be downloaded through their respective public databases via the helper scripts on the scPRINT, BenGRN, GRnnData, and scDataLoader packages. Source data are provided to re-generate the figures. Code to generate the large UMAP of Figure 1 is available as a notebook on GitHub at `https://github.com/cantinilab/scPRINT/blob/1.6.4/figures/nice_umap.ipynb`. Code to re-generate the source data is available as notebooks on our Github.

1.5.18 Code availability

The code and notebooks used to develop the model, perform the analyses, and generate results in this study are publicly available and have been deposited in cantinilab/scPRINT at `https://github.com/cantinilab/scPRINT` under MIT license. The specific version of the code associated with this publication is archived in the same repository under the tag 1.6.4 and is accessible via `https://github.com/cantinilab/scPRINT/tree/1.6.4/` and DOI :10.5281/zenodo.14749466.

Additional developed packages for this analysis are defined in the pyproject file and project submodules. They are available on GitHub :

- **GrnnData** : `https://github.com/cantinilab/GRnnData`, DOI :10.5281/zenodo.10573141
- **BenGRN** : `https://github.com/jkobject/benGRN`, DOI :10.5281/zenodo.10573209
- **scDataLoader** : `https://github.com/jkobject/scDataLoader`, DOI :10.5281/zenodo.10573143
- **scGPT and notebooks to reproduce the results** : `https://github.com/jkobject/scGPT/tree/main/mytests`

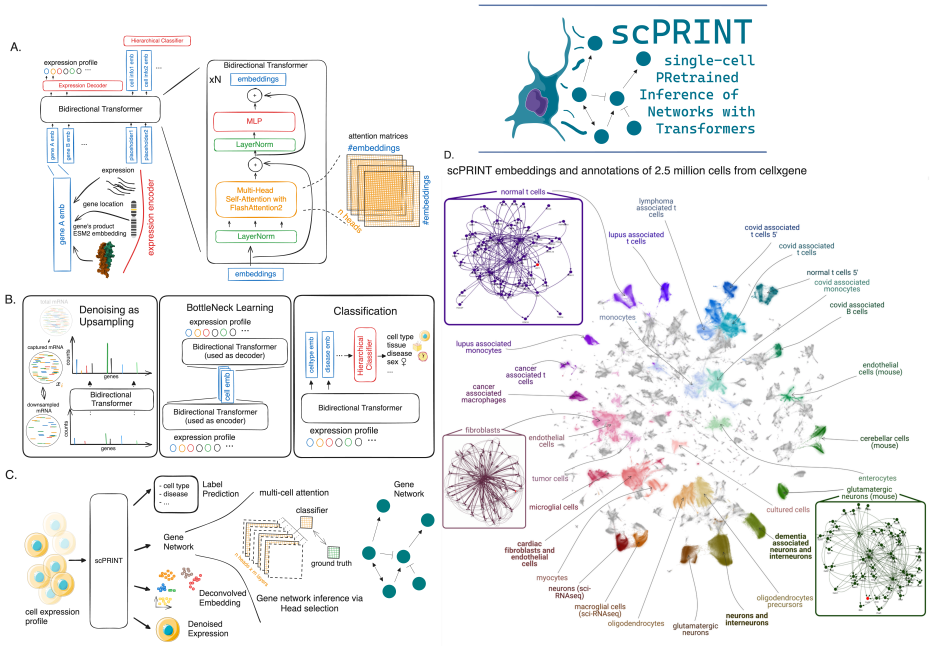


Figure 1.1 – (a) Schematic representation of scPRINT with its bidirectional encoder, gene expression embedding encoding via gene location, matched protein ESM2 embedding, and gene expression. (b) scPRINT pre-training tasks : Denoising task whose goal is to recover the known transcriptomic profile from a purposefully downsampled expression profile. Bottleneck learning reconstructs the expression of requested genes using only their cell embedding. The same model is used for both The encoding and decoding steps. Hierarchical classification is achieved by applying a hierarchical classifier to each disentangled embedding. This pushes the first embedding to contain cell type info, the second embedding to contain disease info, and so on (see 1.5. methods). (c) The different outputs in scPRINT. scPRINT generates label predictions of cell type, tissue, disease, sex, sequencer, ethnicity, and organism. scPRINT generates multiple embeddings (which we call disentangled embedding), a general one, as well as a specific embedding for each class. scPRINT also generates a reconstructed expression profile at any requested sequencing depth (i.e., total transcript count) (denoising). scPRINT also generates a Gene Network by selecting and combining various attention heads into a gene x gene matrix. (d) Example of a scPRINT output from a random subset of 2.5 million cells from the cellxgene database. Embeddings and labels are generated by scPRINT, together with the example cell type-specific gene networks. We show only subparts of the networks extracted from a central node, represented in red.

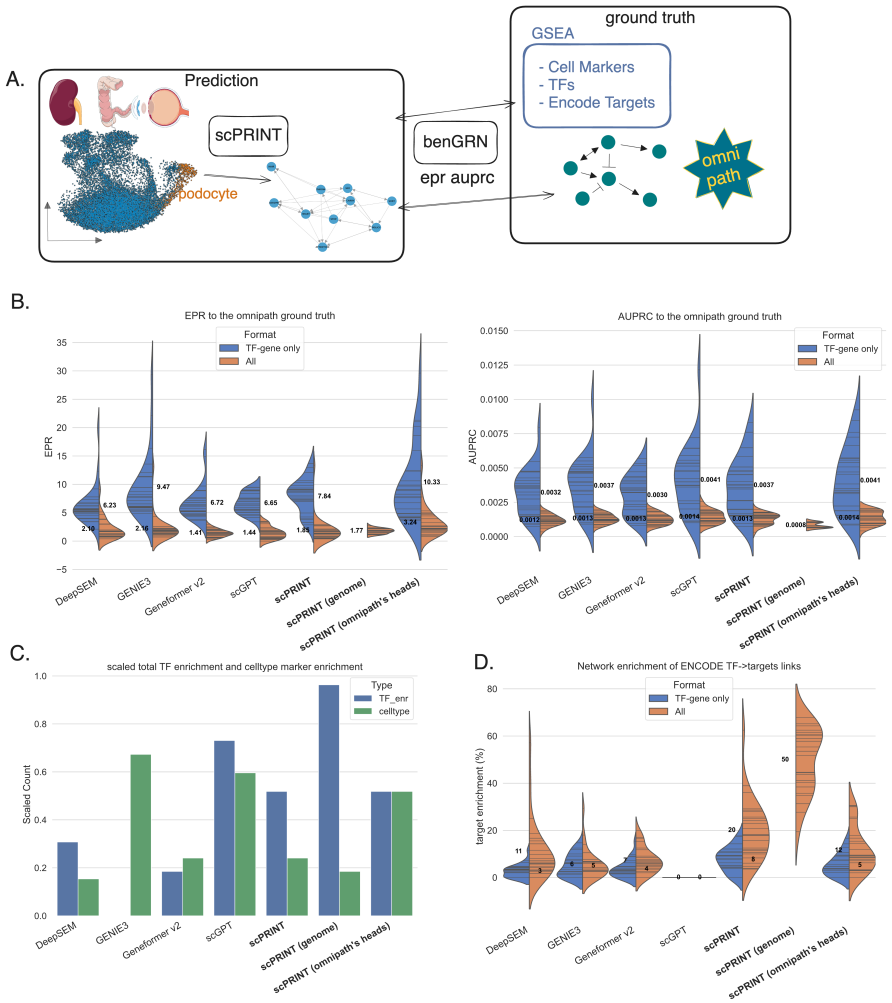


FIGURE 1.2 – (a) We extract cell type-specific gene networks for each cell type in the dataset (n=26 cell types across 3 datasets). We perform Gene Set Enrichment Analysis (GSEA)[**subramanianGeneSetEnrichment2005a**] on the network's nodes (n=4000 genes). We compute the ability of the edges to recover the Omnipath ground truth's connections. (b) Violin plot of the ten different Area Under the Precision Recall Curve (AUPRC) and Early Precision Rratio (EPR) values obtained when comparing the inferred cell type-specific networks with the Omnipath network for scPRINT : average of all attention heads, scPRINT (genome) : same scPRINT version but computing a genome-wide gene network, scPRINT (omnipath's heads) : same scPRINT version but with attention heads selected using a subset of omnipath, scGPT, DeepSEM, Geneformer v2, and GENIE3, when considering only Transcription Factor (TF)-gene connection or all gene-gene connections. (c) Violin plot of the average number of TF with enrichment for their ENCODE target in each cell-type-specific network. (d) Number of GNs with a significant enrichment of TFs and of their cell type's marker genes.

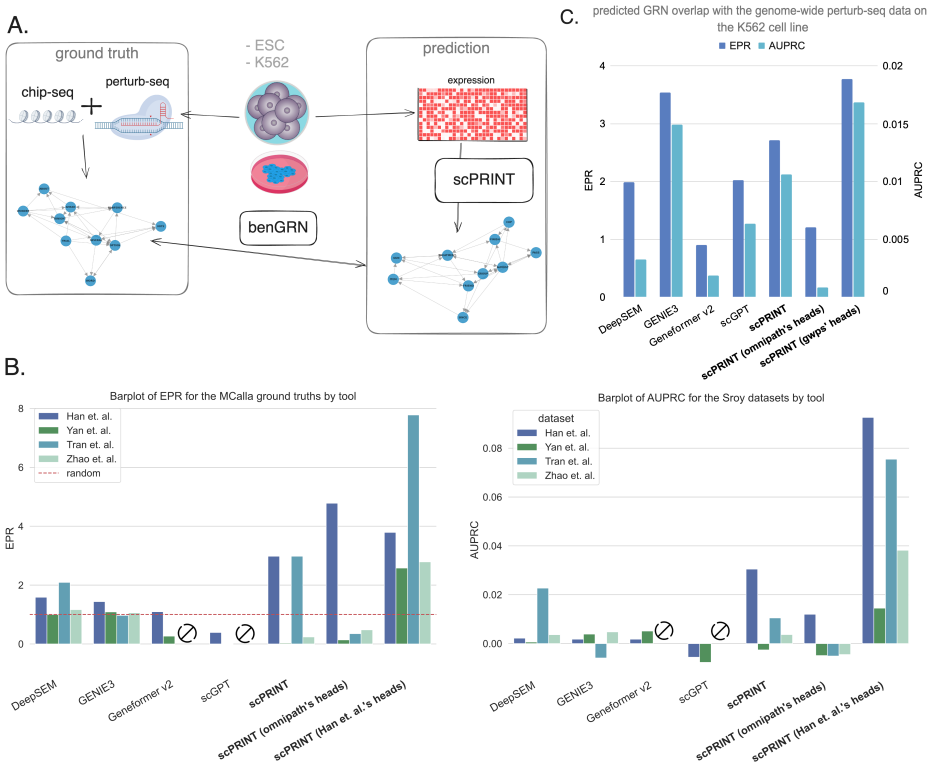


Figure 1.3 – (a) The ground truths are generated via orthogonal sequencing assays on the same cell type. ChIP-seq and perturb-seq are intersected for the MCalla et al. dataset on human (hESCs) and mouse (mESCs) Embryonic Stem Cells, whereas perturb-seq on the K562 cell line is only used for the genome-wide perturb-seq ground truth. (b) Performance of scPRINT, scPRINT (omnipath's heads) : same scPRINT version but with attention heads selected using a subset of omnipath, scPRINT (Han et al.'s heads) : same scPRINT version but with attention heads selected using a subset of the Han et al.'s ground truth dataset, compared to GENIE3, DeepSEM, Geneformer v2, and scGPT on the MCalla et al. ground truth using the AUPRC and EPR on two human and two mouse ESC datasets. (c) Same as (b) but on the genome-wide perturb-seq dataset with scPRINT (Han et. al.'s heads) replaced with scPRINT (gwps' heads) : same scPRINT version but with attention heads selected using a subset of the genome-wide perturb-seq ground truth. Early Precision Ratio (EPR) and Area Under the Precision-Recall Curve (AUPRC) are provided here in one barplot, left to right.

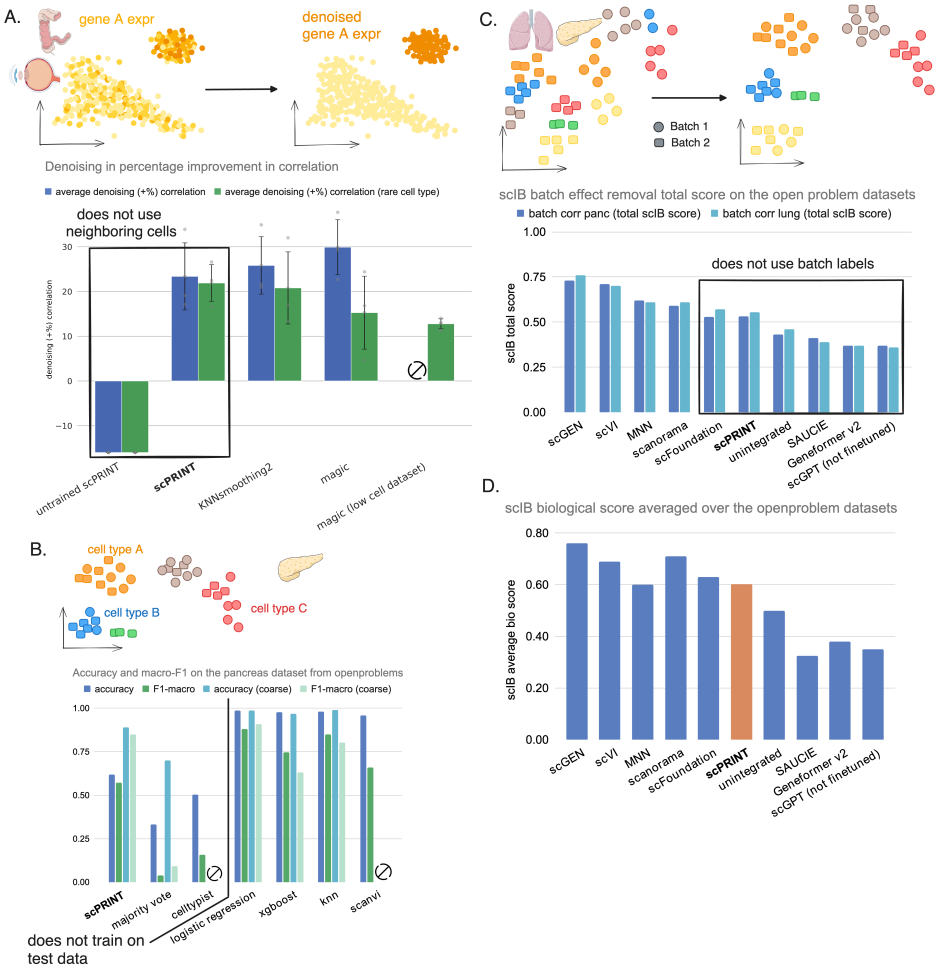


FIGURE 1.4 – (a) Performance for a denoising task compared to state-of-the-art methods MAGIC and knnsmooth2 on 3 datasets (ciliary body, colon, and retina tissues) from cellxgene. Here, we generate a noisy profile by downsampling 70% of the cell transcripts and computing the Spearman correlation increase of the correlation between the denoised and the true profile compared to the one between the noisy and the true profile. (b) Performance on cell-type label prediction compared to state-of-the-art methods as well as CellTypist. Showing accuracy, F1 and macro-F1 scores for the open-problems human pancreas dataset. (c) The performance of scPRINT as well as scGPT and Geneformer v2 on batch effect correction on the human pancreas and lung datasets from the openproblems challenge showing the scIB aggregated score. They are compared to state-of-the-art methods which results were extracted from the openproblems benchmark. Unintegrated means only PCA was applied. (d) The scIB avgBIO score on both datasets.

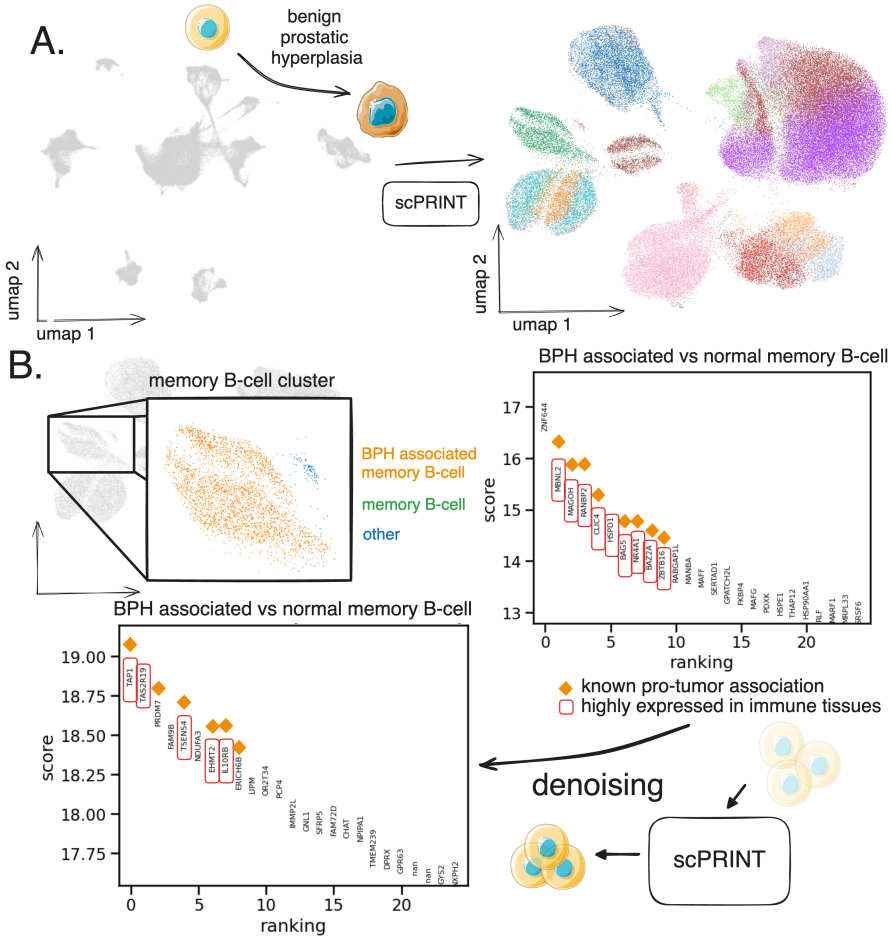


Figure 1.5 – (a) Single-cell RNAseq atlas of benign prostatic hyperplasia (BPH) and normal prostate tissues of 83,000 cells given to scPRINT. scPRINT generates a set of embeddings and label predictions for each cell. To clean our predictions, we drop cell types with less than 400 cells and diseases with less than 1000 cells, replacing them with the “other” label (see 5. Supplementary Figure S8). (b) Zooming in on one cluster, we see annotations of a switched memory B-cell cluster, some labeled "benign hyperplasia" and others "normal". Differential expression analysis on the two groups of B-cells showing enrichment of B-cell & cancer markers when assessing its top 10 genes. We performed upsampling of the transcript count before performing a new differential expression analysis where we now see new genes amongst the top 10 differentially expressed ones some of them also associated with cancer and immune tissues.

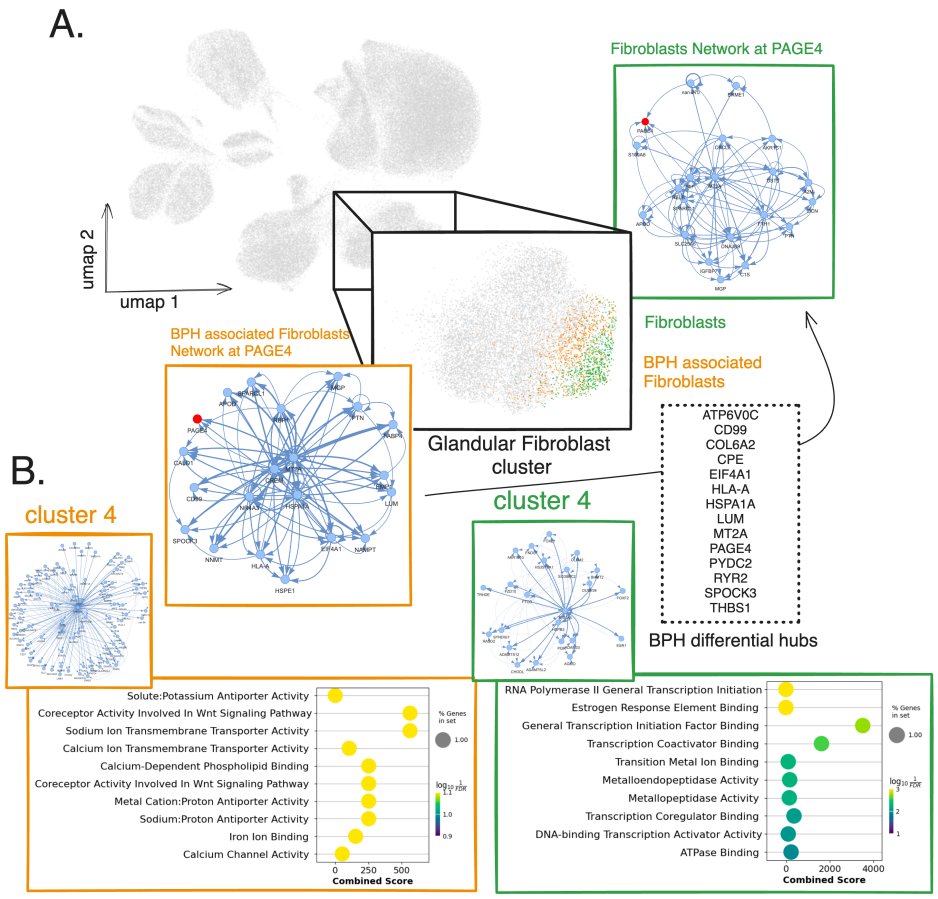


FIGURE 1.6 – Continuing on the single-cell RNAseq atlas of benign prostatic hyperplasia (BPH) and normal prostate tissues of 83,000 cells given to scPRINT (a) Zooming in on another cluster from scPRINT's cell embeddings and annotations, we see a group labeled as "fibroblast of connective tissue of glandular part of prostate", some labeled as "benign prostatic hyperplasia", and others "normal". We generate gene networks from each and highlight a sub-network of the PAGE4 differential hub gene in BPH, showing different connection strengths and patterns between normal and BPH-associated fibroblasts. (b) Left to right : gene-set enrichment analysis, using Enrichr, of the gene community 4 found by the Louvain algorithm in the BPH-associated fibroblast gene network, same but on the normal fibroblast gene network. It shows the top 10 most strongly enriched gene sets from GO_MF_2023 according to q-value (i.e. adjusted p-value).

Xpressor : Towards foundation models that learn across biological scales

2.1 Summary

We have reached a point where many bio foundation models exist across 4 different scales, from molecules to molecular chains, cells, and tissues. However, while related in many ways, these models do not yet bridge these scales. We present a framework and architecture called Xpressor that enables cross-scale learning by (1) using a novel cross-attention mechanism to compress high-dimensional gene representations into lower-dimensional cell-state vectors, and (2) implementing a multi-scale fine-tuning approach that allows cell models to leverage and adapt protein-level representations. Using a cell Foundation Model as an example, we demonstrate that our architecture improves model performance across multiple tasks, including cell-type prediction (+12%) and embedding quality (+8%). Together, these advances represent first steps toward models that can understand and bridge different scales of biological organization.

2.2 Introduction

Biology processes information across different scales, from individual molecules to entire tissues. Recent advances in artificial intelligence have led to the development of foundation models that excel at representing biological data at specific scales, such as protein structures [**esm2**] or cell states [**scprint**, **cuiScGPTBuildingFoundation2024**]. However, these models typically operate in isolation, unable to leverage the rich interconnections between different biological scales. Having models that can learn across biological scales will be crucial to capture the complexity of the biological phenomena.

The main premise of our work is that by using information gained from a lower scale (e.g., molecules), we might improve the input representations of an higher scale phenomena

(e.g., cells) [**bunneHowBuildVirtual2024**, **songAIDrivenDigitalOrganism2024**]. Reciprocally, using relationships learned at the higher scale, we might improve the lower-scale models too. Finally, we would want to use joint representations of molecules, DNA, proteins, cells, and tissues, which are all the elements of the organisms we want to study.

While it is likely infeasible to learn across all scales at once, we might be able to use foundation models that have been trained at specific scales, which we call uniscale models, using only fine-tuning and some architectural changes (see Figure 2.1). We first review the existing uniscale foundation models in depth for each of the four main biological modalities [**siFoundationModelsMolecular2024**].

2.2.1 Foundation models across scales

Molecular foundation models (mFM) try to model with atomistic precision the complex quantum physics-based rules that govern molecules and their interactions [**abramsonAccurateStructure**
They generate embeddings of molecules by encoding their chemical representation, often using SMILES notation. These embeddings should contain information to predict molecular measurements such as binding to a target, potency, solubility, and more [**mendez-lucioMolEFoundationMod**
rossLargescaleChemicalLanguage2022]. The models are often built with invariances concerning the symmetries of molecules (relative positions and angles) [**batznerE3equivariantGraphNeural**
These models can also be paired with ones that learn to predict the structure and dynamics of these molecules. Training data in this context is mostly limited by compute since molecular dynamics simulations can be generated at will [**kozinskyScalingLeadingAccuracy2023**]. The first use cases of such models are in material generation and drug discovery.

However, computing binding affinities and force-fields similar to the most precise molecular dynamics methods remains a frontier [**benali2025pushingaccuracylimitfoundation**, **rhodes2025orbv3atomisticsimulationscale**].

Nucleotide foundation models (nFM) are a category of models designed to analyze sequences of nucleotides or amino acids, which are encoded in triplets of nucleotides, primarily using data derived from sequencing across various life forms. Although new architectures have been introduced to handle large context sizes [**nguyen2023hyenadnalongrangegenomicsequence**], most models generally rely on traditional transformer models with small context sizes and are trained with masking. These models are based on the transformer architecture and language model techniques (LM) [**vaswaniAttentionAllYou2023**] to produce representations of the lengthy and repetitive molecular structures found in DNA and RNA, sometimes termed dnaLM and rnaLM [**dalla-torreNucleotideTransformerBuilding2024**, **wangMultipurposeRNALanguage2024**, **fradkinOrthrusEvolutionaryFunctional2024**, **brixiGenomeModelingDesign2025**].

While protein language models like ESM2 [**esm2**] have shown real-world usage in helping generate 3D models of proteins, dnaLM mainly focused on the task of understanding regulatory mechanisms, such as binding interactions and chemical modifications on DNA. It has been shown however, that representations learned by dnaLM can also contain information about the secondary structures of proteins and even protein-protein interactions [**brixiGenomeModelingDesign2025**, **cornmanOMGDatasetOpen2024a**].

For these reasons, we fold protein language models into the nFM category, proposing that their distinctions will blur in the future.

Numerous challenges still exist in accurately predicting the diverse conformations of RNA, DNA, and proteins, as well as in modeling their intricate interactions [**abramsonAccurateStructureI**
Indeed, it is still hard to measure complexes with the same accuracy as individual proteins. A goal would be to generate nFMs that learn across the very related lexicons, which are DNA, RNA, and proteins, by introducing architectures and training modalities that go beyond what exists today [**xiaNatureLMDecipheringLanguage2025**]. Indeed, there we could use the framework of "learning across scales" by using the representations of molecules, learned and compressed by mFMs, as the very tokens of nFMs, allowing them to talk about ribonucleotides, deoxyribonucleotides, amino acids, and their potential modifications.

Currently, the main applications of nFMs have been in drug, and target discovery, as well as many other fields of biology.

Cell foundation models (cFM) are a class of models trained on a matrix of abundances of the different chemical elements (proteins, RNAs) present in cells. [**bunneHowBuildVirtual2024**, **scprint**, **theodorisTransferLearningEnables2023**, **cuiScGPTBuildingFoundation2024**, **haoLargescaleFoundationModel2024**, **rosenUniversalCellEmbeddings2023**]. Their architecture is often based on bidirectional encoder-based transformers trained on single-cell RNA-sequencing data. While diverse training strategies have been presented, the model's architectures have, for now, remained fairly classical. The goal of these cFMs is to generate an accurate model of the cell that would allow predictions of cell evolution and response to perturbations [**kedzierskaAssessingLimitsZeroshot2023**].

However, immense challenges remain. Current promises have not stood up to experimental validations [**bendidiBenchmarkingTranscriptomicsFoundation2024**, **boiarskyDeepDiveSingleC**
While many reasons can be formulated, issues exist around data quality, diversity, and coverage. Indeed, single-cell data is very noisy, only measures a tiny fraction of the molecular composition of cells, and has been mostly produced on human and model animals [**programCZCELLxGENEDiscover2023**]. While data will remain an important challenge, an area of improvement would be to, again, distill the rules of molecular interactions from sequence learned at the sequence level onto cFMs. This allows them to better learn the complex regulatory mechanisms of the cell.

Tissue foundation models (tFM) strive to understand the interactions between cells that form tissues, mostly in higher-order organisms. Often based on imaging techniques, they consider the 2D structural relationship of cells or group of cells in a tissue slice. The stained microscopy slides allow the prediction of tissue type, organs, and even some protein expression levels. These models are often versions of the famous vision transformer architecture and framework (Dino V2), applied to medical images [**oquabDINOv2LearningRobust2024**, **chief**]. They thus learn on image patches where each pixel has some channels of information (often from 2 to 30 different chemical elements are represented within these channels) [**brayCellPaintingHighcontent2016**, **wencksternAIpoweredVirtualTissues2025**]. The number of channels can go up to tens of thousands in spatial transcriptomics image modalities, where each channel represent a transcripts location at a subcellular level(e.g., xenium) or at a cell-group-level (e.g., visium).

Overall, even more challenges arise in tissue foundation models. Most of the data exists behind institutional barriers, the resolution of high channel count modalities is really poor, while the channel amount of high-resolution modalities is really small, making it hard to predict even the cell state. Slices are often of tiny subparts of tissues. Most of the available data is in 2D slides, and 3D modalities are still burgeoning [**alonExpansionSequencingSpatially2021**]. We lack good measurements of what cells are communicating, but we know that they do, from sequences to molecules and even entire organelles [**hertleHorizontalGenomeTransfer2021**]. tFMs' vocabulary can be seen as made of cells. Their tokens are cell representations and could be be the rich representations learned by cFMs. The goal of a tFM is then to predict the presence of cells given other cells in spatial context.

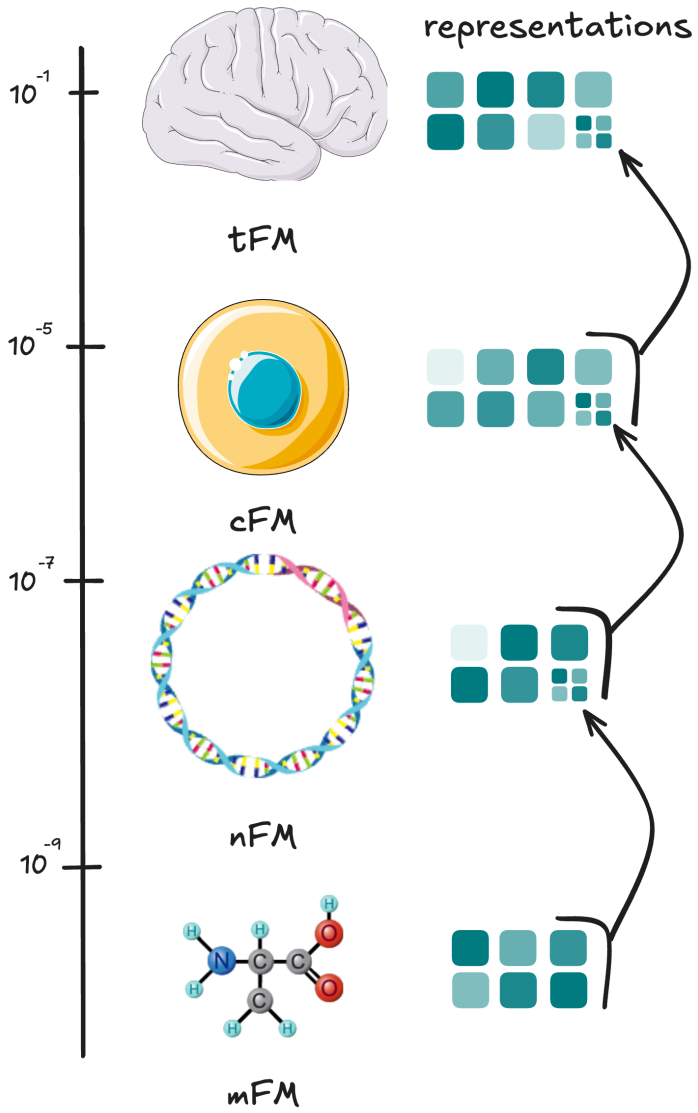


Figure 2.1 – we show how the representation of different foundation models could feed the upper scales and their learning could inform the lower scales' representations.

2.2.2 Architectural modifications : compressed representations

For biological representations, previous methods have leveraged many different methods from matrix factorization, nearest neighbors, and neural networks [**gunawanIntroductionRepresentatio bengioRepresentationLearningReview2014a**]. Popular approaches are Variational Autoencoder (VAE) such as scVI and scArches [**scvi**, **scarches**]. In the domain of protein embedding, the HourGlass embedding method [**cheap**] introduced FSQ [**mentzerFiniteScalarQuantization202** as a framework to encode both amino acid sequences and 3D structural information from a pLM into a quantized latent space. Meanwhile, DNA sequence model embeddings have been mostly restricted to metagenomics, with the exception of DNA-BERT-S [**zhouDNABERTSPioneeringSpec**

Finally, it has been shown not only in biology but also in the NLP community that for transformer models, embeddings based on average,max,sum-pooling of last-layer tokens are very restrictive and do not perform well [**schockaertEmbeddingsEpistemicStates2023**, **leeNVEmbedImprovedTechniques2025**, **ilseAttentionbasedDeepMultiple2018**]. Indeed, current state-of-the-art methods use more complex approaches such as cross-attention mechanism and additional pre-training or fine-tuning tasks.

In the following, we will show that we can use a similar cross-attention mechanisms to compress the output embeddings of a foundation model into a set of lower-dimensional vectors.

2.2.3 Training modifications : fine-tuning

An extensive literature exists on fine-tuning. The simplest and most powerful approach remains to continue training on a small set of epochs and with a lower learning rate [**testft**]. Common tools include low-rank approximations of the Multi-layer perception (MLP) and Query, Key, Values (QKV) matrices using LoRA, QLORA, and COLA [**lora**, **qlora**, **cola**, **tangEvaluatingRepresentationalPower2024**], which allow cheap fine-tuning of large foundation models. Other common approaches also mostly revolve around reducing the memory footprint of fine-tuning by only back-propagating the loss across a specific subset of parameters, from updating only specific layers of the model, only the MLPs, the QKV matrices, or only the biases of the MLPs [**petersTuneNotTune2019**, **chronopoulouEmbarrassinglySimpleApproach2019**]. Finally, adapter layers have also been used for their versatility. They often consist of an additional MLP on top of the large model's output representations [**efficientft**].

In the following, we will show that the adapter layer is a sensible approach to perform multi-scale fine-tuning.

2.2.4 Contributions

Following up on these recent advances, we propose :

— A cross-attention "compressor" block whose goal is to compress a foundation model's

output embeddings into a small set of low-dimensional vectors, called the *Xpressor* (Cross-Attention Compressor transformer). This is learnt using an auto-encoding approach with a reconstruction loss. The Xpressor is modality agnostic and can be used by mFMs, nFMs, cFMs, tFMs, or even other non-biological domains, and can work in addition to other training tasks like masking or denoising (see Figure 2.2A).

— A multi-scale fine-tuning approach using adapter layers. This allows the fine-tuning of models from one level using the upper-scale model's task (see Figure 2.2B).

2.3 Xpressor

2.3.1 Background

scPRINT [**scprint**] is a foundation model trained on more than 50 million unique single-cell RNA-seq profiles, representing around 100B tokens. It learns with a multi-task pre-training loss, allowing state-of-the-art zero-shot abilities in denoising and label prediction. scPRINT builds on previous foundation models, like scGPT [**cuiScGPTBuildingFoundation2024**] and scFoundation [**haoLargescaleFoundationModel2024**]. It improves upon them on multiple benchmarks and is also easier to use and faster to train than many other similar models. Additionally, it comes with a gymnasium of benchmarks presented in **scprint**. For these reasons, we chose to use it as our cFM and the starting point for our work.

ESM2 [**esm2**] is a protein language model that learns embeddings of amino acid sequences. It has been shown to be able to learn the evolutionary constraints of proteins and to be able to predict contact maps. Models like ESMfold [**esmfold**] have been created to predict a protein's 3D structure directly from its output embeddings. It is also simple to use. For these reasons, we chose to use it as our nFM.

2.3.2 Approach

Our first contribution is the compression of output embeddings of foundation models using a transformer block and a bottleneck-learning training modality (see Supplementary Material 2.4.5) : we call it the Xpressor (see Figure 2.2A). Compression / decompression is a key mechanism to transfer representations across scales (see Supplementary Material 2.2.1), we thus models that can compress and decompress their input into a lower-dimensional space. To do so, we introduce an additional set of transformer blocks called "Xpressor blocks". In the context of scPRINT, these blocks represent cell features.

As inputs scPRINT continues to use a set of summed up gene expression and gene ID tokens. The first ones are generated using an MLP on each expression values of genes in a cell j, the other ones are generated from ESM2's output embeddings of each gene sequenced aggregated with mean-pooling. The newly proposed Xpressor block uses as input a set of learned latent tokens $\boldsymbol{T}$. It then performs cross-attention between the last layer of the gene embeddings and the latent tokens (see Figure 2.2A). The goal is for the Xpressor blocks to

be of smaller dimensions and context size than the main blocks, such that we end up with $\boldsymbol{C}_j$ a set of n tokens of dimension d_t generated from the encoded gene expression and ID matrices $\boldsymbol{E}_j$ and $\boldsymbol{G}$. Where $\boldsymbol{G}$ and $\boldsymbol{E}_j$ are sets of m tokens of size d_c representing the IDs of the genes and their corresponding expression in cell j, respectively, where $d_c < d_t$ and $n << m$:

$$\boldsymbol{O}_j = \text{Transformer}(\boldsymbol{E}_j, \boldsymbol{G})$$

$$\boldsymbol{C}_j = \text{Xpressor}(\boldsymbol{O}_j, \boldsymbol{T})$$

for a cell j, with the *Xpressor* being initialized with a learned set of input cell tokens, and $\boldsymbol{C}_j$ being the cell tokens associated with the input $\boldsymbol{E}_j$.

The *Transformer* and *Xpressor* are both transformer with N and M layers, respectively. Indeed, we have designed both blocks to contain a cross-attention architecture (see Figure 2.2C) such that we can also do : $\hat{\boldsymbol{O}}_j = \text{Transformer}(\boldsymbol{C}_j, \boldsymbol{G})$, with $\hat{\boldsymbol{O}}_j$ being the output of the *Transformer* when using the *Xpressor* representation as input. We add an optional MLP after cross-attention to a transformation of the embeddings prior to the self-attention round. In our example, the decompression is done with gene ID tokens as input only ($\boldsymbol{G}$) (see Figure 2.2A). These tokens remain the same for all cells of a given species and thus do not depend on j. In the context of protein language models, for example, this would be replaced by positional tokens.

As can be seen in Figure 2.2A, the *Transformer* blocks are applied twice. The first application act as an encoder, only using self attention, while the *Xpressor* and second application of the *Transformer* blocks act as decoders. We follow these definitions from the original "Attention is All You Need" paper [**vaswaniAttentionAllYou2023**]. It has to be noted that in our case cross-attention is performed first instead of last. Related ideas have also been explored in **leeNVEmbedImprovedTechniques2025**, where the authors propose a cross-attention-based method to update tokens using "latent" embeddings followed by a classical mean-pooling.

The goal of the *Xpressor* and the entire model can be seen as to perform compression of the gene tokens into a set of cell tokens similar to the classical information bottleneck from **tishbyInformationBottleneckMethoda** (see Supplementary Material 2.4.5). This is our main training objective to train the *Xpressor* blocks, while the *Transformer* is also trained with masking.

In our case, each embedding represents different cell components. At training time, we present multiple losses to both regularize it and ensure differences across them, similar to what can be done in VAEs (see Supplementary Material 2.4.6).

2.3.3 Results

We show that such an instantiation of the transformer leads to better performance over the gymnasium of tasks available in the scPRINT cFM.

Table 2.1 – Comparison of cell embedding approaches

Model	Cell Label Pred.	Embed. Quality	Gene-Net Infer.
Class-pooling	0.64	0.48	**4.0,2.3**
Xpressor	**0.72**	**0.52**	4.1,2.1

Indeed, we now look at three specific tasks : cell-type prediction, embedding quality, and gene-network inference. The tasks are the same as presented in **scprint**.

"Embedding quality" refers to the average scIB [**lueckenBenchmarkingAtlaslevelData2022**] score for batch correction and biological consistency of cell embeddings. In this context scIB looks at the quality of the embeddings based on measures of similarity, nearest neighbors, and clustering.

Cell-label predictions are generated using a classifier on top of the cell embeddings generated by each model. We follow the approach of **scprint** here, which was recently presented with a different mechanism in **wangHierarchicalInterpretationOutofDistribution2024**. This classification task allows us to see how one can steer the model's embeddings to represent meaningful biological features.

Finally, we display two different metrics for gene-network inference. The gene network inference benchmark tries to estimate the quality of the self-attention matrices based on similarity to a gene-gene ground-truth matrix. Here we use EPR, an odds-ratio measure where, e.g. a value of N means that the predictions are N-times as likely to be correct as a random guess. One is the EPR score on the genome-wide perturb-seq gene-network from BenGRN [**scprint**], while the second is the average EPR of multiple predicted gene-networks across various cell types compared to the BenGRN's omnipath ground truth gene network [**tureiOmniPathGuidelinesGateway2016**].

In our comparison, the regular transformer's class-pooling is done similarly to scGPT's [**cuiScGPTBuildingFoundation2024**] approach, where a class token is added to the model's input and an additional loss is placed on it : $argmin_{C_j}(||E_j - \boldsymbol{C}_j \boldsymbol{G}_j^T||_2)$. Both models use the same latent dimensions, architectures, training paradigm, and number of input tokens for both genes and cells.

We see that the Xpressor outperforms the simpler class-pooling approach on embedding quality and cell-label prediction, while the gene-network inference results remain roughly similar.

We will now see how we can further train -or fine-tune- these representations using information from the upper scale. While Xpressor layers with their small set of low-dimensional tokens are best suited for this task, we will focus on commonly available foundation models and architectures, presenting a general approach.

2.4 Multi-scale Fine-tuning

2.4.1 Background

To merge foundation models, we need a way to connect the lower-scale models to the upper one. It had been proposed in **rosenUniversalCellEmbeddings2023**, **scprint** to use protein language model-based representations, like those of ESM2, as input tokens for the models. This decreases the number of parameters the model has to learn ; It allows the model to work on genes unseen at training time ; Moreover, it also lets the model use information that it would not have gained otherwise, such as protein structure, homology, and mutations.

2.4.2 Approach

We propose going beyond simply reusing lower-scale models' representations and fine-tuning them during the pre-training of the upper-scale model using an adapter layer (see Figure 2.2B). With such layer, each output embedding $\boldsymbol{e}$ is transformed with a differentiable function f (here, an MLP) :

$$\boldsymbol{i}_k = f(\boldsymbol{e}_k)$$

By using an MLP, the adapter layer not only applies a transformation of its input but also adds information (see Supplementary Material 2.4.4). In our case, we use ESM2 as the lower-scale model and scPRINT as the upper-scale model. The initial ESM2 embedding is known to contain a representation of the protein's sequence, evolutionary similarity, and constraints.

Indeed, this is what allows this representation to replace the multiple sequence alignment (MSA) step in ESMfold [**esmfold**]. We posit that this initial embedding already contains the information necessary to understand some of the rules in gene interactions (homology and similar evolutionary constraints). However, representations from ESM2 are very different from those from single-cell foundation models. Our goal is to enrich these representations with knowledge gained from co-expression information across millions of cells.

2.4.3 Results

We show that a cFM trained using the pooled embeddings of a pretrained nFM performs better in most tasks from the **scprint** gymnasium benchmark than one with learned representations (see Table 2.2). This is possible because we allow the model to start from a very rich representation instead of a random set of vectors, while still giving it the flexibility to incorporate additional knowledge. Each foundation model tested uses the same latent dimensions, architectures, training, and number of input tokens. We report the performance at the best epoch, and the training is stopped after 20 epochs.

Table 2.2 – Comparison of input-gene embedding approaches

Model	Cell Label Pred.	Embed. Quality	Gene-Net Infer.
Random init.	0.62	0.48	4.5,1.0
ESM2 frozen	0.60	0.484	5.2,1.4
ESM2 fine-tuned	**0.70**	**0.49**	**4.8,2.4**

We also show the difference in cell embeddings obtained between the regular transformer and the Xpressor (see Figure 2.3). The dataset is a very challenging mix of modalities with various batch effects and amounts of noise. Cell types are also quite similar, making the task more difficult. We can see that the Xpressor embeddings contain more structure and resolve different cell types better than a transformer with class-pooling.

Using ESM2's embeddings allows scPRINT to work on genes and sequences unseen at training time, to learn from an unlimited number of species, and to integrate DNA, RNA, and protein-level information such as mutations and structural variants.

Finally, contrary to other methods, this version does not require an update to the original model and can be added to the new model. Moreover, with this approach, scPRINT still maintains its ability to work on genes and sequences unseen at training time.

Conclusion

We have proposed a framework towards building compositional hierarchical foundation models for life, from atoms to tissues. We highlighted progress and challenges remaining for each specific scale of biological representations. While data generation efforts focusing on breadth and quality remain paramount to progress, we believe that the composition of foundation models could drive progress forward. Having a vocabulary for biological entities will allow us to better reference them, helping us define the impact of a molecule on a tissue or the interaction between RNA and proteins. Such a model of life should not be seen as one being trained end-to-end but as a set of models distilling the key information that they have learned and that the next one requires.

We have presented one small piece in this approach, where a cell foundation model (scPRINT) uses and fine-tunes a protein sequence foundation model (ESM2). We have also shown how XPressor can compress the output representations of transformers into a small set of lower-dimensional vectors, bridging proteins to cells. Such an approach could be used to bridge molecules to proteins and cells to tissues by using compressed representations that are then fine-tuned. This is a promising back-bone architecture for a general model going from atoms to tissues.

Future work should focus on using Xpressor's representations to power upper scale

models or the ability to learn a Xpressor on top of a pre-trained foundation model. The Xpressor approach could also be extended to decoder-based language models. Finally, fine-tuning using and adaptor layer suffers from a main drawback, the non-additivity of MLPs and therefore the limited use of such fine-tuned models in other contexts than for their compressed representations. Implementing intelligent GPU scheduling and using LoRA-type methods to fine-tune only XPressor blocks will allow for more complex fine-tuning in GPU-rich settings. We will need to show that this can be applied to the other scales of biological representations and generate benchmarks that better capture the diversity of real-world biological tasks across these scales.

Supplementary

2.4.4 proof that fine-tuning ESM2 with an adapter layer is at least sufficient to learn to add co-expression information

We show below that an MLP (with at least one hidden layer and a sufficiently large number of neurons) can learn to map each of D input protein embeddings to an arbitrary desired output, even if that output corresponds to a unique lookup for each protein.

1. **Finite Data Interpolation :** Let the set of D protein embeddings be $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_D\} \subset \mathbb{R}^D$, and suppose that for each $\mathbf{e}_k$ we want the MLP to output $\mathbf{w}_k \in \mathbb{R}^D$. Because the set $\mathcal{E}$ is finite, it is possible to design a function that exactly maps $\mathbf{e}_k \mapsto \mathbf{w}_k$ for all $k = 1, \ldots, D$.

2. **Constructive Argument Using ReLU Networks :** For a ReLU-based MLP, one can construct "bump" functions that are activated only in a small neighborhood around each $\mathbf{e}_k$. For instance, one may define functions of the form $\mathbf{r}_k(\mathbf{x}) = \sigma(-\|\mathbf{x} - \mathbf{e}_k\| + \delta)$, where $\delta > 0$ is chosen so that $\mathbf{r}_k(\mathbf{e}_k) > 0$ and $\mathbf{r}_k(\mathbf{x})$ is nearly zero for $\mathbf{x}$ that are not close to $\mathbf{e}_k$. By associating one or more hidden neurons to each protein embedding $\mathbf{e}_k$, one can form a linear combination $\mathrm{MLP}(\mathbf{x}) = \sum_{k=1}^{D} c_k \, \mathbf{r}_k(\mathbf{x})$, where the coefficients $c_k \in \mathbb{R}^D$ are chosen so that $\mathrm{MLP}(\mathbf{e}_k) = \mathbf{w}_k$ for all k. Because the supports of the functions $\mathbf{r}_k(\mathbf{x})$ can be made nearly disjoint, the MLP can "memorize" the mapping by acting as a lookup table.

3. **Conclusion :** Thus there exists a configuration of weights (and biases) in an MLP that yields $\mathrm{MLP}(\mathbf{e}_k) = \mathbf{w}_k, \quad \text{for } k = 1, \ldots, D$. Hence, even though the MLP is simply performing a transformation, its capacity is sufficient to learn any arbitrary mapping for the D proteins. In other words, at worst, it can learn a mapping that is equivalent to a lookup table, thereby ensuring that each of the D proteins is assigned a specific, learned output value.

2.4.5 argument about the Tishby et al. bottleneck learning approach

The Information Bottleneck (IB) method seeks a stochastic mapping $p(t|x)$ that compresses the input variable X into a representation T, while preserving as much information as

possible about the relevant variable Y. The trade-off is controlled by the Lagrange multiplier $\beta \geq 0$. The IB objective is to minimize the following Lagrangian :

$$\mathcal{L}_{\text{IB}}[p(t|x)] = I(X;T) - \beta\, I(T;Y), \tag{2.1}$$

where $I(\cdot;\cdot)$ denotes mutual information.

Under the Markov constraint $\mathbf{Y} \leftrightarrow \mathbf{X} \leftrightarrow \mathbf{T}$, the optimization leads to the following self-consistent equations :

$$p(t|x) = \frac{p(t)}{Z(x,\beta)} \exp\left(-\beta\, D_{\text{KL}}\big(p(y|x) \,\|\, p(y|t)\big)\right), \tag{2.2}$$

$$p(t) = \sum_x p(x)\, p(t|x), \tag{2.3}$$

$$p(y|t) = \frac{1}{p(t)} \sum_x p(y|x)\, p(x)\, p(t|x), \tag{2.4}$$

where :

- $D_{\text{KL}}\big(p(y|x) \,\|\, p(y|t)\big)$ is the Kullback-Leibler divergence between the conditional distributions $p(y|x)$ and $p(y|t)$,
- $Z(x,\beta)$ is the normalization factor ensuring that $\sum_t p(t|x) = 1$.

2.4.6 FSQ and other contrastive losses on the cell embeddings

While D_{KL} over a non-informative Gaussian prior is a common formulation for regularizing the embedding space in VAEs, other formulations have been used such as with the VQ-VAE and FSQ-VAE. In these contexts, the D_{KL} is replaced with a discretization objective tailored to the respective quantization schemes.

VQ-VAE. Value Quantized (VQ)-VAE employ a *codebook* of size C, where each codebook entry is a d-dimensional vector. The encoder produces a continuous latent vector, which is then mapped to its nearest entry in the codebook (a hard quantization). A commitment loss term encourages the encoder's outputs to stay close to the chosen codebook vector, making the entire latent representation discrete at the vector level.

FSQ-VAE. By contrast, Finite Scalar Quantization (FSQ)-VAE discretizes each latent dimension *independently*. Specifically, the encoder outputs d values, each constrained to lie within a bounded range (e.g., $[-1, 1]$). Each dimension is then quantized into one of M discrete levels within that range. This dimension-wise quantization can be implemented as either a hard nearest-bin assignment or a differentiable approximation thereof. Because FSQ enforces scalar-level discretization, it provides a simpler and more fine-grained alternative to VQ's vector-level codebook approach, while still offering strong regularization of the latent space.

Contrastive regularization across embedding dimensions. We further encourage each of the d embedding dimensions to encode distinct information by adding a contrastive loss between them. Specifically, we compute pairwise similarities among embedding elements and penalize redundancy, thus pushing each dimension to capture complementary features. A general contrastive loss for this purpose can be written as $\mathcal{L}_{\text{contrastive}} = \sum_{i=1}^{d} \sum_{j \neq i} \ell\left(\mathbf{e}_i, \mathbf{e}_j\right)$, where $\mathbf{e}_i$ denotes the i-th embedding dimension and ℓ is a contrastive loss function (e.g., InfoNCE [**oordRepresentationLearningContrastive2019**]) that encourages *dissimilarity* among different embedding components.

Dimension-specific classifiers. To further steer each dimension's content, one can add a separate classifier on top of each dimension to learn about different classes. The classifier for dimension i is trained via a cross-entropy loss $\mathcal{L}_{\text{cls}}^{(i)} = -\sum_c y_c \log p\left(c \mid \mathbf{e}_i\right)$, where y_c is the ground-truth label and $p\left(c \mid \mathbf{e}_i\right)$ is the predicted probability for class c. Summing these per-dimension losses yields an overall classification objective $\mathcal{L}_{\text{cls}} = \sum_{i=1}^{d} \mathcal{L}_{\text{cls}}^{(i)}$. Together, the contrastive and classification losses ensure each embedding dimension captures unique, discriminative information, resulting in more expressive representations.

Software and Data

The software and data for training scPRINT as well as gymnasium tasks and code to reproduce the results of the manuscript are available at `https://github.com/cantinilab/XPressor`.

WandB logs, are available in the following link : `https://api.wandb.ai/links/ml4ig/h370j6io`

Model checkpoints are available in the following link : `https://huggingface.co/jkobject/scPRINT/tree/main`

Checkpoints, wandb logs, and more will be made available after the review and deanonymization process.

Acknowledgments

The project leading to this manuscript has received funding from the Inception program (Investissement d'Avenir grant ANR-16-CONV-0005) L.C. and the European Union (ERC StG, MULTIview-CELL, 101115618) L.C. We acknowledge the help of the HPC Core Facility of the Institut Pasteur and Déborah Philipps for the administrative support. L.C.

The work of G. Peyré was supported by the French government under management of Agence Nationale de la Recherche as part of the 'Investissements d'avenir' program, reference ANR19-P3IA-0001 (PRAIRIE 3IA Institute). G.P.

Impact Statement

This paper presents work whose goal is to advance the fields of computational biology and machine learning. No ethical issues are raised by the work other than what is typically noted in computational biology and foundation model papers. It might have an impact on building better models for drug discovery, target discovery, and improving our understanding of biological systems.

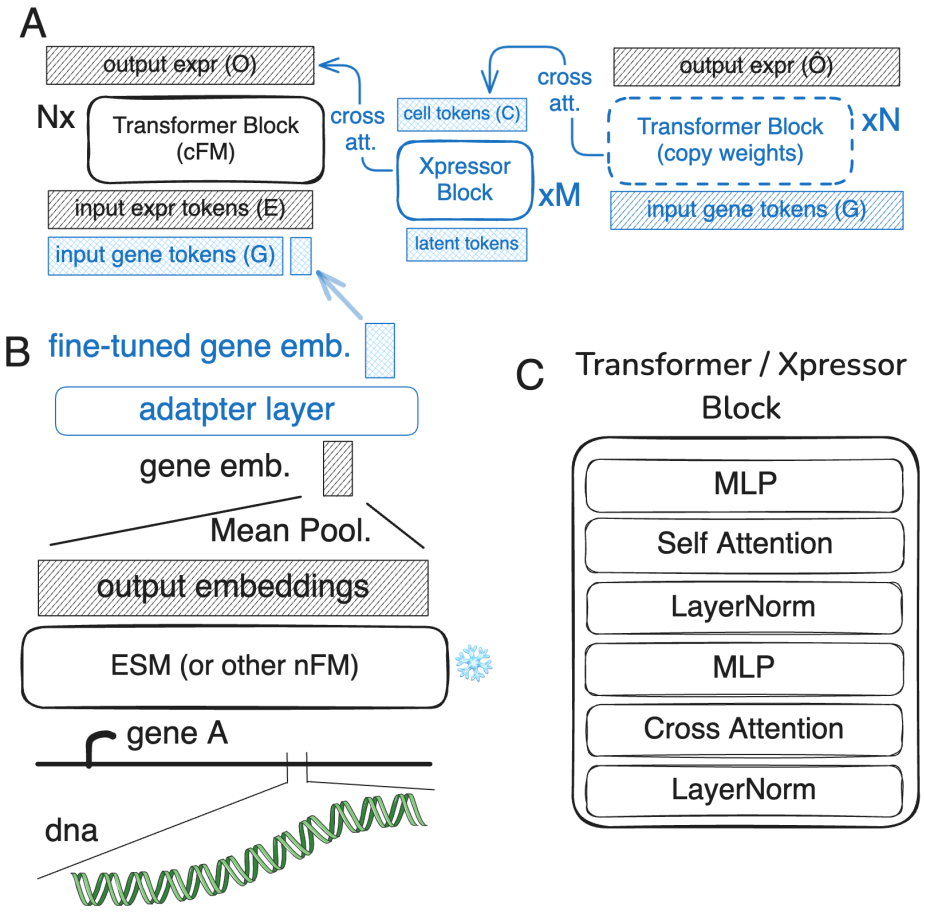


FIGURE 2.2 – A. The Xpressor architecture, composed of M layers, shows how gene-level representations are compressed into cell-state vectors through cross-attention over the output embeddings of a transformer, composed of N layers. These compressed representations are then decompressed back using the same initial transformer model with cross-attention given the initial gene-level tokens. B. Example of the multi-scale fine-tuning setup illustrating how the adapter layer enables joint training of gene-level representations that are then used by a cFM. C. Detailed structure of the transformer and Xpressor blocks showing the cross-attention and self-attention sub-blocks. Blue blocks are our contributions. Shaded blocks indicate inputs and outputs.

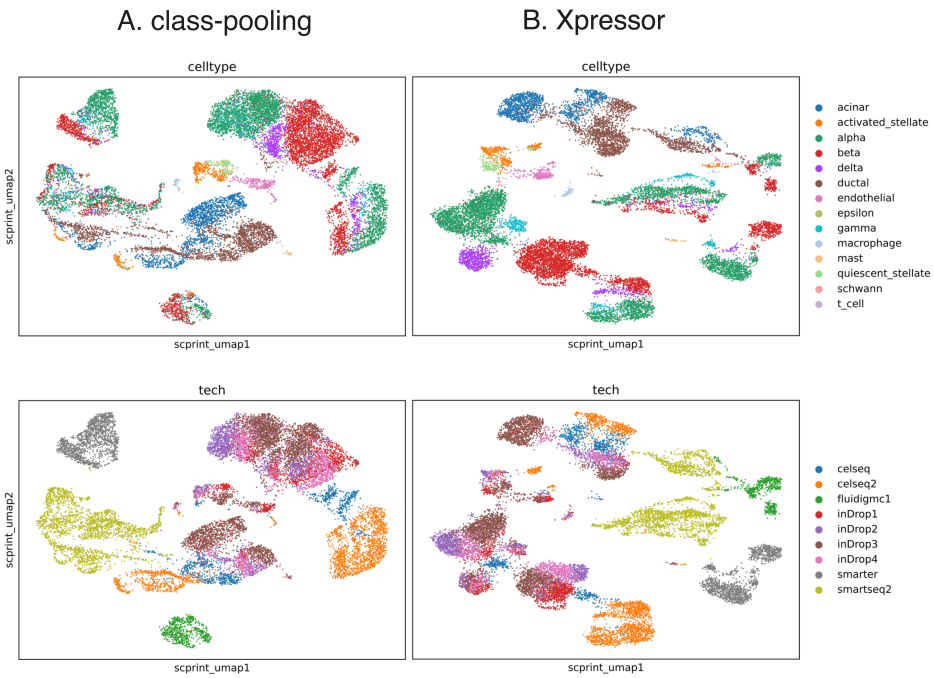


Figure 2.3 – between the regular transformer with class-pooling (left), scIB : 0.43, and the Xpressor (right), scIB : 0.48. The Xpressor embeddings contain more structure and resolve different cell types better.

scPRINT-2

Within this article we will

Discussion and perspectives

By the end of this thesis, we have developed a next-generation foundation models for single-cell data. In our first publication„ we examined a set of initial improvements to the previously presented tools, including novel training and encoding/decoding schemes. We focused our analysis on a set of common benchmarks and comparisons that were lacking in previous works, focusing on the internal representation of genetic interactions that these models possess. We also focused on real-world usage and accessibility with a tool competitive or state-of-the-art on many tasks orthogonal to gene network inference, such as cell type annotation, batch effect correction, and denoising.

In our second publication, we looked at an architectural change that would allow AI models to work across multiple scales of biology, from the molecules to the cells and tissues. We showed how multimodality can improve the unimodal performance of these models.

In our third publication, we reused our proposed architecture, refined our training and encodings paradigms, and scaled our model and training dataset to create a next-generation single-cell foundation model. We also separately assessed the impact of each of the model's components in an additive study across a gymnasium of tasks introduced in our first publication. We also questioned and showed how these tasks could be refined in a study of model generalization across unseen modalities and organisms.

But a lot of work remains at the crossroads of AI and biology. In the following sections, I will discuss some of the challenges and opportunities that I see in this space.

4.1 collecting data in the wild

The first issue to address for a better model will be obtaining cell expression data across a much more diverse genetic background.

This also means sequencing the genome of the tissues we are analyzing, which is rarely done because genomic data has strong laws surrounding patient anonymity and public sharing.

Fortunately, we have seen projects starting around this goal, such as the 10K10K, which aims to sequence 10,000 cells from 10,000 people along with genetic data. The Sanger Institute is also doing something similar with spatial transcriptomics of 10,000s of samples in development, along with their genomes.

But these remain small-scale projects compared to the diversity of life on Earth. In genomics, scBasecamp and other for-profit companies are working on sampling life around the world from barren places to ocean depths, with the stated goal ofdeveloping higherr quality models. Single-cell models would stand to benefit just as much.

Finally, we also want interventional datasets. Currently, perturb-seq datasets with tens of thousands of perturbations exist, such as the genome-wide perturb-seq dataset and Tahoe-100M. However, here too, the scale remains too small. The Broad & Sanger Institute's PRISM and DepMap projects examined millions of perturbations, albeit without deep phenotypic readouts. Recursion assessed image-based perturbations across at least as many. Xaira has started to release a dual gene knockout dataset of unparalleled quality.

Indeed, the second missing important axis is data quality. Genomics is plagued with very low-quality, noisy, or biased, poorly labeled datasets. This is due to the high cost of sequencing, the complex chemistry of the experiments, and the poor academic incentives driving the creation of these datasets.

It leads to an unstate Pareto front, where we want both more depth and more breadth : more diversity versus more quality.

It might be solved by new technologies, indeed we now have technologies like VASA-seq, 10X's Flex, and smart-seq 3 that promise unparalleled definition for a given sequencing budget.

10x's Xenium, BGI's STEREO-seq„ and expansion-based in situ method are promising for sequencing RNAs in their original 2D or even 3D context within sub-cellular locations of millions of cells at once.

But we will also have to be smarter in how we select cells to sequence.

4.2 multi modality & perturbations

Indeed„ these two modalities and their tradeoffs exist within a range of other techniques often needed to make sense of RNA biology itself.

Interventional data is also required for the model to learn causality, especially when assessed at multiple fine-grained timescales and in higher-quality cellular models such as organoids.

But the search space is unfathomable.

Tools like digital microfluidics might allow us to solve some of these problems by providing precise control over which cells receive specific perturbations and obtain particular

readouts, instead of pooling experiments and sequencing budgets randomly.

If paired with AI models and online active learning, we might have a shot at creating a true AI-virtual cell.

4.3 AIVC

One could view such a training modality as reinforcement learning with active feedback (RLAF) of a large pretrained AI model. It would have been pretrained on most of biology, using foundation models of single-cell multimodalities, tissues, molecules, and protein-RNA-DNA sequences, pooled together in the kind of approaches we described in Chapter 2.

LLMs could allow rich reasoning across these representations, results, and the breadth of written human knowledge.

Many challenges remain in bridging fields such as data engineering, machine learning, material engineering, microelectronics, molecular biology, and cell biology, but the rewards are tremendous.

Conclusion

Single-cell Foundation Models while in their infancy, have the power to change the way we do biology and medicine.

During this Thesis, we have shown that :

- We can use the internal workings of scFMs to predict meaningful gene interactions.
- We can update their training tasks, data, losses, as well as their architectures to better capture the underlying biology of the cell.
- We can use them to perform a variety of single cell tasks in a zero-shot of few-shot manner, from cell annotations, denoising, imputation, embeddings generation, batch correction, cross-species integration and counterfactual reasoning
- We can use multiple techniques at inference and fine-tuning time to improve their performance.
- We can leverage the other foundation models pre-trained on other modalities to improve their performance.

In conclusion, the follow-up of these studies should allow to multiply the use-cases of single-cell transcriptomics in clinical applications. To integrate other modalities like sequences, epigenetics, proteomics, spatial, and imaging via multi-scale architecture and fine-tuning. But also allow these models to reason by integrating them to LLMs. Finally one will need to gather more data from novel species, patient contexts and across perturbations. To the later point especially we will want to use active learning to guide the experiments and maybe reach our goal of cellular modeling.

Supplementary Information: What can 50 million cells tell us about gene networks?

Table S1: List of novelties in scPRINT and comparison to scGPT and scFoundation

features	scPRINT	scGPT	scFoundation	Geneformer v2
classification pretraining	v	x	x	x
hierarchical classification	v	x	x	x
denoising pretraining	v	x	v	x
masking pretraining	v	v	v	v
MVC pretraining	v	v	x	x
AE pretraining	v	x	x	x
large cell count GN inference	v	x	x	x
zero-shot classification	v	x	x	x
zero-shot batch correction	v	x	x	x
zero-shot denoising	v	x	x	x
genome-wide GN inference	v	x	x	x
large input context	x	x	v	x
raw count encoding	v	x	v	x
very large model	v	x	x	x
pretraining strategy and dataset	v	x	x	x
low GPU/hours implementation	v	x	x	x
weighted random sampling	v	x	x	x
protein encoding	v	x	x	x
cross-species abilities	v	x	x	x
gene location encoding	v	x	x	x
genome-wide input context	x	x	v	x
xtrimogene architecture	x	x	v	x
train / validate / test strategies	v	x	x	x
flashattention2	v	x	x	x

Comparison of the features and novelties from scPRINT compared to 2 similar published state-of-the-art methods: scGPT and scFoundation.

Table S2: model comparison

model name	model size	training time (hours)	training hardware	num cells	num leaf cell type	dimension (d)	layers	heads	token input size	num species	training	attention
scPRINT-small	7M	24	1xA100	41M (91M before QC)							denoising (60%) + classification + bottleneck	flashattention2
Geneformer v2	? (~50M)	72	12xV100	30M	?	256	6	4	2,048	1	masked (15%)	normal
scPRINT-medium	20M	72	1xA100	41M (91M before QC)	540	256	8	4	2,200	2	denoising (60%) + classification + bottleneck	flashattention2
scGPT	100M	?	?	33M	?	512	12	8	1,200	1	masked (15%)	flashattention1
scFoundation	100M	?	?	50M	?	768	12+12	12+8	20,000	1	masked (30%) + denoising	xtrimogene
scPRINT	90M	96	4xA100	41M (91M before QC)	540	512	16	8	2,200	2	denoising (60%) + classification + bottleneck	flashattention2
GPT2-small	117M	?	?	300B tokens (~150M cells)	x	768	12	12	1200	x	masked (15%)	normal
UCE	650M	960	24xA100	36M	(~1000 ?) likely <500	1280	33	20	1024	5	masked (20%)	normal
cellFM	700M	?	32xAscend910 NPUs	100M	?	1536	40	48	4096	1	masked (20%)	normal + LORA
scPRINT-vlarge	700M	168	24xA100	41M (91M before QC)	540	1280	20	10	2,200	2	denoising (60%) + classification + bottleneck	flashattention2

Table comparing different model sizes and architectures. Comparing scPRINT to other state-of-the-art methods, as well as GPT2-small and GPT3-large models

Table S3: Ablation study and impact on performance across tasks

id	description	denoise/reco2full_vs_noisy2full	emb_lung/ct_class	emb_lung/scib	emb_panc/ct_class	emb_panc/scib	reconstruction loss	classification accuracy	denoising loss	epoch
or46096v	small	0.34	0.31	0.47	0.11	0.41	1.31	0.4	1.16	24
ghqf2hym	medium	0.12	0.58	0.55	0.52	0.51	1.25	0.33	1.125	27
7asy8qpn	large	0.18	0.69	0.56	0.52	0.50	1.23	0.76	1.109	21
24chcp2e	medium-nofreeze	0.15	0.45	0.54	0.52	0.53	1.25	0.33	1.115	23
6o76ew23	medium-2-heads	0.10	0.49	0.55	0.40	0.53	1.25	0.33	1.124	26
lsr3pvnf	medium-MSE	0.21	0.61	0.56	0.51	0.49	1.26	0.33	6.3 (diff)	29
muwj73gx	medium-MVC	0.21	0.51	0.55	0.40	0.47	1.29	0.3	1.132	37
n8jypo8z	medium-noPE	0.09	0.71	0.56	0.35	0.46	1.27	0.33	1.31	23
q0fzpj5g	medium-no-random-weighted	0.17	0.51	0.53	0.19	0.48	1.26	0.26	1.118	27
f5e4qfkr	medium-MLM	0.04	0.53	0.54	0.39	0.46	1.26	0.35	0.999	23

The table shows the results of the ablation study on denoising, embedding with batch correction, and cell-type classification tasks. Results are displayed for the medium-size scPRINT model. Top to bottom: *small, medium, large:* regular models of various sizes. *medium-nofreeze:* a model trained without freezing gene embedding during pre-training. *medium-2-heads:* a model trained with only two heads per layer instead of 4. *medium-MSE*: a model with Mean Squared Error instead of the ZINB loss. *medium-MVC:* a model trained with scGPT's MVC methodology for the creation of the cell embedding. medium-noPE: a model trained without positional encoding for the gene's location. medium-no-random-weighted: a model trained without weighted random sampling. *medium-MLM*: a model trained with masked language modeling instead of denoising.

Table S4: Computational speed of various GN inference methods

model	speed for 1000 cells	speed for a dataset of 12 cell types	scale to #cells	scale to #genes
DeepSEM	10mn	2 hours	linear	quadratic
GENIE3	50mn	10 hours	quadratic	quadratic
GENIE3 (100 trees)	4mn	1 hour	quadratic	quadratic
Geneformer v2	1mn	15mn	linear	linear
scGPT	1mn	15mn	linear	linear
scPRINT	1mn	15mn	linear	linear

The computational speed of running various gene network inference methods on a set of 4000 genes and 1000 cells. It is showing that transformer-based models are far faster than previous methods, owing to their clever use of the GPU and pre-training.

Table S5: Performance of GN inference methods on the Sergio simulated scRNAseq dataset

model	EPR	AUPRC	TF_targ	TF_enr
DeepSEM	0.92601	0.00101	0	FALSE
GENIE3	0.94497	0.00193	5.2	TRUE
Geneformer v2	0.699	0.00409	0	TRUE
scGPT	0.6167	0.00278	10.5	TRUE
scPRINT	1.836	0.00861	13.15	FALSE

We generate a Sergio simulated scRNAseq dataset of 1000 cells for 800 genes from the RegNetwork ground truth network. We here showcase the ability of each model to recover the RegNetwork ground truth from this dataset. It shows how only scPRINT can recover some of RegNetwork's connections.

Table S6: Comparison scPRINT model size on performance across tasks and GN inference abilities

id	description	denoise/recc2full_vs_noisy2full	emb_lung/ct_class	emb_lung/scib	emb_panc/ct_class	emb_panc/scib	grn_gwps/auprc	grn_gwps/omni	grn_gwps/auprc_self	grn_gwps/epr	grn_gwps/epr_omni	grn_gwps/epr_self	grn_omni/auprc	grn_omni/auprc_class	grn_omni/epr	grn_omni/epr_class	grn_omni/tf_enr	grn_omni/tf_enr_class	grn_omni/tf_targ_enr	grn_omni/tf_targ_enr_class	grn_sroy/auprc	grn_sroy/auprc_omni	grn_sroy/auprc_self	grn_sroy/epr	grn_sroy/epr_omni	grn_sroy/epr_self	mean epr	mean auprc	mean ct_class	mean scib	total weigthed mean	reconstruction loss	classification accuracy	denoising loss	epoch
on6096v	small	0.08	0.31	0.47	0.11	0.41	0.03804	0.038	0.039	4.375	4.375	4.420	0.00137	0.00134	2.042	2.020	2	2	9	4	0.099	0.099	0.106	1.303	1.303	1.538	2.429	0.0049527261	0.21	0.44	**0.8502439048**	1.31	0.4	1.16	24
ghqf2hym	medium	0.12	0.58	0.55	0.52	0.51	0.03183	0.032	0.035	3.239	3.040	3.514	0.00148	0.00142	2.111	2.293	1	1	30	21	0.100	0.089	0.088	1.524	0.726	0.540	1.992	0.0040718035	0.55	0.53	**1.023048607**	1.25	0.33	1.125	27
7asy6qpn	large	0.18	0.69	0.58	0.52	0.50	0.03407	0.035	0.038	3.466	3.595	4.126	0.00139	0.00142	2.021	3.257	6	8	32	30	0.100	0.102	0.108	0.589	0.985	1.322	2.252	0.0043660238	0.60	0.53	**1.096761024**	1.23	0.76	1.109	21

here the comparison between scPRINT small (7M params), scPRINT medium (20M params) and scPRINT large (90M) params. We compare across both almost all metrics presented in results section.
While scPRINT small attains great performances on some gene network inference tasks and on the denoising we see that the overall best model remains scPRINT large

Table S7: overlap of different GN ground truths

comparison	precision	recall	random precision
MCalla et al. vs Omnipath	0.0520	0.0074	0.00154
MCalla et al. - T vs Omnipath	0.0155	0.0022	0.00154
gwps vs Omnipath	0.0015	0.0219	0.00129
gwps -T vs Omnipath	0.0030	0.0426	0.00129

Comparison of the overlap, expressed as precision and recall, of the three different ground truth networks used: MCalla, Omnipath, and gwps.

Table S8: Omnipath benchmark results on the genome-wide perturb-seq dataset

tool	EPR	AUPRC	TF target enr.	TF_enr	TF_only	ct_pred	RAND precision
DeepSEM	4.1	0.00192	21.4	FALSE	FALSE	FALSE	0.001633
GENIE3	4.7	0.00188	17.9	TRUE	FALSE	FALSE	0.00163
Geneformer v2	0.2	0.001796	5.9	FALSE	FALSE	FALSE	0.001528
scGPT	1.0	0.00208	14.0	TRUE	FALSE	FALSE	0.00163
scPRINT	2.8	0.00170	8.6	TRUE	FALSE	FALSE	0.00161
scPRINT (omnipath's heads)	4.7	0.00189	3.4	TRUE	FALSE	FALSE	0.00161
scPRINT (gwps' heads)	1.6	0.00190	5.0	TRUE	FALSE	FALSE	0.00161

Omnipath network overlap (EPR, AUPRC), as well as transcription factor enrichment, TF target enrichment, and cell type marker enrichment for gene networks generated by the different tools on the genome-wide perturb seq K562 cells at steady state (no perturbations)

Table S9: Omnipath benchmark results on the MCalla et al. datasets

tool	dataset	EPR	AUPRC	TF target enr.	TF enr.	cell type enr.
DeepSEM	**Han et. al.**	5.54	0.00029	18.9	FALSE	FALSE
DeepSEM	**Yan et. al.**	0.97	-0.00002	7.5	FALSE	FALSE
GENIE3	**Han et. al.**	1.51	0.00016	11.3	FALSE	TRUE
GENIE3	**Yan et. al.**	1.74	0.00020	0.0	FALSE	TRUE
Geneformer	**Han et. al.**	1.63	0.00010	11.3	FALSE	FALSE
Geneformer	**Yan et. al.**	1.99	0.00011	20.0	FALSE	FALSE
scGPT	**Han et. al.**	0.89	0.00016	17.0	TRUE	FALSE
scGPT	**Yan et. al.**	0.16	0.00007	20.0	FALSE	FALSE
scPRINT	**Han et. al.**	2.03	0.00019	23.6	TRUE	FALSE
scPRINT	**Yan et. al.**	1.76	0.00026	31.1	FALSE	TRUE
scPRINT (omnipath's heads)	**Han et. al.**	5.12	0.00004	3.6	TRUE	FALSE
scPRINT (omnipath's heads)	**Yan et. al.**	3.35	0.00019	13.3	FALSE	TRUE
scPRINT (Han et. al.'s heads)	**Han et. al.**	0.94	0.00030	30.9	TRUE	TRUE
scPRINT (Han et. al.'s heads)	**Yan et. al.**	0.57	-0.00004	6.7	TRUE	TRUE

Omnipath network overlap (EPR, AUPRC), as well as transcription factor enrichment, TF target enrichment, and cell type marker enrichment for gene networks generated by the different tools on the 2 human embryonic stem cell datasets used in [scPRINT outperforms GENIE3 and scGPT on cell type-specific ground truths].

Table S10: Denoising results per datasets

tools	denoising (+%) correlation. gNNpgpo6gATjuxTE7CCp	denoising (+%) correlation. R4ZHoQegxXdSFNFY5LGe	denoising (+%) correlation (RElyQZE6OMZm1S3W2Dxi)	denoising (+%) correlation (low cell count: 30). gNNpgpo6gATjuxTE7CCp	denoising (+%) correlation (low cell count: 30). R4ZHoQegxXdSFNFY5LGe	denoising (+%) correlation (low cell count: 30) (RElyQZE6OMZm1S3W2Dxi)	average denoising (+%) correlation	average denoising (+%) correlation (rare cell type)
untrained scPRINT	-16.0	X	X	-16.0	X	X	-16.0	-16.0
scPRINT	19.1	33.9	17.1	22.5	26.6	16.6	23.4	21.9
KNNsmoothing2	21.0	34.9	21.6	17.0	32.0	13.4	25.8	20.8
magic	29.3	34.6	22.7	16.8	24.4	4.6	28.9	15.3
magic (low cell dataset)	X	X	X	11.3	14.0	13.0	X	12.8

This table shows the detail of the denoising results for each of the three datasets for scPRINT-large, KNNsmoothing2, MAGIC, and MAGIC run on only the small cell type cluster. “Random scPRINT model” is the performance of an untrained scPRINT model.

Table S11: highlighted b-cell cluster genes in the BPH study

gene	link	in cancer	in b cell	analysis
MBNL2	https://www.nature.com/articles/s41467-023-44126-w	prostate cancer	high expr in immune tissues	BPH B-cell to normal B-cell diff. expr.
MAGOH	https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9738831/	cancer	high expr in immune tissues	
RANBP2	https://www.nature.com/articles/leu2012286. https://www.genecards.org/cgi-bin/carddisp.pl?gene=RANBP2	B-cell lymphoma	b cell validated	
CLIC4	https://www.nature.com/articles/s41420-022-01003-7	prostate cancer	high expr in immune tissues	
BAG5	https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3598994/	prostate cancer	b cell in cancer	
NR4A1	https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9424640/ https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8081071/	prostate cancer	b cell validated	
BAZ2A	https://www.nature.com/articles/s41598-024-56073-7	prostate cancer	b cell validated	
ZBTB16	https://www.pnas.org/doi/full/10.1073/pnas.0703872104 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5642638/	tumor suppressor in prostate cancer	b cell validated	
TAP1	https://bmccancer.biomedcentral.com/articles/10.1186/s12885-023-10527-9 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5674960/	cancer	b cell validated	BPH B-cell to normal B-cell diff. expr. post denoising
TAS2R19	https://v19.proteinatlas.org/ENSG00000212124-TAS2R19/tissue/B-cells		b cell validated	
PRDM7	https://pubmed.ncbi.nlm.nih.gov/27129774/	cancer		
TSEN54	https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10120902/	cancer	b cell in cancer	
EHMT2	https://www.frontiersin.org/journals/immunology/articles/10.3389/fimmu.2017.00429/full		b cell validated	
ERICH6B	https://platform.opentargets.org/target/ENSG00000163645/associations	cancer		
IL10RB	https://pubmed.ncbi.nlm.nih.gov/37144812/	cancer	b cell in cancer	

Table of the highlighted genes in the differential expression analysis in BPH vs normal B-cells together with their annotation on their relation to cancer and to b-cells, with sources.

Table S12: hub and differential hub genes in the fibroblast GN of the BPH study

TOP 15 hubs in BPH fibroblasts GN	TOP 15 hubs in normal fibroblasts GN	TOP 15 differential hubs in BPH fibroblasts vs normal	TOP 15 eigenvector_centrality differential hubs in BPH fibroblasts vs normal
HSPA1A	S100A6	HLA-A	CD99
MT2A	TGIF2-RAB5IF	MT2A	HLA-A
CREM	MIF	ATP6V0C	HSPA1A
TGIF2-RAB5IF	DNAJB9	DEFA1	LUM
HSPE1	IGFBP7	EIF4A1	ATP6V0C
CALD1	APOD	HSPA1A	CD99
SPOCK3	BRME1	LUM	EIF4A1
HLA-A	SPARCL1	SPOCK3	PAGE4
SPARCL1	TIMP1	nan-99	RYR2
RBP1	DCN	CD99	SERPINF1
C1S	C1S	CPE	C1R
BRME1-1	MGP	THBS1	COL6A2
FABP4	nan-270	LGALS1	HNRNPA0
nan-99	SLC25A6	PYDC2	SERPING1
LUM	BLOC1S5-TXNDC5	SERPING1	SERPINA3

List of the Top-15 elements in different GN analyses. Genes in yellow in the last columns are the new ones found with eigenvector centrality compared to the 3rd columns.

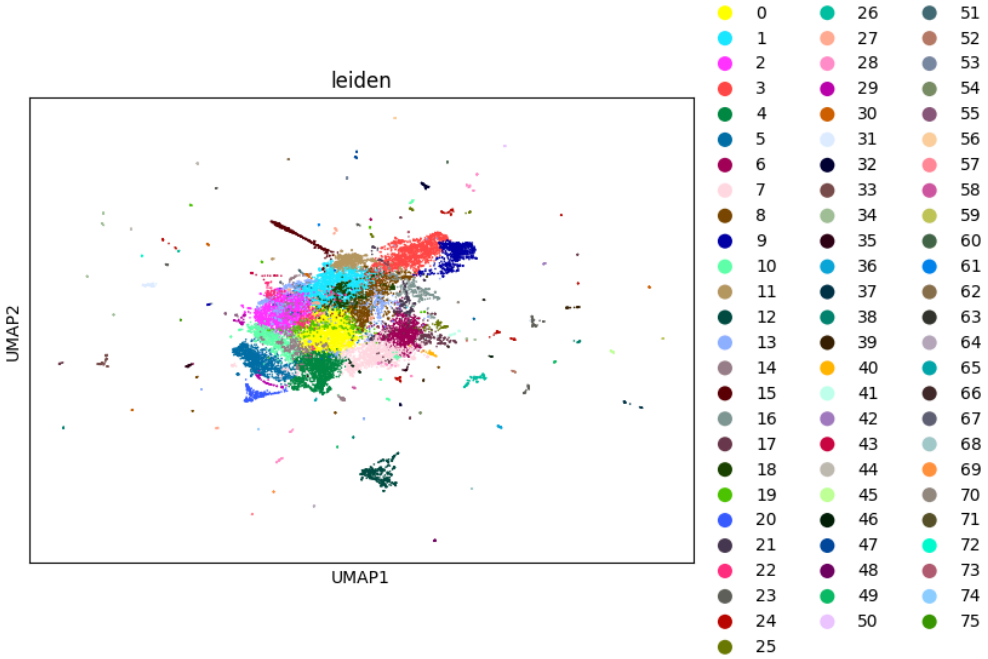
Table S13: number of elements predicted per class

ethnicity	21
sex	2
organism	2
cell type	424
disease	62
assay	26

Number of labels predicted by the model for each class. We use hierarchical classification for cell type, disease, assay, and ethnicity.

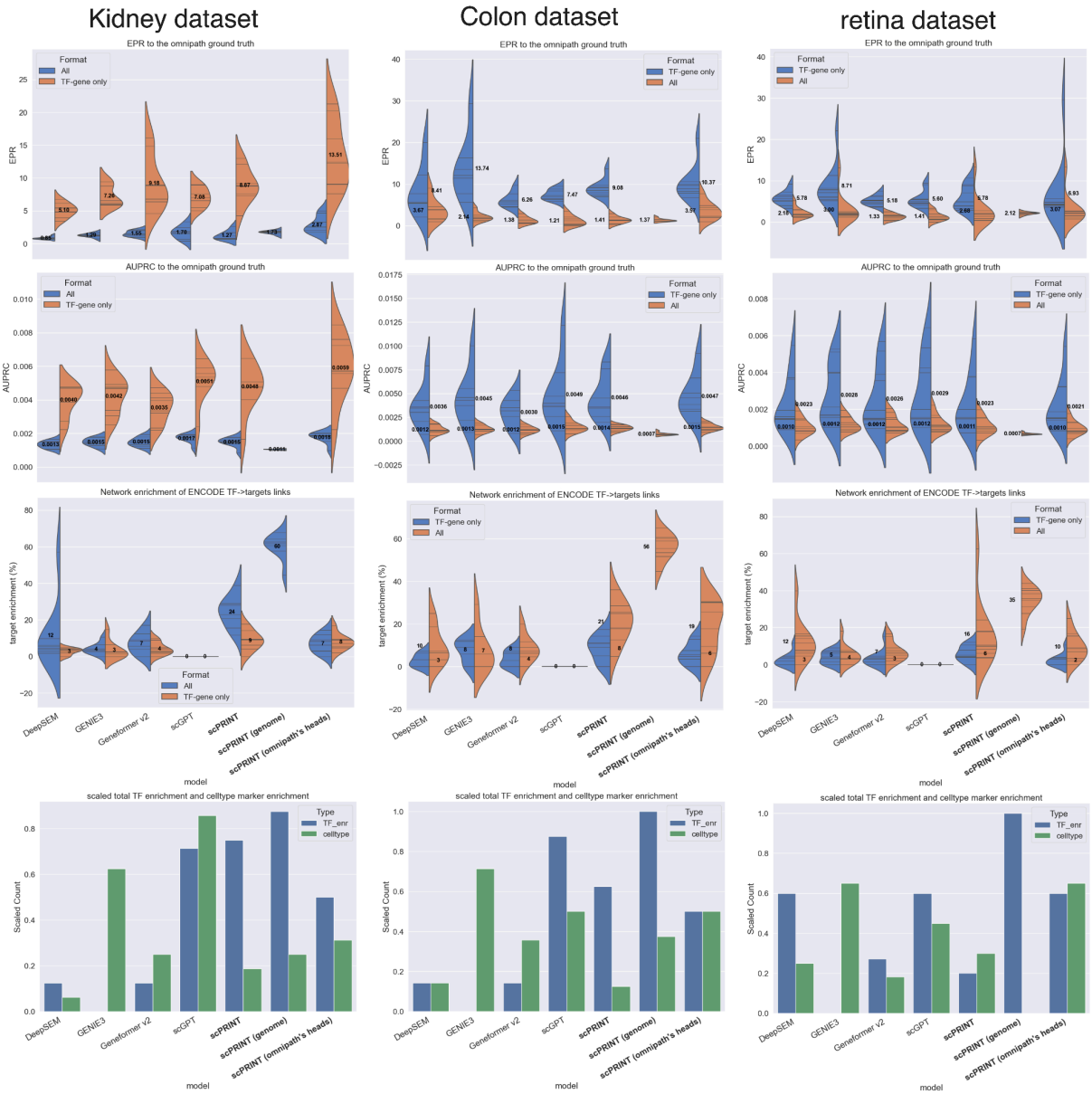
Supplementary figures

FIG S1: visualization of human gene embedding from ESM2



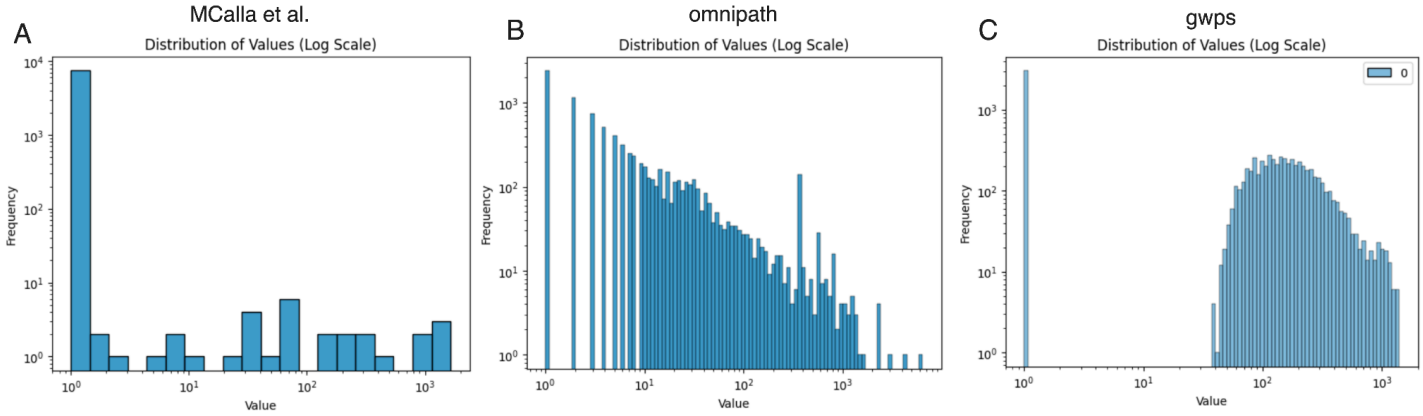
Umap of the ESM2 protein embeddings for the most common protein of all protein-coding genes in Ensembl. The PCA variance ratio is 0.856 for the top 50 principal components. We color it using the Louvain clustering of the embedding.

FIG S2: Gene network inference comparison with Omnipath per datasets



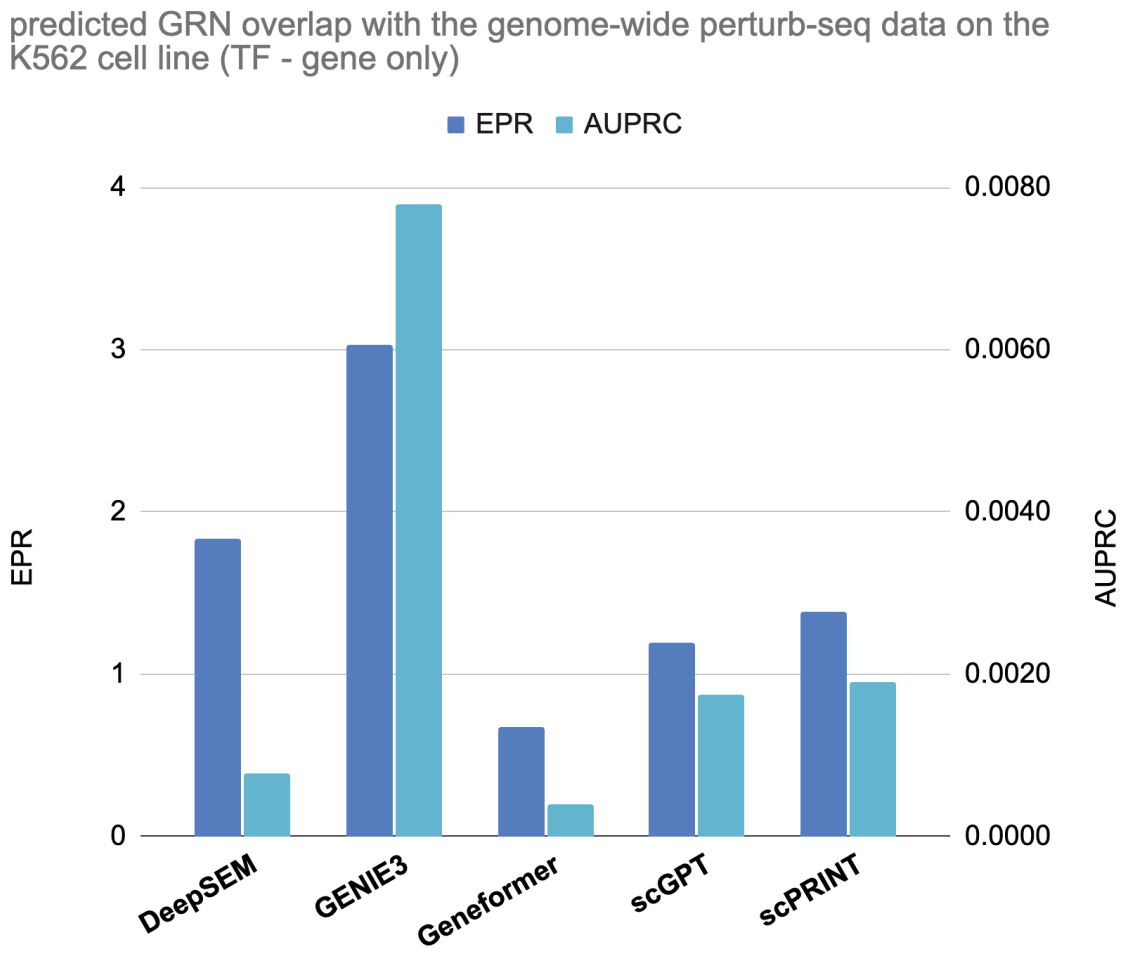
The same plots as in Figures 2B, C, and D, showing the Omnipath and enrichment results per dataset for each of the 3 datasets used. Source data are provided as a Source Data file.

FIG S3: Distribution of connection amongst the three ground truths



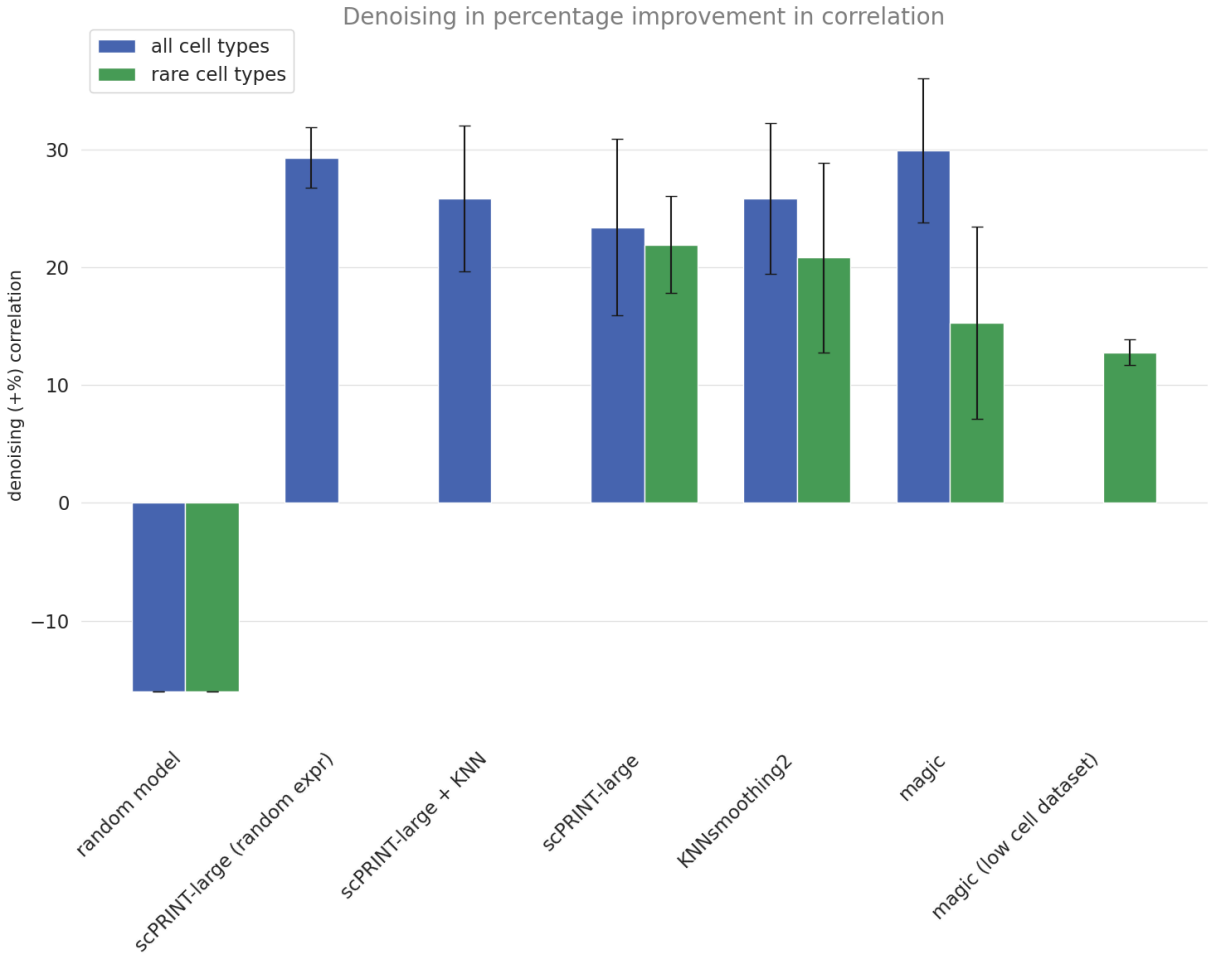
(a) Barplot of the distribution of the number of connections per edge in the MCalla human ground truth network. Most connections are 0, and there is a roughly uniform distribution of connections otherwise. This means most connections belong to the half a dozen most connected edges. (b) Barplot of the distribution of the number of connections per edge in the Omnipath ground truth network. We can see an almost linear relationship on the log-log scale, suggesting a power law distribution. (c) Barplot of the distribution of the number of connections per edge in the genome-wide perturb-seq ground truth network. We can see a very different distribution where only a few genes have little differentially expressed genes post-knock-out, and this trend increases until reaching around 200 connections. Then, it diminishes in what might be a power law. However, some of it is likely caused by the differential expression method and noise in the scRNAseq methodology.

FIG S4: Performance of each GN inference method on predicting the TF-gene only subset of the GWPS ground truth network



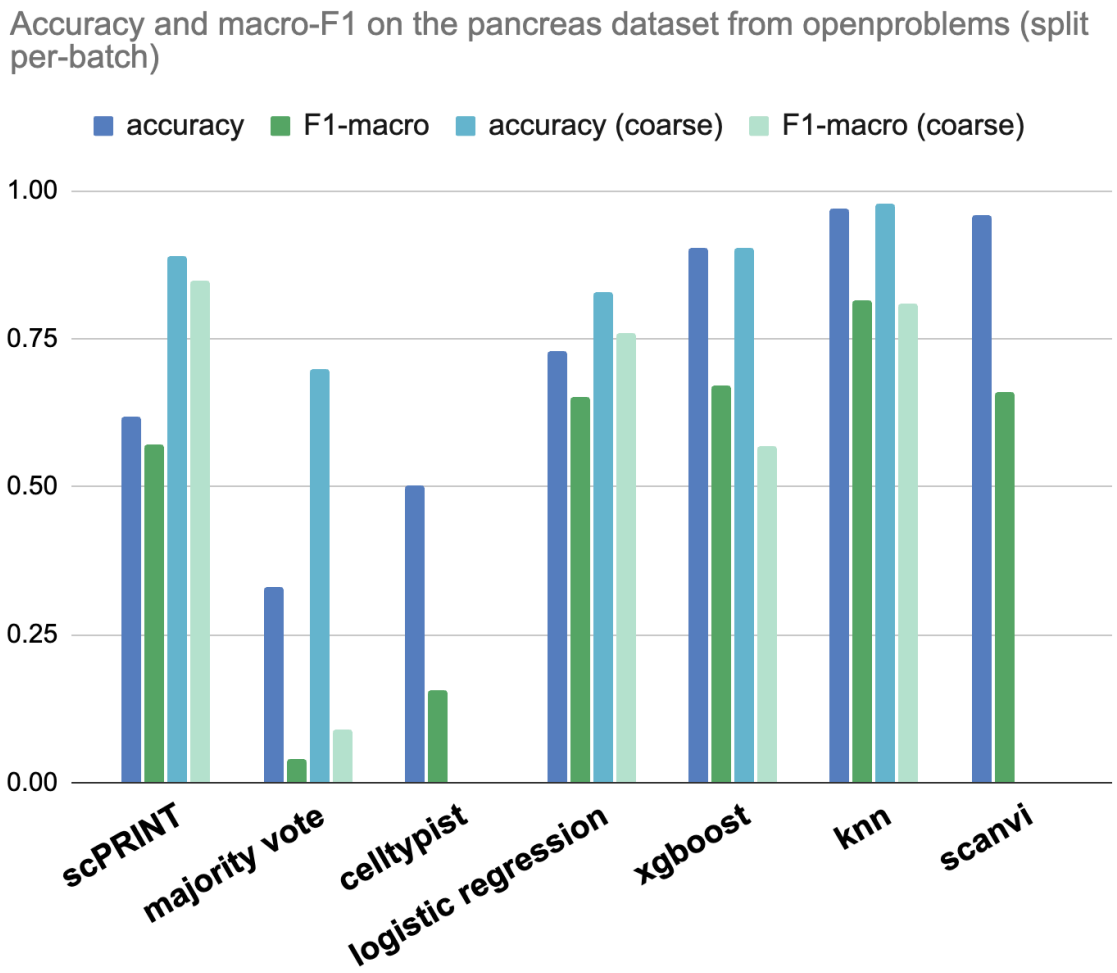
Performances of each model's networks on its overlap with the TF-gene-only subset of the genome-wide perturb seq ground truth. It shows that on this task, most foundation models do not perform well. This could be due to the way their attention matrix is normalized. Source data are provided as a Source Data file.

FIG S5: Full denoising results



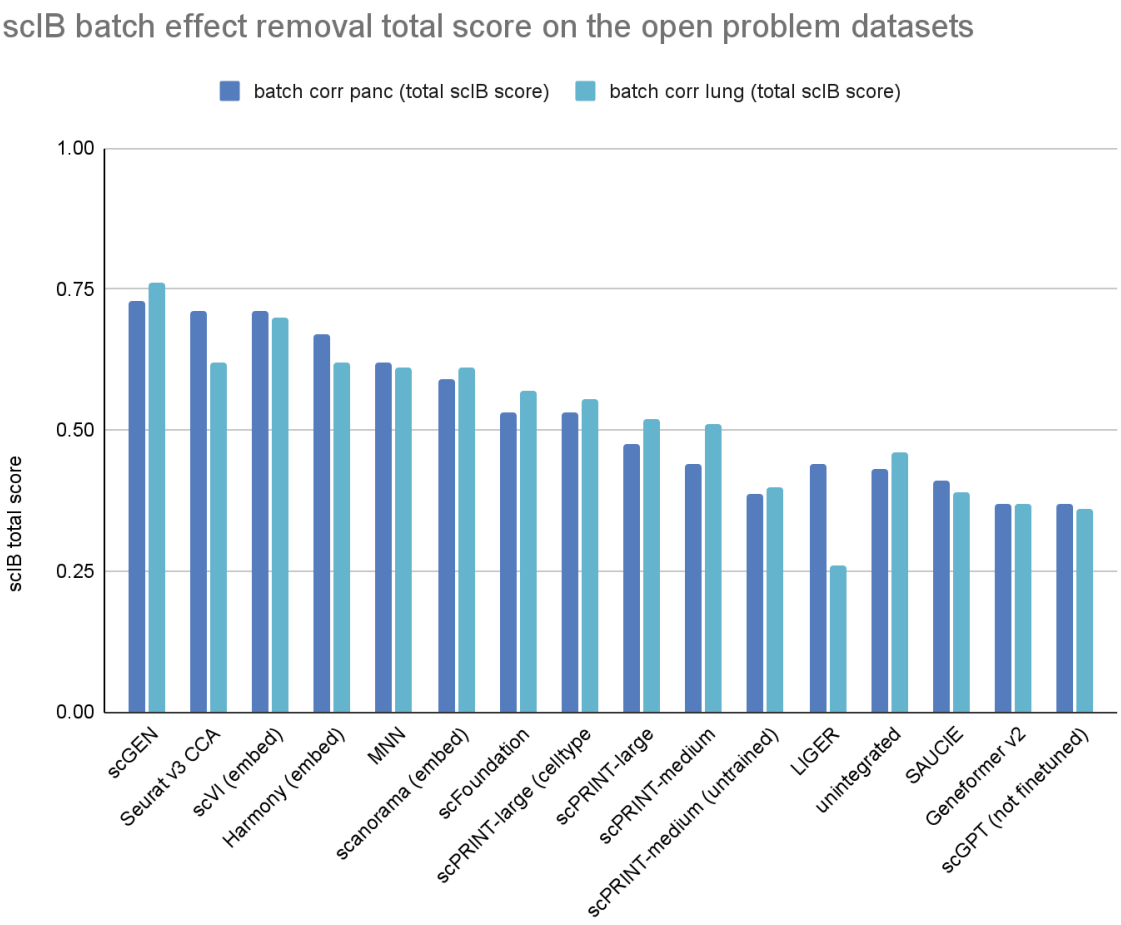
Denoising scores, similar to Figure 4A, but over more tools. “Random model” means a scPRINT model without pre-training. “Random expr” means that scPRINT was using a set of 3000 genes in a similar way as done in pre-training: Taking random expressed genes completed with random unexpressed genes if less than 3000 genes are expressed in the cell. “low cell dataset” means that MAGIC was only using the rare cell population for the dataset as presented in section Denoising validation test of the methods. Source data are provided as a Source Data file.

FIG S6: cell type classification metrics with per-batch split



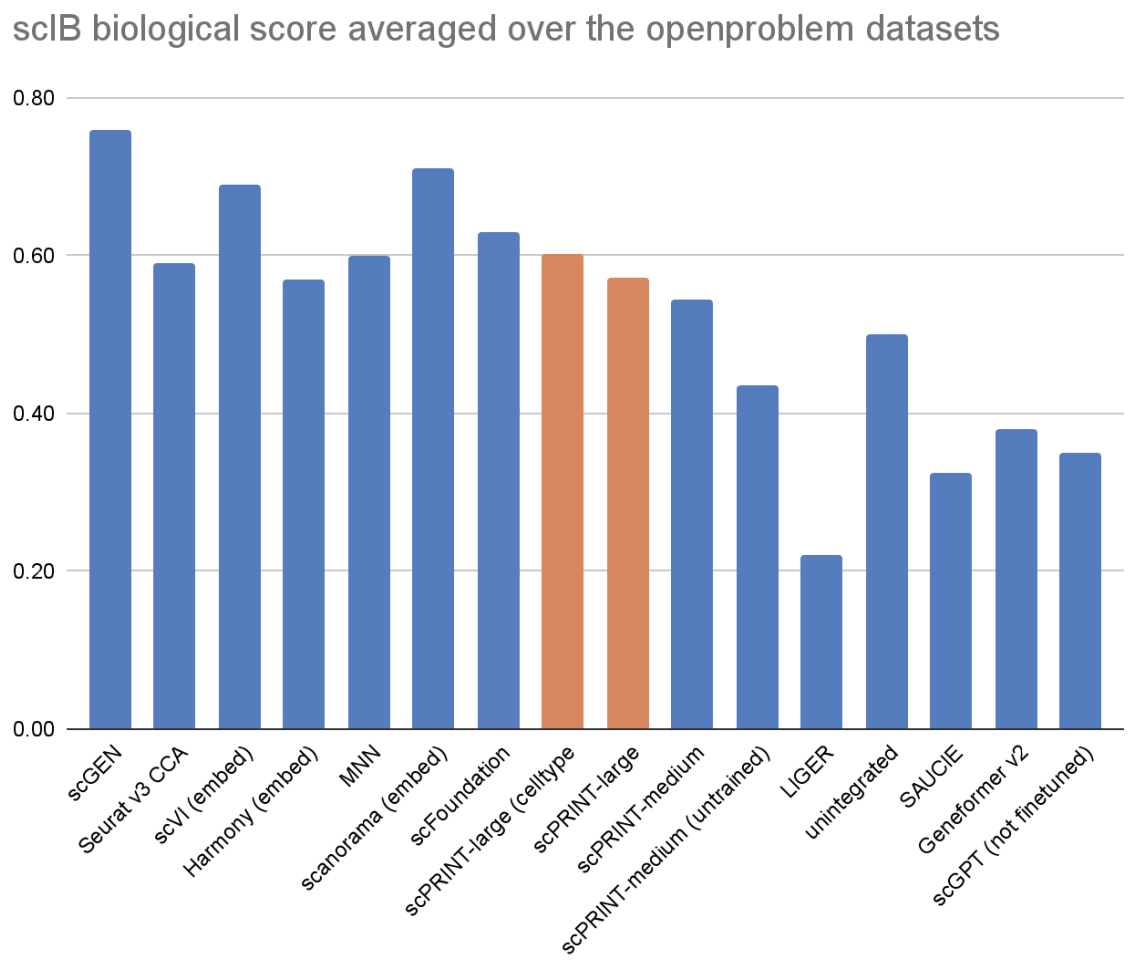
Cell type classification scores over the kidney test dataset of openproblems. Same as Figure 4B, but now the trained methods are trained on a subset of the batches representing roughly 70% of the dataset. The performance is lower in this context, and scPRINT, majority voting, and Celltypist's performance are not changing. Source data are provided as a Source Data file.

FIG S7: Full scIB batch correction scores



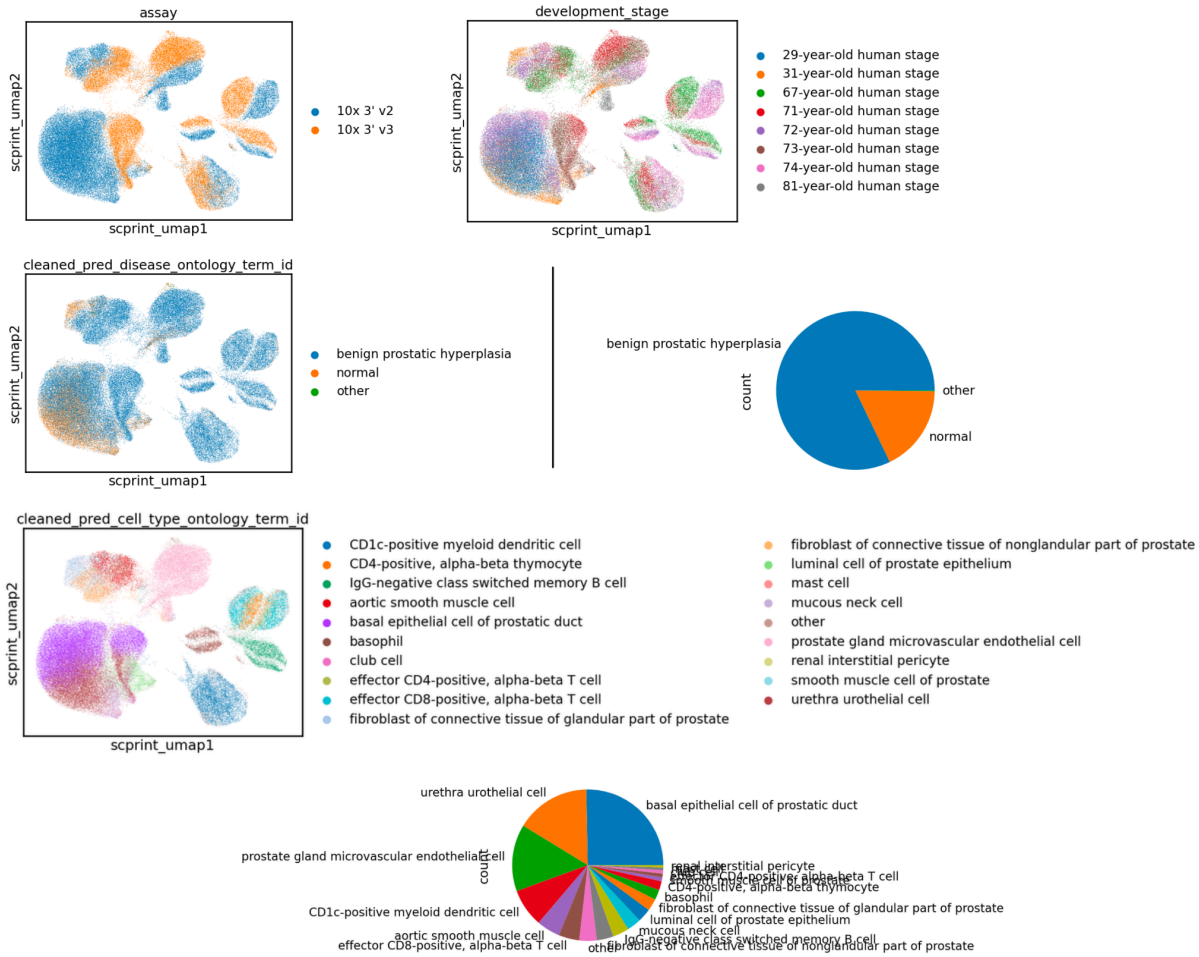
scIB benchmarking scores, averaged for the kidney and lung openproblems test datasets. Same as Figure 4C but over more tools. Cell type logits mean that the logits of the cell type classifier have been used as cell embeddings instead of the cell type embedding itself. Source data are provided as a Source Data file.

FIG S8: Full avgBio scores



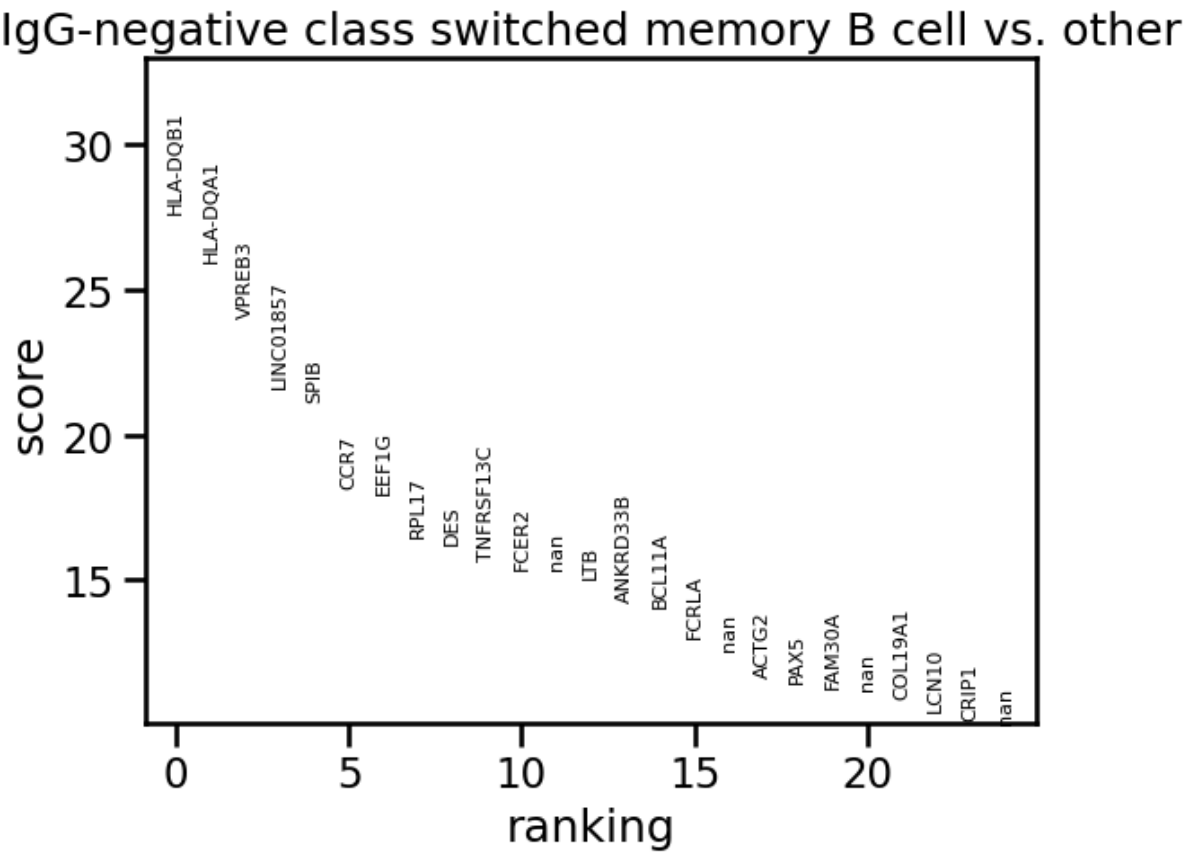
The average Biological score of the scIB benchmark averaged over the kidney and lung openproblems test datasets. Same as Figure 4D but over more tools. Cell type logits mean that the logits of the cell type classifier have been used as cell embeddings instead of the cell type embedding itself. Source data are provided as a Source Data file.

FIG S9: In-depth view of the BPH dataset and its scPRINT-predicted annotations



Detailed view of the assay, development stage, scPRINT-predicted diseases, and scPRINT-predicted cell types. Predicted diseases and cell types have been “cleaned” following the strategy presented in Figure 5. We also add pie charts of the relative abundance of each predicted label.

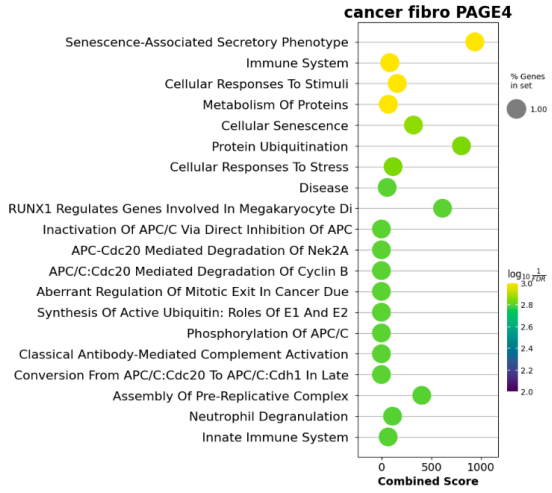
FIG S10: differential expression analysis of the B-cell cluster vs the rest of the cells in the BPH dataset

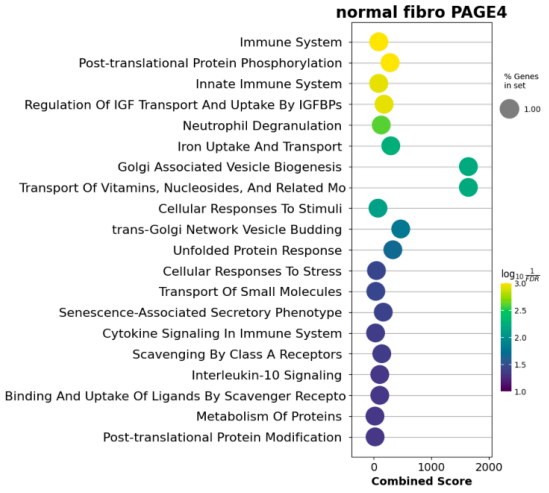


Top genes of the differential expression analysis of the scPRINT inferred B-cell cluster in vs the rest of the cells in the BPH dataset.

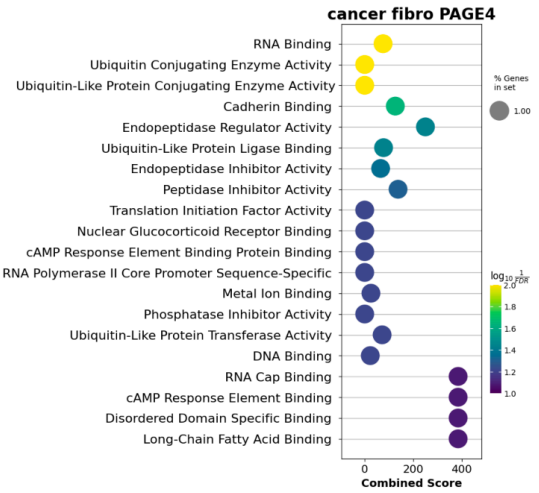
FIG S11: gene enrichment comparison in the PAGE4 GN

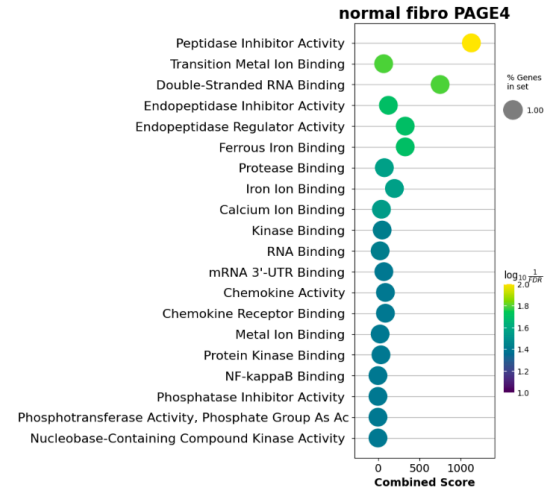
Reactome



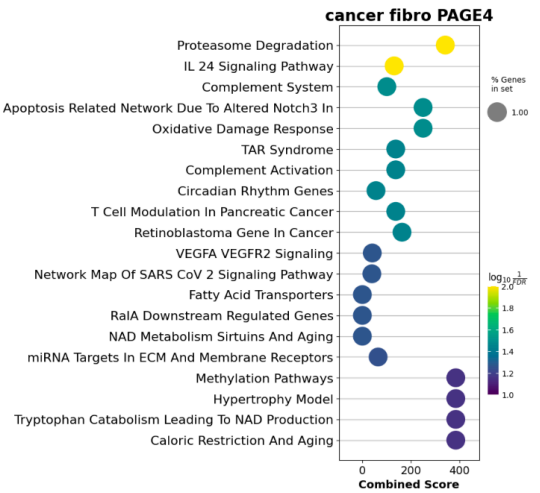


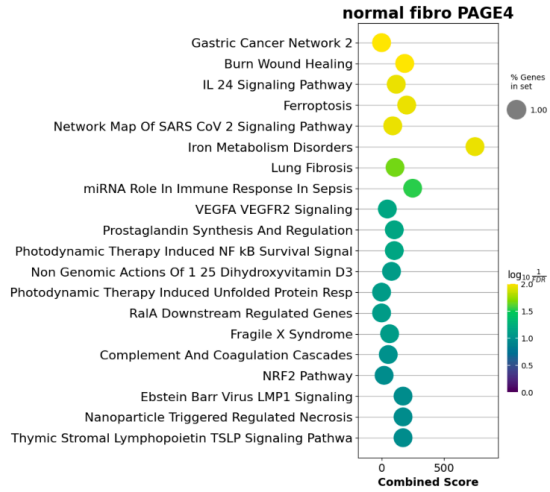
GO Mol. Func.





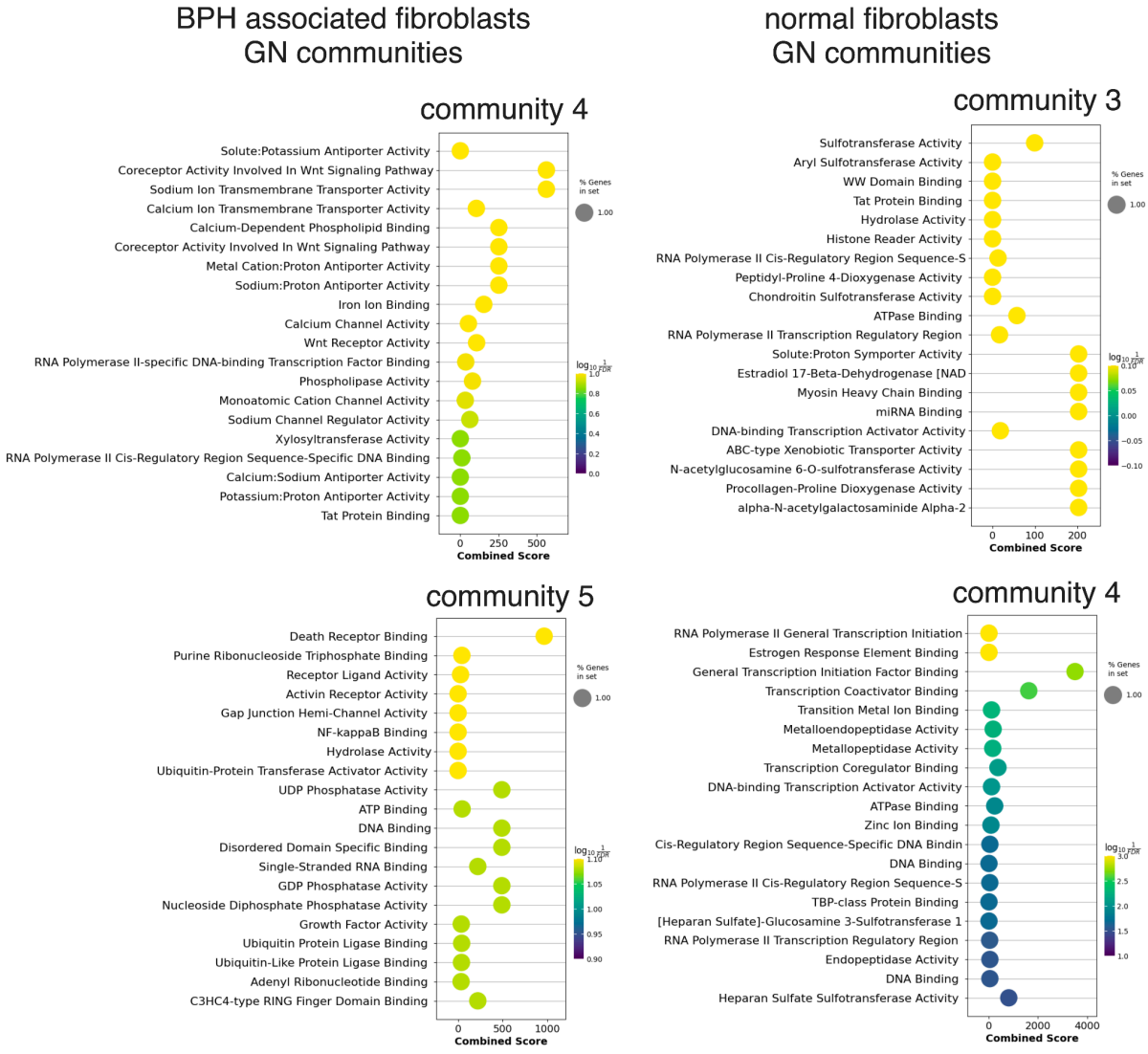
Wikipathways





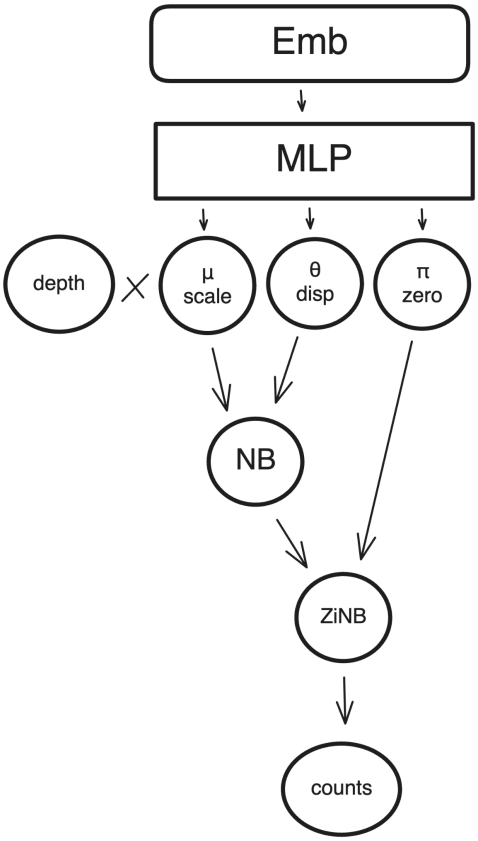
Comparison of the top 20 most enriched terms in Wikipathways, GO molecular function, and Reactome for the 40 most connected genes to PAGE4 in both BPH-associated and normal fibroblast GNs inferred by scPRINT

FIG S12: Gene Network enrichment comparison between the BPH and normal fibroblast on their Louvain communities



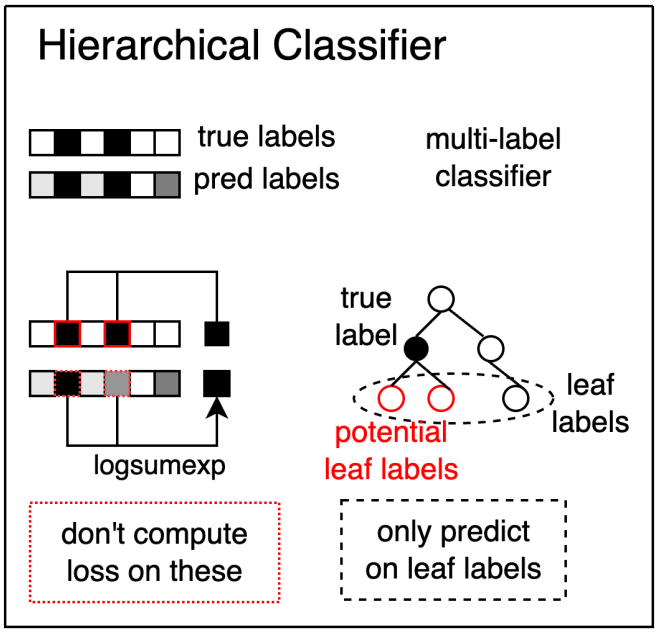
Dotplot of the top 20 GO Molecular function gene sets enriched in the Louvain communities of the BPH and normal fibroblast's Gene Networks.

FIG S13: Graphical Model



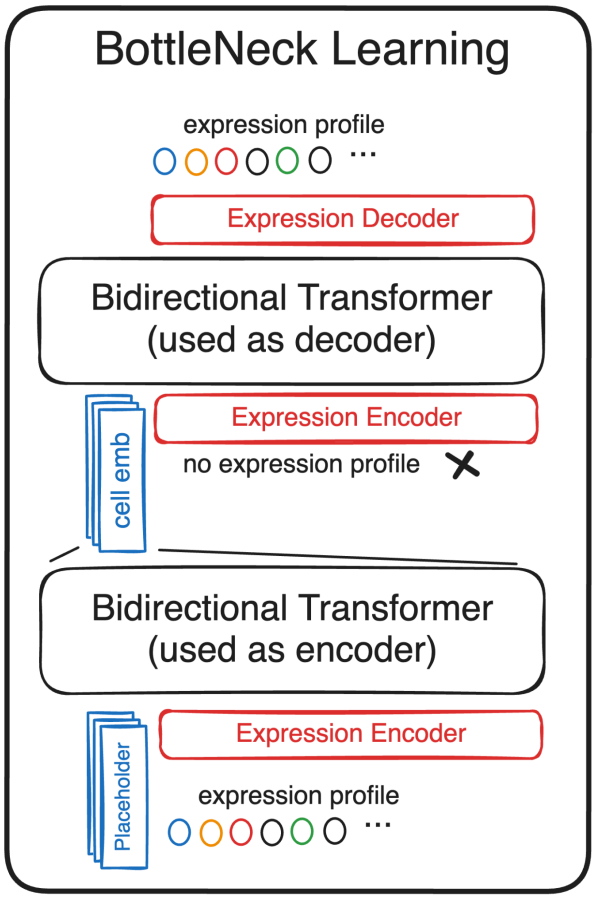
Schematic representation of the zero-inflated negative binomial graphical model of the expression decoder. We generate three values μ, θ, π which are used to model a distribution. We also multiply the μ with the depth (or total count) over the cell.

FIG S14: Hierarchical classifier



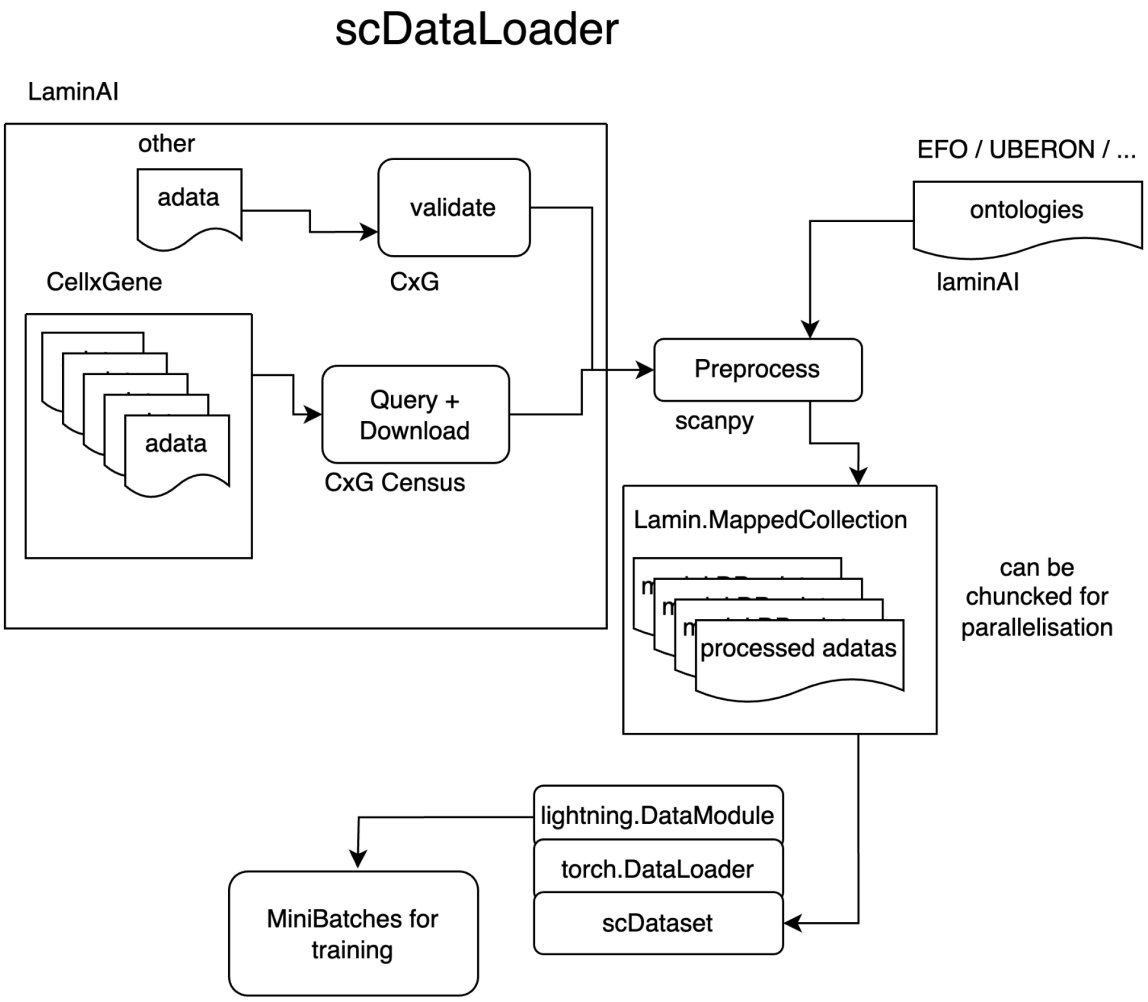
Schematic representation of the hierarchical classifier and its behavior during training. We can train on labels not predicted by the classifier as long as they are parent to one of the predicted labels in the ontological tree.

FIG S15: Detailed representation of the bottleneck learning procedure



Schematic representation of the bottleneck learning procedure where scPRINT's Bidirectional Transformer Encoder is used both as the "Encoder" and "Decoder" of an auto-encoding (AE) bottleneck learning scheme.

FIG S16: Schematic representation of our dataloader



Schematic representation of scDataLoader. Using Lamin.ai, we download and preprocess all cellxgene datasets as AnnDatas. We can also add and validate other expression datasets using lamin.ai. Based on lightning's datamodule framework, torch's dataloaders, our weighted random sampler, and lamin.ai's mapped collection, we can then sample minibatches for pre-training across thousands of datasets and millions of cells with weighted random sampling.