Deep Learning and Numerical Methods for Modeling Complex Biological Systems

A. Ali Heydari

A DISSERTATION
PRESENTED TO THE FACULTY
OF APPLIED MATHEMATICS IN PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE
OF DOCTOR OF PHILOSOPHY

The dissertation of A. Ali Heydari is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Professor Suzanne S. Sindi (chair)

Professor Erica M. Rutter

Professor Maxime Theillard

September 2023

ABSTRACT

Mathematical modeling of biological systems provides a structured framework for exploring and understanding intricate biological phenomena that can be key to unlocking the underlying principles that govern life. Even relatively elementary mathematical models applied to highly complex biological systems have exhibited a capacity to unveil significant patterns and insights, underscoring the substantial utility of computational models in scientific inquiries. In a simplified perspective, mathematical models of biological phenomena can be categorized into three groups: The first group assumes known governing equations and employs mathematical descriptions (*e.g.* Ordinary or Partial Differential Equations) for modeling dynamics and variations. The second group relies on complete observations to learn some or all system components, often utilizing Machine Learning to uncover useful patterns and relationships in the data. The third group aims to use incomplete observations or assumptions to learn from the system, often seeking to generalize findings to unknown agents in the same or different systems.

The goal of this dissertation is to propose novel mathematical techniques to address some of the intricacies inherent in modeling complex biological systems. This dissertation is structured into three parts, each corresponding to the modeling paradigms mentioned earlier, with individual chapters devoted to various methods and applications. The first part of this dissertation introduces a novel numerical method for modeling Partial Differential Equations on deforming geometries. Subsequently, the dissertation transitions to data-driven modeling of biological tissues in humans and animals, leveraging deep learning to capture meaningful insights that can improve the robustness and accuracy of biological analyses. In the final part of this dissertation, novel deep-learning algorithms are formulated to address data or label scarcity, offering substantial improvements in the analysis of systems where prior knowledge or the amount of observations is limited.

Similar to mathematical biology itself, the methods and systems explored in this dissertation encompass a wide spectrum, spanning various scales (from a single cell to tissues) and species (yeast, mice, humans) within the realm of biological sciences. However, they are unified in their pursuit to enable more robust and accurate analysis of complex biological systems. Despite their distinctions, the frameworks presented in this work underscore my dissertation's overarching goal: to provide a comprehensive

toolkit for modeling complex biological systems across different modeling regimes and limitations. Through these efforts, I hope to contribute to gaining a deeper understanding of life and its underlying mathematical principles.

Acknowledgments

First and foremost, I would like to thank my advisor and committee chair, Dr. Suzanne Sindi, for her invaluable help and guidance throughout the years. Dr. Sindi is an incredible mentor who, somehow, can help in any matter or "knowns," someone who can, both academically and personally. This is truly a quality (and a superpower) unlike any I have seen. I cannot recall a single instance when I struggled with something I did not think Dr. Sindi could help fix. Her unwavering support has been instrumental to any success I have had or will have, and for that, I am eternally grateful.

Next, I would like to thank Dr. Elana Fertig, my National Institute of Health F31 co-mentor. Her expertise in mathematical modeling and cancer research has been tremendously helpful in shaping my research, interests, and skillset. I also would like to thank her assistants who handled our one-on-one and mentorship meetings with Dr. Sindi, a task that seemed impossible at times over the years due to our schedules.* I also must thank the funding agencies that made my research possible through the years. I am thankful to the National Science Foundation, Grant DMS- 1840265, and the National Human Genome Research Institute for selecting me as a *National Research Service Award* (F31) recipient, F31HG012718.

I want to thank my committee, Dr. Erica Rutter and Dr. Maxime Theillard, who have provided exceptional guidance and invaluable feedback throughout my PhD. I am grateful to both for always finding time to meet with me to answer any questions, even if that entailed looking at my long and poorly coded C++ code. I would like to acknowledge my immunologist collaborators, Dr. Oscar Davalos and Dr. Katrina Hoyer, who were patient enough to teach a mathematician about perhaps the most complex biological system knownthe immune system. Parts of my dissertation contain material I co-authored with many outstanding scientists, to whom I am grateful. I also would like to thank all of my industry mentors for their invaluable help and support at different stages of my PhD (listed chronologically): Dr. James Bently Brown

*Dr. Sindi and Dr. Fertig are two of the busiest humans I have ever known.

and Dr. Peter Zwart at Lawrence Berkely National Laboratory, Dr. Asif Mehmood at the U.S. Air Force Research Laboratory, Dr. Srevatsan Muralidharan and Dr. Hamid Mohammadi at Amazon, Dr. Juan-Arturo Herrera at Microsoft, Google Research, and Dr. Naghmeh Rezaei and Dr. Javier Prieto at the Consumer Health Research team at Google. There have been many other great mentors and collaborators who I, unfortunately, cannot list due to space constraints; to you all, I want to express my sincere gratitude.

In my more personal acknowledgments, I want to thank my parents, sisters, friends, and family. *Maman, Baba, Zahra, and Fatemeh*: Thank you for setting the sky as the limit for me, believing in me endlessly, and always having my back (except if it costs too much money). To my incredibly loving, caring, patient, and fashionable partner, *Alina*: Thank you for allowing me to be my true self and inspiring me every single day to be a better person. You are the sunshine that brightens my days. My cousin *Amir*: Thank you for helping me fall in love with math. Without your help, I would have definitely failed my 8th-grade math. Thank you *Maia* for being a great friend and my extra pair of eyes, helping me fix my mistakes, whether in writing or my personal life. *Matteo, Oscar, and Alex*: I cannot imagine going through the past few years without you! You all have been extraordinary friends, and I am excited to keep waking up to the most random messages from our group chat in the years to come.

Last but certainly not least, I want to acknowledge the fantastic faculty and my colleagues at the Applied Math department. You all have been instrumental in my personal and academic growth.

To all who kindled my mathematical dreams, and to all whose support made those dreams a remarkable reality.

Contents

Listing of Figures

Everything should be made as simple as possible, but

no simpler.

Albert Einstein

1

Introduction

The history of mathematics and its applications to various modeling stands as a testament to its pervasive and transformative influence across diverse scientific disciplines. Mathematics' origins can be traced back to ancient civilizations, where early mathematicians and astronomers, such as the Babylonians, Persians, Egyptians, and Greeks[3], developed equations to describe celestial motion and geometric principles. However, it was not until the Renaissance that mathematical modeling began to take a more structured form[4], with pioneers like Galileo Galilei and Johannes Kepler using mathematical principles to model planetary orbits and motion[4]. The 17th cen-

tury saw the emergence of calculus, courtesy of Isaac Newton and Gottfried Wilhelm Leibniz, which provided powerful tools for modeling and solving complex scientific problems. In the 19th century, mathematical modeling extended its reach into physics, engineering, and chemistry, with luminaries like Carl Friedrich Gauss, Joseph Fourier, Pierre-Simon Laplace, and Lord Kelvin making significant contributions to the advancements[5]. The 20th century witnessed unprecedented breakthroughs in mathematical modeling across fields like biology, economics, and environmental science, propelled by computing breakthroughs and the realization that complex natural phenomena could be unraveled through quantitative approaches[5]. Today, mathematical modeling remains at the forefront of scientific inquiry, facilitating exploring, understanding, and predicting intricate phenomena in fields as varied as climate science, epidemiology, finance, and beyond. Its historical trajectory highlights its enduring significance in shaping how we comprehend and engage with the natural world.

With the advent of mathematical modeling, the field of *Applied Mathematics* emerged to address complex real-world problems by utilizing mathematical principles and techniques to formulate practical solutions. Applied Mathematics aims to bridge the gap between theoretical mathematics and the tangible challenges faced in various scientific and engineering domains. Applied Mathematics has been instrumental in advancing our understanding of complex phenomena, improving the optimization of processes, predicting future outcomes, and making informed decisions across diverse fields. Applied Math's interdisciplinary approach has not only expanded the boundaries of mathematical curiosity but has also significantly contributed to technological advancements, scientific breakthroughs, and the overall enhancement of our modern society. Within the realm of Applied Mathematics, few branches have exhibited as much unexpected potential and promise as *Mathematical Biology* and *Machine Learning*, which are the main focuses of this dissertation.

Both Mathematical Biology and Machine Learning have experienced a paradigm shift propelled by the availability of high-throughput biological data, sophisticated computational tools, and interdisciplinary collaborations[6]. These developments have empowered researchers to construct increasingly intricate models that capture the intricacies of biological systems, from cellular processes to ecosystems. Mathematical Biology has played a pivotal role in developing personalized medicine, enabling tailored treatment strategies based on an individual's genetic makeup and disease profile. In epidemiology, mathematical modeling has been instrumental in predicting disease outbreaks, guiding public health responses, and optimizing vaccination campaigns, as exemplified during the COVID-19 pandemic. As a result, the interdisciplinary nature of Mathematical Biology continues to shape our understanding of life sciences and offers invaluable insights for addressing pressing challenges in healthcare, environ-

mental conservation, and beyond.

In the following sections, I provide a brief overview of the recent advances in Mathematical Biology and Machine Learning, motivating the main contributions of my dissertation. I discuss the specifics of each chapter and how they fit in the grand scheme of modeling complex biological systems.

1.1. THE NEW FRONTIERS IN MATHEMATICAL BIOLOGY

Classical mathematical biology models predominantly used differential equations to describe changes occurring in a system over time[7]. These differential equations served as powerful tools to describe dynamic processes, allowing researchers to quantify the rates of change of various biological entities. This foundational approach paved the way for a deeper understanding of biological systems, such as population dynamics, biochemical reactions, and the spread of diseases. Moreover, it established a strong mathematical framework for investigating biological phenomena and laid the groundwork for developing more sophisticated modeling techniques and computational methods that have since expanded the field's scope and capabilities.

As computational resources and data availability increased, mathematical biologists began exploring new areas of applications. Additionally, technological advancements of the late 20th century paved the way for computational biologists to research the human genome, a feat deemed impossible just half a century ago[8]. These efforts, spearheaded by the Human Genome Project[9], aimed at deciphering the chemical makeup of the entire human genetic code. The Human Genome Project played a crucial role in driving advancements in DNA sequencing, ultimately paving the way for *high-throughput sequencing technologies* for the genome and transcriptome. These technologies allowed for the rapid and cost-effective generation of vast amounts of biological data, including DNA sequences, gene expression profiles, and protein interactions. In response to this influx of data, mathematical biologists have turned to leveraging statistical and machine-learning techniques in their modeling approaches.

The utilization of Machine Learning in modeling complex biological systems is catalyzing transformative shifts across various branches of Mathematical Biology, including Computational Biology[10], Genomics[6], and Bioinformatics[11]. The integration of machine learning techniques has unlocked new avenues for data-driven discovery and decision-making in these areas, enabling the extraction of valuable insights from vast and complex biological datasets. However, the application of machine learning in the context of biological data is not without its challenges. Biological datasets often exhibit inherent nuances, including technical and biological noise, high dimensionality, and sparsity, which pose substantial hurdles to efficient and effective learning[12].

I provide a more extensive biological background of the challenges associated with biological data in Chapter 3.

1.2. THE RISE OF DEEP LEARNING

The inception of Machine Learning dates back to the mid-20th century, marked by the emergence of initial concepts and rudimentary algorithms in the hopes of recognizing letters of the alphabet[13]. These early days were characterized by limited computational resources and relatively simple models, such as the *perceptron* and *linear regression*[13]. Progress was modest, and the field faced considerable challenges due to the scarcity of high-quality data and computational power. However, Machine Learning has experienced remarkable advancement over time, to the point of being referred to as *Artificial Intelligence*. The integration of more sophisticated algorithms, the explosion of big data, and the advent of powerful hardware have tremendously contributed to the fields' remarkable growth. Notable milestones include the development of neural networks, reinforcement learning techniques, and the rise of *Deep Learning*.

Deep Learning is a subset of Machine Learning that uses deep neural networks (DNNs) to analyze large and complex datasets. DNNs comprise layers of artificial neurons inspired by how human neurons work. Each neuron takes the weighted summation of all inputs and passes it through a non-linear activation function, creating a stack of hidden layers. The input information flows from the input layer through the hidden layers, and the model generates an output at the last layer, the output layer. The large set of trainable weights of the neurons and the non-linear transformations enable DNNs to capture underlying complex patterns of the data. Training a DNN is the process of determining these trainable weights to optimize model performance. Recent advances in Deep Learning have revolutionized almost all areas of science, notably Computational Biology and health-related research. I provide a more complete background of various Deep Learning architectures used in Computational Biology in Chapter 3.

1.3. THREE MATHEMATICAL MODELING REGIMES

In a simplified view, any mathematical model of biological phenomena is grouped into three cases: First, models that assume the underlying governing equations are known. These models often leverage classical mathematical frameworks (*e.g.* Ordinary or Partial Differential Equations) to model change over time (or space), or to infer important biological parameters (*e.g.* tumor growth rate). The second group con-

sists of models that assume enough observations are present to learn the specifics of system components, often utilizing some form of Machine Learning to *learn* key parameters or to identify the agents within the system. An example of such a model is a classifier that learns to predict the specific condition of patients based on blood work through supervised learning[14]. The last group is made up of algorithms that are meant to model incomplete observations or assumptions, aiming to generalize the learnings to other systems or improve the initial set of assumptions. Primary examples of such models are *few-shot* or *zero-shot* models[15] trained on very few examples of different classes within the observations, aiming to learn structural similarities that can be exploited on future unseen inputs to the model.

Given the tremendous utility of mathematical and machine learning models in modeling biological systems, this dissertation aims to propose novel frameworks that enable researchers to study complex biological systems, particularly those relating to diseases, in a more robust and accurate manner.

1.4. MODELING PHENOMENA USING GOVERNING EQUATIONS

Mathematical modeling of systems based on prescribed governing equations has been instrumental in scientific and engineering applications. Such models are built on the premise that natural phenomena and physical systems can be described by mathematical equations, typically in the form of differential equations that aim to capture change in dynamics. By formulating and solving these equations, researchers gain insights into the fundamental principles governing the behavior of the desired system. Models based on governing equations are widely used to model the flow of fluids[16,17], the spread of diseases[18,19], or the behavior of financial markets[20,21]. In the real world, most systems vary both in time *and* space, thus necessitating the need for Partial Differential Equations (PDEs). PDEs are frequently used to model biological systems, allowing researchers to describe and analyze complex biological processes that experience spatial and temporal variations on complex geometries.

Solving some of these models analytically, *e.g.* PDEs, often fail due to the inherent complexities and nonlinearity of many real-world phenomena. The challenges of deriving closed-form solutions highlight the need for developing numerical methods that approximate these systems' solutions. Among the numerical methods for solving PDEs, finite difference (FD), finite elements (FE), and finite volumes (FV) are perhaps the widely used methods. Finite differences, the simplest and most intuitive method among the three, discretize the domain into a grid of points and then approximate the derivatives in the partial differential equation at each point. Though FD

methods are often well-suited for simple problems, they can be inaccurate for problems with more complex geometries. Biological systems inherently possess interesting geometries that require FE or FV: FE methods to discretize the domain into a set of elements and approximate the solution to the partial differential equation over each element. Though constructing different elements provides much-needed flexibility, it can potentially be the most computationally expensive method. FVs methods, on the other hand, start by discretizing a domain into a set of *control volumes*, then approximating the integral conservation law on each control volume. Due to its construction, FV is ideal for solving PDEs over complex geometries as they can be more efficient than FEs but potentially less accurate.

Given the geometrical complexities of biological systems, FV are frequently used to solve biologically-motivated PDEs. However, it is known that FV methods are not conservative on deforming geometries, which is a critical challenge given that many of the most basic biological phenomena include some deformation in their initial geometry (*e.g.* cell division). To address this issue, I present a novel *conservative* formulation of the FV method in Chapter 2. Motivated by experimental observations of protein aggregation in dividing yeast cells, I present a conservative finite volume approach for reaction-diffusion systems defined over deforming geometries. The key idea of this approach is to use spatio-temporal control volumes instead of integrating the time-discretized equations in space, as it is common practice. The theoretical and computational results demonstrate the convergence of this method and highlight how traditional approaches can lead to inaccurate solutions. In this chapter, I present the results of employing our approach to investigate the partitioning of protein aggregates in dividing yeast cells, leveraging the flexibility of the level set method to construct realistic biological geometries. Using a simple reaction-diffusion model, we find that spatial heterogeneity in yeast cells during division can alone create asymmetries in the concentration of protein aggregates. Moreover, we find that obstructing intracellular entities, such as nuclei or insoluble protein compartments, amplify these asymmetries, suggesting they may be essential in regulating molecular partitioning. Beyond these findings, our results illustrate the flexibility of our approach and its potential to design realistic predictive tools to explore intracellular bio-mechanisms.

1.5. LEARNING FROM LARGE-SCALE BIOLOGICAL OBSERVATIONS

Mathematical and machine learning frameworks are key to unlocking the full potential of large-scale biological data analysis. In the current age of biological “big data" characterized by massive, intricate datasets, computational models offer a structured and effective means to unveil biological systems’ underlying dynamics and

patterns. Machine learning models have shown tremendous potential in extracting meaningful features, relationships, and predictive insights from such data. Moreover, mathematical frameworks provide a foundation for integrating diverse data types, facilitating the creation of comprehensive models encompassing genomics, proteomics, metabolomics, and more. These frameworks not only enhance our understanding of complex biological processes but also empower advancements in personalized medicine, disease diagnosis, drug discovery, and the elucidation of intricate biological networks. Such mathematical models have the potential to bridge the gap between raw biological data and actionable knowledge, making them an indispensable tool in modern mathematical biology.

The era of big biological data is most prominently marked by high-throughput genomics[22]. This data is a valuable resource for biologists, helping them identify patterns, discover trends, and test their hypotheses. Additionally, big data has opened up exciting research possibilities, such as creating data-driven theories to improve biological predictions, studying the effects of global changes on different organisms, and developing tools that make it faster to use models with data. The marriage of Machine Learning with Biology has revolutionized genomics, enabling the rapid and cost-effective sequencing of genomes, transcriptomes, and epigenomes. In biomedicine, machine learning has empowered the development of predictive models for disease diagnosis, drug discovery, and treatment optimization, fostering personalized healthcare approaches. The multifaceted nature of genomics data demands continual advancements in machine learning algorithms that are specialized for biological applications. Researchers grapple with the need for efficient yet robust methods to handle noisy datasets, extract meaningful features from high-dimensional spaces, and address the challenges of data *scarcity*. Overcoming these challenges remains central to harnessing the full potential of machine learning in unraveling the mysteries of biology and advancing our ability to tackle pressing issues in healthcare, genetics, and beyond. The later chapters of my dissertation aim to propose new methods that leverage deep learning to improve the robustness and accuracy of transcriptomics data, particularly single-cell RNA sequencing (scRNAseq). ScRNAseq technologies allow gene expression measurements at a single-cell resolution. This gives researchers a tremendous advantage for detecting heterogeneity, delineating cellular maps, or identifying rare subpopulations. Chapter 3 is dedicated to providing the necessary mathematical and biological background for the methods I later propose.

Despite the significant advances in deep-learned models for scRNAseq, the lack of interpretability is a limitation of current deep-learning approaches for scRNAseq analysis. Moreover, existing pipelines are designed and trained for specific tasks used

disjointly for different stages of analysis. In Chapter 4, I propose a novel *interpretable* deep learning framework for enhancing the analysis of scRNAseq data. Our approach, called *scANNA*, leverages neural attention to learn gene associations. After training, the learned gene importance (interpretability) is used to perform downstream analyses (e.g., global marker selection and cell-type classification) without retraining. I demonstrate that ScANNAs performance is comparable to or better than state-of-the-art methods *designed and trained for specific standard scRNAseq analyses*, even though scANNA was *not* trained for these tasks explicitly. ScANNA enables researchers to discover meaningful results without extensive prior knowledge or training separate task-specific models, saving time and enhancing scRNAseq analyses.

1.6. MATHEMATICAL MODELING OF BIOLOGICAL SYSTEMS IN DATA-LIMITED REGIMES

Data collection from biological systems is inherently challenging due to existing biological technical variations. Despite the rapid advancements in data generation technologies, numerous biological domains still grapple with insufficient or sparse datasets. The intricate nature of biological systems, ethical considerations, cost-intensive experimental procedures, and the immense diversity of life forms contribute to this scarcity. As a result, researchers often face limitations when attempting to build robust models or draw statistically significant conclusions, impeding the depth and breadth of research insights and hindering the development of accurate predictive models. Addressing these challenges requires innovative approaches that leverage advanced statistical techniques and machine learning algorithms.

Given the formidable challenges associated with extracting biological insights and constructing reliable predictive models from complex datasets, there has been a notable upsurge in adopting deep learning-based models to augment data quality and analysis. Deep learning has exhibited significant promise by effectively handling vast volumes of high-dimensional biological data, such as single-cell RNA sequencing (scRNAseq) data, allowing researchers to uncover intricate patterns and relationships. However, it is crucial to recognize that adapting deep learning techniques from other domains, like Computer Vision and Natural Language Processing, to the unique intricacies of biological processes can present challenges and limitations. To address these issues, Chapters 5 and 6 are dedicated to the development of novel deep learning frameworks specifically tailored for the intricacies of computational biology, offering innovative solutions to enhance the modeling and analysis of biological studies with limited observations or knowledge of the system. The methodologies introduced in my dissertation exhibit a robust and versatile nature, offering potential utility in mod-

eling observations from diverse biological systems. While my dissertation primarily centers on developing these deep learning techniques in the context of scRNAseq studies, their adaptability extends to various other biological domains. This emphasis on scRNAseq data is driven by its broad applicability and the distinctive hurdles it presents, given its inherent complexity, high feature dimensionality, and the presence of technical and biological noise.

A critical complication in scRNAseq studies remains the scarcity of detailed information about individual samples in biological systems, including scRNAseq studies. Among the most critical limitations in scRNAseq research is the absence of reliable labels for individual cells, which can significantly hinder our ability to perform crucial downstream tasks, limiting our ability to infer cell-cell communication, differentiation trajectories, and disease progression. Consequently, numerous models have been proposed to predict cell types and fate[6]. However, most such models demand extensive data with reliable labels, both of which are often challenging to obtain in real-world applications. In Chapter 5, I introduce a semi-supervised deep learning approach inspired by Partial Differential Equations (PDEs), capable of accurately predicting cell types even when provided with limited training data. Beyond its significant biological implications, the proposed model called *Drift-Diffusion Graph Neural Network* (DD-GNN) offers a valuable mathematical framework for designing graph neural networks founded on dynamics prescribed on the model's latent manifold. This framework introduces a high degree of adaptability, enabling the modeling of various biological systems by incorporating specific dynamics.

In addition to the inherent challenge of limited sample sizes, researchers face another major hurdle: the low number of single-cell observations stemming from the rarity of subpopulations, tissue degradation, or cost. This absence of sufficient data may cause inaccuracy or irreproducibility of downstream analysis. Chapter 6 of my dissertation proposes *Automated Cell-Type-informed Introspective Variational Autoencoder (ACTIVA)*: a novel framework for generating realistic synthetic data using a single-stream adversarial variational autoencoder conditioned with cell-type information. Within a single framework, ACTIVA can enlarge existing datasets and generate specific subpopulations on demand instead of two separate models. Data generation and augmentation with ACTIVA can enhance scRNAseq pipelines and analysis, such as benchmarking new algorithms, studying the accuracy of classifiers, and detecting marker genes. ACTIVA will facilitate the analysis of smaller datasets, potentially reducing the number of patients and animals necessary in initial studies. Chapters 5 and 6 propose novel techniques tailored to addressing the challenges posed by scRNAseq datasets with extremely limited data, having the potential to revolutionize our com-

prehension of intricate biological processes. Such an advancement has far-reaching implications, spanning disciplines from developmental biology to disease research, where it could expedite groundbreaking discoveries and innovative applications.

Portions of my dissertation contain research that has been published in peer-reviewed venues, with me as the first author in all papers [17,23,24,25]. Therefore, per the University of California, Merced's guidelines, each chapter has been written to be self-contained while the common thread in all chapters remains focused on modeling complex biological systems. The following freestanding chapters will adhere to the standard structure of scientific publications, as directed in university dissertation guidelines, each containing an introduction, background, methodology, results, and discussion sections, though certain sections may recall more complete descriptions from previous chapters as a reference to the reader.

Part I

Systems with Known Governing Equations

Truth is the biggest lie of all. All truths are rooted in assumption. In fact, real truth isn't a stagnant point, but a dynamic force of eternal correction.

Abhijit Naskar

2

Solving Governing Equations on Deforming Geometries

2.1. INTRODUCTION

Cells are often considered to be the smallest unit of life because they group critical constituents within a single compartment. Inside the boundary of a cell, the environment is continually changing as molecular species are constantly created, degraded, and interacting with one another[26]. Mathematical models have proven to be powerful tools in biology through their ability to provide abstract representations of the cellular environment, generate simulations under distinct hypotheses, and provide quantitative output that can be validated through comparisons with experiments[27,28]. However, mathematical modeling necessarily involves simplifying assumptions about the cell itself and its environment.

One common assumption is that the cell environment is well-mixed and spatially homogeneous, allowing for the use of deterministic ordinary differential equations or stochastic simulations of the chemical master equation, which offer both analytical and computational advantages over their partial differential equation counterparts[29,30,31,32]. However, even if the cell began as a well-mixed compartment, reaction, diffusion, and cellular reconfiguration can create spatial heterogeneities, which

may cause unequal partitioning after cellular division[33,34]. More recently, structured population equations have proven useful to consider populations of cells where intracellular constituencies may be partitioned unequally[34,35]. However, these models still assume that the environment is well-mixed within individual cells. With the increasing ability to experimentally observe distributions of biochemical species within individual cells, it becomes necessary to favor mathematical methods that model spatial heterogeneity. For such a model to be realistic, it must reproduce observed cell deformations while preserving essential physical properties such as mass conservation.

Over the last decades, the level set method[36,37] has been recognized as a versatile interface representation, virtually applicable to any interfacial problem. The interface is defined as the zero contour of a continuous function (*i.e.* the level set function), and an advection equation models its evolution. In doing so, the costly mesh conformation inherent to any method explicitly tracking the geometry is replaced by the numerical resolution of a standard partial differential equation. The level set method has been employed to simulate a broad range of applications in computer, engineering, and natural sciences, such as the dynamic multi-phase flows[38], the electrostatic of biomolecules[39], or the response of elastic structures[40]. In computational biology, it has for example been used to simulate the behavior of self-healing[41], growing[42] and shear stress-stimulated[43,44] tissues, tumor growth[45,46,47], wound healing[48] or the protrusion of cells in micropipettes[49]. Even though the level set formalism seems to be the natural mathematical tool for studying the dynamics of single cells, to the best of our knowledge, it has not been used to study protein aggregation in three-dimensional yeast cells.

In this work, we present a framework for studying the reaction-diffusion process in deforming cells and employ it to simulate prions aggregation in dividing yeast cells. First, we construct a level-set-based model to represent the budding cell cycle of yeasts; that is, our model begins with a single compartment (mother cell), which, through the budding processes, produces a growing daughter cell that ultimately disconnects completely (see Fig. 2.2). Second, we construct a novel finite volume formulation that ensures mass conservation even on deforming geometries. Third, we validate our model using analytical and practical examples and finally employ it to study the impact of intracellular material and biochemistry on the prions transmission process. Our results both illustrate the limitations of the well-mixed assumptions and the ability of the reaction-diffusion mechanisms to create asymmetric distribution alone. In addition to providing insights into the segregation of proteins in yeast, our work demonstrates the power of our framework to serve as a tool for modeling cell division and intracellular dynamics, providing a powerful testbed for mathematical biologists to generate predictions on an increasingly relevant experimental scale.

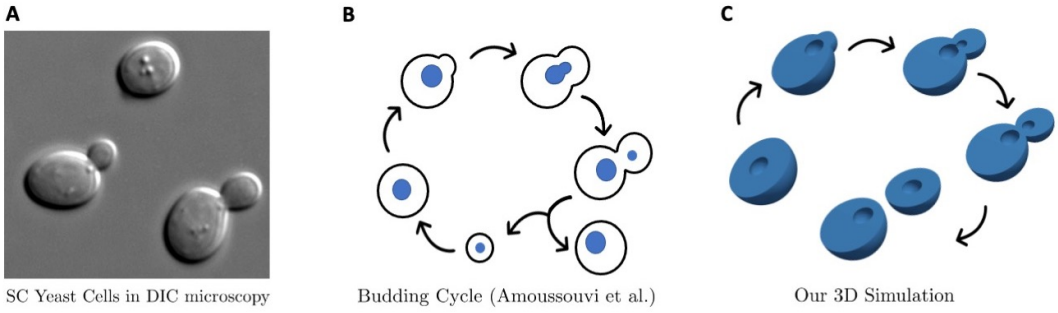


Figure 2.1: **Asymmetric yeast cell division**. **(A)** Saccharomyces cerevisiae yeast cells under Differential Interference Contrast (DIC) microscopy showing two yeast cells (at the bottom) budding[1]. **(B)** Illustration of a yeast cell budding, as depicted by Amoussouvi *et al.*[2]. **(C)** Half-space rendering of our three-dimensional simulation of single yeast cell budding.

We start in Section 2.2 by providing background for our biological motivations. Section 2.3 describes our mathematical model, both the reaction-diffusion system and the level set representation. We then introduce our conservative finite volume formulation in section 2.4 and validate it in section 2.5. We apply our framework to simulate prions dynamics in dividing yeast cell in Section 2.6 and conclude in Section 2.7.

2.2. BIOLOGICAL BACKGROUND

2.2.1. PROTEINS AGGREGATION AND PRIONS DISEASES

Proteins are linear sequences of amino acids, which then fold into a three-dimensional shape or conformation. The function of a protein is tightly connected to its conformation[50]. As such, cells have developed protein-quality control mechanisms, including molecular chaperones, which degrade misfolded or damaged proteins[51]. Prions are a special class of proteins that are capable of adopting multiple stable conformations, which not only fail to be removed by protein quality control mechanisms but which themselves can induce proteins in the normal configuration to change to the alternate (prion) conformation[52].

More specifically, the proteins in the prion conformation form aggregates. These aggregates then convert normally folded protein to the prion state by acting as a template. The newly misfolded protein monomer is then incorporated into the aggregate, thus increasing its size. Then, rather than be cleared by protein quality control mecha-

nisms, these aggregates can be split (fragmentation), increasing the number of aggregates and rate of further misfolding[52].

In mammals, prions have only been associated with progressive, untreatable, and fatal neurodegenerative disease[53,54]. Prions disease in mammals can be infectious between members of the same species (scrapie in sheep, kuru in humans), occur spontaneously in an individual (Creutzfeldt-Jakob disease and fatal familial insomnia), and can even jump between species as when humans acquire variant Creutzfeldt-Jakob disease from consuming meat from cattle with bovine spongiform encephalopathy (Mad Cow disease)[55]. Fortunately, in humans, prion diseases are rare. There are less than 400 reported cases of Creutzfeldt-Jakob disease per year, but approximately 70% of those affected die within a year of exposure[56]. Fatal familial insomnia, though rare, is typically fatal within only 18 months of initial symptoms[57].

While prion diseases are rare, the biochemical processes of prion disease are closely related to other protein-misfolding disorders such as Alzheimer's and Parkinson's. All these diseases are related by the common amyloid structure of their corresponding protein aggregates. As such, much of the research within the prion disease aims to characterize the dynamics of the amyloid protein aggregates, which will provide insights into classes of much more common disease[55,58]. Much of the research within the field aims to characterize the dynamics of protein aggregates that cause prion and amyloid diseases[55,58]. One powerful biological tool for studying prion and protein aggregation processes is the single-celled yeast *Saccharomyces cerevisiae*. However, because yeast cells are quite different from mammalian cells, mathematical models of protein aggregation in yeast need to include other processes from those models designed for mammals.

2.2.2. YEAST CELLS AND ASYMMETRIC DIVISION

The single-celled yeast *Saccharomyces cerevisiae* has emerged as a model eukaryote in biological research and, as such, is the subject of our study. Yeast biologists have a host of experimental manipulations at their disposal, making yeast an ideal system to study a host of biological processes, including protein misfolding and aging[59,52,60].

Yeast is an attractive system for studying protein misfolding because, unlike for mammals, protein misfolding in yeast is not fatal or harmful. Moreover, in yeast, also unlike mammals, protein misfolding processes can be turned on and off[61,62,63]. That is, protein misfolding can be reliably induced in healthy cells[64] and, for certain protein mutants, protein misfolding phenotypes can be destabilized (i.e., all of the mis-

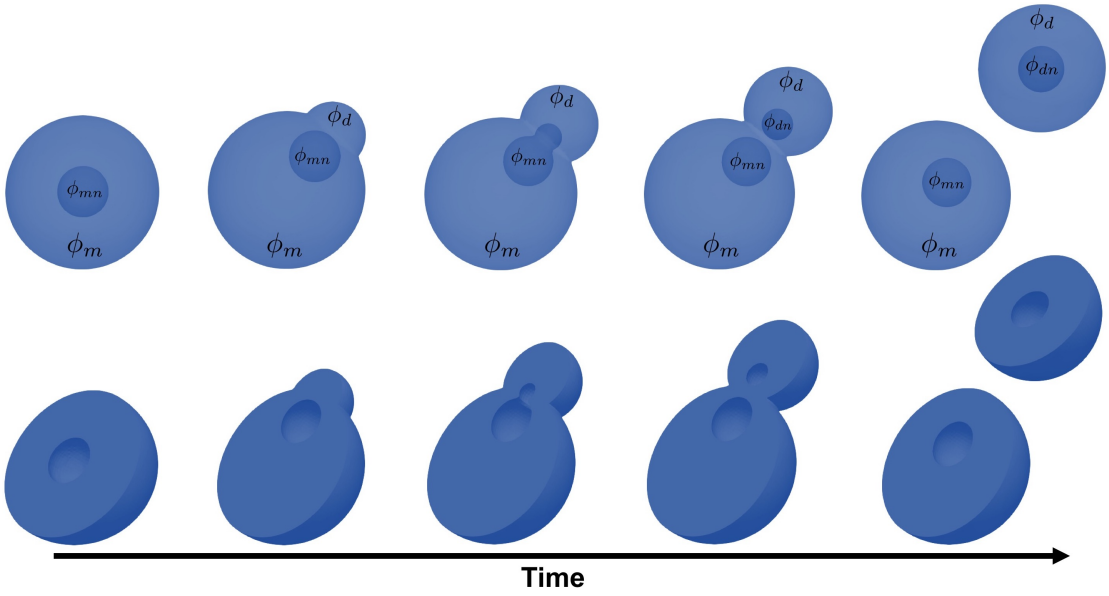


Figure 2.2: **Level-set based simulation of a budding yeast cell**. **Top row**: full three-dimensional simulation of a yeast cell (larger sphere ϕ_m) with its nucleus (the middle sphere ϕ_{mn}) at its center. As time progresses, the daughter cell, ϕ_d, and its nucleus,ϕ_{dn} starts budding off. **Bottom row**: half-space rendering of the yeast cell budding.

folded protein will return to normal.) There are even studies that indicate misfolded protein aggregates may have benefits for yeast[65].

The cell-division process in yeast makes them a valuable tool for studying aging. Unlike bacteria, which divide into two identical (or nearly identical) cells, yeast divide asymmetrically by budding into a mother cell and a daughter cell (see Fig. 2.2). In budding cell division, a new (daughter) cell is grown as an outgrowth of the old (mother) cell. At the time of separation, the mother cell is larger than the daughter cell. Finally, the daughter does not inherit the replicative age of the mother[66]. Although intracellular constituencies, including the nucleus, are transmitted from mother to daughter cell through a narrow bud neck connecting them, not all constituents transmit with equal efficiencies. Mother cells have been shown to retain a variety of "damaged" protein species preferentially, and this bias in retention can not be explained by differences in volume[67].

Intriguingly, prion aggregates are one such cellular constituency shown to not transmit efficiently between mother and daughter cells[61]. A variety of theories have emerged to explain this bias: (1) Misfolded proteins are larger and therefore less mo-

bile, especially in a crowded environment such as the intracellular matrix, and therefore have a harder progressing toward the daughter cell; (2) "Chaperone" proteins are present in the cell and forcibly restrict the damaged proteins to particular compartments, preventing them from reaching the daughter cell. The mathematical modeling framework we propose offers the capability to investigate these causes. However, we will focus on the passive assumptions alone for the present scope.

2.3. BIOLOGICAL MODELING

Our mathematical model is built as a two-species reaction-diffusion system defined over a deforming geometry, represented by a level set function. To study the influence of intracellular compartments, we will consider a succession of geometries with increasing complexity and discuss their construction.

2.3.1. GOVERNING EQUATIONS

We consider two protein species, A and B, living inside a deforming cell $\Omega(t)$ (see Fig. 2.2), and track their respective concentrations ψ_A and ψ_B. We assume that A is a monomer and can aggregate with itself to form the dimer B, which can degrade into two monomers. We define the dimerization and degradation rate as γ_{AB} and γ_{BA}, respectively. Additionally, both species are diffusing with the corresponding diffusivities D_A and D_B. The concentrations ψ_A and ψ_B are therefore solutions of the reaction-diffusion equations

$$\frac{\partial \psi_A}{\partial t} - D_A \triangle \psi_A = 2\gamma_{BA}\psi_B - \gamma_{AB}\psi_A^2 \quad \forall \mathbf{x} \in \Omega(t), \tag{2.1}$$

$$\frac{\partial \psi_B}{\partial t} - D_B \triangle \psi_B = \frac{1}{2}\gamma_{AB}\psi_A^2 - \gamma_{BA}\psi_B \quad \forall \mathbf{x} \in \Omega(t). \tag{2.2}$$

We assume the cell membrane to be hermetic and, therefore, enforce a no-flux boundary condition on the contour of the cell $\partial\Omega(t)$

$$\nabla \psi_A \cdot \mathbf{n} = 0 \quad \forall \mathbf{x} \in \partial\Omega(t), \tag{2.3}$$

$$\nabla \psi_B \cdot \mathbf{n} = 0 \quad \forall \mathbf{x} \in \partial\Omega(t), \tag{2.4}$$

where $\mathbf{n}$ denotes the normal vector to $\partial\Omega(t)$.

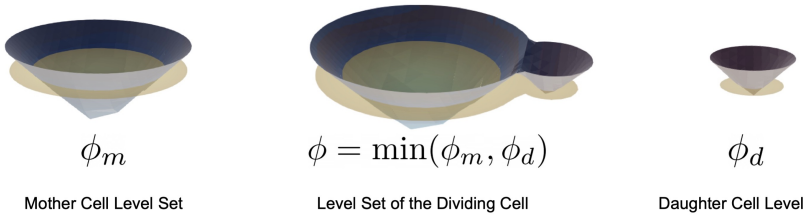


Figure 2.3: **level set representation of a dividing yeast cell in two dimensions**. The beige plane depicts the zero level set, and the three-dimensional surface illustrates the level set functions of mother and daughter cells (in two dimensions), denoted by ϕ_m and ϕ_d, respectively. The union of the two biological cells is reproduced by taking the minimum of their level-set function.

2.3.2. LEVEL SET REPRESENTATION

For the biological motivation, we construct $\Omega(t)$, the inside of the dividing yeast cell, by decomposing it into multiple subdomains (see Fig. 2.2). To each subdomain, we associate a level set function. Then, as illustrated in Fig. 2.3, we assemble the complete geometry by taking unions and intersections of these subdomains, equivalent to taking $\min$ and $\max$ of the corresponding level set functions.

This setup allows us to construct a succession of four geometrical representations of increasing complexity to evaluate the impact of each geometrical feature on the prion aggregation problem. From the initial representation containing the mother ϕ_m and daughter ϕ_d cells only (Fig. 2.4-(A)), we successively introduce the mother nucleus ϕ_{mn} (Fig. 2.4-(B)), the daughter nucleus ϕ_{dn} (Fig. 2.4-(C)), and finally two compartments found in yeast cells, the *Juxtanuclear Quality Control Compartment* (JUNQ) and *Insoluble Protein Deposit* (IPOD)[68] (Fig. 2.4-(D)), which have shown impact the budding and aggregation process[68]. We will use the convention that the level set function is negative inside the domain of interest.

A No Nuclei

In this first case, we represent the dividing mother cell as the stationary domain Ω_m and its detaching daughter cell $\Omega_d(t)$ as two spheres represented by the following level set functions

$$\phi_m(\mathbf{x},t) = |\mathbf{x}-\mathbf{x}_m|-r_m, \tag{2.5}$$
$$\phi_d(\mathbf{x},t) = |\mathbf{x}-\mathbf{x}_d(t)|-r_d(t), \tag{2.6}$$

where $\mathbf{x}_m$ and $\mathbf{x}_d(t)$ are the center of the mother and daughter cell and r_m and $r_d(t)$

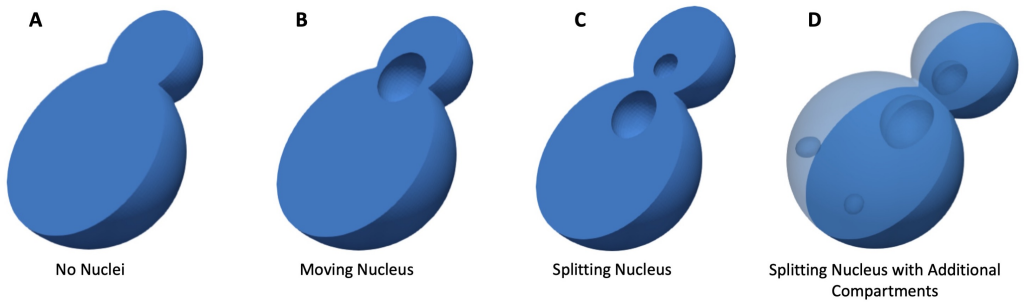


Figure 2.4: **Succession of geometric representations**. We start by modeling the cell budding of the mother cell without any nucleus **(A)**. Then, we introduce a moving nucleus as shown in **(B)**, allow the nucleus to split **(C)**, and finally consider additional moving compartments in the mother cell **(D)**. We describe the specifics of the geometry for our problem in Section 2.3.2.

are the respective radii. The entire dividing cell is defined as the union of these two domains and is represented by the level set function $\phi = \min(\phi_m, \phi_d)$, as Fig. 2.3 illustrates.

B Moving Nucleus

Similarly we define the center $\mathbf{x}_{mn}(t)$ of the nucleus of the mother cell, its radius $r_{mn}(t)$ and corresponding level set function

$$\phi_{mn}(\mathbf{x},t) \quad = |\mathbf{x} - \mathbf{x}_{mn}(t)| - r_{mn}(t). \tag{2.7}$$

The upgraded domain $\Omega(t)$ over which the governing equations are valid is now the intersection of the previous domain with the outside of the mother nucleus

$$\Omega(t) = (\Omega_m \cup \Omega_d(t)) \cap \Omega_{mn}^c(t), \tag{2.8}$$

or equivalently in terms of the level set functions

$$\phi = \max(\min(\phi_m, \phi_d), -\phi_{mn}). \tag{2.9}$$

C Splitting Nucleus

Following case (i), we define a new level set function for the daughter's nucleus, which we model detaching from the mother cell's nucleus, as observed in nature.

The corresponding level set function is

$$\phi_{dn}(\mathbf{x},t) = |\mathbf{x}-\mathbf{x}_{dn}(t)|-r_{mn}(t)-r_{dn}(t), \tag{2.10}$$

leading to

$$\Omega(t) = (\Omega_m \cup \Omega_d(t)) \cap (\Omega_{mn}(t) \cup \Omega_{dn}(t))^c \quad \Longleftrightarrow \quad \max(\min(\phi_m,\phi_d),-\min(\phi_{mn},\phi_{dn})) \tag{2.11}$$

D Additional Cellular Compartments (IPOD and JUNQ)

Finally, for our most comprehensive model, we include two cellular compartments representing Insoluble Protein Deposit (IPOD) and Juxta Nuclear Quality (JUNQ). We model these compartments as moving and rotating ellipsoids of centers $\mathbf{x}_I(t), \mathbf{x}_J(t)$ and parameters $\mathbf{p}_I(t), \mathbf{p}_J(t)$

$$\phi_I(\mathbf{x},t) = \left(\left(\frac{(x-x_I(t))}{p_I^x(t)}\right)^2 + \left(\frac{(y-y_I(t))}{p_I^y(t)}\right)^2 + \left(\frac{(z-z_I(t))}{p_I^z(t)}\right)^2\right)^{\frac{1}{2}} - 1, \tag{2.12}$$

$$\phi_J(\mathbf{x},t) = \left(\left(\frac{(x-x_J(t))}{p_J^x(t)}\right)^2 + \left(\frac{(y-y_J(t))}{p_J^y(t)}\right)^2 + \left(\frac{(z-z_J(t))}{p_J^z(t)}\right)^2\right)^{\frac{1}{2}} - 1. \tag{2.13}$$

The final geometry and level set function are

$$\Omega(t) = (\Omega_m \cup \Omega_d) \cap (\Omega_{mn} \cup \Omega_{dn})^c \cap \Omega_I^c \cap \Omega_J^c \qquad \forall \mathbf{x} \in \mathbb{R}^3, \forall t \geq 0, \tag{2.14}$$

$$\phi = \max(\min(\phi_m,\phi_d),-\min(\phi_{mn},\phi_{dn}),-\phi_I,-\phi_J) \tag{2.15}$$

In our implementation, we use the above formulas to initialize at each iteration the level set function, which we then reinitialize by reaching the steady state of the following equation

$$\frac{\partial \phi}{\partial \tau} + \text{sign}(\phi_0)\,(|\nabla \phi|-1) = 0, \tag{2.16}$$

defined in fictitious time τ and over the entire computational domain. This is achieved using the second-order Total Variation Diminishing algorithm detailed in[69].

2.4. NUMERICAL METHODS

This section presents our novel finite volume method and compares it to the traditional finite volume method, focusing, in particular, on mass conservation. While we

will limit our analysis to the reaction-diffusion system described in the above section, we should point out that it can easily be extended to other systems of partial differential equations. The last part of this section focuses on using and constructing adaptive non-graded octree grids.

2.4.1. FINITE VOLUME APPROACH

The numerical solution of the system (2.1)-(2.2)-(2.3)-(2.4) is constructed using a semi-implicit finite volume method[70], which final form is given by Eqs. (2.33) and (2.34). For stability, the diffusive effects are treated implicitly while the non-linear reactive terms are treated explicitly. All diffusive fluxes are approximated using the second-order antisymmetric discretization proposed by Lossaso *et al.*[71] and used in our previous studies[72,73,74,38,75].

NOTATIONS, APPROXIMATIONS, AND REMARKS

All quantities are stored at the centers of the octree grids. The control volumes V_i^n are defined as the intersection of the i^{th} computational cell C_i with the biological complex Ω^n at time step t^n (see Fig. 2.5)

$$V_i^n = C_i \cap \Omega^{n+1}. \tag{2.17}$$

The integrals over these volumes for any cell-based quantity q are approximated using the approximations

$$\int_{V_i^n} f = f_i |V_i^n| + \mathscr{O}(\Delta x^3). \tag{2.18}$$

The contour integrals of the normal fluxes $\nabla q \cdot \mathbf{n}$ over ∂V_i^n are decomposed as

$$\int_{\partial V_i^n} \nabla q \cdot \mathbf{n} = \sum_{f \in \text{faces}(C_i)} \int_{f \cap \Omega^n} \nabla q \cdot \mathbf{n} + \int_{\partial \Omega^n \cap C_i} \nabla q \cdot \mathbf{n}, \tag{2.19}$$

and since the boundary conditions (2.3)-(2.4) are homogeneous, all interfacial fluxes are null, and we approximate the above as

$$\int_{\partial V_i^n} \nabla q \cdot \mathbf{n} = \sum_{f \in \text{faces}(C_i)} \nabla q \cdot \mathbf{n}|_f |f \cap \Omega^n| + \mathscr{O}(\Delta x^3). \tag{2.20}$$

The volumes $|V_i^n|$ and the faces fractions $|f \cap \Omega^n|$ are calculated using the second-order quadrature rules presented in[76]. The resulting linear systems are solved using a Multigrid[77] preconditioned Bi-Conjugate Gradient Stabilized solver. For the model presented in section 2.3, the total mass of protein at any given time t is

$$M_T(t) = M_A(t) + 2M_B(t) = \int_{\Omega(t)} \psi_A + 2\int_{\Omega(t)} \psi_B = \sum_{i=1}^{M} \int_{V_i^n} \psi_A^n + 2\sum_{i=1}^{M} \int_{V_i^n} \psi_B^n, \quad (2.21)$$

where $M_A(t)$ and $M_B(t)$ are the total mass of species A and B respectively. The factor 2 in front of the second integral accounts for species B being the dimer composed of two monomers and, therefore, twice as heavy. While this conservation property is easily achieved on fixed geometry, additional care is required to ensure it holds in the deforming case, as we will see next.

In practice, the dynamic mesh refinement imposes to interpolate the solution between consecutive grids. To preserve the overall accuracy, we use a third-order Least Square interpolation, which is non-conservative. This interpolating step can generate local spurious converging mass variations. However, as our results illustrate, these variations are converging and, in practice, reasonably small.

Another key feature of our proposed finite volume formulation is that it does not require extending the solution across the interface as it is often needed with interfacial problems (see[72,38]). Instead, our formulation only involves the solution where it is formally defined (*i.e.* inside $\Omega(t)$).

TRADITIONAL FORMULATION

Here, we consider a common way of applying the finite volume method to a time-dependent problem, which we will refer to as the *traditional formulation*. In this formulation, we start by discretizing the conservation equations (2.1) and (2.2) in time using the semi-implicit scheme

$$\frac{\psi_A^{n+1} - \psi_A^n}{\Delta t} - D_A \triangle \psi_A^{n+1} = 2\gamma_{BA}\psi_B^n - \gamma_{AB}\left(\psi_A^n\right)^2, \quad (2.22)$$

$$\frac{\psi_B^{n+1} - \psi_B^n}{\Delta t} - D_B \triangle \psi_B^{n+1} = \frac{1}{2}\gamma_{AB}(\psi_A^n)^2 - \gamma_{BA}\psi_B^n, \quad (2.23)$$

where the upper-scripts n and $n+1$ indicate the semi-discrete quantities being evaluated at $(\mathbf{x}, t^n)$ and $(\mathbf{x}, t^{n+1})$ respectively. We construct the finite volume formulation by

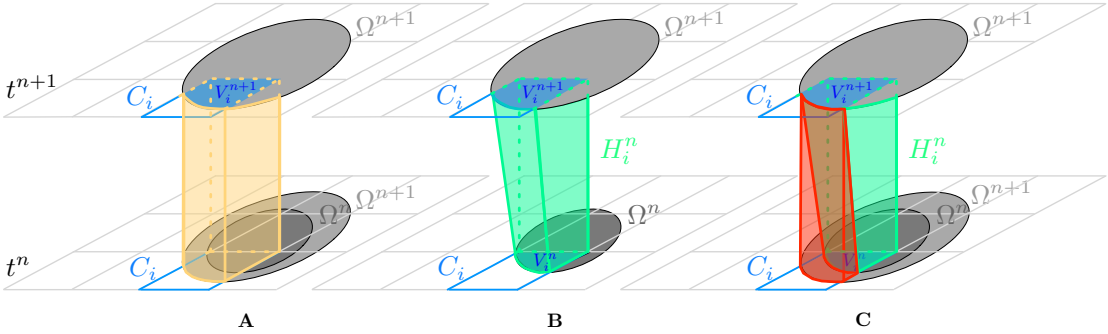


Figure 2.5: **Comparison between the traditional and conservative finite volume formulations**. (**A**) With the traditional approach, the equation is first discretized in time and then spatially integrated over each control volume $V_i^{n+1} = C_i \cap \Omega^{n+1}$ (depicted in blue) at the future time t^{n+1}. In the spatio-temporal domain, this is an integral over the orange hypervolume. (**B**) With our proposed formulation, the partial differential equation is directly integrated over the hypervolume $H_i^n = V_i(t) \times [t^n, t^{n+1}]$, leading to mass conservation. (**C**) The local mass loss with the traditional method directly relates to the difference between the two hypervolumes (in red).

integrating the above Eqs. over the control volumes $V_i^n = C_i \cap \Omega^{n+1}$ (see Fig. 2.5). For each cell $C_{i=1,\cdots,M}$, we obtain

$$\begin{aligned}
\int_{V_i^{n+1}} \psi_A^{n+1} - D_A \Delta t \int_{V_i^{n+1}} \triangle \psi_A^{n+1} &= \int_{V_i^{n+1}} \left(\psi_A^n + \Delta t \left(2\gamma_{BA} \psi_B^n - \gamma_{AB} \psi_A^{n2} \right) \right), \\
\int_{V_i^{n+1}} \psi_B^{n+1} - D_B \Delta t \int_{V_i^{n+1}} \triangle \psi_B^{n+1} &= \int_{V_i^{n+1}} \left(\psi_B^n + \Delta t \left(\frac{1}{2} \gamma_{AB} \psi_A^{n2} - \gamma_{BA} \psi_B^n \right) \right),
\end{aligned} \tag{2.24}$$

which, using Gauss' theorem, we rewrite as

$$\begin{aligned}
\int_{V_i^{n+1}} \psi_A^{n+1} - D_A \Delta t \int_{\partial V_i^{n+1}} \nabla \psi_A^{n+1} \cdot \mathbf{n} &= \int_{V_i^{n+1}} \left(\psi_A^n + \Delta t \left(2\gamma_{BA} \psi_B^n - \gamma_{AB} \psi_A^{n2} \right) \right), \\
\int_{V_i^{n+1}} \psi_B^{n+1} - D_B \Delta t \int_{\partial V_i^{n+1}} \nabla \psi_B^{n+1} \cdot \mathbf{n} &= \int_{V_i^{n+1}} \left(\psi_B^n + \Delta t \left(\frac{1}{2} \gamma_{AB} \psi_A^{n2} - \gamma_{BA} \psi_B^n \right) \right).
\end{aligned} \tag{2.25}$$

To quantify the total mass (shown in (2.21)) we start by summing Eq. (2.25) over all grid cells,

$$\sum_{i=1}^{M}\left(\int_{V_i^{n+1}}\psi_A^{n+1}-D_A\Delta t\int_{\partial V_i^{n+1}}\nabla\psi_A^{n+1}\cdot\mathbf{n}\right) \;=\; \sum_{i=1}^{M}\int_{V_i^{n+1}}\left(\psi_A^n+\Delta t\left(2\gamma_{BA}\psi_B^n-\gamma_{AB}\psi_A^{n2}\right)\right). \tag{2.26}$$

Because the diffusive fluxes are antisymmetric and the boundary condition (2.3) is homogeneous, the contour integrals vanish, leaving us with

$$\sum_{i=1}^{M}\int_{V_i^{n+1}}\psi_A^{n+1} \;=\; \sum_{i=1}^{M}\int_{V_i^{n+1}}\left(\psi_A^n+\Delta t\left(2\gamma_{BA}\psi_B^n-\gamma_{AB}\psi_A^{n2}\right)\right), \tag{2.27}$$

which in terms of the total mass M_A of the monomer A reads

$$M_A^{n+1} \;=\; \sum_{i=1}^{M}\int_{V_i^{n+1}}\psi_A^n+\Delta t\sum_{i=1}^{M}\int_{V_i^{n+1}}\left(2\gamma_{BA}\psi_B^n-\gamma_{AB}\psi_A^{n2}\right). \tag{2.28}$$

Similarly, we obtain for the second specie

$$M_B^{n+1} \;=\; \sum_{i=1}^{M}\int_{V_i^{n+1}}\psi_B^n+\Delta t\sum_{i=1}^{M}\int_{V_i^{n+1}}\left(\frac{1}{2}\gamma_{AB}\psi_A^n-\gamma_{BA}\psi_B^{n2}\right). \tag{2.29}$$

Multiplying Eq. (2.29) by two and adding the result to Eq. (2.28), the reactive terms cancel and it results

$$M_T^{n+1}=M_A^{n+1}+2M_B^{n+1} \;=\; \sum_{i=1}^{M}\int_{V_i^{n+1}}(\psi_A^n+2\psi_B^n). \tag{2.30}$$

Because the volumes over which the integral is performed are evaluated at time step t^{n+1}. In contrast, the integrands are evaluated at t^n; the right-hand side is not the total mass at t^n and therefore, with this semi-discrete formulation, $M_T^{n+1}\neq M_T^n$. Another issue resulting from this asynchronicity is that the quantities ψ_A^n,ψ_B^n may not be formally defined over the control volumes V_i^{n+1}. Therefore, they may have to be extended over the interface $\partial\Omega^n$ before being integrated in Eq. (2.25). Such extensions are often expensive as they involve costly geometric reconstruction [78,72] or solving non-linear propagation equations [79,80], and may not preserve the stability of the method.

CONSERVATIVE FORMULATION

Inspired by the work of Gaburro *et al.*[81], we construct the conservative formulation by integrating Eqs. (2.1) and (2.2) over the space-time hyper volume $H_i^n = V_i(t) \times [t^n, t^{n+1}]$ (see Fig. 2.5). Focusing on the monomer A, we obtain

$$\int_{H_i^n} \left(\frac{\partial \psi_A}{\partial t} - D_A \triangle \psi_A \right) \quad = \quad \int_{H_i^n} \left(2\gamma_{BA} \psi_B^n - \gamma_{AB} \left(\psi_A^n\right)^2 \right), \tag{2.31}$$

which in virtue of Gauss's theorem, can be rewritten as

$$\int_{V_i^{n+1}} \psi_A^{n+1} - \int_{V_i^n} \psi_A^n - D_A \int_{t^n}^{t^{n+1}} \int_{\partial V_i(t)} \nabla \psi_A \cdot \mathbf{n} = \int_{H_i^n} \left(2\gamma_{BA} \psi_B - \gamma_{AB} \left(\psi_A\right)^2 \right). \tag{2.32}$$

Integrating the diffusive terms implicitly and the reactive ones explicitly, the semi-discrete formulation becomes

$$\int_{V_i^{n+1}} \psi_A^{n+1} - \int_{V_i^n} \psi_A^n - D_A \Delta t \int_{\partial V_i(t)} \nabla \psi_A \cdot \mathbf{n} = \Delta t \int_{V_i^n} \left(2\gamma_{BA} \psi_B - \gamma_{AB} \left(\psi_A\right)^2 \right), \tag{2.33}$$

and similarly

$$\int_{V_i^{n+1}} \psi_B^{n+1} - \int_{V_i^n} \psi_B^n - D_B \Delta t \int_{\partial V_i(t)} \nabla \psi_B \cdot \mathbf{n} = \Delta t \int_{V_i^n} \left(\frac{1}{2} \gamma_{AB} \psi_A^n - \gamma_{BA} \psi_B^{n2} \right). \tag{2.34}$$

To prove that this formulation is conservative, we again sum the formulation overall computational cells, use the anti-symmetry of the fluxes and the homogeneity of the boundary condition (2.3) to conclude that

$$M_A^{n+1} \quad = \quad \sum_{i=1}^{M} \int_{V_i^n} \psi_A^n + \Delta t \sum_{i=1}^{M} \int_{V_i^n} \left(2\gamma_{BA} \psi_B^n - \gamma_{AB} \psi_A^{n2} \right). \tag{2.35}$$

The only difference between the above equation and (2.28) is that the first integral in the right-hand side is now evaluated on the geometry at t^n. It implies that the sum is indeed the total mass of A at t^n

$$M_A^{n+1} \quad = \quad M_A^n + \Delta t \sum_{i=1}^{M} \int_{V_i^n} \left(2\gamma_{BA} \psi_B^n - \gamma_{AB} \psi_A^{n2} \right), \tag{2.36}$$

and similarly

$$M_B^{n+1} = M_B^n + \Delta t \sum_{i=1}^{M} \int_{V_i^{n+1}} \left(\frac{1}{2} \gamma_{AB} \psi_A^n - \gamma_{BA} {\psi_B^n}^2 \right), \tag{2.37}$$

leading to the total conservation

$$M_A^{n+1} + 2M_B^{n+1} = M_A^n + 2M_B^n \quad \Longleftrightarrow \quad M_T^{n+1} = M_T^n. \tag{2.38}$$

We conclude that this formulation is mass-conservative.

CONVERGENCE LIMITATIONS OF THE TRADITIONAL FORMULATION

As we exposed, the traditional finite volume method fails to preserve mass. We find here an upper bound for its accuracy. From this result, we demonstrate that the total mass loss, and by extension, the concentration, is not guaranteed to converge. We conduct this analysis on a uniform grid for readability and denote the spatial resolution by Δx.

To quantify the total mass variation Δ_M^n between two consecutive time steps

$$\Delta_M^n = M_T^{n+1} - M_T^n = \sum_{i=1}^{M} \left(\int_{V_i^{n+1}} \psi_A^n + 2\psi_B^n - \int_{V_i^n} \psi_A^n + 2\psi_B^n \right), \tag{2.39}$$

we start by reformulating Eq. (2.30) as

$$M_T^{n+1} - \sum_{i=1}^{M} \int_{V_i^{n+1}} (\psi_A^n + 2\psi_B^n) = 0, \tag{2.40}$$

and decomposing each integrals over the domains V_i^n and $V_i^{n+1} \setminus V_i^n = \Lambda_i^n$ (visualized in Fig. 2.5-(C) as the red shaded region).

$$M_T^{n+1} - \sum_{i=1}^{M} \int_{V_i^n} (\psi_A^n + 2\psi_B^n) = \sum_{i=1}^{M} \int_{\Lambda_i^n} (\psi_A^n + 2\psi_B^n). \tag{2.41}$$

The left-hand side term is exactly Δ_M^n, and so after taking the absolute value, we ob-

tain

$$|\Delta_M^n| \leq \sum_{i=1}^{M} \int_{\Lambda_i^n} |\psi_A^n + 2\psi_B^n|, \tag{2.42}$$

Because only for the computational cells that are crossed by the interface during the interval $[t^n, t^{n+1}]$ the integral over Λ_i^n is non-zero, we can further simplify the above inequality and write that

$$|\Delta_M^n| \leq M_\Gamma^n \quad \max_i |\Lambda_i^n| \quad ||\psi_A^n + 2\psi_B^n||_\infty, \tag{2.43}$$

where M_Γ^n is the total number of cells crossed by the interface during $[t^n, t^{n+1}]$. Provided that the time step is small enough, this total number scales as $\frac{1}{\Delta x^{d-1}}$, d is the spatial dimension. The size of the volume variation $|\Lambda_i^n|$ is the order of the local interface displacement multiplied by the area of the surface contained in V_i^n, therefore $|\Lambda_i^{n+1}| = \mathscr{O}(\Delta x^{d-1}\Delta t)$. Assuming that the concentration field remains bounded, we obtain the following approximation for the local mass loss

$$\Delta_M^n = \mathscr{O}(\Delta t), \tag{2.44}$$

telling us that the global mass loss is $\mathscr{O}(1)$. This proves that the mass losses are non-diverging, but we cannot conclude whether they converge. However, we can construct a simple example * where the upper bound in Eq. (2.42) is reached, proving that in general the total mass is not converging and therefore that concentration is also not converging in L^∞ -norm.

2.4.2. ADAPTIVE MESH REFINEMENT

The computational domain is represented as a non-graded octree grid[82]. At each iteration, the mesh generation starts with the root cell representing the entire domain (corresponding to level 0), which we subdivide into eight identical cells of level 1 (see Fig. 2.6). We then recursively divide each newly created cell C if either

$$\min_{v \in \text{nodes}(C)} |\phi(v)| \leq \text{Lip}(\phi) \cdot D(C) \qquad \text{and} \qquad \text{level}(C) < \max_{\text{level}}, \tag{2.45}$$

*Consider, for example, a flat interface moving at a constant speed V, with uniform concentration and fields, and a fixed time step $\Delta_t = \frac{\Delta x}{V}$. In this example, there are exactly $\frac{1}{\triangle x}$ many Λ_i^{n+1}, which are all identical and of size Δx^2, and thus the upper bound in Eq. (2.42) is reached.

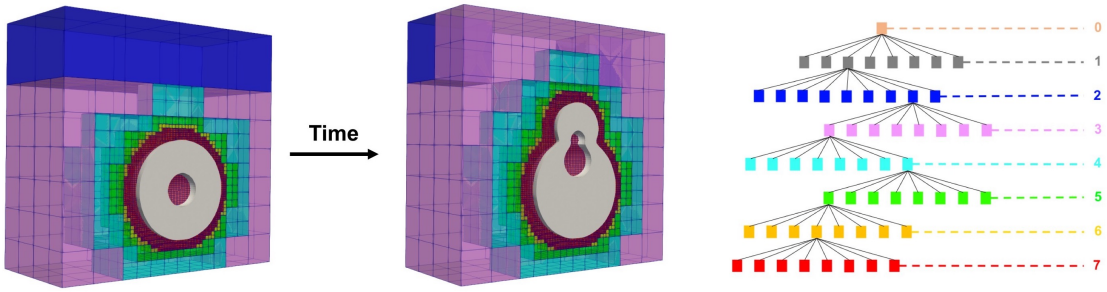


Figure 2.6: **Adaptive grid generation and representation**. The level of a computational cell is defined as the number of successive subdivisions to create it. In this example, the maximum level is 7 (red cells), and the minimum level is 2 (blue cells). The automatic refinement is done according to criterion (2.45)-(2.46).

or

$$\text{level}(C) \quad < \quad \min_{\text{level}}, \tag{2.46}$$

where $\text{Lip}(\phi)$ is an estimation of the minimal Lipschitz constant for the level set function $\phi(\mathbf{x})$, set to 1.2 in practice, $D(C)$ is the length of the diagonal C, $\min_{\text{level}}$ and $\max_{\text{level}}$ are the prescribed minimum and maximum tree level. As done in a previous study [73], we store the concentrations at the cell centers and the level set values at the nodes. Because the interface is evolving between iterations, the grid is, too, and the solution must be interpolated between grids. This is achieved using third-order Least Square regression as done in previous studies [72,73,38].

2.5. NUMERICAL VALIDATIONS

In this section, we compare our conservative formulation to the traditional method in two and three spatial dimensions. We first consider a simpler test problem for which we construct an analytic solution, allowing us to investigate the convergence of the solution. We then return to our motivating application and focus on total mass conservation. In both cases, we find that our method converges with second-order accuracy in space while the traditional approach rapidly stalls.

2.5.1. TEST PROBLEM: EXPANDING SPHERE

To study the spatio-temporal convergence of the two formulations, we consider a one-species diffusion system with an exact solution on an expanding and translating

sphere. We define this spherical domain by the level set function

$$\phi_{exact}(t) = |\mathbf{x} - \mathbf{x}_0(t)| - r(t), \tag{2.47}$$

where $r(t) = 0.25t$ is the expanding radius and $\mathbf{x}_0(t) = (0,t,0)$ is the translating center. In this domain, we consider the following test problem for a single concentration field $\psi(\mathbf{x},t)$

$$\begin{aligned} \frac{\partial \psi}{\partial t} - \triangle \psi &= f(\mathbf{x},t), \quad \forall \mathbf{x} \in \Omega(t), & (2.48)\\ \nabla \psi \cdot \mathbf{n} &= g(\mathbf{x},t), \quad \forall \mathbf{x} \in \partial\Omega(t). & (2.49) \end{aligned}$$

We use for the exact solution $\psi_{exact}(\mathbf{x},t)$, forcing term $f(\mathbf{x},t)$ and boundary flux $g(\mathbf{x},t)$ the functions

$$\begin{aligned} \textbf{2D} \quad & \psi_{Exact}(\mathbf{x},t) = e^{x+y+t}, \quad f(\mathbf{x},t) = -e^{x+y+t}, \quad g(\mathbf{x},t) = e^{x+y+t}, \\ \textbf{3D} \quad & \psi_{Exact}(\mathbf{x},t) = e^{x+y+z+t}, \quad f(\mathbf{x},t) = -2e^{x+y+z+t}, \quad g(\mathbf{x},t) = e^{x+y+z+t}, \end{aligned} \tag{2.50}$$

and run the simulation for $t \in [0,5]$. Because we expect our new formulation to be first-order in time and second-order in space - *i.e.* $\mathscr{O}(\Delta t + \Delta x^2)$ -, we let $\Delta t = 1000\Delta x^2$, so that the measured numerical error is $\mathscr{O}(\Delta x^2)$. To decrease the resolution, we increase the grid's minimum and maximum levels. By doing so, we ensure that the spatial resolution diminishes everywhere. Increasing the maximum level would only refine the grid close to the interface.

Fig. 2.7 depicts the time evolution of the L^∞- error for increasing grid resolutions. In two dimensions, our method is converging at an apparent constant rate, while the traditional finite volume formulation appears to stall after two refinements ($\max_{\text{level}} = 7$). Our conservative formulation results in a solution orders of magnitude more accurate. The same observations apply to the three-dimensional results, although the differences between the two methods are less striking as the analysis is limited to a $\max_{level} = 10$.

The corresponding order of convergence for the global L^∞- errors are reported in Table 2.1. The conservative formulation is second-order accurate in space and at least first-order in time in two and three dimensions. With the non-conservative method, the order of convergence decreases as the resolution increases. This accuracy drop is flagrant in two dimensions, where the order rapidly falls around 0.5. In comparison, the three-dimensional order only drops to 1.18.

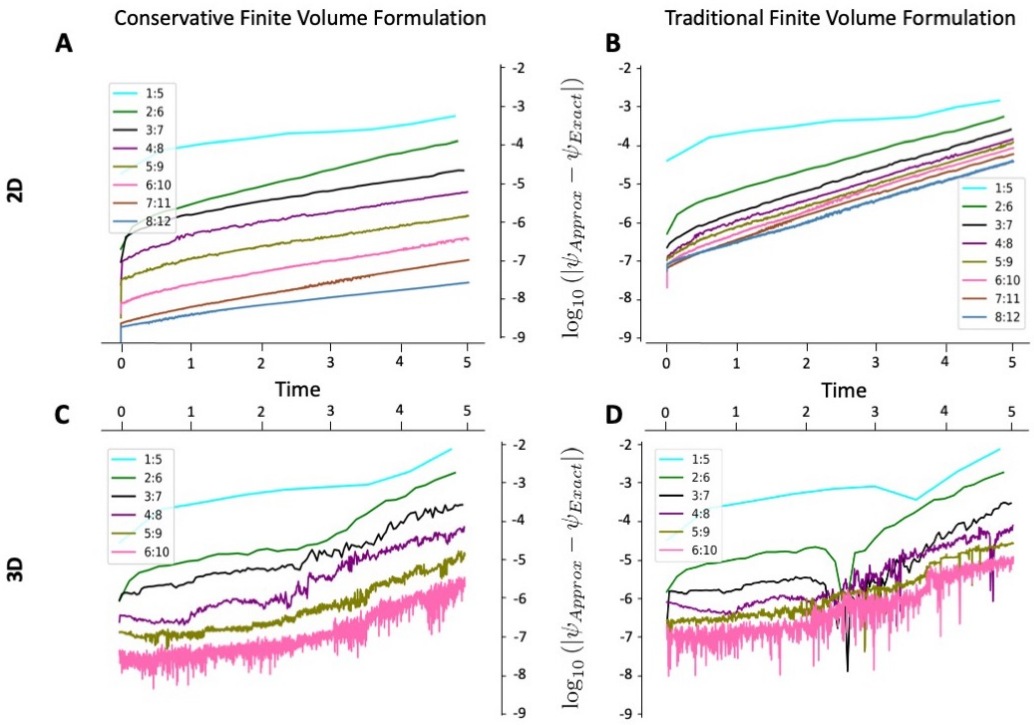


Figure 2.7: **Expanding sphere simulation reults**. Convergence of our conservative (left column) and the traditional (right column) formulations for the expanding sphere test 2.5.1, in two (**A-B**), and three spatial dimensions (**C-D**).

2.5.2. PRACTICAL MASS CONSERVATION

To quantify the mass conservation of both methods, we go back to the original biological motivation described by the system in Eq. (2.1)-(2.2) and consider the three-dimensional Splitting Nucleus geometry with the geometric parameters listed in 2.3. Simulations are carried until the final time $T = 90$ min, at which point the daughter cell has fully detached from the mother cell. Following the biologically relevant ranges (Table 2.3), we set the diffusion coefficients to be $D_A = 10^3\ \mu\text{m}^2 \cdot \text{min}^{-1}$ and $D_B = 1\ \mu\text{m}^2 \cdot \text{min}^{-1}$, and the reaction rates between $\gamma_{AB} = 10^{-2}\ \mu\text{m}^3 \cdot \text{min}^{-1}$ and $\gamma_{BA} = 10^{-3}\ \text{min}^{-1}$, which are biologically relevant. We define the relative total mass variations at any given discrete time t^n in terms of the initial and current total mass

$$e_M(t^n) = \left| \frac{M_T^n - M_T^0}{M_T^0} \right|, \tag{2.51}$$

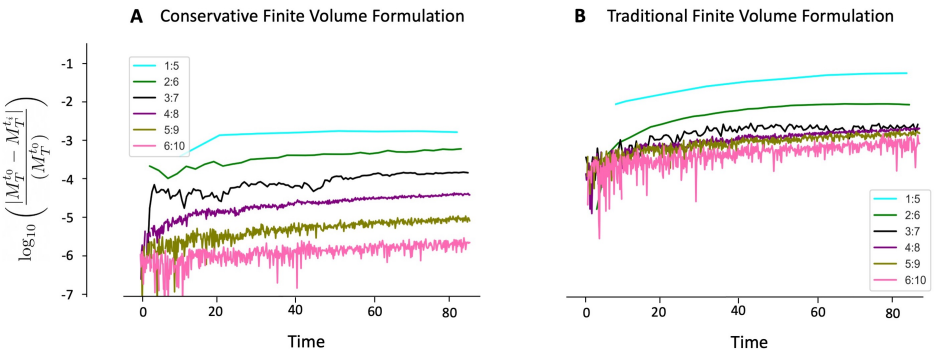


Figure 2.8: **Semi-log plot of relative mass loss**. Here we present the semi-log relative mass loss plots for the conservative (**A**) and traditional FV (**B**). The curves are labeled by their minimum and maximum grid level. We observe that the traditional method rapidly stalls while the conservative one provides converging mass loss over the considered range of grid resolution.

Table 2.2: Mass loss for the biological application, using either our methods. Results are indexed by the minimum and maximum grid levels.

	3D			
	Conservative FV		**Traditional FV**	
Levels	Mass Loss	Order	Mass Loss	Order
1:5	2.56×10^{-3}	-	6.06×10^{-2}	-
2:6	8.74×10^{-4}	1.55	9.66×10^{-3}	2.65
3:7	2.13×10^{-4}	2.03	3.02×10^{-3}	1.68
4:8	5.68×10^{-5}	1.91	2.31×10^{-3}	0.38
5:9	1.46×10^{-5}	1.95	1.98×10^{-3}	0.22
6:10	3.86×10^{-6}	1.92	1.64×10^{-3}	0.28

Table 2.1: Convergence of the global $L^\infty-$error with the proposed and traditional formulations, in two and three spatial dimensions. Results are indexed by the minimum and maximum grid levels.

	2D			
	Conservative FV		**Traditional FV**	
levels	error	order	error	order
1:5	9.76×10^{-4}	-	1.12×10^{-3}	-
2:6	2.15×10^{-4}	2.18	4.36×10^{-4}	1.37
3:7	3.74×10^{-5}	2.52	2.03×10^{-4}	1.09
4:8	1.02×10^{-5}	1.87	1.14×10^{-4}	0.83
5:9	2.46×10^{-6}	2.06	9.74×10^{-5}	0.23
6:10	6.52×10^{-7}	1.92	6.86×10^{-5}	0.51
7:11	1.71×10^{-7}	1.93	4.78×10^{-5}	0.52
8:12	4.41×10^{-8}	1.96	3.21×10^{-5}	0.57

	3D			
	Conservative FV		**Traditional FV**	
levels	error	order	error	order
1:5	6.83×10^{-3}	-	6.83×10^{-3}	-
2:6	1.69×10^{-3}	2.01	1.69×10^{-3}	2.01
3:7	2.56×10^{-4}	1.96	2.72×10^{-4}	2.64
4:8	6.86×10^{-5}	2.07	6.85×10^{-5}	1.99
5:9	1.64×10^{-5}	2.06	2.43×10^{-5}	1.50
6:10	4.24×10^{-6}	1.95	1.07×10^{-5}	1.18

where the current total mass is calculated as

$$M_T^n = \int_{\Omega^n} \psi_A^{n+1} + 2\psi_B^{n+1}. \tag{2.52}$$

The mass loss as a function of time and for increasing grid resolution, using either method, is depicted in Fig. 2.8, and the estimated orders of convergence are reported in Table 2.2. As for the previous error analysis, our formulation converges at an apparent second-order rate, while the traditional method stalls after a couple of refinements. Again, the error differences between the two methods are striking: they reach three orders of magnitudes on the finest grid ($\max_{\text{level}} = 10$). The mass loss with our method

on the coarsest grid is comparable to the one obtained with the traditional on the finest grid.

For the rest of our biological study, we will set the minimum and maximum grid levels to 5 and 8, respectively. In light of the measurement reported in Fig. 2.8, we are confident that the typical mass loss will be well below 0.1%.

2.6. SIMULATED PRIONS DYNAMICS IN DIVIDING YEAST CELLS

2.6.1. SIMULATION PARAMETERS

We provide all parameter values in Table 2.3 and summarize how they were obtained here. Bryne *et al.*[83] found the cell reproduction time to be 1.46 hours on average (for the strains YJW512 [PSI +]); based on this, we took the cell division time (*i.e.* the final time) to be $T = 90$ min. Next, we chose $37\mu\text{m}^3$ as the volume of the mother cell at the initial time according to Tyson *et al.*[84]; from the same reference, we have that the volume of the daughter cell must be two-thirds of the volume of the mother cell when budding is complete. For the nucleus size, we use the estimation provided in[85], which reasonably agrees with the experimental observations of Wang *et al.*[86]. We model the additional compartments in the mother cell to resemble the IPOD and JUNQ compartments found in yeast cells, which are known to affect protein aggregation. Although the size and shape of these compartments are known to vary, we chose to represent these compartments as ellipsoids for simplicity. We selected their characteristic lengths to be smaller than the nucleus, based on microscopic observations found in[87].

We estimate the characteristic prion concentration ψ_0 from the value reported on the Saccharomyces Genome Database[88]. For the typical rates of fragmentation and aggregation, γ_{BA} and γ_{AB} respectively, we chose values in agreement with our previous study[89]. We note that these findings were obtained in a different context, and so the typical rates (reported in table 2.3) will only be used as guiding information during our computational exploration. In all our simulations, we use constant initial concentration fields with random spatial noise. These initial profiles are re-scaled so that the initial total mass is identical across all examples. For the rest of this chapter, we will use the minute and micrometer as our characteristic time and length scales and non-dimensionalize all concentrations by the characteristic concentration ψ_0. From now on, all quantities will be reported in dimensionless form.

In this last section, we return to the original biological motivation, the simulation of prion dynamics in dividing yeast cells, and characterize the impact of the biochemical properties and geometric features on the protein distributions. We conduct this

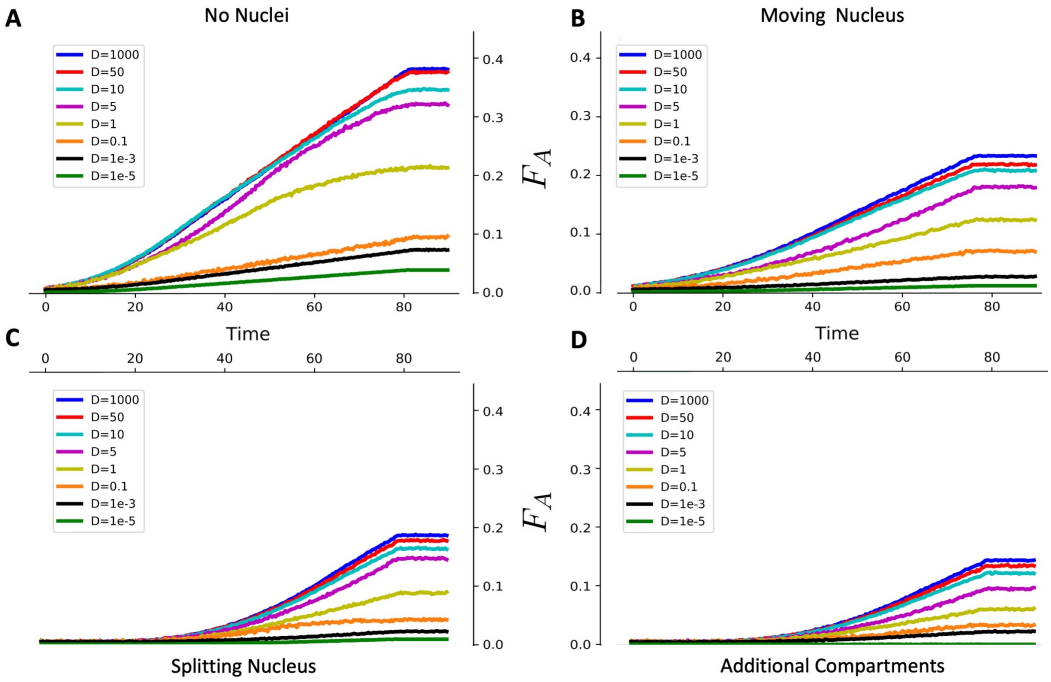


Figure 2.9: **Fraction of total mass in the daughter cell as a function of time for various diffusion rates**. In **(A)**, where there is no nucleus, about 40% of the mass is transferred to the daughter cell for fast diffusion rates. The addition of obstacles drastically reduces the amount of transmitter material. **(B)** Moving Nucleus. **(C)** Splitting Nucleus. **(D)** Additional compartments in the cell.

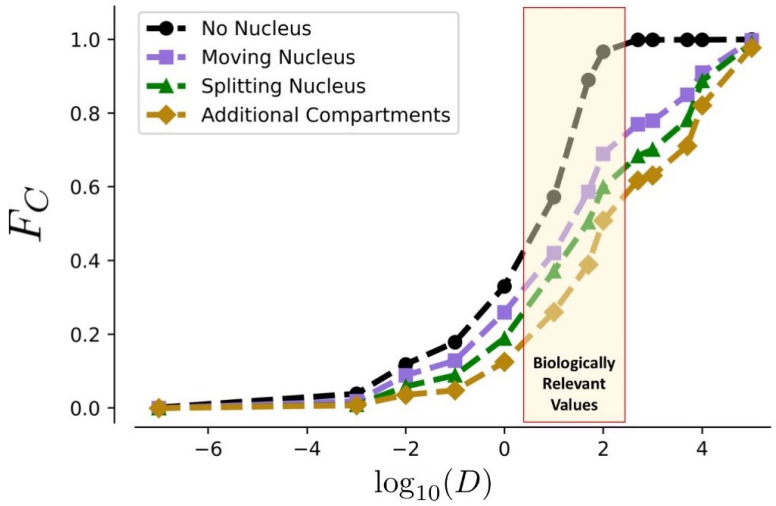


Figure 2.10: **Ratio of the final average concentration in the daughter and mother cells (F_C) for varying diffusion coefficient (D).** As expected, almost no mass is transferred between the mother and daughter cells for very slow diffusion rates. At the other extreme, the system is well-mixed, and the concentrations are identical (ratio of 1). We find that the system is far from an ideal well-mixed environment for biologically relevant diffusion rates. The system's complexity amplifies this discrepancy.

Table 2.3: Simulation parameters for the dividing yeast cell (section 2.6.1)

	Parameter	Symbol	Value	References
Geometry	Budding time	T	90 min	[83]
	Mother radius	r_m	2.07 μm	[84]
	Daughter final radius	r_d	1.81 μm	[84]
	Nucleus radius	r_n	0.89 μm	[85,86]
	Estimated IPOD and JUNQ length	-	0.28 μm	[68,87]
Biochemistry	Characteristic concentration	ψ_0	713 μm^{-3}	[88]
	Typical diffusivity	-	24-120 $\mu\text{m}^2\text{min}^{-1}$	[85]
	Typical fragmentation rate	γ_{BA}	1.35×10^{-3} min^{-1}	[89]
	Typical aggregation rate	γ_{AB}	$2.57 \times 10^{-4} \mu\text{m}^3 \cdot$ min^{-1}	[89]
Computational	Domain length	-	7.5 μm	
	Grid levels	-	5:8	
	Grid resolution	-	29 nm - 234 nm	
	Time step	Δt	12 s	

analysis in two steps, focusing first on a purely diffusive system and exposing the limitations of the well-mixed assumption as the system representation complexifies. We then focus on the full reaction-diffusion system and quantify the asymmetries in the species repartition.

2.6.2. DIFFUSIVE SYSTEM - LIMITATIONS OF THE WELL-MIXED ASSUMPTION

To quantify the impact of the diffusion alone, we first simulate a purely diffusive system (*i.e.* , $\gamma_{AB} = \gamma_{BA} = 0$). Because the two protein species are now decoupled, we will focus on species$A$ only and assume that $\psi_B = 0$. For each geometrical representation, we vary the diffusion rate from 10^{-7} to 10^5, scanning in particular through the biologically relevant range ($\approx 10^1 - 10^2$, shown in Table 2.3), and measure the amount of transmitted material in two ways. First, we compute $F_A(t)$ the fraction of total mass transferred to the daughter cell

$$F_A(t) = \frac{\int_{\Omega_D(t)} \psi_A}{\int_{\Omega_D(t) \cup \Omega_M(t)} \psi_A} = \frac{\int_{\Omega_D(t)} \psi_A}{M_T^0} \tag{2.53}$$

where $\Omega_M(t)$ and $\Omega_D(t)$ denote the inside of the mother and daughter cells at time t, and M_T^0 is the initial total mass. Second, we compute F_C, the final ratio of average

concentrations between the mother and daughter cells

$$F_C = \frac{\int_{\Omega_D(T)} \psi_A}{\int_{\Omega_M(T)} \psi_A} \frac{|\Omega_M(T)|}{|\Omega_D(T)|}, \tag{2.54}$$

For a symmetric transfer from the mother to the daughter cell, we expect the average concentration in both final cells to be identical, and so the concentration ratio F_C to be 1, and the final mass ratio $F_M(T)$ to only depend on the cells volume and be $\frac{|\Omega_D(T)|}{|\Omega_D(T)|+|\Omega_M(T)|}$, which for the chosen radii (see table 2.3), and in the absence of nuclei or compartments, is 0.4.

Fig. 2.11 depicts the diffusion process for all four geometries. The fraction of final mass for all geometries and all considered diffusion coefficients is represented in Fig. 2.9. Unsurprisingly, the fraction of mass remains under 0.4 (the ideal value for a perfectly symmetric transfer) even for the fastest diffusion and the least obstructed geometry (for case **A**, $D_A = 10^3$, we measure $F_M(T) = 0.38$). The final separation between the two cells happens around $t = 80$. After this time, the daughter mass cannot change despite the daughter cell still moving and growing. Indeed, we observe the daughter's mass to be constant after this point in all cases. We interpret this as another illustration of the conservation property of our finite volume formulation.

Adding the nucleus seems to have the most dramatic impact on the transfer process, as it reduces the daughter's mass by around 50%. Furthermore, including the split of the nucleus or additional cellular compartments reduces the transmission by another 10% to 15%. Ultimately, the transmission is largely asymmetric for the most realistic representations (B, C, D), suggesting that the diffusive process alone can generate asymmetries and that the well-mixed assumption is irrelevant.

To further investigate these asymmetries and the validity of the well-mixed assumption, we turn our attention to Fig. 2.10, where the final concentration ratios are reported. Again, an ideal well-mixed system would lead to an ideal transfer and a ratio of 1. A ratio of 0 indicates that no transfer occurred. For the most realistic geometries (B, C, D), the ideal transmission limit is only approached for the largest diffusion coefficients, likely an order of magnitude larger than the biological ones. The well-mixed assumption for such a system is, therefore, largely inaccurate. Most interestingly, the transition from a non-transferring system ($F_C = 0$) to an ideal well-mixed system ($F_C = 1$) appears to be centered around the biologically relevant diffusion values. This observation suggests that the biological system may lie where the diffusion coefficient variations have their largest impact. In other words, this hints that the biological system may be hypersensitive to the prions' diffusivity.

2.6.3. *FULL SYSTEM - ASYMMETRIC DISTRIBUTIONS*

For this final study, we consider the full systems and quantify the impacts of the reaction rates on the transfer asymmetry for each protein species. We set the diffusion coefficients ($D_A = 10^3$, $D_B = 1$) to be close to the biologically relevant range. We chose D_B to be less than D_A because species B is bigger and therefore expected to be less mobile. We will consider the splitting nucleus geometry only and quantify the transfer process by computing the final mass fraction of each species

$$F_A(T) = \frac{\int_{\Omega_D(T)} \psi_A}{M_T^0} \qquad \text{and} \qquad F_B(T) = \frac{2\int_{\Omega_D(T)} \psi_B}{M_T^0}. \tag{2.55}$$

For a single species, the final mass fraction in the daughter cell does not exceed 0.2 (see Fig. 2.9 **C**). Because all our initial conditions are re-scaled to have the same total mass, the final mass fraction of either species cannot exceed 0.2. If either of them reaches this value, it indicates that the other species has been depleted. In the current context, we will define a perfectly symmetric transfer as one where the above final ratios are identical. The reaction-diffusion process is illustrated in Fig. 2.12. Both concentrations appear quasi-uniform in each cell, sharply varying over the bud neck. These localized spatial heterogeneities seem to explain the large concentration asymmetries we observed at the final stage. This suggests that the area of the bud neck where the cellular material is transferred is crucial in the transmission process.

In the same figure, we display the final fraction of each species for varying reaction rates. As expected, when one of the reaction rates becomes extremely large (top left and bottom right corners of the diagrams), one of the species will be almost depleted. At the same time, the mass ratio of the other one will approach the maximum ratio (0.2). Furthermore, in the vicinity of the biologically relevant rates, we observe that the fraction of species A is about two times larger than that of species B, indicating a clear asymmetry in the transmission process.

We expect this unbalanced transfer will occur each time a new cell is created. Therefore, the ratio of asymmetries between the youngest and oldest cells will magnify with the generational gap's length. We note that these observations are consistent with previous experimental and computational studies of the yeast [PSI^+] prion system[61].

2.7. DISCUSSION AND CONCLUSION

Motivated to understand asymmetric transfers in dividing yeast cells, we proposed a novel numerical framework for reaction-diffusion systems in a three-dimensional deforming domain. Using finite volume discretizations, level set functions, and adaptive octree grids, our framework can produce accurate simulations at an enhanced computational cost while offering extreme modeling flexibility. The cornerstone of our approach is our novel finite volume formulation, where the partial differential equations are integrated over spatio-temporal control volumes. As we demonstrated, this procedure ensures mass conservation and produces a converging solution, which a traditional finite volume discretization may not achieve.

Using this new computational tool, we demonstrated how spatial heterogeneity can cause asymmetric protein transfer in dividing yeast cells and studied the effect of the yeast geometry, the mobility of the prions, and the reaction rates. We found that diffusion alone can create asymmetries, even more so for realistic parameters and geometries. This leads us to conclude that the well-mixed assumption is not pertinent to such systems. Looking at the full reaction-diffusion model, we were able to quantify the transfer of each protein species from the mother to the daughter cell for a wide range of reaction rates. We found that our system produces large asymmetries reminiscent of these observed in experimental setups for plausible estimations of these rates.

Our reaction-diffusion model is probably too simple to simulate the entire transmission process comprehensively. Yet, it succeeds at reproducing experimentally observed features and provides valuable insights for future modeling strategies. Our exploration revealed sharp spatial variations across the bud neck, intuiting that the geometry of the neck may be crucial for the transmission process. This suggests that either the model or the computational grid may need to be refined in that area or that perhaps a reduced two-dimensional model of the bud neck can capture the essence of this problem.

The flexibility of our framework makes it a method of choice for studying complex intracellular biophysical processes and virtually any reaction-diffusion system on a deforming domain. The reaction-diffusion system can easily be modified to include more protein species, different initial populations, production and destruction rates, or other biological processes while preserving mass conservation. In addition, the cell shape can be refined to better match experimental observations, and other cellular entities can be integrated to better reflect how complex a cell's environment is.

As microscopic imaging technologies advance, high-fidelity modeling strategies for studying sub-cellular protein dynamics aggregation models and simulations are

required to match and support experimental data. This work lays the foundation of a new class of continuum modeling techniques for efficient and accurate simulation of intracellular processes, which we believe can help the scientific community shine the light on some essential biophysical mechanisms, a necessary first step to understanding diseases and developing new treatments.

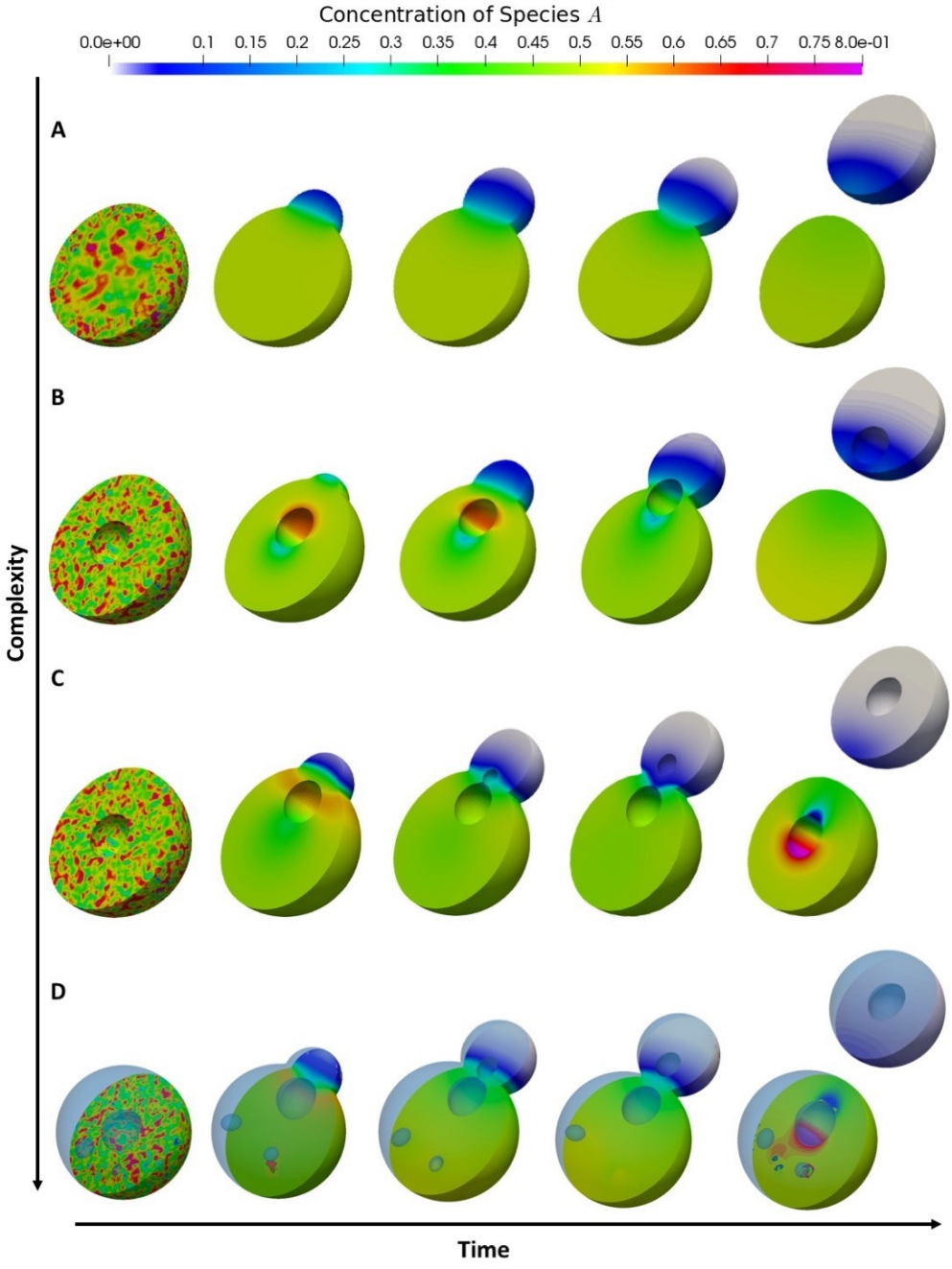


Figure 2.11: **Prions diffusion in dividing yeast cells for geometries of increasing complexity and** $D_A = 10^{-3}$. The concentration profile is represented in characteristic concentration units ($\psi_0 = 713\ \mu\text{m}^{-3}$). **A** No nucleus, **B** Moving nucleus, **C** Splitting nucleus, and **D** Splitting nucleus with additional moving compartments.

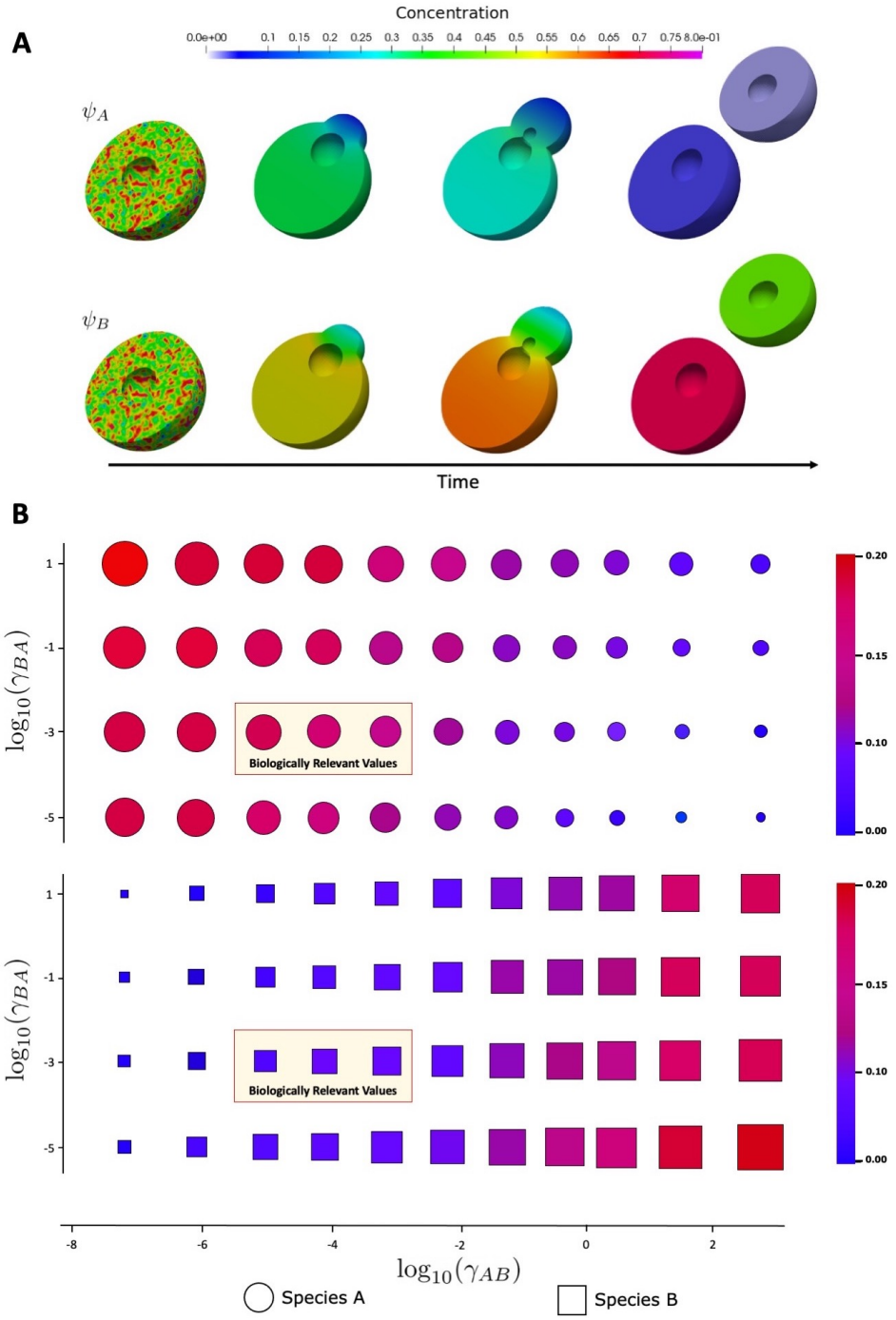


Figure 2.12: **Asymmetric mass transfer in dividing yeast cell**. **(A)** Concentration profiles for $\gamma_{BA} = 10^{-3}$, $\gamma_{AB} = 1$, $D_A = 10^3$ and $D_B = 1$. **(B)** Final mass fractions in the daughter cell ($F_A(T)$, $F_B(T)$).

Part II

Data-Driven Analysis of Biological Systems

The capacity to blunder slightly is the real marvel of DNA. Without this special attribute, we would still be anaerobic bacteria, and there would be no music.

Lewis Thomas

3

Mathematical and Biological Background

Although multicellular organisms contain a common genome within their cells, the morphology and gene expression patterns of cells are largely distinct and dynamic. These differences arise from internal gene regulatory systems and external environmental signals. Cells proliferate, differentiate, and function in tissues while sending and receiving signals from their surroundings. These environmental factors cause cell fate to be highly dependent on the environment in which it exists. Therefore, monitoring a cell's behavior in the residing tissue is crucial to understanding cell function and its past and future fate[90].

Advancements in single-cell sequencing have transformed the genomics and bioinformatics fields. The advent of single-cell RNA sequencing (scRNAseq) has enabled researchers to profile gene expression levels of various tissues and organs, allowing them to create comprehensive atlases in different species[91,92,93,94,95]. Moreover, scRNAseq enables the detection of distinct subpopulations present within a tissue, which has been paramount in discovering new biological processes, the inner workings of diseases, and effectiveness of treatments[96,97,98,6,99,100,101,102]. However, high-throughput sequencing of solid tissues requires tissue dissociation, resulting in the loss of spatial information[103,104]. To fully understand cellular interactions, data on

tissue morphology and spatial information is needed, which scRNAseq alone can not provide. The placement of cells within a tissue is crucial from the developmental stages (*e.g.* , asymmetric cell fate of mother and daughter cells[105]) and beyond cell differentiation (such as cellular functions, response to stimuli, and tissue homeostasis[106]). These limitations would be alleviated by technologies that could preserve spatial information while measuring gene expression at the single-cell level.

Spatial Transcriptomics (ST) provides an *unbiased* view of tissue organization crucial in understanding cell fate, delineating heterogeneity, and other applications[107]. However, many current ST technologies suffer lower sensitivities than scRNAseq while lacking the single-cell resolution that scRNAseq provides[108]. Targeted *in situ* technologies have tried to solve the issue of resolution and sensitivity but are limited in gene throughput and often require *a priori* knowledge of target genes[108]. More specifically, *in situ* technologies (such as *in situ* sequencing[109], single-molecule fluorescence *in situ* hybridization (smFISH)[110,111,112], targeted expansion sequencing[113], cyclic-ouroboros smFISH (osmFISH)[114], multiplexed error-robust fluorescence *in situ* hybridization (MERFISH)[115], sequential FISH (seqFISH+)[116], and spatially resolved transcript amplicon readout mapping (STARmap)[117]), are typically limited to pre-selected genes that are on the order of hundreds, with the accuracy potentially dropping as more probes are added[117]. We will refer to these methods as *image-based* techniques.

On the other hand, Next Generation Sequencing (NGS)-based technologies (such as 10x Genomics' Visium and its predecessor[118,119], Slide-Seq[120], HDST[121]) barcode entire transcriptomes but have limited capture rates, and resolutions that are larger than a single cell[122] (50 μm - 100 μm for Visium and 10 μm for Slide-Seq). Moreover, unlike image-based technologies, NGS-based methods allow for unbiased profiling of large tissue sections without necessitating a set of target genes[123,124]. However, using computational approaches, NGS-based technologies do not have a single-cell resolution, requiring cellular features to be inferred or related to the histological scale. Many current algorithms use traditional statistical or medical image processing frameworks that require human supervision[122,125,126], which is not ideal for large-scale analyses. Additionally, many algorithms are not generalizable across different sequencing platforms, which limits their utility and restricts multi-omics integration efforts.

Deep Learning (DL) methods can use raw data to extract useful representations (or information) needed for performing a task, such as classification or detection[127]. This quality makes this Machine Learning (ML) algorithm class ideal for applications with large, higher-dimensional, noisy data, such as single-cell omics. DL models have been

extensively used in scRNAseq studies (*e.g.* preprocessing[128,129], clustering[130,131], cell-type identification[132,133,25,134] and data augmentation[23,135]), and have shown to significantly improve upon traditional methods[6], suggesting the potential of such methods in ST analysis. Moreover, DL models can leverage multiple data sources, such as images and text data, to learn a set of tasks[136]. Since spatially-resolved transcriptomics are inherently multimodal (*i.e.* , they consist of images and gene expression count data), and that downstream analysis consists of multiple tasks (*e.g.* clustering and cell-type detection), researchers have sought to develop ST-specific DL algorithms.

3.1. RNA SEQUENCING TECHNOLOGIES

RNA sequencing (RNA-seq) provides comprehensive insights into cellular processes (such as identifying up or down-regulated genes, etc.). However, traditional bulk RNA-seq is limited to revealing the average expression from a collection of cells and not disambiguation of single-cell behavior. Thus, it is difficult to delineate cellular heterogeneity with traditional RNA-seq, which is a disadvantage since cellular heterogeneity has been shown to play a crucial role in understanding many diseases[137]. Therefore, researchers have turned to single-cell RNA-seq (scRNAseq) to identify cellular heterogeneity within tissues. ScRNAseq technologies have been instrumental in the study of key biological processes in many diseases, such as cancer[138], Alzheimer's[139], cardiovascular diseases[140], etcetera (see[137] for more details). RNA sequencing of cells at a single-cell resolution, scRNAseq, generally consists of four stages:

(i) **Isolation of Single-Cells and Lysing**: Cells are selected through laser microdirection, fluorescence-activated cell sorting (FACS), microfluidic/microplate Technology (MT) or a combination of these methods[141], with MT being highly complementary to NGS-based technologies[142]. MT encapsulates each single cell into an independent microdroplet containing unique molecular identifiers (UMI), lysis buffer for cell lysis (to increase the capturing of as many RNA molecules as possible), oligonucleotide primers, and deoxyribonucleotide triphosphates (dNTPs) in addition to the cells themselves. Due to MT's higher isolation capacity, thousands of cells can be simultaneously tagged and analyzed, which is beneficial for large-scale scRNAseq studies.

(ii) **Reverse Transcription**: One challenge in RNA sequencing is that RNA can not be directly sequenced from cells, and thus RNA must first be converted to complementary DNA (cDNA)[143]. Although dist technologies employ different techniques, the reverse transcription phase generally involves capturing mRNA

using poly[T] sequence primers that bind to mRNA ploy[A] tail before cDNA conversion. Based on the sequencing platform, other nucleotide sequences are added to the reverse transcription; for example, in NGS protocols, UMIs are added to tag unique mRNA molecules to trace them back to their originating cells, enabling the combination of different cells for sequencing.

(iii) **cDNA Amplification**: Given that RNA can not be directly sequenced from cells, single-stranded RNAs must first be reverse-transcribed to cDNA. However, limited cDNA is produced due to the small amount of mRNA in cells, which is not optimal for sequencing. Therefore, the limited quantity of cDNA must be amplified before library preparation and sequencing [144]. The amplification is often done by either PCR (exponential amplification process with its efficiency being sequence dependent) or IVT (a linear amplification method which requires an additional round of reverse transcription of the amplified RNA) before sequencing [143,145]. The final cDNA library has an adaptor-ligated sequencing library attached to each end.

(iv) **Sequencing Library Construction**: Finally, every cell's tagged and amplified cDNA is combined for library preparation and sequencing, similar to bulk RNA sequencing methods, followed by computational pipelines for processing and analysis. [146].

Fig. 3.1(A) illustrates an example of the workflow for scRNAseq. For more details of each stage and various scRNAseq workflows, we refer the reader to references [147,145,148,149,149].

With the technologies now defined, we describe common *Machine Learning* methods used to analyze sequencing data. In this section, we first discuss the algorithmic development of ML and *Deep Learning* models and then discuss common architectures used for spatially-resolved transcriptomics (and scRNAseq data).

3.2. INTRODUCTION TO MACHINE LEARNING AND DEEP LEARNING

Machine Learning (ML) refers to a computer algorithm's ability to acquire knowledge by extracting patterns and features from raw data [150]. All ML algorithms depend on data, which must be available before the methods can be used, and a defined mathematical objective. ML models' lifecycle consists of two phases, namely *training* and *evaluation*. During training, ML algorithms analyze the data to extract patterns and adjust their internal parameters based on optimizing their objective (known as *loss function*). In the evaluation (or inference) stage, the trained model makes predictions (or performs the task it was trained to do) on *unseen* data.

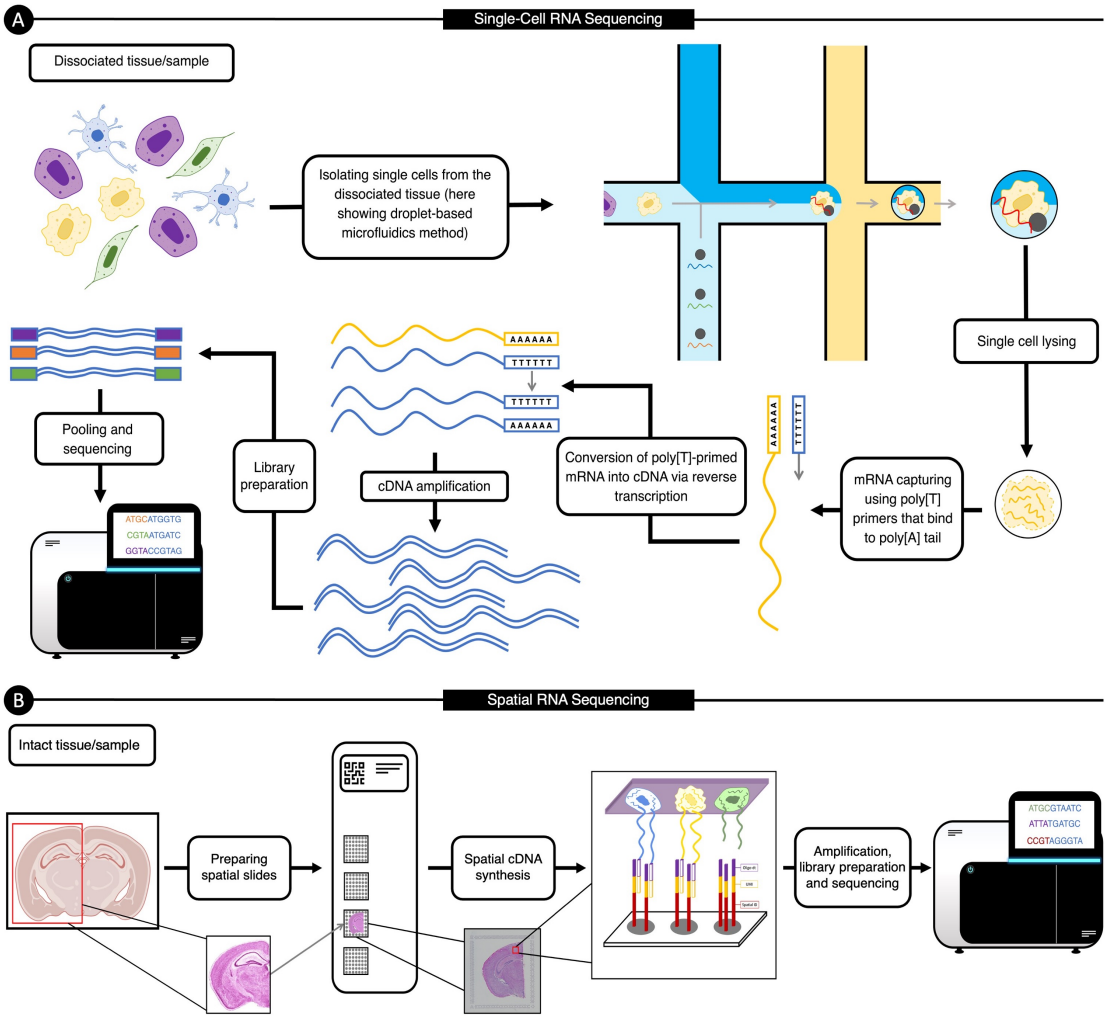


Figure 3.1: **Example Single-Cell RNA Sequencing and Spatial Transcriptomics Workflows**. **(A)** An illustration of droplet-based microfluidics single-cell RNA sequencing which consists of (1) dissociating a tissue or biological sample, (2) isolating single cells, unique molecular identifiers and lysis buffer, (3) cell lysis, (4) mRNA capture and reverse transcription, (5) cDNA amplification, and (6) library construction. **(B)** A visualization of the steps for next-generation sequencing-based spatial transcriptomic, which include (1) tissue preparation, staining, and imaging, (2) permeabilizing the tissue, (3) cDNA synthesis, amplification, and library construction, followed by (5) sequencing.

There are four main types of ML algorithms: *supervised*, *unsupervised*, *semi-supervised* and *self-supervised*. An ML algorithm is considered to be *unsupervised* if it utilizes raw inputs without any labels to optimize its objective function (an example would be the K-Means clustering algorithm[151]). Conversely, if an algorithm uses raw data and the associated labels (or targets) in training, it is a *supervised* learning algorithm. Supervised learning is the most common form of ML[127]. An example of supervised learning in scRNAseq analysis would be classifying cell subpopulations using prior annotations: this requires a labeled set of cell types for training (the available annotations), an objective function for calculating learning statistics (teaching the model), and testing data for measuring how well the model can predict the cell-type (label) on data it has not seen before (*i.e.* generalizability of the model). Another common example of supervised learning is regression, where a model predicts continuous values instead of outputting labels or categorical values in classification. For supervised tasks, a model is trained on the majority of the data (known as *training set*) and then evaluated on held-out data (*test set*). Depending on the size of our dataset, there can also be a third data split known as a *validation set*, which is used to measure the performance of the model throughout training to determine *early stopping*[152]: Early stopping is when we decide to stop the training of a model due to overfitting (or over-optimization) on the training set. Overfitting training data worsens the model's generalizability on unseen data, which early stopping aims to avoid[152]. In addition to supervised and unsupervised algorithms, there are also *semi-supervised* learning, where a model uses a mix of both supervised and unsupervised tasks, and *self-supervised*, where the computer algorithm generates new or additional labels to improve its training or to learn a new task.

Raw experimental data typically contains noise or other unwanted features, which present many challenges for ML algorithms. Therefore, it is often necessary to carefully preprocess data or to rely on domain-specific expertise to transform raw data into some internal representation from which ML models can learn[127]. However, Deep Learning (DL) algorithms aim to use only raw data to automatically extract and construct useful representations required for learning the tasks at hand. In a broad sense, DL models can learn from observations by constructing a hierarchy of concepts, where each concept is defined by its relation to simpler concepts. A graph representation of the hierarchy of concepts (and learning) will consist of many layers, with many nodes and edges connecting the vertices, resembling a human neural network. This graph is referred to as an Artificial Neural Network (ANN). ANNs are composed of interconnected nodes ("artificial neurons") that resemble and mimic our brains' neuronal functions. An ANN is considered a DL model if it consists of many layers–often more than three, hence being called *deep*.

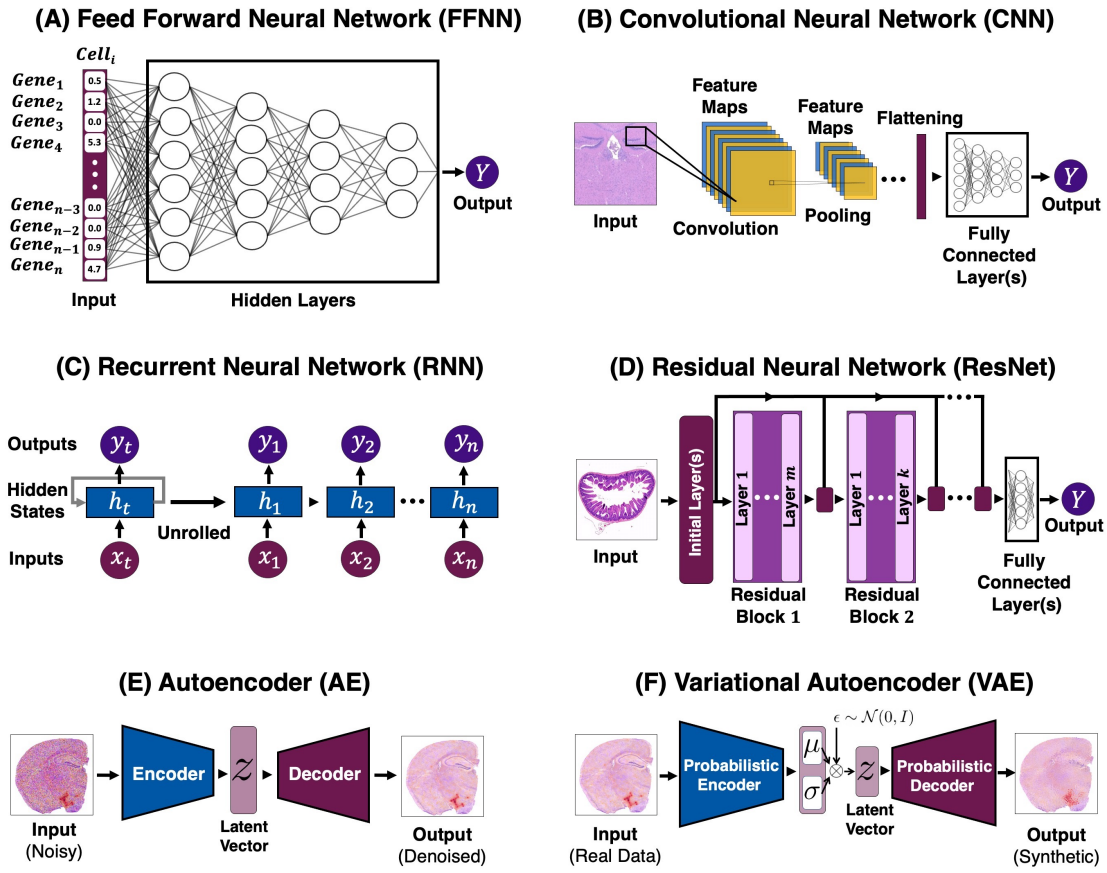
Many tasks humans perform can be viewed as mappings between sets of inputs and outputs. For example, humans can take a snapshot image of their surroundings (input) and detect the relevant objects (the outputs). DL, more generally Artificial Intelligence, aims to learn such mappings to model human-level intelligence. Mathematically, ANNs are universal function approximators, meaning that, theoretically, they can approximate any (continuous) function[153,154,155]. Cybenko[153] proved this result for a one-layer neural network with an arbitrary number of neurons (nodes) and a sigmoid activation function by showing that such architecture is dense within the space of continuous functions (this result has now been extended to ANNs with multiple layers[154]). While constructing arbitrarily-long single-layer ANNs is not possible, it has been shown that ANNs with many many layers (deeper) generally learn faster and more reliably than ANNs with few wide (many neurons) layers[156]. This has allowed researchers to employ deep networks for learning very complex functions through constructing simple non-linear layers, which can transform the representation of each module (starting with the raw input) into a representation at a higher, slightly more abstract level[127].

DL models' ability to approximate highly non-linear functions has revolutionized many domains of science, including Computer Vision[157,158,159], Natural Language Processing[160,161,162] and Bioinformatics[163,164,165]. DL is becoming increasingly incorporated in many computational pipelines and studies, especially in genomics and bioinformatics, including scRNAseq and spatial transcriptomics analysis. The following sections provide a brief overview of essential deep learning architectures used in spatial transcriptomics and scRNAseq analysis. In Fig. 3.2, we present illustrations of the architectures discussed in the following sections. Note that for simplicity, we have categorized all Graph Convolution Networks (GCN)[166] as DL models; this is because (i) GCNs can easily be extended to include more layers (deeper networks), and (ii) lack of other existing methods which incorporate some elements of DL. A more comprehensive description of each architecture can be found in the seminal textbook by Goodfellow *et al.*[150].

3.3. A MATHEMATICAL OVERVIEW OF DEEP LEARNING ARCHITECTURES

3.3.1. FEED FORWARD NEURAL NETWORK (FFNN)

FFNNs, the quintessential example of Artificial Neural Networks (ANNs), aim to approximate a function mapping a set of inputs to their corresponding targets (see Fig. 3.2(A)). More specifically, given an input $\mathbf{x} \in \mathbb{R}^n$ and a target $\mathbf{y} \in \mathbb{R}^m$, where $n, m \in \mathbb{R}$, FFNNs aim to *learn the optimal parameters* $\boldsymbol{\theta}$ *such that* $\mathbf{y} = f(\mathbf{x}; \boldsymbol{\theta})$. FFNNs



(Caption on next page)

are the building blocks of many more advanced architectures (*e.g.* convolutional neural networks) and, therefore, of paramount importance in the field of ML[150]. As mentioned previously, ANNs are universal function approximators, representing a directed acyclic graph of function composition hierarchy within the network. Each layer of an FFNN, $f^{(i)}(\mathbf{x};\boldsymbol{\theta})$ ($i \in \mathbb{N}$ being the i-th layer), is often a simple linear function: For example, we can have a linear function for outputting $y \in \mathbb{R}$ of the form Eq. (3.1), with weight parameters $\mathbf{w} \in \mathbb{R}^n$ and a bias $b \in \mathbb{R}$:

$$y = f^{(1)}(\mathbf{x};\boldsymbol{\theta}) = f^{(1)}(\mathbf{x};\mathbf{w},b) = \mathbf{x}^T\mathbf{w} + b. \tag{3.1}$$

Figure 3.2: **Examples of Deep Learning Architectures**. Models depicted in **(A), (B), (C), (D)** are examples of supervised learning, and networks shown in **(E), (F)** are unsupervised. **(A)** An example of an FFNN architecture with gene expression count as its input. **(B)** An example of CNN architecture, where the model passes the inputs through the three stages of a CNN (with non-linear activation not depicted) to extract features. Then, outputs are flattened and fed into a fully connected layer (or layers). **(C)** The general training flow of an RNN, with the unrolled version showing the timestep-dependent inputs, hidden state, and outputs. The inputs to RNNs need a sequential structure (*e.g.* time-series data). **(D)** An illustration of a ResNet. In traditional ResNets, identity mappings (or skip connections) pass the input of a residual block to its output (often through addition). **(E)** Here, we show the general architecture of a *trained* denoising AE in the inference stage, with a noisy histology slide as its input, yielding a denoised version of the input image. Note that this approach is unsupervised since no labels are required during training (only the "clean" images). **(F)** A depiction of a traditional VAE in the inference stage. VAE aims to generate synthetic data that closely resembles the original input. This is done by regularizing an AE's latent space with a probabilistic encoder and decoder.

However, a model composed of *only* linear functions can *only* approximate linear mappings. As such, we must consider *non-linear activation* functions to increase model capacity, enabling the approximation of complex non-linear functions. In the simplest case, Neural networks (NNs) use an affine transform (controlled by learned parameters) followed by a non-linear activation function, which, theoretically, enables them to approximate any non-linear function [167]. Moreover, we could compose many non-linear transformations to avoid infinitely wide neural networks when approximating complex functions. However, in this context, finding a set of optimal functions $f^{(i)} : \mathbb{R}^{q_i} \to \mathbb{R}^{d_i}$ $(q_i, d_i \in \mathbb{R})$ is a practically impossible task. As such, we restrict the class of function that we use for $f^{(i)}$ to the following form in Eq. (3.2):

$$f^{(i)}(\mathbf{x}^{(i-1)}; \boldsymbol{\theta}^{(i)}) = \sigma^{(i)}(W^{(i)}\mathbf{x}^{(i-1)} + \mathbf{b}^{(i)}), \tag{3.2}$$

where superscript i enumerates the layers, $\sigma(\cdot)$ is a non-linear activation function (usually a Rectified Linear Unit [168]), $\mathbf{x}^{(i-1)} \in \mathbb{R}^{q_i}$ denotes the output of the layer $(i-1)$ (with $\mathbf{x}^{(0)}$ indicating the input data), weights $W \in \mathbb{R}^{d_i \times q_i}$ and biases $\mathbf{b}^{(i)} \in \mathbb{R}^{d_i}$. Note that because of the dimensionality of the mapping, $W^{(i)}X^{(i-1)} \in \mathbb{R}^{d_i}$ and we must have a vector of biases $\mathbf{b}^{(i)} \in \mathbb{R}^{d_i}$). FFNNs are composed of such functions in chains; to il-

lustrate, consider a three-layer neural network:

$$\mathbf{y} = f(\mathbf{x};\boldsymbol{\theta}) \tag{3.3a}$$
$$= f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x};\boldsymbol{\theta}^{(1)});\boldsymbol{\theta}^{(2)});\boldsymbol{\theta}^{(3)}) \tag{3.3b}$$
$$= f^{(3)}\left(h^{(2)}\left(h^{(1)}\left(\mathbf{x};\mathbf{w}^{(1)},\mathbf{b}^{(1)}\right);\mathbf{w}^{(2)},\mathbf{b}^{(2)}\right);\mathbf{w}^{(3)},\mathbf{b}^{(3)}\right). \tag{3.3c}$$

with h representing the *hidden states or hidden layers*.

FFNNs find the optimal contribution of each parameter (*i.e.* weights and biases) by minimizing the desired objective. The goal is to generalize the task to data the model has never seen before (testing data). Although the non-linearity increases the capacity of FFNNs, it causes most objective functions to become non-convex. In contrast to convex optimization, non-convex loss functions do not have global convergence guarantees and are sensitive to the initial starting point (network parameters)[169]. Therefore, such optimization is often done through stochastic gradient descent (or some variant). Moreover, given the sensitivity to initial values, weights are typically chosen as small random values, with biases initialized to zero or small positive values[150,170,171].

3.3.2. CONVOLUTIONAL NEURAL NETWORK (CNN)

Learning from images, such as detecting edges and identifying objects, has been of interest for some time in computer science[172]. Images contain a lot of information. However, only a small amount of that information is often relevant to the task at hand. For example, an image of a stained tissue contains important information, namely the tissue itself, and irrelevant pixels, such as the background. Prior to DL, researchers would hand-design a feature extractor to learn relevant information from the input. Much of the work had focused on the appropriate feature extractors for desired tasks (*e.g.* see the seminal work by Marr and Hildreth[173]). However, one of the main goals of ML is to extract features from raw inputs without hand-tuned kernels for feature extraction. CNNs[174,175] are a specialized subset of ANNs that use the convolution operation (in at least one of their layers) to learn appropriate kernels for extracting important features beneficial to the task at hand. Mathematically, convolution between two functions f and w is defined as a commutative operation shown in Eq. (3.4)

$$(f * w)(x) \triangleq \int_{-\infty}^{\infty} f(s)w(s-x)ds. \tag{3.4}$$

Using our notation, we intuitively view convolution as the area under $f(s)$ weighted by $w(-s)$ and shifted by x. In most applications, discrete functions are used. For example, assume we have a 2D kernel K that can detect edges in a 2D image I with dimension $m \times n$. Since I is discrete, we can use the discrete form of Eq. (3.4) for the convolution of I and K over all pixels:

$$E(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n). \tag{3.5}$$

However, since there is less variation in the valid range of m,n (the dimensions of the image) and the operation is commutative, most algorithms implement Eq. (3.5) equivalently:

$$E(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n). \tag{3.6}$$

Typical CNNs consist of a sequence of layers (usually three), which include a layer performing convolution, hence called a *convolutional layer* (affine transform), a detector stage (non-linear transformation), and a pooling layer. The learning unit of a convolutional layer is called a *filter or kernel*. Each convolutional filter is a matrix, typically of small dimensions (*e.g.* 3x3 pixels), composed of a set of weights that acts as an object detector, with the weights being continuously calibrated during the learning process. CNNs' objective is to learn an optimal set of filters (weights) to detect the needed features for specific tasks (*e.g.* image classification). The result of convolution between the input data and the filter's weights is often referred to as a *feature map* (as shown in Fig. 3.2(B)). Once a feature map is available, each value of this map is passed through a non-linearity (*e.g.* ReLU). The output of a convolutional layer consists of as many stacked feature maps as the number of filters present within the layer.

Two key ideas are behind the design of CNNs: First, local neighbors have highly correlated information in data with grid-like topology. Second, equivariance to translation can be obtained if units at different locations *share weights*. In other words, sharing parameters in CNNs enabled the detection of features regardless of the locations where they appear. An example of this would be detecting a car. In a dataset, a car could appear at any position in a 2D image, but the network should be able to detect it regardless of the specific coordinates[172]. These design choices provide CNNs with three main benefits compared to other ANNs: (i) sparse interactions, (ii) shared weights, and (iii) equivariant representations[174].

Another way of achieving equivariance to translation is to utilize pooling layers. Pooling decreases the dimension of learned representations and makes the model in-

sensitive to small shifts and distortions[127]. In the pooling layers, we use the outputs of the detector stage (at certain locations) to calculate a summary statistic for a rectangular window of values (*e.g.* calculating the mean of a 3x3 patch). There are many pooling operations, with common choices being max-pooling (taking the maximum value of a rectangular neighborhood), mean-pooling (taking the average), and L2 norm (taking the norm). In all cases, rectangular patches from one or several feature maps are inputted to the pooling layer, where semantically similar features are merged into one. CNNs typically have an ensemble of stacked convolution layers, non-linearity, and pooling layers, followed by fully connected layers that produce the network's final output. The backpropagation of gradients through CNNs is analogous to FFNNs, enabling the model to learn an optimal set of filters for the task(s) at hand. CNNs have been effectively used in many applications in computer vision and time-series analysis[176] and are being increasingly utilized for the analysis of ST data since spatial omics are multimodal, with one of the modalities being images[122].

3.3.3. RECURRENT NEURAL NETWORK (RNN)

Just as CNNs are specialized to process data with a grid-like topology, RNNs'[177] special characteristics make them ideal for processing sequential data $X = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(n)}\}$, where $\mathbf{x}^{(i)}$ denotes the i-th element in the ordered sequence X. Examples of such sequence-like structures include time series and natural language. RNNs process sequential inputs one at a time and implicitly maintain a history of previous input sequence elements. We present an illustration of the conventional RNN architecture in Fig. 3.2(C). Similar to FFNNs or CNNs, RNNs can be composed of many layers, with each layer depending on the previous hidden state, $h^{(t-1)}$, and a shared set of parameters, $\boldsymbol{\theta}$. A deep RNN with n hidden states can be expressed as follows:

$$h^{(n)} = f(\mathbf{x}^{(n)}, h^{(n-1)}; \boldsymbol{\theta}); \boldsymbol{\theta}) \tag{3.7a}$$

$$= f(\mathbf{x}^{(n)}, f(\mathbf{x}^{(n-1)}, h^{(n-2)}; \boldsymbol{\theta}); \boldsymbol{\theta}) \tag{3.7b}$$

$$= f(f(\cdots f(\mathbf{x}^{(2)}, h^{(1)}(\mathbf{x}^{(1)}; \theta); \boldsymbol{\theta}) \cdots; \boldsymbol{\theta}); \boldsymbol{\theta}). \tag{3.7c}$$

The idea behind sharing $\boldsymbol{\theta}$ in RNN states is similar to CNNs: parameter sharing across different time points allows RNNs to generalize the model to sequences of variable lengths, and share statistical strengths at different positions in time[150,*]. Similar

*Note that if the model chose a separate parameter for each $\mathbf{x}^{(i)}$, for $i = 1, \cdots, n$, then the model could not generalize to any inputs where $|X| > n$ (size of X is greater than n).

to FFNNs, RNNs learn by propagating the gradients of each hidden state's inputs at discrete times. This process becomes more intuitive if we consider the outputs of hidden units at various time iterations as if they were the outputs of different neurons in a deep multi-layer network. However, due to the sequential nature of RNNs, the backpropagation of gradients shrinks or grows at each time step, causing the gradients to vanish or blow up potentially. This fact and the inability to parallelize training at different hidden states (due to the sequential nature of RNNs) makes RNNs notoriously hard to train, especially for longer sequences [160,178]. However, when these issues are averted (via gradient clipping or other techniques), RNNs are powerful models and gain state-of-the-art capabilities in many domains, such as natural language processing. The training challenges combined with the nature of scRNAseq data have resulted in fewer developments of RNNs for single-cell analysis. However, recently, some studies have used RNNs and Long Short-Term Memory [179] (a variant of RNNs) for predicting cell types and cell motility (*e.g.* see Kimmel et al. [180]).

3.3.4. RESIDUAL NEURAL NETWORK (RESNET)

As mentioned above, deep RNNs may suffer from vanishing or exploding gradients. Such issues can also arise in other deep neural networks, where gradient information could diminish as the depth increases (through approaches such as Batch Normalization [181] aim to help with gradient issues). One way to alleviate vanishing gradients in very deep networks is to allow gradient information from successive layers to pass through, helping maintain information propagation even as networks become deeper. ResNets [182] achieve this by *skip* (or *residual*) connections that add the input to a block (a collection of sequential layers) to its output. For a FFNN, consider function f in Eq. (3.2). Using the same notation as in Eq. (3.2), ResNet's inner layers take the form shown in Eq. (3.8):

$$f^{(i)}(\mathbf{x}^{(i-1)}) = \mathbf{x}^{(i-1)} + \sigma^{(i)}(W^{(i)}\mathbf{x}^{(i-1)} + \mathbf{b}). \tag{3.8}$$

The addition of $\mathbf{x}^{(i-1)}$, the input of the current layer (or the output of $(i-1)$-th layer), to the current i-th layer output is the *skip or residual* connection helps flow the information from the input deeper in the network, thus stabilizing training and avoiding vanishing gradient in many cases [182,183]. Indeed, this approach can be contextualized within the traditional time integration framework for dynamical systems. For example, consider Eq. (3.9):

$$\dot{x}(t) = \frac{dx}{dt} = \mathscr{F}(t, x(t)), \quad x(t_0) = x_0. \tag{3.9}$$

In the simplest case, this system can be discretized and advanced using $x(t_n)$ and some scaled value of $\mathscr{F}(t_n, x(t_n))$, or a combination of scaled values of $\dot{x}(t_n)$. Forward Euler, perhaps the simplest time integrator, advances the solution as shown through the scheme in Eq. (3.10)

$$x_{n+1} = x_n + h\mathscr{F}(t_n, y_n), \tag{3.10}$$

where h is a *sufficiently small* real positive value. ResNets uses this idea to propose a different way of calculating the transformations in each layer, as shown in Eq. (3.8).

ResNets consist of residual blocks (also called modules), each containing a series of layers. For visual tasks, these blocks often consist of convolutional layers, followed by activation functions, with the skip connection adding the input information to the output of the residual blocks (as opposed to the individual layers inside). ResNets have different depths and architectures, with a number usually describing the depth of the model (*e.g.* ResNet50 means there are 50 layers [there are 48 convolution layers, one MaxPool, and one AveragePool layer]).

ResNets have transformed DL by enabling the training of *very* deep neural networks, setting the state-of-the-art performance in many areas, particularly in computer vision[182]. The pre-trained ResNets on ImageNet dataset [†] are widely used for transfer learning, where the network is either used as is or further fine-tuned on the specific dataset. As discussed in this chapter, pre-trained ResNet models have also been used in spatial transcriptomics analysis.

3.3.5. GRAPH NEURAL NETWORKS

Graph Neural Networks (GNNs) have emerged as a powerful class of machine learning models designed to tackle complex data structures represented as graphs. GNNs aim to produce a representation of graphs in a continuous space (depicted in Fig. 3.3. GNNs have recently gained prominence in various domains recently, from social network analysis to recommendation systems and computational biology. Unlike traditional deep learning models that operate on grid-structured data, GNNs are uniquely equipped to capture the intricate relationships and dependencies within graph-structured data, making them an invaluable tool for tasks that involve connectivity patterns, such as node classification, link prediction, and graph classification, tasks that can answer numerous biological questions. The mathematical foundation of GNNs rests on the principles of graph theory and deep learning, enabling them to generalize and propagate information effectively across graph nodes.

[†] ImageNet[184] is the standard dataset for benchmarking the performance of machine learning algorithms in classification and object recognition. ImageNet contains more than 14 million hand-annotated images.

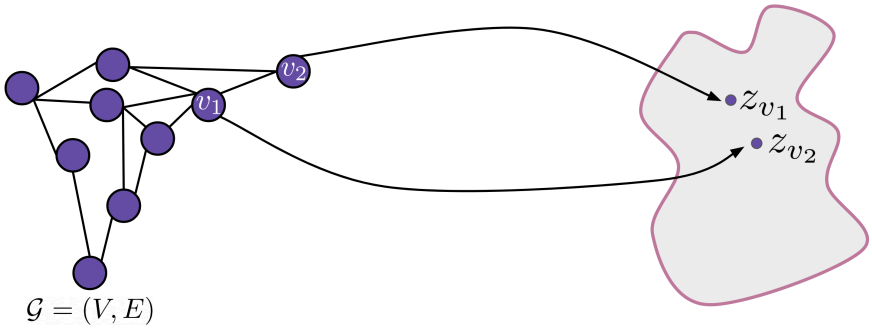


Figure 3.3: Graph Neural networks aim to learn a mapping from graph-structured data (on the left) to a low-dimensional continuous vector space (on the right). These representations, instead of the original graph, can then be used for various downstream tasks.

3.3.6. AUTOENCODER (AE)

AEs [185,186] are neural networks that aim to reconstruct (or copy) the original input via a *non-trivial mapping*. Conventional AEs have an "hour-glass" architecture (see Fig. 3.2(E)) consisting of two networks: (i) an encoder network, $Enc(\cdot)$, which maps an input $\mathbf{x} \in \mathbb{R}^n$ to a latent vector $\mathbf{z} \in \mathbb{R}^d$ where, ideally, $\mathbf{z}$ contains the most important information from $\mathbf{x}$ in a reduced space (*i.e.* $d \ll n$), (ii) the decoder network, $Dec(\cdot)$, which takes $\mathbf{z}$ as input and maps it back to $\mathbb{R}^n$, ideally, reconstructing $\mathbf{x}$ exactly; *i.e.* $\mathbf{x} = AE(\mathbf{x}) = Dec(Enc(\mathbf{x}))$. AEs were traditionally used for dimensionality reduction and denoising, trained by minimizing a mean squared error (MSE) objective between the input data and the reconstructed samples (decoder outputs).

Over time, the AE framework has been generalized to stochastic mappings, *i.e.* probabilistic encoder-decoder mappings, $p_{Enc}(\mathbf{z}|\mathbf{x})$ and $p_{Dec}(\mathbf{x}|\mathbf{z})$. A well-known example of such generalization is Variational Autoencoders (VAEs) [187], where by using the same hour-glass architecture, one can use probabilistic encoders and decoders to generate new samples drawn from an approximated posterior. Both traditional AEs and VAEs have practical applications in many biological fields and have been used extensively in scRNAseq (see reference [6] for an overview of these models), and are becoming more frequently employed in spatial transcriptomics analysis, which we overview later in this work.

3.3.7. VARIATIONAL AUTOENCODER (VAE)

One can describe VAEs[187] as AEs that regularize the encoding distribution, enabling the model to generate new synthetic data. The general idea behind VAEs is to encode the inputs as a *distribution over the latent space* instead of a single point (which is done by AEs). From a neural network perspective, VAEs are autoencoders that use variational inference to reconstruct the original data, having the ability to generate new data that is "similar" to those already in a dataset x. Mathematically, VAEs are motivated through traditional Bayesian inference, where we aim to find a likelihood function for generating x given latent variables z, $p(x|z)$. However, the question is what latent parameters $\mathbf{z}$ one must use, given the data x. VAEs assume that observed data and latent representation are jointly distributed as $p_\theta(x,z) = p_\theta(x|z)p(z)$. In deep learning, the log-likelihood $p_\theta(x|z)$ is modeled through non-linear transformations, thus making the posterior probability distribution,

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}, \tag{3.11}$$

which is intractable due to the *marginal*. More specifically, the issue arises from the denominator,

$$p_\theta(x) = \int p(x,z)dz \tag{3.12}$$

which is possible to compute directly. *Variational inference* aims to approximate the posterior through a family of functions $q_\gamma(z|x)$. Using this idea, we then find that:

$$p(x) = \int p(x,z)dz = \int p_\theta(x|z)p_\theta(z)dz \tag{3.13}$$

$$\Rightarrow \log p(x) = \log \int p_\theta(x|z)p_\theta(z)dz \tag{3.14}$$

$$= \log \int p_\theta(x|z)p_\theta(z)\frac{q_\gamma(z|x)}{q_\gamma(z|x)}dz \tag{3.15}$$

$$= \log\Big(\mathbb{E}_{q_\gamma(z|x)}[p(x,z)]\Big) - \log\Big(\mathbb{E}_{q_\gamma(z|x)}[\log q_\gamma(z|x)]\Big) \tag{3.16}$$

$$\Rightarrow \log p(x) \geq \mathbb{E}_{q_\gamma(z|x)}[\log p(x,z)] - \mathbb{E}_{q_\gamma(z|x)}[\log q_\gamma(z|x)], \tag{3.17}$$

where $\mathbb{E}$ denotes the expected value, *i.e.* $E[X] = \int u f_X(u)du$ for f_X being the probability density function of X. Therefore, we can maximize the evidence lower bound

(ELBO) to get a good approximation of the evidence $p_\theta(x)$:

$$\underbrace{\mathbb{E}_{q_\gamma(z|x)}\left[\log\left(\frac{p_\theta(x|z)p(z)}{q_\gamma(z|x)}\right)\right]}_{\text{ELBO}(\theta,q_\gamma)} \leq \log p_\theta(x), \tag{3.18}$$

where $q_\gamma(z|x)$ being the mentioned auxiliary variational distribution with parameters γ that is dependent on the family of distributions chosen. Note that by maximizing the lower, we can better approximate the evidence and thus obtain a close approximation for the true posterior $p_\theta(z|x)$. We find the variational parameters γ for new inputs x using an inference network $Enc(\cdot)$ with parameters ϕ, such that $Enc_\phi(x) = \gamma(x)$. But the question is how can we measure the difference between ELBO and the true evidence. This can be done by computing the *Kullback-Leibler* (KL) divergence, as shown in Eq. (3.19).

$$\mathbb{KL}\left(q_\phi(z|x)||p_\theta(z|x)\right) := \mathbb{E}_{z\sim q_\phi(z|x)}\log\left[\frac{q_\phi(z|x)}{p_\theta(z|x)}\right] \tag{3.19}$$

Using our definition of KL divergence, we can write:

$$\begin{aligned}
\mathbb{KL}(q||p) &= \mathbb{E}_{z\sim q_\phi(z|x)}[\log q_\phi(z|x)] - \mathbb{E}_{z\sim q_\phi(z|x)}[\log \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}] && (3.20)\\
&= \mathbb{E}_{z\sim q_\phi(z|x)}[\log q_\phi(z|x)] - \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x,z)] + \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x)] && (3.21)\\
&= \log p_\theta(x) - \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] && (3.22)\\
&= \log p_\theta(x) - \text{ELBO}(\theta,q_\phi) && (3.23)\\
&\Rightarrow \log p(x) = \text{ELBO}(\theta,q_\phi) - \mathbb{KL}\left(q_\phi(z|x)||p_\theta(z|x)\right). && (3.24)
\end{aligned}$$

Given that KL divergence is non-negative, this means that $\log p(x) \geq \text{ELBO}(\theta,q_\phi)$. Therefore, we have:

$$\log p_\theta(x) \geq \text{ELBO}(\theta,q_\phi) = \mathscr{L}(x;\theta,\phi) = \mathbb{E}_{z\sim q_\phi(z|x)}\left[-\log q_\phi(z|x) + \log p_\theta(x|z)p_\theta(z)\right],$$

which can be written as shown in Eq. (3.3.7):

$$\mathscr{L}(x;\theta,\phi) = \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x|z)] - \mathbb{KL}\left(q_\phi(z|x)||p(z)\right)\right].$$

A common assumption in many frameworks is that the posterior and the prior are isotropic Gaussian distributions, *i.e.* $q_\phi(z|x^{(i)}) = \mathcal{N}(z;\mu^{(i)},(\sigma^{(i)})^2 I)$ and $p_\theta(z) = \mathcal{N}(z;0,I)$. This assumption simplifies the computation of the KL divergence, as shown in Eq. (3.25):

$$\mathbb{KL}\left(q_\phi(z|x)||p_\theta(z)\right) = \frac{1}{2}\sum_j^J \left(1 + (\log\sigma_j^{(i)})^2 - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) \tag{3.25}$$

(refer to Kingma *et al.* [188] for the proof). In this scenario, the VAE objective function becomes:

$$\mathscr{L}(x^{(i)};\theta,\phi) = \frac{1}{2}\sum_j^J \left(1 + \log(\sigma_j^{(i)})^2 - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \frac{1}{N}\sum_l^N \log p_\theta(x^{(i)}|z^{(i,l)}) \tag{3.26}$$

where $z^{(i,l)} \sim q_\phi(z|x^{(i)})$ with the parametrization trick $z^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \varepsilon^{(l)}$, with $\varepsilon^{(l)} \sim \mathcal{N}(0,I)$. Furthermore, if we parametrize the likelihood function with another Gaussian, our objective function becomes even simpler. The most common form of VAEs' objective is shown in Eq. (3.27), though this is not the exact resulting loss. We provide the derivation of the simplified reconstruction loss in Appendix B.

$$\mathscr{L}(x^{(i)};\theta,\phi) = \frac{1}{2}\sum_j^J \left(1 + \log(\sigma_j^{(i)})^2 - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + MSE(x^{(i)},\hat{x}^{(i)}) \tag{3.27}$$

where $MSE(\cdot)$ denotes the *Mean Squared Error* between the input $x^{(i)}$ and the reconstruction by the probabilistic decoder $\hat{x}^{(i)}$.

Compared to other generative models (*e.g.* Generative Adversarial Networks (GANs) [189]), VAEs have desirable mathematical properties and training stability [150]. However, they suffer from two major weaknesses: (i) classic VAEs create "blurry" samples (those that adhere to an average of the data points) rather than the sharp samples that GANs generate due to GANs' adversarial training. One possibility for this blurriness is the effect of maximum likelihood. That is, the model may assign a high probability to other points rather than just the training data, and these points may include "blurry" samples/images. This issue has often been addressed by defining adversarial training between the encoder and the decoder, as done by our work in Chapter 5. (ii) The other major issue with VAEs is posterior collapse: when the variational posterior and actual

posterior is nearly identical to the prior (or collapse to the prior), which results in poor data generation quality[190]. To alleviate these issues, different algorithms have been developed, which have been shown to significantly improve the quality of data generation[191,192,193,193,194,195,196]. VAEs are used extensively to analyze single-cell RNA sequencing (see Erfanian et al.[6]), and we anticipate them to be applied to a wide range of spatial transcriptomics analyses as well.

3.3.8. GENERATIVE ADVERSARIAL NETWORKS (GANS)

GANs[197] are capable of generating realistic synthetic data and have been successfully applied to a wide range of machine learning tasks[198,199,200,201] and bioinformatics[202,135]. GANs consist of a generator network (G) and a discriminator network (D) that train adversarially, which enables them to produce high-quality fake samples. During training, D learns the difference between real and synthetic samples, while G produces fake data to "fool" D. More specifically, G produces a distribution of generated samples P_g, given an input $z \sim P_z$, with P_z being a random noise distribution. The objective of GANs is to learn P_g, ideally finding a close approximation to the real data distribution P_r so that $P_g \approx P_r$. To learn the approximation to P_g, GANs play a "min-max game" of

$$\min_G \max_D \mathbb{E}_{x\sim P_r} \log[D(x)] + \mathbb{E}_{z\sim P_z} \log[1 - D(G(z))],$$

where both players (G and D) attempt to maximize their own payoff. This adversarial training is critical in GANs' ability to generate realistic samples. Compared to other generative models, GANs' main advantages are (i) the ability to produce any type of probability density, (ii) no prior assumptions for training the generator network, and (iii) no restrictions on the size of the latent space.

Despite these advantages, GANs are notoriously hard to train since it is highly non-trivial for G and D to achieve Nash equilibrium[203]. Another disadvantage of GANs is vanishing gradients where an optimal D cannot provide enough information for G to learn and make progress. If D learns the distinction between real and generated data too well, then G will fail to train, as shown by[204]. Another issue with GANs is "mode collapse" when G generates only a small set of outputs that can trick D. More explicitly, this occurs when G has learned to map several noise vectors z to the same output that D classifies as real data. In this scenario, G is over-optimized, and the generated samples lack diversity. Quantifying how much GANs have learned about real data distribution is often complicated, measuring the dissimilarity between P_g and P_r when P_r is not known or assumed. Therefore, common ways of evaluating

GANs involve directly evaluating the output[205], which can be arduous.

Although some variations of GANs have been proposed to alleviate vanishing gradients and mode collapse (*e.g.* Wasserstein-GANs (WGANs)[206] and Unrolled-GANs[207]), the convergence of GANs remains a major problem. During the training progression, the feedback of D to G becomes meaningless. If GANs continue to train past this point, the quality of the synthetic samples can be affected and ultimately collapse. Common variations of GANs cannot be trained as single-stream networks, and a necessary step is to define a training schedule for G and D separately, adding another layer of complexity. Although all deep learning models are sensitive to hyperparameter choices,[208] show all experimented GANs (including WGANs) are much more sensitive to these choices than VAEs. This can be a drawback in using GANs for scRNAseq generation since the hyperparameters may need to be re-tuned for every new dataset.

If you're not careful what you focus on, you might move towards something you never intended.

Unknown

4

Boosting Single-Cell RNA Sequencing Analysis with Neural Attention

4.1. INTRODUCTION

Single-cell RNA sequencing (scRNAseq) technologies have been instrumental in studying biological patterns and processes in development and disease[24,209]. ScRNAseq datasets are high-dimensional and noisy, often requiring advanced machine learning (ML) methods for accurate and efficient analysis[6]. Recent years have seen a surge in deep learning (DL) methods, setting the state-of-the-art performance in many pre- and post-processing tasks, such as batch correction, dimensionality reduction, and multi-modal integration (see[6]). While DL methods have become the standard for accuracy, a limitation of most DL approaches for scRNAseq analysis is a lack of biological interpretability; traditional DL approaches transform scRNAseq data into low dimensional representations that cannot directly be associated with specific gene signatures. Moreover, current pipelines require researchers to develop and train separate disjoint models for each specific downstream analysis pipeline. In this work, we introduce scANNA (single-cell analysis using Neural-Attention). This flexible DL model utilizes simple neural attention to provide unique biological insights and facilitate the discovery process through a single training procedure for a suite of downstream tasks.

The major advance of scANNA is the innovative use of a neural attention module within a three-module DL core (see Fig. 4.1**(A)**). This DL core is trained once on raw single-cell RNA sequencing counts and can subsequently be applied without retraining to various user-specified downstream tasks, such as automatic cell type classification, optimal feature selection, unsupervised scRNAseq annotation, and transfer learning (Fig. 4.1**(B-E)**). The DL core of scANNA has three serial modules (depicted in Fig. 4.1**(A)**), which take raw scRNAseq counts as input. The parameters of scANNA, *i.e.* the neural network weights of each module, are learned by optimizing the auxiliary objective (*e.g.* , predicting pseudo labels). For this training of the model, the auxiliary objective is designed to predict pseudo-labels, *i.e.* cell type labels generated through an unsupervised algorithm.

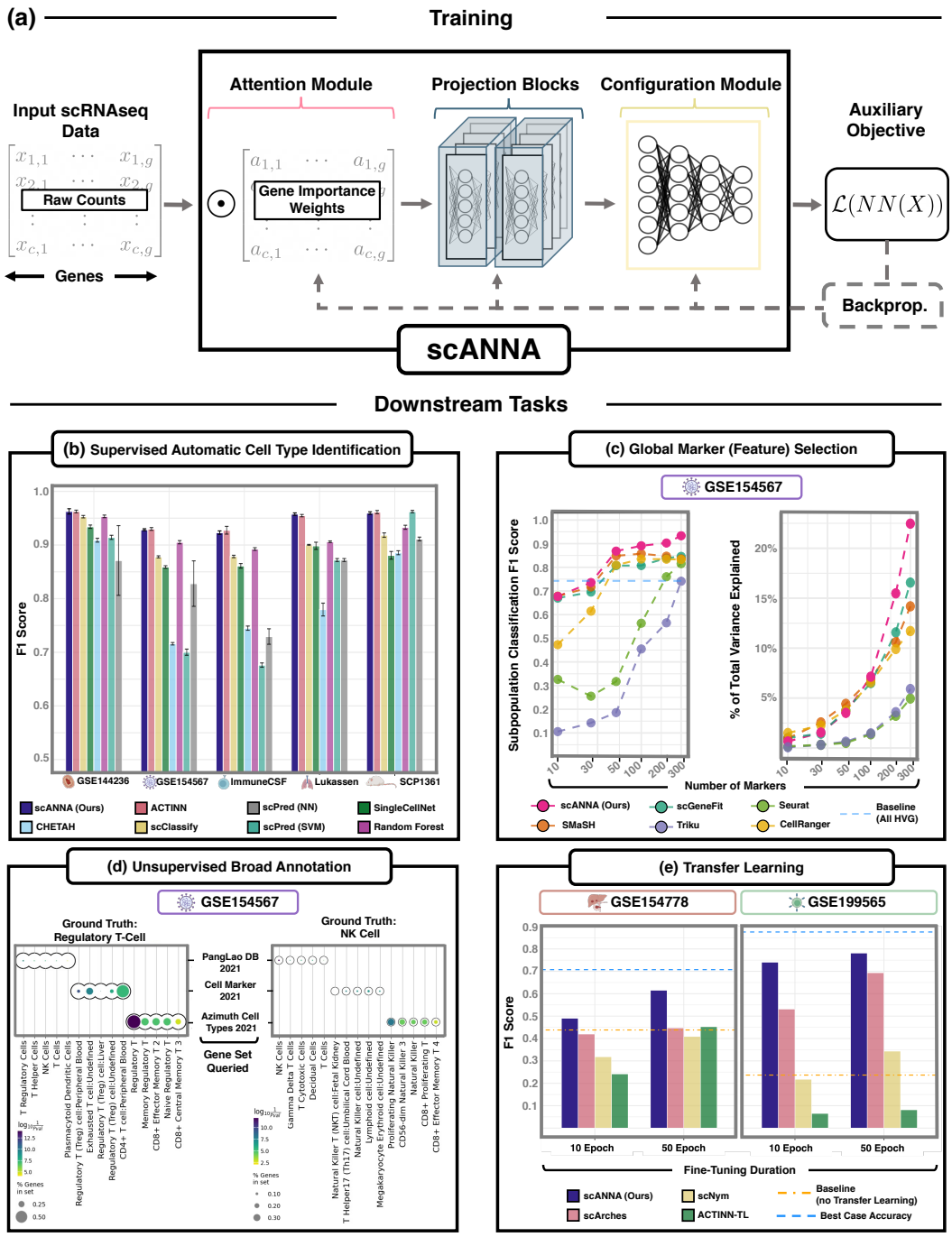


Figure 4.1: (Caption on next page)

Figure 4.1: **Overview of scANNA and its application to various downstream tasks**. **(A)** General workflow utilizing scANNA for scRNAseq studies and illustration of scANNAs novel three-module DL architecture. ScANNA can be used for a wide range of downstream tasks. In this chapter, we focus on four important tasks: **(B)** Automated Cell Type Identification **(C)** Global Marker Selection, **(D)** Unsupervised Broad Annotation, and **(E)** Transfer Learning. These results show that scANNA learns complex relations from data that can be exploited for scalable, accurate, and interpretable analysis in scRNAseq studies.

The first component of scANNA is the *Additive Attention* module, which learns optimal weights for each gene based on their contribution to the auxiliary objective. These weights are used to calculate gene scores, a scaled version of the raw counts. After training scANNAs DL core, the gene attention weights from the Additive Attention Module are used as input for most standard downstream tasks. This common training is achieved by inputting the gene scores into the second component, the *Deep Projection Blocks*, which are an ensemble of operators learning a nonlinear mapping between gene scores. This mapping is designed to increase model capacity and connect the gene associations to the auxiliary objective. A concatenated representation of the ensemble defined in this step forms the input to scANNAs third component, the *Configuration Module*. The Configuration Module is the last layer of the DL core made to align with the chosen auxiliary objective function, allowing for flexibility and potential adjustments based on the dataset (*e.g.* , the number of neurons that correspond to the number of known populations).

This chapter compares scANNA to state-of-the-art methods on various standard scRNAseq tasks using nine public single-cell datasets. We show that scANNA attains comparable to better performance despite not being explicitly trained for these standard tasks. Based on our findings, we propose scANNA as an accurate and effective method for various scRNAseq analysis settings. ScANNAs unbiased pipelines can rapidly and accurately identify new and noteworthy genes in large-scale studies with limited prior knowledge. For small-scale studies (min 2000 cells), scANNAs DL core can be pre-trained on existing atlases to transfer knowledge from existing repositories. As such, scANNA is a valuable tool for novice and expert users in the initial steps of scRNAseq analysis.

4.2. METHODS AND RELATED WORK

SCANNA: EFFICIENT NEURAL ATTENTION WITH BRANCHING PROJECTION BLOCKS

Our approach utilizes a simple neural attention (a feed-forward version of the additive attention introduced in[210]) with ensembles of linear operators, which we call *branching projections*. Multiple branching projections combined form a *projection block*. ScANNA is a much simpler version of the popular Transformers models[160] that revolutionized many fields, such as natural language processing and computer vision[211,212]. Despite being simpler than Transformers, scANNA is still a large-scale model with a large capacity for learning nonlinear relations in complex datasets, such as scRNAseq, while training faster than traditional transformer-based models and being easier to interpret. We describe scANNAs different components below (visualized in Figure 4.2).

Additive Attention

Attention is a weighting scheme that aims to mimic the way humans understand context in sentences or details in images by focusing on a subset of significant features for a given objective[213]. The use of attention-based NN for scRNAseq analysis is still in its infancy, with only a few successful scRNAseq applications to date[214,215,216]. To identify salient genes (markers), we use an additive attention module in a feed-forward NN aiming to learn the importance (weights) of all genes for each cell, given a downstream task. Analyzing the learned associations between genes and cells post hoc provides our models biological interpretability.

The first step in the DL core, the Attention Module, is used to calculate a gene-score matrix (weighted version of scRNAseq count matrix), representing expression data in later layers. These importance scores enable gene prominence quantification for the downstream task, allowing interpretation of the models decision-making. Given a gene expression matrix $X \in \mathbb{R}^{C \times N}$, where C and N denote the number of cells and genes, respectively, we define the gene-score matrix Γ and the attention weights A as shown below:

$$\Gamma = A \odot X, \text{ where } A_{i,j} = \frac{e^{L_{i,j}}}{\sum_{j=1}^{N} e^{L_{i,j}}}$$

with $L = NN(\cdot)$ denoting a linear neural network. After training, the learned operator A is leveraged to identify salient genes for interpretability. Gene scores have the

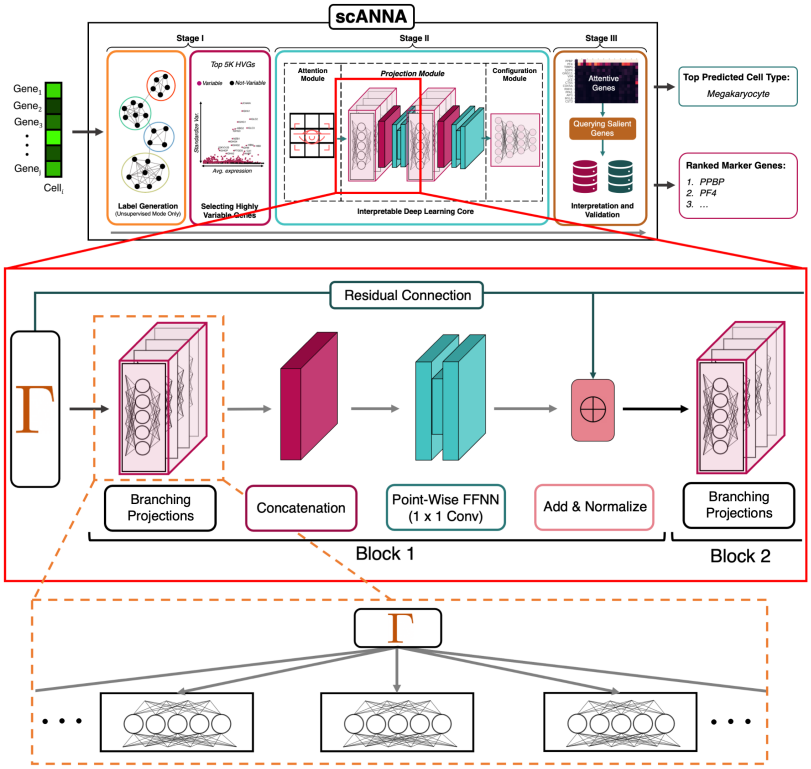


Figure 4.2: **Overview of scANNAs projection block concerning other components**. Here, we use the unsupervised annotation as an example to show scANNAs projection blocks in action.

same dimension as input data, *i.e.* , $\Gamma \in \mathbb{R}^{C \times N}$).

Branching Projections

The second module in scANNA is the projection mechanisms, which are intermediate layers between the attention layer and the Configuration Module (see Fig. 4.1(**A**)). The goal of using projection modules is to strike a balance between model capacity and efficiency: Too much capacity could lead to significant over-fitting, while insufficient capacity prevents the model from learning the correct representations. We design the projection blocks to allow for branching, *i.e.* , consisting of $h \in \mathbb{N}$ separate linear operators in each level, a concept shown by[217] to improve optimization and overall learning. Such design allows efficient consideration of different gene subsets and improves model performance without requiring numerous nonlinear layers and additional computational costs. Outputs from each branch are concatenated and inputted to a subnetwork consisting of two linear operators ($L_1 \in \mathbb{R}^{N \times 128}$ and $L_2 \in \mathbb{R}^{128 \times N}$, respectively) with a Rectified Linear Unit (ReLU) in between. Based on careful ablation studies (Table 4.1), we found projection blocks with $h = 4$ branches provide the appropriate balance of accuracy and efficiency. We hypothesized that adding residual connections from the gene scores to the input of each block would improve interpretability (by reducing the chance of learning a complex, unrelated nonlinear mapping). Through ablation studies, we found that the residual connections, followed by LayerNorm[218] operation, increased accuracy and improved interpretability. We present the architecture of a residual block in Figure 4.2.

Table 4.1: **Ablation Study on the Number of Projection Heads.** We fixed all other hyperparameters and studied the effect of head number on accuracy and interpretability for various datasets, here shown for *Immune CSF*. Training times are the average of 5 training runs (on an A100 GPU). Accuracy of each model remained the same across different training settings, since all random parameters were initialized with the same random seed. *Abbreviations*: *W-F1*: Weighted F1 score; *NW-F1*: Non-Weighted F1 score (Macro F1); *Hit@5*: Binary success measure of whether the correct cell type was retrieved within the 5 most probable cell types predicted by our model.

Number of Heads	W-F1	NW-F1	Hit@5	Avg. Training Time
1	0.9278	0.8968	0.85	9.11 ± 0.14 (Min)
4	0.9317	0.9077	0.85	9.38 ± 0.31 (Min)
8	0.9307	0.9156	**1.00**	9.31 ± 0.22 (Min)
10	0.9322	**0.9173**	**1.00**	9.56 ± 0.27 (Min)
20	**0.9324**	0.9156	**1.00**	9.87 ± 0.24 (Min)

Configuration Module

The last stage of scANNA consists of the Configuration Module, a linear operator. Our goal in separating the Configuration Module from the other components was to provide flexibility for different tasks or for transferring labels between datasets with different numbers of cells. To show scANNAs generality, we trained one model and used a cross-entropy objective to predict labels (actual annotations in a supervised setting or pseudo-labels for the unsupervised training). The Configuration Module used for our work consisted of a linear layer mapping the gene space (number of highly variable genes) to the number of cell types, followed by a Leaky ReLU activation.

TRAINING SCANNA

Results presented for all downstream tasks used one trained model, except for TL between reference and query datasets with different cell populations. For all datasets, we trained scANNA by minimizing a standard cross entropy loss using the Adam gradient-based optimizer at a learning rate $lr = 10^{-4}$ for 50 epochs. Our experiments showed training scANNA for more epochs can result in a slight increase in prediction accuracy; however, we chose to train all models for 50 epochs for additional computational efficiency. We also employed an exponential learning rate scheduling, starting at epoch ten and decaying every five epochs after with $\gamma = 0.95$ to avoid overfitting (more relevant when training over 100 epochs, which we did not do in this study).

FEATURE SELECTION EXPERIMENTS

Five methods, SMaSH[219], scGeneFit[220], Triku[221], Seurat[222], and CellRanger[223], were compared against scANNA in evaluating its feature selection performance (*i.e.* , selecting the most important global features[224]). Following common metrics in this space, we measured performance by calculating the classification accuracy given top n features, with $n = \{10, 25, 50, 100, 200, 300\}$, using K-Nearest Neighbors (KNNs), XGBoost and Nearest Centroid. As a baseline, we also trained each classifier using the 5000 HVGs. Our results only report KNN classification accuracy since KNN on 5000 HVGs performed best for most datasets. Additionally, we computed the fraction of variance on the original gene space (total variance) and compared that with the different approaches. Based on the evidence in the SmaSH manuscript and our own validation, ensemble learning (XGBoost), we have provided the most accurate approach we employed in our experiments. Of note, minor but necessary adjustments were made to the SMaSH package (located at `https://gitlab.com/cvejic-group/smash`) to make SMaSH compatible with `Tensorflow 2.11` (adjustments

noted in the tutorial and reproducibility notebooks). For a fair comparison across all methods, we modified SMaSH to use pre-defined train and test split, which were determined a priori. Our last minor modification was to allow for the ensemble method to train in parallel (by adding `n_jobs=-1` to the `XGBoostClassifer` arguments in the `ensemble_learning` class method). For scGeneFit[220], we used the provided software package (`https://github.com/solevillar/scGeneFit-python`) with the same parameters as in[225] (*i.e.* the parameters for `get_markers` function were set to `method='centers'`, `redundancy=0.25`, `epsilon = 1`). For Triku, we used the provided python package (https://github.com/alexmascension/triku) following the suggested usage protocol with only one modification (*i.e.* , the parameter for `tk.tl.triku` function was set to `use_raw=True`). For Seurat and CellRanger, we leveraged scanpys function `scanpy.pp.highly_variable_genes` to identify important features based on the dispersion-based methods Seurat (Seurat) and CellRanger (CellRanger)[226,227,223].

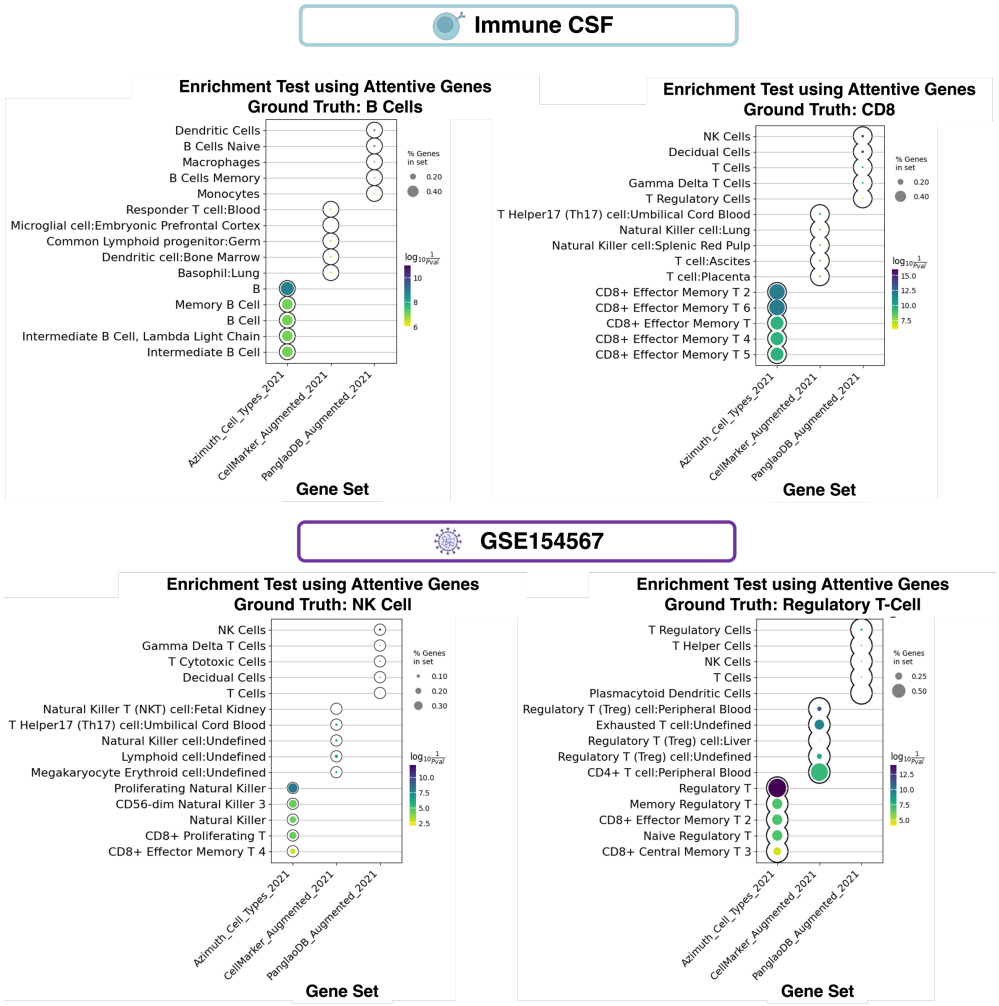


Figure 4.3: **ScANNAs unsupervised broad annotation example (enrichment of local marker genes identified by scANNA)**.

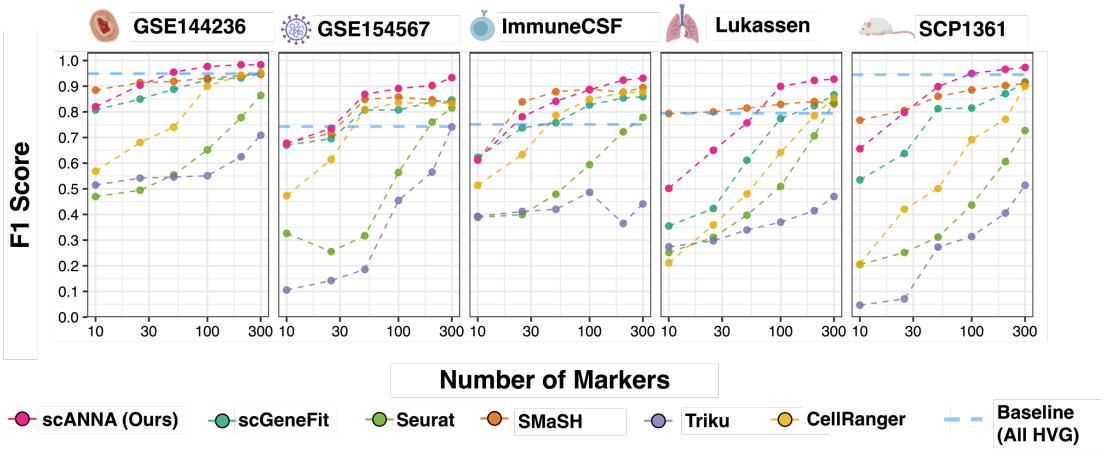


Figure 4.4: **Evaluating global marker selection performance through measuring classification accuracy of cell populations with *n* selected markers from each model**.

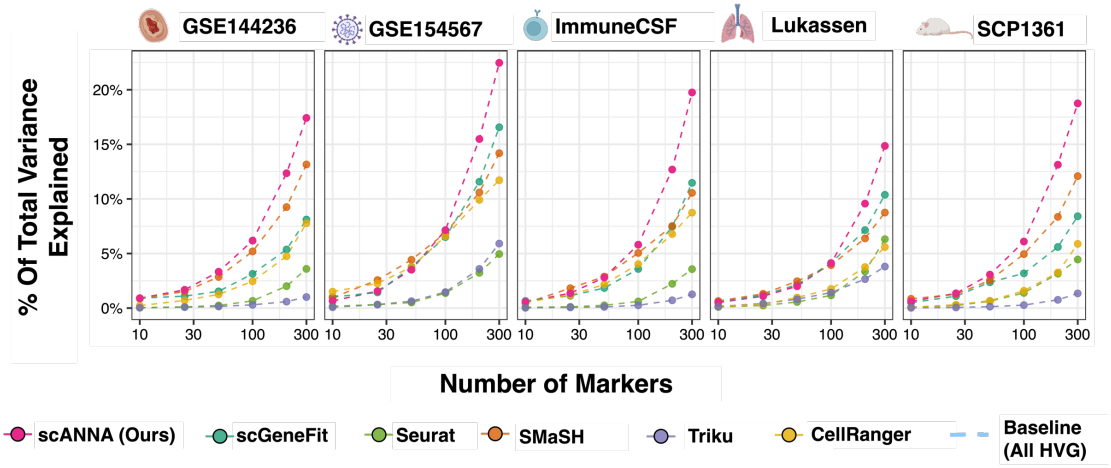


Figure 4.5: **Evaluating global marker selection performance through measuring classification accuracy of cell populations with *n* selected markers from each model**.

SUPERVISED CLASSIFICATION METHODS

Seven models were compared against scANNA to evaluate performance on supervised classification ACTINN[228], scClassify[229], SingleCellNet[230], CHETAH[231],

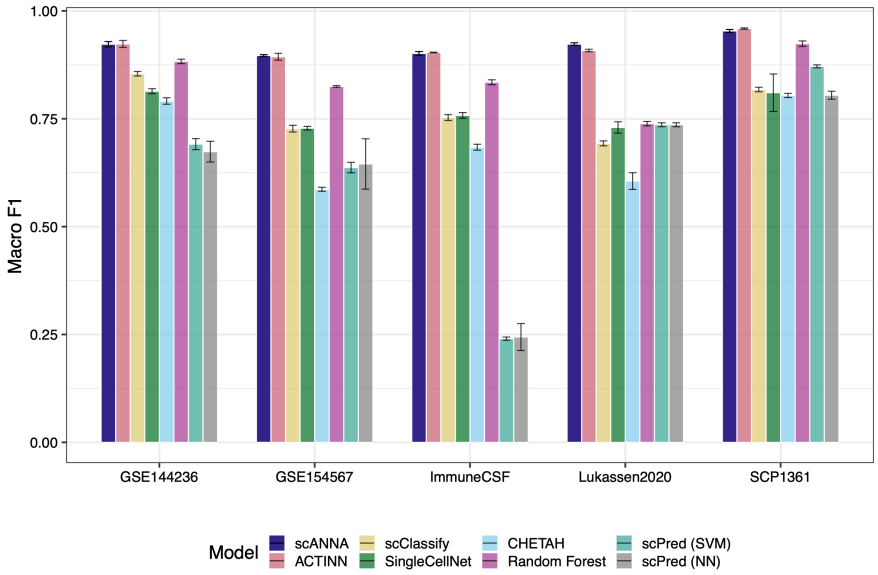


Figure 4.6: **Accuracy of various supervised annotation methods reported in macro F1 score**.

Random Forest, scPred[232] using the svmRadial (support vector machines) model, and scPred using the neural net (NN) model. ScPreds parameters were as follows: `resampleMethod` was set to `none`, `tuneLength` was set to 1, and genes containing zero counts for all cells had a pseudo count of 2 added to a cell randomly to allow scPred to run. ACTINN was run using a PyTorch implementation (https://github.com/SindiLab/ACTINN-PyTorch). Lastly, for Random Forest, we used `sklearn.ensemble.RandomForestClassifier` function with default parameters. We generated five distinct train and test splits (each containing 80/20% of all data) for each dataset using the `ShuffleSplit` function from the Sci-Kit Learn package. Each model was applied to all splits with the following metric: collected runtime, accuracy, weighted F1, and macro F1. Mean and standard deviation were calculated for each metric based on the dataset and model.

SUPERVISED ANNOTATIONS

We chose six large immune datasets to evaluate scANNAs capabilities across multiple diseases and tissues. Five datasets consist of human cells and one of the mouse cells. The human datasets included three SARS-CoV-2 viral infection studies and a human cutaneous squamous cell carcinoma study. We included a mouse dataset

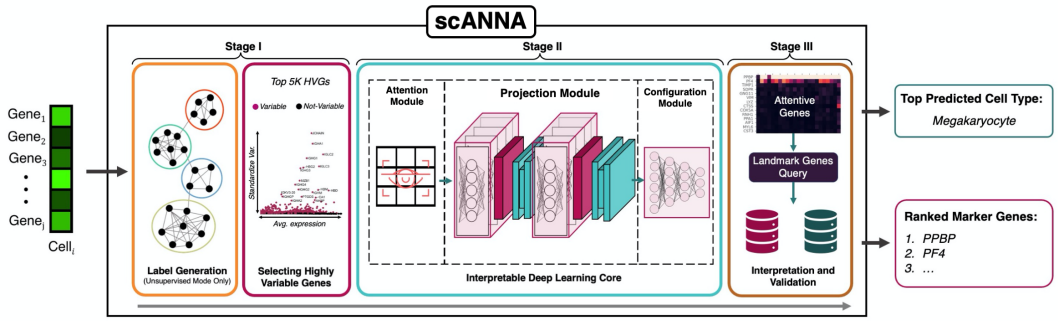


Figure 4.7: **Overview of scANNAs workflow of performing unsupervised annotation used in this study**.

to demonstrate our models effectiveness on non-human and non-immune datasets. All datasets were generated using the 10X Genomics platform. A summary for each dataset is provided in §4.3.

UNSUPERVISED ANNOTATIONS

Since labels are not available in an unsupervised setting, pseudo-labels keep the same supervised objective as in the dataset publications. Pseudo-labels can be generated using traditional methods, such as graph-based clustering or self-supervised contrastive learning (for which scANNA can also be used). To avoid any improvements in annotation not caused by scANNA, we generated pseudo-labels for each dataset using the same clustering approach described in the originating manuscript. After clustering, the arbitrary cluster numbers were used as pseudo-labels, and scANNA was trained to predict the cluster number given to a cell. No information about the cell types, known marker genes, or common housekeeping genes were used during training. After training, we extracted the top n attentive genes and used those genes as markers for querying cell type. using GSEAPYs (`https://github.com/zqfang/GSEApy`) enrichr method. We provide an overview of our workflow in Fig. 4.7.

TRANSFER LEARNING EXPERIMENTS

For transfer learning, we focused on labeling two small-scale query datasets using larger source data (references). These experiments were as follows:

Pancreatic ductal adenocarcinoma (PDAC) Transfer Learning

As an initial assessment for transfer learning, we pre-trained scANNA on a PDAC atlas, an integration of five publicly available datasets (see §4.3), which were split 80% and 20% for training and testing, respectively. As the target data, we chose Lin et al. (GSE154778)[233] since it contained scRNAseq of the pancreatic primary tumor while containing important distinctions and differences from the source data, making it appropriate for transfer learning. From this data, we used 20% of randomly selected cells for fine-tuning and 80% for testing, aiming to mimic a small-scale study. Note that the fine-tuning process uses the reverse training/testing proportions of pre-training splits to test the performance of the TL experiment.

T cell atlas to CD8+ T cell Populations

As a more challenging task, we aimed to perform TL from a broad T cell atlas (GSE188666) to a CD8+ T cell-specific dataset (GSE199565) containing more subpopulations than the reference data. Similar to the PDAC experiment, we used 80% of the source data for pre-training (and 20% for testing the pre-training), 20% of the query target data for fine-tuning, and 80% for testing (see §4.3).

FINE-TUNING SCANNA

Fine-tuning scANNA consists of freezing all weights in the attention and projection modules, with the only trainable parameters being in the configuration module (the last layer of scANNA mapping projections to the corresponding cell types). We fine-tuned scANNA (and other tested models) for 10 or 50 epochs using the same hyperparameters as pre-training.

SCARCHES FOR TRANSFER LEARNING

ScArches[234] allows transferring labels to query data by integrating it with a reference atlas. ScArches relies on an underlying reference-building method. For our application, the appropriate underlying models are scANVI[235] and scGen[236] as described in (https://scarches.readthedocs.io/); however, we were unable to use scGen due underlying maintenance issues. Therefore, all results presented in this chapter for scArches utilize scANVI (which uses scVI at its core). Though the scGen method is recommended, we note that scANVI is also appropriate since it can be trained unsupervised and fine-tuned on target data using labels, thus making the comparison relevant and fair. Using the source data, we trained the scVI core for 100 epochs, with an

additional ten epochs for the annotation, which achieved desirable accuracy (comparable to other methods). Then, similar to scANNA, we fine-tuned the model for 10 and 50 epochs on the query data.

ACTINN-TRANSFER LEARNING

We hypothesized that scANNAs advantage for TL stems from our architecture design. To test the importance of architecture, we applied our methodology for freezing weights and fine-tuning to an existing model, ACTINN, with the following rationale: If scANNAs architecture is not unique and more beneficial for transfer learning, then a pre-trained ACTINN model (on the source data) should perform comparably to scANNA (once fine-tuned on a query dataset) given its comparable performance in the supervised annotation regime. Similar to the other experiments, we trained ACTINN-TL on source data for 50 epochs, then froze the weights in the hidden layers and fine-tuned the last layer responsible for predicting the correct cell type. For simplicity, we call this approach ACTINN-TL.

SCNYM FOR TRANSFER LEARNING

ScNym[237] model is a semi-supervised method combined with MixMatch framework[238] and domain adversarial training for transferring labels from a source to a target dataset. ScNym generates pseudo-labels for target cells and randomly pairs source and target observations, with the weighted average of each being computed. ScNym then aims to minimize a supervised classification loss on the paired mixed training examples while minimizing the interpolation consistency loss on the mixed target cells. We trained scNym with the source and target data using the default setting for 100 epochs and then continued training for an additional 10 or 50 epochs with the pre-trained model using the test target data.

4.3. DATA DESCRIPTION AND DATA AVAILABILITY

PREPARATION OF SCRNASEQ DATASET FOR ANALYSIS WITH SCANNA

All data are publicly available from NCBI gene expression omnibus (GEO) and the Broad Institute Single Cell Portal (SCP), with links provided below. Datasets were processed using the Seurat[239] package (`v4.1.0`) in R. Manual annotations were merged with count matrices using a variety of tidyverse (`v1.3.1`) functions, and subsequently added to the Seurat object as metadata. Data filtering consisted of removing cells with fewer than 200 expressed genes and removing genes present in fewer than

three cells. Next, we retained cells with less than 10% mitochondrial reads to mitigate cellular debris. Lastly, cell types containing less than 100 cells were removed and excluded from the dataset. After filtering, following similar works, we identified each dataset's top 5000 highly variable genes (HVGs) using Seurats FindVariableFeatures function, described in[224]. To minimize biological and technical effects in each dataset based on patient or biological conditions (such as normal versus disease state), we utilized Harmony[240] (`v0.1.0`) to perform integration when necessary. The lower dimensional plots were generated using Uniform Manifold Approximation and Projection (UMAP)[241]. SeuratDisk (`v.0.0.0.9019`) was used to convert the Seurat data object into an AnnData object compatible with scanpy. To perform clustering and generate cell labels (in the unsupervised case), we used Scanpys pipeline for clustering (consisting of dimensionality reduction using principal component analysis (PCA), followed by Leiden clustering). As mentioned, we found Leiden resolutions that led to the same number of clusters as the annotated populations to compare our predictions to the ground truth labels.

SCP1361

SCP1361 consists of aortic cell scRNAseq from mice fed a normal or high-fat diet, resulting in 24K cells (after processing)[242]. The authors identified 27 clusters for ten different cell populations. Original data for SCP1361 can be downloaded from Single Cell Portal `https://singlecell.broadinstitute.org/single_cell.`

ImmuneCSF

ImmuneCSF (GSE163005) profiles scRNAseq data in cerebrospinal fluid[243]. Cells were isolated from 31 patients: 8 Neuro-COVID patients, nine non-inflammatory, nine autoimmune neurological diseases, and five viral encephalitis, resulting in a total of 70K cells after processing, with 15 populations. Raw data can be downloaded from Gene Expression Omnibus (GEO) with accession number GSE163005.

GSE154567

GSE154567 evaluates the transcriptional immune dysfunction in patients triggered during moderate and severe COVID-19 using scRNAseq[244]. Peripheral blood mononuclear cells were isolated and sequenced from 20 patients. Patients ranged from healthy ($n = 3$), moderate COVID ($n = 5$), acute respiratory distress syndrome (ARDS-Severe, $n = 6$), and recovering (ARDS-Recovering, $n = 6$), resulting in 69K cells. After pre-processing the dataset, we retained 64K cells. The authors identified

nine populations in the dataset. Raw data can be downloaded from Gene Expression Omnibus (GEO) with accession number GSE154567.

GSE144236

GSE144236 evaluates scRNAseq of normal skin and cutaneous squamous cell carcinoma (cSCC) tumors[245]. Normal and cSCC tumor cells were sequenced from each patient (n =10), resulting in 48K cells and seven major cell populations. We retained 47K cells after pre-processing. The myeloid cell population (CD14+Hi) is composed of various subpopulations. Raw data can be downloaded from Gene Expression Omnibus (GEO) with accession number GSE144236.

Lukassen Lung

Lukassen (EGAS00001004419) evaluates single nuclei RNA sequencing of lung tissue to evaluate the expression of ACE2 and TMPRSS1[246]. Primary lung tissue was sampled from male and female smokers and non-smokers ($n = 12$ total), resulting in 39K cells and nine cell populations. We retained 39K cells after pre-processing. Raw data is available from the European Genome-Phenome Archive (EGA) with study ID EGAS00001004419.

PDAC atlas

PDAC atlas consists of five scRNAseq published datasets[247,248,249,250,251] from normal and pancreatic ductal adenocarcinoma patients. After pre-processing and integration (as described by[252]), we retained 85,437K cells with 11 subpopulations.

PDAC small

PDAC small (GSE154778) is a specific PDAC study consisting of individual cells (57K with ten subpopulations) from dissociated primary tumors or metastatic biopsies obtained from patients with PDAC[233]. Raw data can be downloaded from Gene Expression Omnibus using accession number GSE154778.

T cell atlas

T-cell atlas (GSE188666) evaluates CD8 T cell exhaustion during viral infection by scRNAseq resulting in 96K cells (retained 96K cells after pre-processing)[253]. Raw data can be downloaded from Gene Expression Omnibus (GEO) with accession number GSE188666.

GSE199565

CD8 specific data (GSE199565) evaluates CD8+ T cell temporal differentiation by scRNAseq resulting in 29K cells (retained 29K cells after pre-processing)[254]. The authors identified 19 CD8 T cell populations (many intermediate populations). Raw data can be downloaded from Gene Expression Omnibus (GEO) with accession number GSE199565.

4.4. RESULTS

Supervised Automated Cell Type Identification: While we primarily intend scANNA to be a tool for unsupervised analysis, as an initial test, we considered scANNA in a supervised learning framework. We compared scANNA against standard state-of-the-art supervised automatic cell type identification (ACTI) models * on five complex datasets (four human datasets with distinct disease conditions and one mouse dataset). ScANNA was trained on quality-controlled raw data, and for computational efficiency, the gene space was reduced to the 5000 (5K) most variable genes, although larger gene spaces (8K, 10K, 15K) achieve similar results (a performance drop of approximately 4% from 15K to 5K). Figure 4.1**(B)** and Fig. 4.6 show that scANNA performs comparably to current state-of-the-art models for ACTI (ACTINN[228], scPred[232], SingleCellNet[230], scClassify[229], CHETAH[231]) while providing decision making interpretability (*i.e.* , attention tensor).

Next, we evaluated scANNAs capabilities in two different scRNASeq tasks: global feature selection (5 datasets), unsupervised annotations (5 datasets), and transfer learning from large-scale data to small datasets (2 pairs of datasets). For each scRNASeq data, we trained scANNA once and then applied it to each corresponding downstream task without retraining the core DL model. To train scANNA in these unsupervised analysis tasks, for each data set, we: (1) Generated pseudo-labels of cell types using an unsupervised clustering† (2) trained the DL core to predict pseudo-labels, and (3) extracted attention weights from the Attention Module or fine-tuned the Configuration Module if transferring the learnings to a different biological context.

Global Marker Selection: We considered scANNA for global feature selection.

*Although this is not scANNAs primary application. Here, we use the ground truth labels; in the other studies, we use only our generated pseudo-labels.

†While it is possible to use more advanced methods, such as self-supervised contrastive algorithms, we chose a naïve unsupervised clustering process to assess improvements offered by scANNA more directly. For the five datasets used in feature selection and unsupervised annotation, we conditioned the number of clusters to be the same as the number of manually annotated populations for a fair comparison.

That is, we used scANNAs learned gene weights to identify global markers, *i.e.* a unique and small subset of genes that are most informative and useful for additional analyses[228]. Despite the importance of global marker selection in scRNAseq studies, computational methods are nascent and often limited in the number of markers that can be selected[232]. In contrast to methods that exhaustively search for n features ranked in a one versus all task[‡], scANNAs Attention Module implicitly performs feature selection by assigning weights to each gene during training. After training, we extracted attention values and ranked top genes accordingly (§4.2).

We compared scANNA-selected genes against markers selected by state-of-the-art ML methods (scGeneFit[220] and SMaSH[225], Triku[221]) and statistical techniques (CellRanger and Seurat) for global feature selection. Following[225]; we assessed methods by evaluating classification accuracy (Weighted F1) and the fraction of total variance explained, using $n = \{10, 25, 50, 100, 200, 300\}$ (Fig. 4.1 **(C)**, Figures 4.4 and 4.5). Our results show that scANNA outperforms these methods in both metrics in the biologically relevant regime (between 50 - 200 genes[225]). As such, scANNA has the potential to be a powerful addition to biological applications such as the detection of pertinent markers (100-200 genes) required for designing padlock probes used *in situ* sequencing[24,225].

Unsupervised Annotation: Next, we set out to perform cell type annotation, another critical challenge in scRNAseq studies[255], in an unsupervised manner. A significant limitation in most computational pipelines is the reliance on manual cell type annotation based on curated lists of marker genes, which is laborious, time-consuming, partially subjective, and requires expertise[256,23]. In this experiment, we considered whether scANNAs Attention Module gene weights provide a powerful companion to manual annotation. Note that, as before, we use pseudo-labels for training and use the existing annotations for validation and testing purposes after training. Once scANNA was trained, we extracted the top attentive genes in each cluster as markers to query the gene set for identifying cell type embedded in our software package (§4.2).

[‡]One v *n* marker selection refers to evaluating the importance of each gene against n other genes (potentially all). Then, top genes are selected as markers based on some metric (*e.g.* fraction of variance explained).

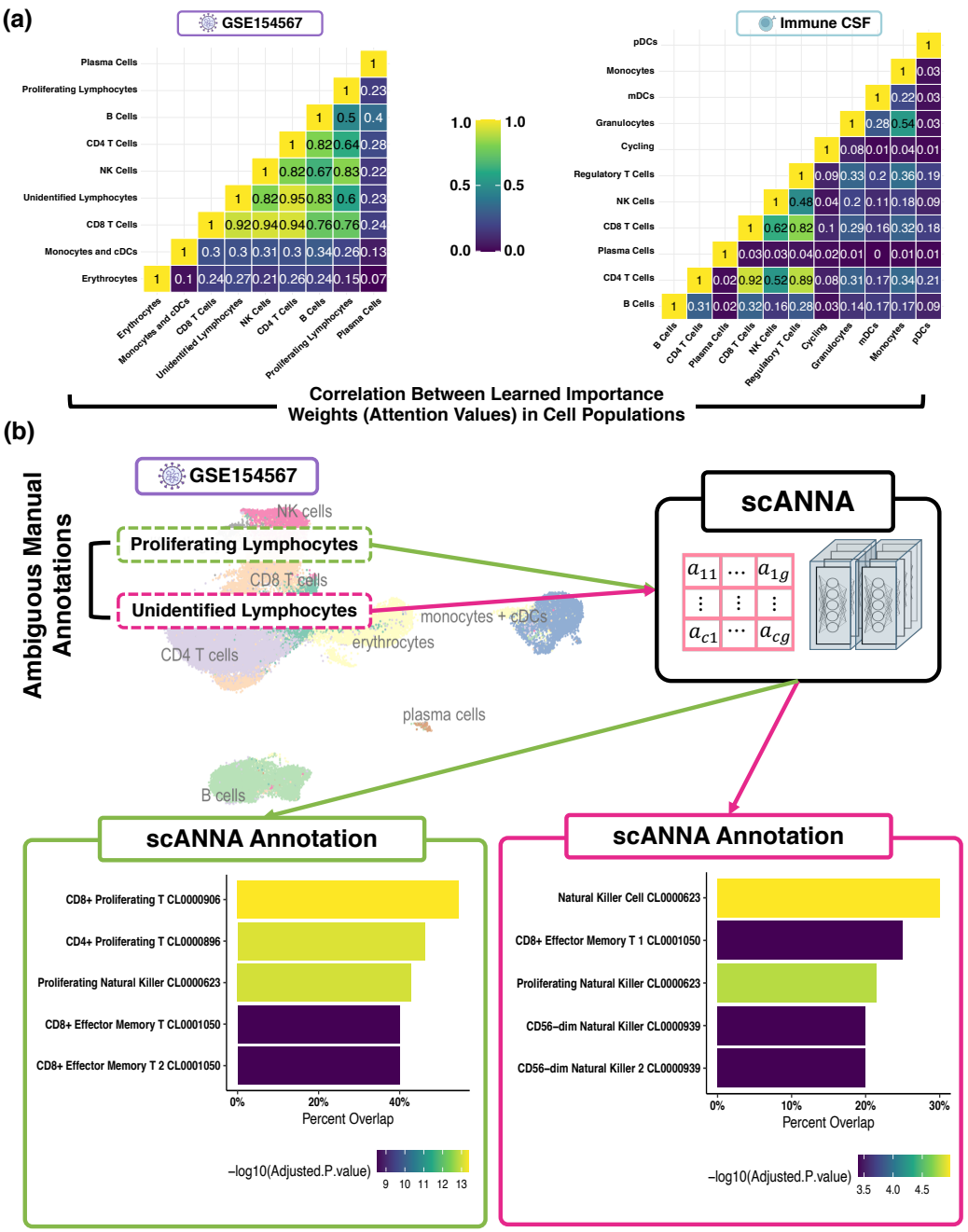


Figure 4.8: (Caption on next page)

Figure 4.8: **Utility of scANNA for Unsupervised Annotations**. **(A)** scANNA learns complex relations between genes across cells. Here, we show the correlation between attention values (learned importance weights) among different cell types in two datasets. The learned gene associations in similar cell types (*e.g.* , CD8 and CD4 cells) have the highest correlations. **(B)** Enrichment analysis identified significant populations in Proliferating Lymphocytes, with CD8+ proliferating and effector memory T cells being the most prominent, while Unidentified Lymphocytes were mainly composed of natural killer cells, along with some CD8+ effector memory T cells.

Our results (Fig. 4.1**(D)**, Fig. 4.8 **(A)**, and 4.3) show that scANNA learns salient genes for each cluster enabling accurate and scalable unsupervised annotations without prior knowledge or expertise. Since many manual annotations are performed with only a few marker genes[257], it is possible to define broad and ambiguous populations; for example, two populations of GSE154567 data (see Data Description and Data Availability) annotated as Proliferating Lymphocytes and Unidentified Lymphocytes. We disambiguate these broad annotations using scANNAs gene scores without re-clustering or further complex analyses. Our enrichment analysis for Proliferating Lymphocytes resulted in multiple significant populations, with CD8+ proliferating and effector memory T cells as the most prominent populations (Fig. 4.8 **(B)**). Enrichment analysis for Unidentified Lymphocytes yielded primarily natural killer cells with some CD8+ Effector Memory T cells (Fig. 4.8 **(B)**). These results signify scANNAs utility for unsupervised annotation and the applicability of our framework in tandem with other annotation forms to provide interpretability and validation, and scANNA performs well even with difficult-to-assign mixed populations.

Transfer Learning: Lastly, we considered the application of scANNA to transfer learning, the accurate and robust annotation transfer of knowledge from a larger reference set to a smaller query set. Linear machine learning methods have had broad applicability due to their interpretability[258]. In contrast, existing DL approaches for transfer learning (TL) on scRNSseq data typically require complex transformations prior to fine-tuning or fall short when the query dataset contains a different number of cell types than the source. To study scANNAs ability to perform TL, we pretrained scANNA to predict cell types on two previously annotated atlases, (i)T cell atlas[253] with 96750 cells and (ii) pancreatic ductal adenocarcinoma[252] (PDAC) and 136807 cells. For target queries, we selected (i) a more specific CD8 T cell study (GSE199565)[254] with 29616 cells and (ii) a specific PDAC study (GSE154778) with 57530 (see Data Description and Data Availability).

To simulate small-scale studies and to test TL capabilities, we fine-tuned scAN-

NAs Configuration Module on only 20% of target datasets (5911 and 11520 cells) and used 80% for testing (23705 and 46020 cells). We also analyzed scANNA performance on out-of-distribution data without fine-tuning. We compared our approach against the current state-of-the-art TL models, scArches[234] and scNym[237]. To evaluate the importance of scANNAs architecture, we modified ACTINN for TL (ACTINN-TL; §4.2) and used this as an additional benchmarking method. Our results (4.1**(E)**) show that fine-tuned scANNA achieves higher accuracy than the other tested methods, performing within 10% F1 score of the best model (supervised classifier trained on 80% data [as opposed to 20%], denoted as Best Case in 4.1**(E)**), showing tremendous potential for TL tasks. ScANNAs improvement over ACTINN-TL showcases the strength of attention in identifying salient genes and supporting transfer learning. ScANNAs enhanced transfer learning can potentially improve the accuracy and robustness of small-scale single-cell analysis.

UTILITY OF SCANNA FOR DISAMBIGUATION OF BROAD ANNOTATIONS

Given that many manual annotations are typically performed with only a few genes (marker genes)[256], it is possible to have broad and ambiguous populations. This was the case with two populations of GSE154567 data (§4.3): The original annotations Proliferating Lymphocytes and Unidentified Lymphocytes. (Fig. 4.8 **(B)**). We aimed to utilize scANNA to disambiguate these broad annotations without re-clustering or performing other complex analyses. Therefore, we investigated the two broad annotations in GSE154567 data and queried our top 50 attentive genes from the Azimuth Cell Type 2021 database33 using Enrichr R Package (`v3.0`)[259,260,261] to perform enrichment analysis.

Our enrichment analysis for proliferating lymphocytes resulted in multiple significant populations, with the most prominent type being CD8+ proliferating T cells (4.8 **(B)**). Intuitively, these results are expected given this populations quantitative and qualitative similarity with the CD8+ T cell population. The difference in gene overlap percentages of the top three predictions, namely CD8+ proliferating T, CD4+ proliferating T, and proliferating natural killer cells, is very small due to the similar lineage and function of these populations, which may explain why the original annotations were left broadly as proliferating lymphocytes. Enrichment analysis for unidentified lymphocytes yielded natural killer cells as the most probable cell type, with other viable populations also being statistically significant. Similar to the previous population, these results are intuitive since CD8+ T cells and natural killer cells are both lymphocyte subsets and proliferate. There is extensive overlap in the gene sets between CD8+ T cells and natural killer cells due to their similar lineage and function. Lastly, we

note that the second enriched population based on attentive genes is CD8+ effector memory T, suggesting that the unidentified lymphocyte population could be refined into two or more populations. These results further signify the utility of scANNA for unsupervised annotation and the applicability of our framework in tandem with other annotation forms to provide interpretability and validation and the ability to identify functional linage differences between similar cell subsets that can be exploited in experimental studies.

4.5. DISCUSSION AND CONCLUSION

We presented scANNA, a DL model employing a novel core that leverages simple neural attention for multiple scRNAseq analyses. We show that scANNA performs comparably or better than state-of-the-art methods designed and trained for specific standard scRNAseq tasks (global features selections, unsupervised annotation, and transfer learning), even though scANNA was not trained for these tasks explicitly. We have demonstrated scANNAs effectiveness at identifying cell types in a supervised and unsupervised manner, reducing subjectivity while minimizing the time for annotating scRNAseq datasets. Our results indicate that leveraging attention in global feature selection outperforms current methods in identifying features that explain the most variance in a dataset. Lastly, we demonstrated that scANNA outperforms state-of-the-art models in TL tasks, highlighting the importance of interpretable and generalized models in the scRNAseq space. Moreover, scANNAs unique flexibility extends beyond using cell-type pseudo-labels (as we did in this work), enabling the utilization of other groupings for learning the desired gene associations, such as leveraging hierarchies for trajectory inference.

Our results with scANNA demonstrate the potential of attention-based DL methods to offer significant improvement compared to traditional DL methods when applied to scRNAseq. Researchers can replace task-specific DL pipelines with scANNA and potentially other interpretable DL methods that are likely to emerge. As attention-based DL methods are not yet widely used in scRNAseq analyses, our work suggests significant discoveries may yet exist in the abundance of existing public scRNAseq datasets. Finally, we note that scANNAs architecture can be generalized to design further interpretable models for scRNAseq, including spatial transcriptomics. We envision using attention layers in DL models will allow the field to transition from single-purpose DL models designed for specific tasks to interpretable multi-tools capable of enabling researchers without extensive computational training to accelerate the discovery process.

Part III

Systems with Limited Observations and Assumptions

How seldom we weigh our neighbor in the same balance with ourselves!

Thomas a Kempis

5

Biological Representation Learning with PDE-Inspired Graph Neural Networks

5.1. INTRODUCTION

In the era of precision medicine and the ever-expanding breadth of biological data, applying advanced computational techniques to decipher complex biological processes has become indispensable[262,263]. Among the myriad data types and modalities, single-cell RNA sequencing (scRNAseq) data stands out as a powerful tool for exploring cellular heterogeneity and unraveling the intricacies of gene regulation at a single-cell resolution. However, the effective analysis of scRNAseq data poses unique computational challenges due to its inherent high-dimensionality, sparse structure, and innate variability. Moreover, many real-world scRNAseq studies suffer from data scarcity or limited reliable knowledge of the system due to factors such as the high cost and complexity of scRNAseq experiments, the need for specialized equipment, and other technical challenges in sample preparation. Consequently, researchers often grapple with small sample sizes and sparse datasets, which can impede the robustness and generalizability of analytical models. Addressing the challenge of data limitation is essential for unlocking the full potential of scRNAseq studies. These challenges necessitate the development of innovative mathematical frameworks to unveil hidden biological patterns and interactions from such data.

Graphs, representing a collection of nodes and edges, are fundamental in capturing relational information present in various domains, including social networks, molecular structures, citation networks, and knowledge graphs. Biological data often follow a graph structure, with the agents (*e.g.* cells) forming the nodes and the interaction between them (*e.g.* signaling pathways) forming the edges[264]. To analyze and process such graph structures, Graph Neural Networks (GNNs)[265,266] were developed to harness the rich interconnectedness and relationship between the nodes and edges. Each vertex in a graph encapsulates distinct features. GNNs employ iterative information propagation mechanisms, allowing nodes to refine their representations based on their inherent attributes and information exchanged with neighboring nodes[166]. Through successive iterations, GNNs aggregate and fuse information from the local neighborhood of each node, progressively enriching the node representations with an evolving understanding of the global graph structure[267]. This iterative neighborhood-aware processing makes GNNs proficient in discerning complex patterns, uncovering latent features, and making informed predictions on various tasks ranging from node classification and link prediction to entire graph property inference.

GNNs aim to use the information about nodes and edges of a graph to produce meaningful representations in the Euclidean space. GNNs represent a crucial embodiment of this idea, as they are artificial neural networks tailored for data with inherent graph structures. GNNs fundamentally aim to transform features associated with each node in the graph and subsequently combine these transformations, often employing various aggregation schemes such as averaging. Very often, the difference btween GNN architectures lie in their approach to transforming node features and aggregating information from neighboring nodes[267]. Such versatility allows GNNs to be highly adaptable to diverse domains, providing a flexible framework to model and analyze complex interactions within graph-structured data. Recently, researchers have proposed GNN architectures for analyzing biological networks and, more specifically, for studying scRNAseq data.

In this work, our approach is inspired by two general concepts. First, we aim to mimic how humans make predictions: humans tend to rely on their past experiences, making assessments based on previous observed similarities. Recent studies have shown that learning similarity between agents in a system (*e.g.* by learning a similarity metric in an embedded space) can significantly improve patient representation and downstream tasks[14]. However, such models rely on a large number of similar samples to adequately learn their similarities, thus not being feasible for rare or underrepresented conditions[268]. In our approach, we tackle the notion of similarity in scRNAseq by constructing a *single cell graph*, where the nodes represent each cell's features (*i.e.* gene expression), and the edges express similarity or relatedness between

the cells. Specifically, we leverage a simple nearest-neighbor kernel for learning similarities between cells, resulting in a sparse adjacency matrix. Our second inspiration is to prescribe dynamics on the data manifold, which the model must follow. These dynamics may be different based on the desired task at hand. For example, if we want to classify different classes, we can prescribe samples to *diffuse* or *advect* on the latent manifold based on their similarity with other data points (depicted in Fig. 5.1). This idea serves as the main motivation for our methodology.

This chapter of my dissertation proposes a novel GNN containing PDE-inspired recursions. More specifically, we discretize a drift-diffusion (also called advection diffusion) PDE on a graph to find an update rule based on time discretization, thus constructing the layers of GNNs. This construction allows us to prove valuable properties about the network and conditions on its convergence, enabling us to design the neural architecture based on the mathematical guarantees. In contrast to most conventional GNNs, our model does not suffer from over-smoothing even as we increase the number of layers (*i.e.* forming a deep network). To demonstrate the versatility and effectiveness of our proposed approach, we conducted extensive experiments on standard benchmarking datasets and, notably, scRNAseq studies. Our results showcase that our method consistently outperforms or is on par with the existing state-of-the-art models across all test cases, underscoring its potential for advancing future research in graph-based machine learning and computational biology.

5.2. BACKGROUND AND RELATED WORK

Neural Partial Differential Equations: Recent years have seen an increased interest in using neural networks for solving differential equations, spearheaded by Neural Ordinary Differential Equations[269] and Physics-Informed Neural Networks[270] for solving PDEs. These breakthroughs have primarily focused on leveraging neural networks to solve various differential equations, most prominently PDEs. Numerous innovative approaches have emerged to enhance and extend these foundational concepts as the field has evolved. Among these advancements, integrating GNNs into the realm of differential equations[271,272] is noteworthy. It's important to distinguish our work from these methods: while these approaches employ neural networks to solve PDEs directly, our novel approach involves embedding dynamics, specifically PDEs, onto a data manifold to design an architecture based on GNNs. This unique perspective marks a departure from traditional paradigms and holds the promise of opening new avenues for solving complex mathematical and computational problems. We next discuss techniques that are more closely related to our work.

GNNs for Data Limited Regimes: The most common type of GNNs for semi-supervised

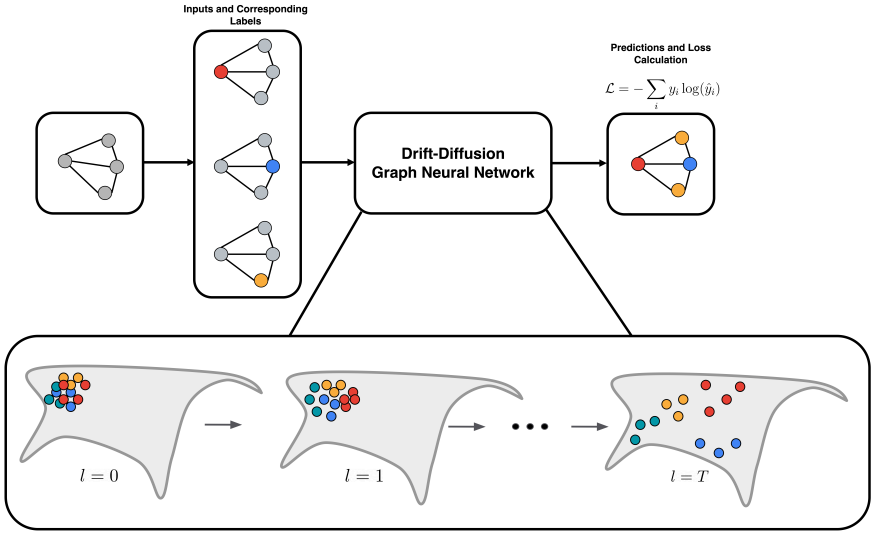


Figure 5.1: **Illustration of our proposed Diffusion-Drift GNN.** Here, we present an illustration of how our model propagates through the layers (denoted by l) the dynamics on the latent manifold for classification.

node classification and edge prediction is the Graph Convolution Network (GCN)[166], which defines a simple convolution to aggregate information from a neighborhood to compute an embedding for each node. GCNs can be generally divided into spectral[166,273,274] or spatial methods[275,276], most of which can be viewed as a Message Passing Neural Networks[277]. Given a feature matrix X with its adjacency matrix A and degree matrix D, each layer of a GCN can be written as:

$$GCN(X,A) = \sigma(\tilde{A}XW) \tag{5.1}$$

where $\tilde{A} = D^{-1/2}AD^{-1/2}$ denotes the symmetrically normalized adjacency matrix, and $\sigma(\cdot)$ denotes a pointwise activation function (*e.g.* sigmoid or ReLU), with the learnable weights W. Though GCN-based techniques have changed how we learn from and process graph-structured data, they suffer from two main limitations: First, GNNs consisting of GCN layers do not provide any theoretical guarantees for design choices or convergence criteria, an issue that has been attributed to GCN's lack of consistent performance across different datasets[278]. The second issue is that GNNs consisting of GCN layers *cannot be deep* due to the well-known *oversmoothing* issue, where the node representations become almost identical across all vertices, thus significantly hindering performance[279,280]. In this work, we aim to alleviate the mentioned issues

for learning biological representations with GNNs through constructing layers that propagate a dynamic on the latent manifold.

PDE-Inspired GNNs: Though the connection between continuous manifolds and graphs has been known, the association of partial differential equations (PDEs) and GNNs were uncovered and utilized more recently, prominently in Chamberlain *et al.* (GRAND)[281] and Eliasof et al. (PDE-GCN)[278]. Similar to our approach, these studies present GNNs as discrete PDEs, particularly the diffusion equation*. Building on these methods, our work introduces a GNN based on the drift-diffusion equation, which requires different (and more stringent) stability conditions, as we prove in §5.3, and can alleviate the common issues with GNNs such as oversmoothing[279,280] and bottlenecks[282]. Moreover, due to our PDE formulation, the layer propagation introduced in our work differs from the existing approaches, aiming to provide additional flexibility for training and discretization. For example, our approach models the drift-diffusion process explicitly in contrast to GRAND, which uses multi-headed attention[160] to model diffusivity. Additionally, the main focus of our work is modeling single-cell data, which requires careful construction of cell graphs.

GNNs for Single-Cell RNA Sequencing Data: As more challenges of scRNAseq datasets have come to light, such as relatively few and sparse observations with thousands of genes (large feature spaces), researchers have sought to leverage GNNs to improve existing analysis pipelines. GNNs have shown promising results in imputing missing expression values, unsupervised clustering, and predicting cell fate. One of the most popular and successful GNN-based algorithms for scRNA-seq analysis is scGNN[283], which we use in this chapter for comparison with our approach. ScGNN is a novel framework that integrates GNN and a left-truncated mixture Gaussian model for learning meaningful representations of scRNAseq. ScGNN consists of three iterative multi-modal autoencoders. Each autoencoder contains a graph neural network encoder and a graph neural network decoder. The encoder learns to represent each cell as a vector of features that capture its gene expression and its relationships with other cells. The decoder then learns to reconstruct the original gene expression of the cell from its encoded representation. The left-truncated mixture Gaussian model is used to model the distribution of gene expression values. This model is well-suited for scRNA-seq data, which is often characterized by a large number of zero values. ScGNN is trained to minimize the reconstruction error between the original and reconstructed gene expressions. This forces scGNN to learn representations of cells that capture both their gene expression and their relationships with other cells, which can

*While Chamberlain et al. only considers the diffusion equation in GRAND, PDE-GCN introduces a convex combination of diffusion and wave equations where the coefficient is learned during training

provide valuable insights when used for downstream tasks. One drawback of scGNN is the use of GCN layers, potentially making the model suffer from the same issues as traditional GCNs. To show the utility of our approach and potential improvements to scGNN (and other similar models), we utilize scGNN's training scheme but change the GCN layers with various GNN architectures, including our proposed DDGNN layers.

5.3. DRIFT-DIFFUSION GRAPH NEURAL NETWORKS

5.3.1. PROPOSED DRIFT-DIFFUSION EQUATION

We will first start with the continuous form to describe the Drift-Diffusion Partial Differential Equation on a graph. On a general continuous manifold $\mathscr{M}$, a system governed by drift-diffusion is described using (5.2),

$$u_t = \nabla \cdot (D\nabla u) - \nabla \cdot (Vu) + R, \quad u(t=0) = u_0\ ,\ 0 \leq t \leq L, \tag{5.2}$$

where D denotes diffusivity, R is a source or sink, and V describes a velocity field in which quantity u moves (*e.g.* particles moving through an electromagnetic field). In our application of learning patient representations, we assume $R = 0$ and assume that V has zero divergence since individual samples will be incompressible[†]. Therefore, we can simplify Eq. (5.2) to be:

$$u_t = \nabla \cdot (D\nabla u) - V \cdot \nabla u, \quad u(t=0) = u_0\ ,\ 0 \leq t \leq L. \tag{5.3}$$

Equivalently, using the same initial conditions, we can express (5.2) as shown in Eq. (5.4):

$$u_t = \nabla \cdot \tilde{D}^* \sigma(\tilde{D}\nabla u) - \tilde{V}^* \psi(\tilde{V} \cdot \nabla u). \tag{5.4}$$

where $\sigma(\cdot)$ denotes a pointwise activation function. To show the stability of our formulation (*i.e.* the PDE is well-behaved), following the approach in Ruthotto *et al.*[284], we prove the conditions that yield non-decreasing energy of the Eq. (5.4).

Theorem 5.3.1. *Let $\sigma(\cdot)$ and $\psi(\cdot)$ be monotonic non-decreasing element-wise sign preserving functions which satisfy $\{\sigma(x), \psi(x)\} \leq x$, for $x \in \mathbb{R}$, with potentially $\sigma = \psi$. Then, the energy of the system described by* (5.4) *with homogeneous Dirichlet boundary conditions is non-increasing.*

[†]The assumption of zero divergence for the fluid is often the case in fluid dynamics as well, where V is assumed to describe the vector field for an incompressible fluid

Proof. Let $\partial\Omega$ denote the boundary of Ω. We want to show that the energy of the system, *i.e.*

$$E = \frac{1}{2}\int_{\partial\Omega} u^2 dS,$$

is non-decreasing throughout time. Therefore, we have that

$$2\frac{dE}{dt} = \int_{\partial\Omega} u \cdot \frac{\partial u}{\partial t} dS = \int_{\partial\Omega} u\left(\nabla \cdot \tilde{D}^* \sigma(\tilde{D}\nabla u) - \tilde{V}^* \psi(\tilde{V}\nabla u)\right) dS$$

$$\frac{\partial}{\partial t}\|u\|^2 = \int_{\partial\Omega} u\left(\nabla \cdot \tilde{D}^* \sigma(\tilde{D}\nabla u)\right) dS - \int_{\partial\Omega} u\left(\tilde{V}^* \psi(\tilde{V}\nabla u)\right) dS$$

From the specifications of σ and ψ, we have $\sigma(\cdot) \leq I(\cdot)$ and $\psi(\cdot) \leq I(\cdot)$ where $I(\cdot)$ is the linear activation (identity mapping).[‡]

Therefore

$$\begin{aligned}
\frac{\partial}{\partial t}\|u\|^2 &\leq \int_{\partial\Omega} u\left(\nabla \cdot \nabla u\right) dS - \int_{\partial\Omega} u\left(\nabla u\right) dS \\
&\leq u\left(\nabla u\right)\Big|_{\partial\Omega} - \int_{\partial\Omega} \left(\nabla u\right)^2 dS - \frac{1}{2}u^2\Big|_{\partial\Omega} \\
&\leq -\int_{\partial\Omega} \left(\nabla u\right)^2 dS \leq 0
\end{aligned}$$

□

In the next theorem, we show that more general activation functions for σ could be used, while only one family of activation functions is permissible for ψ.

Theorem 5.3.2. *Let $\sigma(\cdot)$ and $\psi(\cdot)$ be monotonic non-decreasing element-wise vector functions, where $\sigma(\cdot)$ being sign-preserving and $\psi(\cdot) \leq x$, for $x \in \mathbb{R}$. Then the energy of the partial differential equation system defined in* (5.4) *with homogeneous Dirichlet boundary conditions is non-increasing.*

[‡]*e.g.* Leaky ReLU, $f(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha x & \text{otherwise} \end{cases}$, with $\alpha < -1$.

Proof. Let $< \cdot, \cdot >$ denote the inner product. Similar to the previous proof, we have:

$$\begin{aligned}\frac{\partial}{\partial t}\|u\|^2 &= \left\langle u, \nabla \cdot \tilde{D}^* \sigma(\tilde{D}\nabla u)\right\rangle - \left\langle u, \tilde{V}^* \psi(\tilde{V}\nabla u)\right\rangle \\ &= -\left\langle \nabla u, \tilde{D}^* \sigma(\tilde{D}\nabla u)\right\rangle - \left\langle u, \tilde{V}^* \psi(\tilde{V}\nabla u)\right\rangle \\ &= -\left\langle \tilde{D}\nabla u, \sigma(\tilde{D}\nabla u)\right\rangle - \left\langle \tilde{V}u, \psi(\tilde{V}\nabla u)\right\rangle\end{aligned}$$

Given the sign-preserving specifications of σ and ψ, we have:

$$sign(\tilde{D}\nabla u) = sign(\sigma((\tilde{D}\nabla u))$$

Therefore, we have that:

$$\begin{aligned}\frac{\partial}{\partial t}\|u\|^2 &= -\left\langle \tilde{D}\nabla u, \sigma(\tilde{D}\nabla u)\right\rangle - \left\langle \tilde{V}u, \psi(\tilde{V}\cdot\nabla u)\right\rangle \\ &\leq 0 - \left\langle \tilde{V}u, \psi(\tilde{V}\cdot\nabla u)\right\rangle \leq 0 - \left\langle \tilde{V}u, \tilde{V}\cdot\nabla u\right\rangle \\ &\leq 0\end{aligned}$$

□

Theorem 5.3.2 shows that a recursive algorithm based on the proposed PDE in Eq. (5.4) will be stable. Using this, we formulate the specifics of Diffusion-Drift GNN in the next section.

5.3.2. DIFFUSION-DRIFT INSPIRED GRAPH NEURAL NETWORK

Graphs are considered to be the discretization of continuous manifolds[285,278]. We define a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ to approximate $\mathscr{M}$, and let $\mathbf{u}$ to be the discretized realization of u on $\mathscr{G}$. Therefore, to approximate the solution to Eq. (5.4), we formulate a recursion on a graph (which describes the operations for each layer) based on the discretization of the equation. Following graph theory conventions, we define the discretization of ∇, $\mathbf{B}$, to be the *incidence matrix*, which for two connected nodes i and j is defined as:

$$(\mathbf{Bu})_{ij} = W_{ij}(\mathbf{u}_i - \mathbf{u}_j) \tag{5.5}$$

where W_{ij} is an edge weight matrix, and $\mathbf{u}_{\{i,j\}}$ denotes the features on each node, respectively. Note that the edge weights in W can either be learned or set to a quantity that would directly approximate the true gradients; for example, using the inverse distances between two nodes, *i.e.* $W_{ij} = d_{ij}^{-1}I$, will provide the second-order approxi-

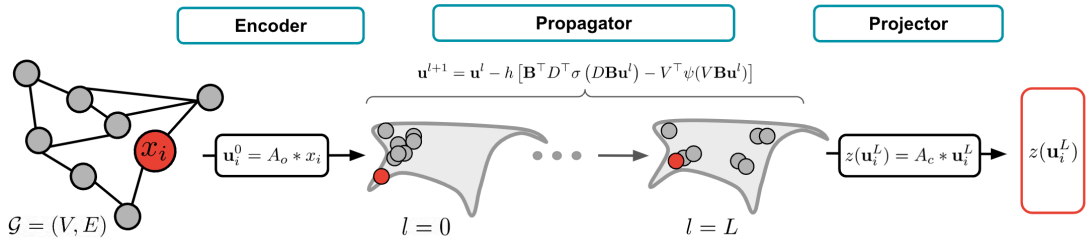


Figure 5.2: **Illustration of the architecture and propagation scheme of the proposed Diffusion-Drift GNN.** Here, we show the process of generating a continuous embedding $z(G)$ for a graph input G. We provide additional details of the network architecture in §5.3.3.

mation to the true gradients. Given the discrete gradient operator, we can define the discrete Laplacian operator (Kirchhoff matrix) as $\Delta \sim B^\top B$.

Given the discretization of the gradients and divergence, we can now formulate the recursion describing the operations in each layer of the DDGNN. Using the Euler discretization, we have:

$$\mathbf{u}^{l+1} = \mathbf{u}^l - h\left[\mathbf{B}^\top D^\top \sigma\left(D\mathbf{B}\mathbf{u}^l\right) - V^\top \psi(V\mathbf{B}\mathbf{u}^l)\right] \tag{5.6}$$

Eq. (5.6) defines the recursion that our model aims to propagate in each layer. Note that the dynamics are prescribed on the hidden space and do not follow a physical intuition in the original data manifold. We present an illustration of the proposed DDGNN in Fig. 5.1 and 5.2.

5.3.3. ARCHITECTURE OF DRIFT-DIFFUSION GRAPH NEURAL NETWORK

Given a graph $G = (V, E)$, with each node v_i having features x_i, our model starts by mapping the input features to the latent input at layer $l = 0$, i.e. $\mathbf{u}_i^0 = A_0 * x_i$, where A_0 is a learnable set of weights. We chose the dimensions of Q_0 so that $\mathbf{u}_i^0 \in \mathbb{R}^{128}$. Then, given a fixed number of layers L, our model advances the solution forward in time according to Eq. (5.6). We note that in our model, the graph's adjacency matrix is used in computing the discrete gradients, which can be viewed as a weighted directional derivative as formulated in Eq. (5.5). Moreover, h denotes the time step of the scheme (Eq. (5.5)), which is constrained by the CourantFriedrichsLewy (CFL) condition. We chose h based on a random grid search for $h = \{0.01, 0.05, 0.1, 0.2, 0.5\}$, finding that $h = 0.2$ provides the best results on the Cora dataset (described in §5.4). For the num-

ber of layers L, we perform an ablation study to investigate over-smoothing (as presented in §5.5), finding that our model performs similarly as depth (*i.e.* number of layers L) increases. However, for computational efficiency, we chose $T = 8$ layers (unless otherwise noted). Due to our mathematical guarantees proved in theorems 5.3.2, we let $\sigma(\cdot) = \tanh(\cdot)$ and $\psi(\cdot) = \text{LeakyReLU}(\cdot)$. Lastly, we pass the hidden representation $\mathbf{u}^L$ into a projector $z(\mathbf{u}_i^L) = A_c * \mathbf{u}_i^L$, where A_c is a tensor of learnable weights, resulting in the final continuous representation of node v_i as $z(\mathbf{u}_i^L)$, which is used in the downstream tasks. The dimension of A_c is chosen so that the final representation is a 64-dimensional vector.

To classify the nodes, we add a softmax layer after producing $z(\mathbf{u}_i^L)$, which assigns a probability to each class. The outputs of the softmax layer are used to optimize a standard cross-entropy loss. We then identify the class with the highest probability as the predicted class for v_i. We train our model for 1000 epochs with Adam[286] optimizer, leveraging early stopping if model performance on validation data does not improve in 20 epochs. We aim to reconstruct the input adjacency matrix for unsupervised representation learning after pruning the graph (refer to Wang et al.[283] for description). Therefore, we leverage scGNN's decoder module instead of the classification layer and use the embeddings before this layer to perform the downstream analyses. Additional details on this experiment are provided in §5.5.

5.4. DATA AND DATA PROCESSING

5.4.1. STANDARD BENCHMARKING DATASETS

To show the utility of our approach on general use cases, we test our method on the standard benchmark datasets for evaluating node classification accuracy of GNNs, namely *Cora, Pubmed*, and *CiteSeer*:
The **Cora dataset** contains 2708 scientific publications classified into one of seven classes, with the citation network containing 5429 links. Each publication in the dataset is described by a 0- or 1-valued word vector indicating the absence or presence of the corresponding word from the dictionary. The dictionary contains 1433 unique words.

The **Pubmed data** consists of 19717 scientific publications from the PubMed database of diabetes classified into one of three classes. The citation network consists of 44338 links. Each publication in the dataset is described by a Term Frequency - Inverse Document Frequency weighted word vector from a dictionary of size 500.

The **CiteSeer dataset** comprises 3327 scientific classified into one of six classes. The citation network consists of 44338 links. Each publication in the dataset is described by a 0- or 1-valued word vector indicating the absence or presence of the correspond-

ing word from the dictionary of 1433 unique words.

5.4.2. SINGLE CELL RNA SEQUENCING DATA

Given our true intention of developing a model for data-limited studies, we aimed to choose scRNAseq data with few observations (cells). Therefore, we use the following three datasets (as used by Wang et al.[283] for scGNN): (1) **Chung data**[287], a transcriptome profiling study of breast cancer tissues from 11 patients, with only 515 cells, (2) **Klein et al.**[288] data containing expression profile of about 10000 cells, and (3) **Zeisel et al.**[289] dataset which contains scRNAseq profiling of the primary somatosensory cortex (S1) and the hippocampal CA1 region, resulting in 9 distinct clusters based on 3005 single-cell transcriptomes.

We followed the steps described in Wang et al.[283] for preprocessing. That is, for each scRNAseq data, we kept genes that had non-zero expression in more than 1% of cells, and cells that had more than 1% non-zero genes. After filtering, we selected the top 2000 highly variable genes[224] as feature inputs and log-transformed the filtered expression profiles. After preprocessing, we computed the single-cell graph as input to graph-based models. The graph construction step is described next.

5.4.3. CONSTRUCTING SINGLE CELL GRAPHS

The cells and their interactions with their neighbors can be used to construct a graph. Following previous approaches, we construct the cell graph utilizing a k-nearest neighbors (KNN) graph framework, where each node represents an individual single cell, and edges signify relationships between these cells. The number of neighbors, k, is a set hyperparameter that sets each cell's connectivity with its neighbors. To calculate the weights of each edge, we utilize the local connectivity within each neighborhood, as described in McInnes et al.[241].

5.5. RESULTS

5.5.1. DEPTH AND OVERSMOOTHING ANALYSIS

To evaluate our mathematical intuition and validate our mathematical guarantees, we set out to test the effect of increasing the number of layers in our model (*i.e.* depth). As mentioned in §5.1, GCN-based models suffer from over-smoothing, constraining such models to shallow networks. To test the effect of depth increase, we compared the classification accuracy of our model against GCN[166] on the Cora dataset for varying layers: $l = \{2, 8, 16, 64, 128\}$. Our results, presented in Table 5.1, demon-

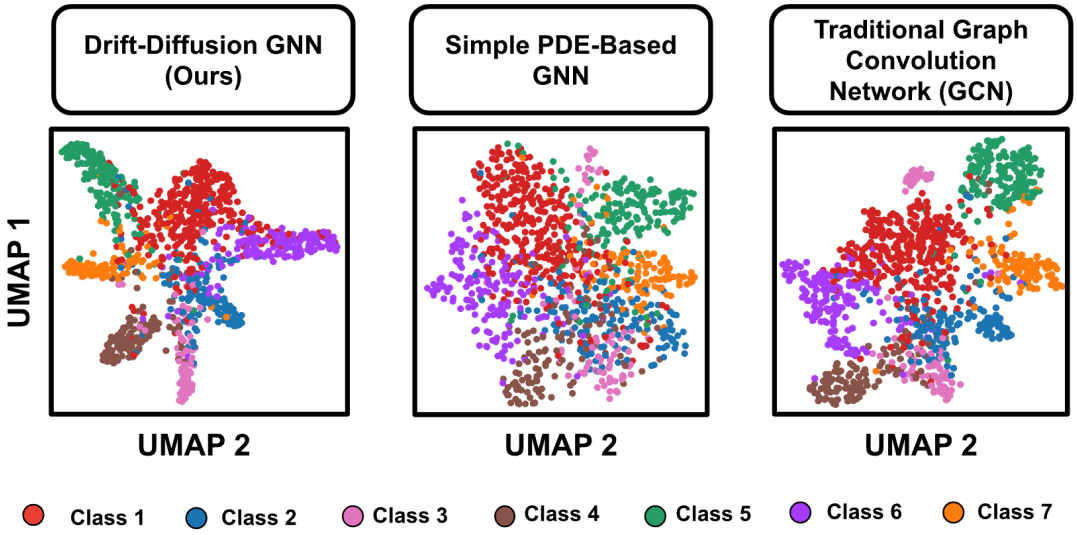


Figure 5.3: **A qualitative comparison of learned node representations for Cora dataset produced by different approaches**.

strate that GCN's performance decreases significantly as more layers are added. DDGNN's performance, on the other hand, remains consistently high for node classification across different layers (and is slightly improved as more layers are added). A qualitative representation of the embeddings qualities is presented in Fig. 5.3. We attribute this improvement to the formulation of our algorithm, rooted in the PDE dynamics, which serves as an implicit regularizer. This implicit regularization forces the model to learn distinct representations for each node, thereby preserving the model's robustness and the capacity to adapt to an array of layer depths.

5.5.2. DATA LIMITED EXPERIMENT: STANDARD BENCHMARKS

For each dataset, we only consider 20 samples from each class and use the remaining samples in testing. We compare our model against the current state-of-the-art (SOTA) universal methods, namely *GCN*, *GAT*[267], and *PDE-GCN*[278], as well as *sigGCN* and modified scGNN, which are the single-cell specific SOTA models. Our results are presented in Table 5.2.

Table 5.1: **Comparison of node classification accuracy for GCN and DDGNN (ours) when increasing depth.** Results of node classification accuracy (Median Micro F1 score from 5 random runs) on the Cora dataset, demonstrating our model's robustness to depth increase. Boldface values indicate better performance.

Number of Layers	**GCN**	**DDGNN (Ours)**
$l = 2$	0.8024	**0.8232**
$l = 4$	0.8081	**0.8383**
$l = 16$	0.6497	**0.8371**
$l = 64$	0.2282	**0.8397**
$l = 128$	0.1668	**0.8414**

Table 5.2: **Comparison of node classification accuracy for various models in data-poor scenarios on standard benchmarks.** Results of node classification accuracy (Median Micro F1 score from 5 random runs) on standard benchmark datasets. Boldface values indicate better performance. ScRNAseq-specific models were not tested on standard benchmarks since the manuscripts did not test their models' applicability in general settings.

Model Name	**Cora**	**CiteSeer**	**Pubmed**
GCN	80.81	70.5	79.51
GAT	82.7	74.36	68.06
PDE-GCN	81.67	77.59	79.15
DDGNN (Ours)	**83.83**	**79.51**	**81.26**

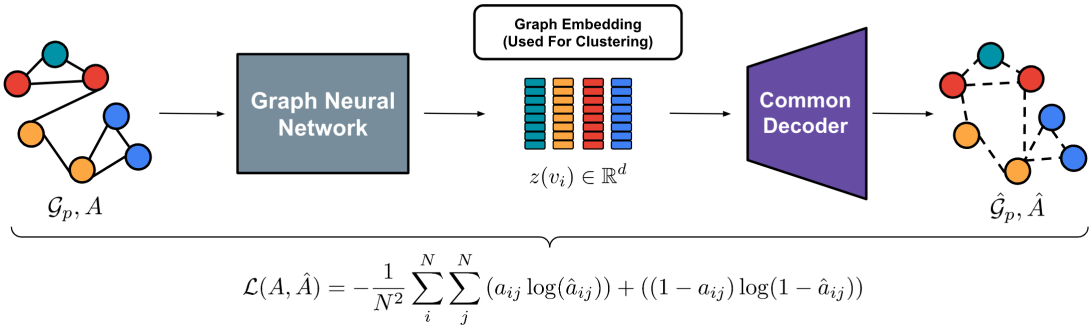


Figure 5.4: **Schematic of training procedure for data-limited scRNAseq experiment described in §5.5.3.** We use our model as well as other state-of-the-art techniques to learn to learn representations in an unsupervised manner. After training, we perform graph-based clustering on the latent representation and measure the clustering accuracy against the ground truth (annotations provided from the actual datasets). Note that the "common decoder" is used for all graph-representation methods to reduce decoding bias when learning the reconstruction of the original graph.

5.5.3. DATA LIMITED EXPERIMENT: SCRNASEQ BENCHMARKS

Following Wang et al.[283], we motivated this experiment by scRNAseq downstream analysis: Given our goal of representation learning in very low data regimes, we train DDGNN and other graph-based neural networks in an *unsupervised* manner. We train all models to produce *embeddings* instead of classification probability. The embeddings generated from all models are then inputted to a decoder (specifically, scGNN's decoder), aiming to reconstruct the adjacency matrix of the input graph. This is done to carefully assess the impact of model architecture and update schemes across various models. The reconstructed adjacency matrices, $\hat{A}$ are used to optimize the loss function introduced by Wang et al.[283], as shown in Eq. (5.7):

$$L(A,\hat{A}) = -\frac{1}{N^2}\sum_{i}^{N}\sum_{j}^{N}\left(a_{ij}\log\left(\hat{a}_{ij}\right) + \left(1-a_{ij}\right)\log\left(1-\hat{a}_{ij}\right)\right) \tag{5.7}$$

where N denotes the number of nodes (cells) $\{a,\hat{a}\}_{ij}$ denote the elements of A or $\hat{A}$, respectively. Since most scRNAseq labels are generated through clustering, we evaluate the accuracy of each model by calculating the Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI) between the true clustering and Louvain clustering performed on the learned embeddings. We depict the training scheme for this

Table 5.3: **Comparison of node clustering accuracy for various models in an unsupervised manner for scRNAseq studies.** Results of clustering latent embeddings of each model (reported median ARI and NMI scores from 5 random runs) on scRNAseq studies with various scales. Boldface values indicate better performance. Our results show that DDGNN-based model outperforms other methods in poor-data scenarios (when the number of observations are very low).

	scRNAseq Studies					
	Chung et al.		**Klein et al.**		**Zeisel et al.**	
Model	*ARI*	*NMI*	*ARI*	*NMI*	*ARI*	*NMI*
GCN	0.5391	0.7146	0.9307	0.8947	0.6401	0.6746
GAT	0.5649	0.7247	0.9271	0.8932	0.6671	0.6895
PDE-GCN	0.5847	0.7588	**0.9347**	**0.9101**	0.6722	0.7052
scGNN	0.5451	0.7214	0.9321	0.9095	0.6513	0.6850
DDGNN (Ours)	**0.6274**	**0.7805**	0.9314	0.8967	**0.7049**	**0.7288**

experiment in Fig. 5.4. As presented in Table 5.3, our results indicate that our model can learn meaningful biological representations for scRNAseq studies, even when the number of observations is limited.

5.6. DISCUSSION AND CONCLUSION

In this work, we introduced the Drift-Diffusion Graph Neural Network (DDGNN), an innovative framework inspired by partial differential equations (PDEs) that offers a universal approach to the design of graph neural networks (GNNs). Our approach capitalizes on the profound connection between PDEs and graphs, utilizing a specific formulation derived from the drift-diffusion equation. When solved on a graph and discretized in time, we obtain a propagation scheme that resembles classical GNNs' update rule. This bridge between PDEs and graph structures is a significant departure from conventional GNN methodologies and underscores the potential for unifying mathematical principles from diverse domains. DDGNN provides an adaptable and versatile framework for modeling complex systems with limited data, paving the way for more insightful analyses and predictions across various disciplines, from computational biology to network analysis, where graph-based representations play a pivotal role in deciphering intricate relationships and patterns.

In our first experiment, we validated our mathematical intuition on our model's

performance when adding more layers. We trained DDGNN on standard benchmark data and showed that increasing the number of layers from 2 to 128 does not hinder model performance, improving the node classification results. In contrast, we found that GCN's performance significantly drops with increased depth, verifying the fact that GCNs must remain shallow. Designing GNNs with the capability to achieve greater depth is paramount to advancing machine learning algorithms for graph-structured data. Building deeper GNN architectures enables these models to capture more intricate and complex relationships within the data, which are often crucial for making accurate predictions and uncovering hidden patterns. Shallow GNNs may struggle to capture complex dependencies that span multiple graph layers, limiting their capacity to learn nuanced representations, especially for biological data. By contrast, deeper GNNs can better exploit the hierarchical structure of data, facilitating the extraction of high-level features and abstractions. Such capability can be leveraged to better explain the complex nature of scRNAseq studies where the expression of thousands of genes depend on one another.

Given the goal of my PhD research and this dissertation, we next applied our methodology to datasets with limited observations, training our model in a semi-supervised and unsupervised manner to test its representation learning. To demonstrate the model's versatility outside of biology, we conducted semi-supervised training with just *20 nodes per class* and evaluated it on a substantial test set of 1000 samples, resembling few-shot learning. Our results showed that DDGNN outperforms existing state-of-the-art methods on these standard benchmark datasets. This highlights our model's adaptability and robustness in situations where data is scarce, making it a valuable tool in broader machine-learning applications.

To evaluate our model's application to biological representation learning, we next trained DDGNN in an unsupervised manner to learn continuous embeddings from scRNAseq data. To demonstrate the value of our PDE-inspired architecture, we followed the training procedure of Wang et al.[283], with the only difference being the GNN architecture. Using the learned embeddings, we performed unsupervised clustering and compared the results with the initial annotations. We found DDGNN to perform significantly better than the competing methods in the low-data experiments (two out of three datasets). These results signify our approach's potential to accurately and robustly identify cell types, even when the number of observations is less than 1000 cells (*e.g.* Chung et al. study[287]). Our models' ability to improve the analysis of such small-scale experiments is particularly noteworthy, as it enables researchers to study rare cell populations or to investigate the effects of perturbations on cell type composition with limited resources. For example, our approach could be used to identify rare cell populations in tumor samples or to study the effects of a new drug on

cell type composition in a small animal model. Our model limitations include longer training time than traditional GCNs and sensitivity to propagation-specific parameters, such as the time step, which could destabilize training if not set correctly. We believe additional improvements are needed to enhance our model's training efficiency and to potentially mitigate the effects of PDE-specific hyperparameters.

Looking ahead, we aim to extend our approach to encompass other single-cell data modalities as well. While this chapter has primarily focused on applying our methodology to scRNAseq data, we believe that considering other single-cell modalities can further enhance the ability of our model to elucidate the underlying biology. Single-cell protein sequencing and single-cell chromatin accessibility profiling are two such data modalities that may improve the analyses and offer insightful perspectives of the system of interest. Expanding our approach to accommodate these diverse data types presents an exciting opportunity to further our understanding of cellular heterogeneity. Another key avenue of exploration for DDGNN is data imputation, where accurate and reliable imputation of missing gene expression values can significantly improve the completeness and quality of scRNAseq studies. Additionally, we plan to leverage DDGNN's capabilities to infer cell-cell interactions, another critical component of understanding the dynamic behaviors of cells within complex biological systems. By applying DDGNN to these important downstream tasks, we anticipate that our research will contribute to developing more comprehensive computational frameworks for scRNAseq data analysis. Such applications will ultimately enable researchers to gain deeper insights into cellular processes, unravel intricate regulatory networks, and further our understanding of the underlying biology in health and disease, even without large-scale observations.

Reality is ultimately a selective act of perception and interpretation.

David Simon

6

Generating Realistic Synthetic Biological Data

6.1. INTRODUCTION

Traditional sequencing methods are limited to measuring the average signal in a group of cells, potentially masking heterogeneity and rare populations[290]. Single-cell RNA sequencing (scRNAseq) technologies allow for amplifying and extracting small RNA quantities, enabling sequencing at a single-cell level[291]. The single-cell resolution thus enhances our understanding of complex biological systems. For example, scRNAseq has been used in immunology to discover new immune cell populations, targets, and relationships, which have been used to propose new treatments[290].

While the number of tools for analyzing scRNAseq data increases, one limiting factor remains the low number of cells, potentially related to financial, ethical, or patient availability[135]. Large, well-funded projects have generated the Human Cell Atlas[292] and the Mouse Cell Atlas[92], which characterized cell populations in organs and tissues in their respective species. Although a tremendous amount of scRNAseq data is available from such projects, they are limited to a broad overview of the cell populations in these tissues and organs. The Atlases overlook sub-populations of smaller, rarer, and important cells in normal and dysregulated states. As Button *et al.* note[293],

small numbers of observations reduce the reproducibility and robustness of experimental results since they may not represent the behavior of the underlying population well. This is especially important for benchmarking new tools for scRNAseq data, as the number of features (genes) in each cell often exceeds the number of samples.

Given limitations on scRNAseq data availability and the importance of adequate sample sizes, *in-silico* data generation and augmentation offers a fast, reliable, and cost-effective solution. Synthetic data augmentation is a standard practice in many fields of machine learning such as text and image classification[294]. Traditional data augmentation techniques, geometric transformations, or noise injection, are being replaced by more recently developed generative models, variational autoencoder (VAE)[188] and Generative Adversarial Networks (GANs)[197], for augmenting complex biological datasets. However, GANs and VAEs remain less explored for data augmentation in genomics and transcriptomics. We briefly overview GANs and VAEs in Section 6.2, with a more detailed review presented in Chapter 3 of this dissertation.

There are many statistical frameworks for generating in-silico scRNAseq data[295,296,297,298,299], but recently Marouf *et al.*[135] introduced the first deep generative models (GAN-based) for scRNAseq data generation and augmentation (called single-cell GAN [scGAN] and conditional scGAN [cscGAN]), and demonstrated that they outperform other state-of-the-art models. While scGAN augments the entire population by creating "holistic" cells, cscGAN is conditioned to generate cells from specific subpopulations.

In this chapter, we will extend and generalize their approach by employing an introspective VAE for data augmentation. The motivation and application of our generative model are closely related to Marouf *et al.*[135], focusing on improving training time, stability, and generation quality using *only one* framework. We compare our proposed model, ACTIVA, with scGAN and cscGAN, showing how it can be leveraged to augment rare populations, improving classification and downstream analysis. In contrast to these previously published GANs, our novel cell-type conditioned introspective VAE model allows us to generate either "holistic" or specific cellular subpopulations in a single framework.

In § 6.2 of this chapter, we provide an overview of GANs (both generally and in the context of scRNAseq data) and VAEs (for more details on the mathematics behind these approaches, refer to Chapter 3). Section 6.3 details ACTIVA, our proposed conditional introspective VAE. Section 6.4 describes our training data and associated processing steps. Section 6.5 compares ACTIVA with competing methods - scGAN and cscGAN. We demonstrate that augmenting rare cell populations with ACTIVA improves classification over GANs while providing a more computationally tractable

framework, mirroring both scGAN and cscGAN in a single model. In comparison with scGAN and cscGAN, ACTIVA generates cells that are harder for classifiers to identify as synthetic (*i.e.* having Areas Under the Curve [AUC] closer to 0.5), with better pair-wise correlation between genes. ACTIVA-generated cells allow for improved classification of rare subtypes (more than 4% improvement over cscGAN), all while reducing run-time by an order of magnitude compared to both models. Finally, we review our approach, findings, and limitations in § 6.6.

6.2. BACKGROUND ON GANS AND VAES

6.2.1. GENERATIVE ADVERSARIAL NETWORKS

GANs[197] are capable of generating realistic synthetic data and have been successfully applied to a wide range of machine learning tasks[198] and bioinformatics[202]. GANs consist of a generator network (G) and a discriminator network (D) that train adversarially, which enables them to produce high-quality fake samples. During training, D learns the difference between real and synthetic samples, while G produces fake data to "fool" D. More specifically, G produces a distribution of generated samples p_g, given an input $z \sim p_z$, with p_z being a random noise distribution. The objective of GANs is to learn p_g, ideally finding a close approximation to the real data distribution p_r, so that $p_g \approx P_r$. To learn the approximation to p_g, GANs play a "min-max game" of $\min_G \max_D \mathbb{E}_{x\sim p_r} \log[D(x)] + \mathbb{E}_{z\sim p_z} \log[1 - D(G(z))]$, where both players ($G$ and D) attempt to maximize their own payoff. This adversarial training is critical in GANs' ability to generate realistic samples. Compared to other generative models, GANs' main advantages are (i) the ability to produce any probability density, (ii) no prior assumptions for training the generator network, and (iii) no restrictions on the size of the latent space.

Despite these advantages, GANs are notoriously hard to train since it is highly non-trivial for G and D to achieve Nash equilibrium[203]. Another disadvantage of GANs is vanishing gradients where an optimal D cannot provide enough information for G to learn and make progress, as shown by Arjovsky *et al.*[204]. Another issue with GANs is "mode collapse," when G has learned to map several noise vectors z to the same output that D classifies as real data. In this scenario, G is over-optimized, and the generated samples lack diversity. Although some variations of GANs have been proposed to alleviate vanishing gradients and mode collapse (*e.g.* Wasserstein-GANs (WGANs)[206] and Unrolled-GANs[207]), the convergence of GANs still remains a major problem. Furthermore, quantifying how well GANs have learned the distribution of real data is often complicated, consisting of measuring the dissimilarity between

p_g and p_r when p_r is not known or assumed. Therefore, common ways of evaluating GANs involve directly evaluating the output[205], which can be arduous. During the training progression, the feedback of D to G becomes meaningless, and if GANs continue to train past this point, the quality of the synthetic samples can be affected and ultimately collapse. Common variations of GANs cannot be trained as single-stream networks, and a necessary step is to define a training schedule for G and D separately, adding another layer of complexity. Although all deep learning models are sensitive to hyperparameter choices, Lucic *et al.*[208] show all experimented GANs (including WGANs) are much more sensitive to these choices than VAEs. This can be a drawback in using GANs for scRNAseq generation since the hyperparameters may need to be re-tuned for every new dataset.

6.2.2. SINGLE CELL GANS

scGAN and cscGAN are state-of-the-art deep learning models for generating and augmenting scRNAseq data. Marouf *et al.*[135] trained scGAN to generate single-cell data from all populations and cscGAN to produce cluster-specific samples, with the underlying model in both being a WGAN. For scGAN, the objective is to minimize Wasserstein distance between real cells distribution, p_r, and generated data, p_g:

$$W(p_r, p_g) = \inf_{\gamma \in \Pi(p_r, p_g)} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\gamma} \|\mathbf{x} - \mathbf{y}\|, \tag{6.1}$$

where $\mathbf{x}$ and $\mathbf{y}$ denote random variables, $\Pi(p_r, p_g)$ is the set of all joint probability distributions $\gamma(\mathbf{x}, \mathbf{y})$ with marginals p_r and p_g. Intuitively, Wasserstein distance is the cost of optimally transporting "masses" from $\mathbf{x}$ to $\mathbf{y}$ such that p_r is transformed to p_g[206]. However, since the infimum in Eq. (6.1) is highly intractable, Arjovsky *et al.*[206] use Kantorovich-Rubinstein duality to find an equivalent formulation of Wasserstein distance with better properties:

$$W(p_r, p_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{\mathbf{x}\sim p_r} \mathbf{f}(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\sim p_g} \mathbf{f}(\mathbf{x}),$$

where the set of 1-Lipschitz functions is denoted by $\|f\|_L \leq 1$, with the solution being a universal approximator (potentially a fully connected neural network) to approximate f. This function is approximated by D, which we denote as f_d. Similarly, f_g denotes the function approximated by the generator. Using these notations, we arrive at the adversarial objective function of WGANs (used in scGAN):

$$\min_{f_g} \max_{\|f_d\|_L \leq 1} \mathbb{E}_{\mathbf{x}\sim p_r} \mathbf{f_d}(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\sim f_g(p_n)} \mathbf{f_d}(\mathbf{x}),$$

where p_n denotes a multivariate noise distribution.

Although WGANs can alleviate the vanishing gradient issue, most of the GANs' training instabilities can still occur, making WGANs less flexible and transferable between different datasets or domain-specific tasks. CscGAN uses a projection-based conditioning[300], which adds an inner product of class labels (cell types) at the discriminator's output. Based on the instruction given by the authors in the implementation, scGAN and cscGAN must be trained separately; however, our model learns to generate specific cell populations (cell types) or collective cell clusters with one training.

6.2.3. VARIATIONAL AUTOENCODERS

VAEs[188] are generative models that jointly learn deep latent-variable and inference models. Specifically, VAEs are autoencoders that use variational inference to reconstruct the original data, having the ability to generate new data that is "similar" to those already in a dataset x. VAEs assume that observed data and latent representation are jointly distributed as $p_\theta(x,z) = p_\theta(x|z)p(z)$. In deep learning, the log-likelihood $p_\theta(x|z)$ is modeled through non-linear transformations, thus making the posterior probability distribution, $p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}$, intractable. Due to the intractability of maximizing the expected log-likelihood of observed data over θ, $\mathbb{E}_{p(x)}[\log \int p_\theta(x,z)dz]$, the goal is to instead maximize the evidence lower bound (ELBO):

$$\underbrace{\mathbb{E}_{q_{\gamma(x)}(z)}\left[\log\left(\frac{p_\theta(x|z)p(z)}{q_{\gamma(x)}(z)}\right)\right]}_{\text{ELBO}(\theta,\gamma)} \leq \log p_\theta(x),$$

where $q_{\gamma(x)}(z)$ is an auxiliary variational distribution (with parameters $\gamma(x)$) that tries to approximate the true posterior $p_\theta(z|x)$. We provide a more complete derivation of VAE's objective in Chapter 3.

The main issue with VAEs arises when the training procedure falls into the trivial local optimum of the ELBO objective; that is, when the variational posterior and the true posterior closely match the prior (or collapse to the prior). This phenomenon often causes issues with data generation since the generative model ignores a subset of latent variables that may have meaningful latent features for inputs[190]. In our experiments, we did not encounter posterior collapse. However, our package provides an option for modifying objective function weights adaptively using SoftAdapt[194]. VAEs

have also been criticized for generating samples that adhere to an average of the data points instead of sharp samples that GANs produce because of adversarial training. This issue has often been addressed by defining an adversarial training between the encoder and the decoder, as done in introspective VAEs (IntroVAEs)[301] which we use in our framework*. IntroVAEs have been used mostly in computer vision, which has performed comparably to their GAN counterparts in applications such as synthetic image generation[301] and single-image super-resolution[192]. We describe the formulations of IntroVAEs in Section 6.3.

VAEs' natural ability to produce both a generative and an inference model presents them as an ideal candidate for generating and augmentation of omics data. In this work, we demonstrate the ability of our deep VAE-based model to produce realistic in-silico scRNAseq data. Our model, ACTIVA, performs comparably to the state-of-the-art GAN models, scGAN and cscGAN, and trains significantly faster and maintains stability. Moreover, ACTIVA learns to generate specific cell types and holistic population data in one training (unlike scGAN and cscGAN, which train separately). Our model trains at least 6 times faster than scGAN on the same datasets and in the same environment. Moreover, ACTIVA can produce 100K samples in less than 2 seconds on a single NVIDIA Tesla V100 and 87 seconds on a common research laptop. ACTIVA provides researchers with a fast, flexible, and reliable deep-learning model for augmenting and enlarging existing datasets, improving the robustness and reproducibility of downstream analyses.

6.3. METHODS

Our proposed model, ACTIVA, consists of three main networks, with a self-evaluating VAE as its core and a cell-type classifier as its conditioner. In this section, we formulate the objective functions of our model and describe the training procedure.

6.3.1. ENCODER NETWORK

The ACTIVA encoder network, *Enc*, serves two purposes: (i) mapping (encoding) scRNAseq data into an approximate posterior to match the assumed prior, and (ii) acting as a discriminator, judging the quality of the generated samples against training data. Therefore, *Enc*'s objective function is designed to train as an adversary of the generator network, resulting in realistic data generation. To approximate the prior distribution, KL divergence is used as a regularization term (denoted as L_{REG}) which regularizes the encoder by forcing the approximate posterior, $q_\phi(z|x)$, to match the

*Here we follow Huang *et al.*[301] and categorize IntroVAEs as a type of VAE.

prior, $p(z)$. We assume a center isotropic multivariate Gaussian prior since it can be reparameterized differently into arbitrary multivariate Gaussian random variables, thus simplifying the inference process[302]. Although $p_\theta(x|z)$ can be parameterized in many ways, we choose an isotropic multivariate Gaussian for simplicity. However, choosing a scRNAseq-specific counts distribution as the conditional likelihood (*e.g.* , zero-inflated negative binomial) may lead to some improvements in the generation process[302].

The posterior probability is $q_\phi(z|x) = \mathcal{N}(z;\mu,\sigma^2)$, where μ and σ are the mean and standard deviation, respectively, computed from the outputs of *Enc*. As in traditional VAEs, z is sampled from $\mathcal{N}(0,I)$, which will be used as an input to the generator network (decoder in VAEs). Due to the stochasticity of z, gradient-based back-propagation becomes difficult, but using the reparameterization trick in Kingma *et al.*[188] makes this operation tractable. That is, define $z = \mu + \sigma \odot \varepsilon$ with $\varepsilon \sim \mathcal{N}(0,I)$ which passes the stochasticity of z onto ε. Now given N cells and a latent vector in a D-dimensional space (*i.e.* $z \in \mathbb{R}^D$), we can compute the KL regularization:

$$L_{REG} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{D}\left(1+\log(\sigma_{ij}^2)+\mu_{ij}^2-\sigma_{ij}^2\right). \tag{6.2}$$

Like traditional VAEs, the encoder network aims to minimize the difference between reconstructed and training cells (real data). We denote the expected negative reconstruction error as L_{AE}, defined as:

$$L_{AE} = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]. \tag{6.3}$$

Following Huang *et al.*[301], we choose the reconstruction loss L_{AE} to be the mean squared error between the training cells and reconstructed cells (more details in Appendix A).

As the last part of the network, we introduce a cell-type loss component that encourages x_r to have the same cell type as x. That is, given a classifying network C, we want to ensure that the identified type of reconstructed sample $C(x_r) = t_r$ is the same as real cell $C(x) = t$; we denote this as L_{CT} shown in Eq. (6.6). We provide additional detail on the conditioning in Section 6.3.3. During the development of ACTIVA, Zheng *et al.*[303] introduced similar conditioning of the IntroVAE framework for image synthesis, which significantly improved the generation of new images. Given our model's objectives, the loss function for *Enc*, $\mathscr{L}_{Enc}$, must encode training data and

self-evaluate newly generated cells from the generator network *Gen*:

$$\mathscr{L}_{Enc} \;=\; L_{REG}(z) \;+\; \alpha_1 \sum_{s=r,g} [m - L_{REG}(z_s)]^+ \;+\; \alpha_2 \left(L_{AE}(x,x_r) + L_{CT}(t,t_r)\right), \quad (6.4)$$

where subscripts r and g denote reconstructed and generated cells from *Gen*, respectively. Note that reconstructed cells x_r correspond directly to training data x, but generated cells x_g are newly produced cells. In Eq. (6.4), $[\cdot]^+ = \max(0,\cdot)$, and $m \in \mathbb{R}^+$ determines our network's adversarial training, as described in Section 6.3.4.

6.3.2. GENERATOR NETWORK

The generator network, *Gen*, aims at learning two tasks. First, *Gen* must learn a mapping of encoded training data, $z \in \mathbb{R}^D$, from the posterior, $q_\phi(z|x)$, back to the original feature space, $\mathbb{R}^M$. In the ideal mapping, reconstructed samples x_r would perfectly match training data x. To encourage learning of this objective, we minimize the mean squared error between x and x_r, as shown in Eq. (6.3), and force cell types of reconstructed samples to match with original cells, as shown in Eq. (6.6). The second task of *Gen* is to generate realistic new samples from a random noise vector $z_n \in \mathbb{R}^D \sim p(z)$ (sampled from the prior $p(z)$) that "fool" the encoder network *Enc*. After producing new synthetic samples x_g, we calculate $Enc(x_g) = z_g$ to judge the quality of generated cells. Given these two objectives, the generator's objective function is defined as

$$\mathscr{L}_{Gen} = \alpha_1 \sum_{s=r,g} L_{REG}\left(Enc(x_s)\right) + \alpha_2 (L_{AE}(x,x_r) + L_{CT}(t,t_r)). \quad (6.5)$$

6.3.3. AUTOMATED CELL TYPE CONDITIONING

Minimizing L_{AE} alone does not enforce our model to generate more cells from the rare populations, so we introduce a cell-type matching objective. This objective aims to encourage the generator to generate cells that are classified as the same type as the input data. More explicitly, the loss component L_{CT} will penalize the network if reconstructed cell types differ from the training data. Given a trained classifier $C(\cdot)$, we can express this objective as

$$L_{CT} = \frac{1}{2} \sum_{i=1}^{N} \|t - t_r\|_F^2 = \frac{1}{2} \sum_{i=1}^{N} \|C(x) - C(x_r)\|_F^2 . \quad (6.6)$$

For ACTIVA's conditioning, we use an automated cell-type identification introduced by Ma *et al.* [133]. This network, called ACTINN, uses all genes to capture features and focuses on the signals associated with cell variance. We chose ACTINN because of its accurate classification and efficiency in training compared to other existing models [304]; we provide an overview of ACTINN in Appendix A. Our model is also flexible to use any classifier as a conditioner, as long as an explicit loss could be computed between predicted and true labels[†]. With ACTINN as the classifier, t and t_r are logits (output layer) for x and x_r, respectively. Our implementation of ACTINN is available as a stand-alone package at `https://github.com/SindiLab/ACTINN-PyTorch`.

6.3.4. ADVERSARIAL TRAINING

The generator produces two types of synthetic cells: reconstructed cells x_r from x and newly generated cells x_g from a noise vector. While both the *Enc* and *Gen* attempt to minimize L_{AE} and L_{CT}, the encoder tries to minimize $L_{REG}(z)$ and maximize $L_{REG}(z_{r,g})$ to be greater than or equal to m. However, the generator tries to minimize $L_{REG}(z_{r,g})$ to minimize its objective function. This is the min-max game played by *Enc* and *Gen*. Note that choosing m is an important step for the network's adversarial training; we describe the strategies for choosing m in A.

6.3.5. ADAPTIVE WEIGHTING OF LOSS COMPONENTS

Our objective functions Eqs. (6.4)-(6.5) have multiple components, which require weights for the linear combination; in our formulation, these weights are denoted by α_1, α_2. A typical way of combining these loss components is to have fixed weights (α_k) that do not change throughout training. However, each part may require a different optimization level based on their performance throughout the training. As mentioned in §6.1, when the L_{REG} term in Eq. (6.4)-(6.5) becomes too close to zero, meaning that the variational distribution collapses towards the prior. For this reason, we use SoftAdapt [194], a family of methods for adaptive weighting multi-part loss functions. SoftAdapt can use live training statistics to prioritize optimization of each part of the loss function, *i.e.* , the parts of the loss function that perform *the poorest* get most of the attention from the optimizer. Using SoftAdapt can also reduce training time for VAE-based models since it can reduce the stability of spurious local minima or maxima [305,194]. This can also improve the quality of training since the loss parts that have reached a minimum and remain at that minimum are no longer considered by the optimizer, placing more importance on the parts that perform poorly. For our

[†]This is true for most classifiers.

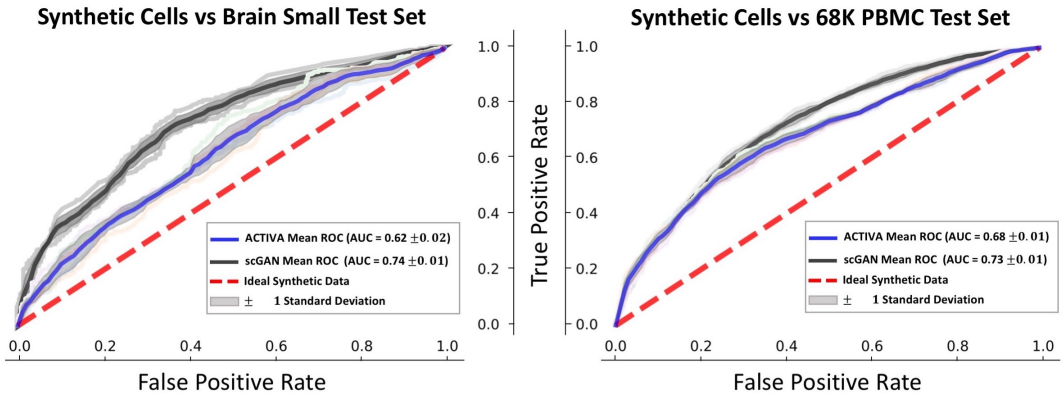


Figure 6.1: **Classifying synthetic data (ACTIVA and scGAN) from real data (test set) for Brain Small (left plot) and 68K PBMC (right plot)**. The metrics are reported using a random forest classifier (detailed in Section 6.5.2) with 5-fold cross-validation (marked by pastel colors in each plot). An area under the curve (AUC) of 0.5 (chance) is the ideal scenario (red dash line), and an AUC closer to this value is better.

method, we use the *Loss Weighted* variant of SoftAdapt, which means that the weights α_k^i in Eq. (6.4)-(6.5) are determined by

$$\alpha_k^i = \frac{f_k^i e^{\beta s_k^i}}{\sum_{\ell=1}^{m} f_\ell^i e^{\beta s_\ell^i}}, \tag{6.7}$$

for an objective function $\mathscr{L} = \alpha_1 f_1 + \alpha_2 f_2 + \cdots + \alpha_m f_m$. In Eq. (6.7), β is a hyperparameter set to 0.1[‡], $k \in 1, 2, \cdots, m$ denotes different coefficients used for different loss parts and i indicates the current iteration for which SoftAdapt is being computed. In the same equation, f_k^i stands for the value of the k-th loss part in the objective function at the i-th iteration; similarly, s_k^i indicates the approximated rate of change for each f_k^i through finite-difference approximation.

[‡]choice of β is discussed in Heydari et al.[194]

Table 6.1: Average training time with sample std. dev. (in seconds), of five iterations for the generative models. ACTIVA (which has the capabilities of scGAN and cscGAN combined) trains much faster than both scGAN and cscGAN. Individual run-times for each iteration are provided in Appendix ??.

	Brain Small			68K PBMC		
	ACTIVA	**scGAN**	**cscGAN**	**ACTIVA**	**scGAN**	**cscGAN**
Average	8074.91 ± 135.9 ($\approx$ 2.2 hours)	142238.10 ± 705.18 ($\approx$ 39.5 hours)	145855.98 ± 335.71 ($\approx$ 40.5 hours)	26025.95 ± 127.68 ($\approx$ 7.2 hours)	164839.14 ± 503.73 ($\approx$ 45.7 hours)	176014.49 ± 192.82 ($\approx$ 48.9 hours)

6.4. DATASETS USED AND PROCESSING STEPS

6.4.1. DATASETS

68K PBMC: To compare our results with the current state-of-the-art deep learning model, scGAN/cscGAN, we trained and evaluated our model on a dataset containing 68579 peripheral blood mononuclear cells (PBMCs) from a healthy donor (68K PBMC)[306]. 68K PBMC is an ideal dataset for evaluating generative models due to the distinct cell populations, data complexity, and size[135]. After pre-processing, the data contained 17789 genes. We then performed a balanced split on this data, which resulted in 6991 testing and 61588 training cells.

Brain Small: In addition to 68K PBMC, we used a randomly selected subset of a larger dataset called Brain Large (both by 10x Genomics). Brain Small contains 20,000 random samples (out of approximately 1.3 million cells) from the cortex, hippocampus, and subventricular zone of two embryonic day 18 mice. Compared to 68K PBMC, this dataset has fewer cells and varies in complexity and organism. The full dataset and and its subset, Brain Small, are available on 10X Genomics portal. After performing the pre-processing steps, the data contained 17970 genes, which we then split (via "balanced split") into 1997 test cells and 18003 training cells.

NeuroCOVID: This dataset[243] contains scRNAseq data of immune cells from the cerebrospinal fluid (CSF) of Neuro-COVID patients and patients with non-inflammatory and autoimmune neurological diseases or with viral encephalitis. Our pre-processing resulted in data of dimensions 85414 cells $\times$ 22824 genes, which we split into testing and training subsets as mentioned above.

6.4.2. PRE-PROCESSING

We use the pipeline provided by Marouf *et al.*[135] to pre-process the data. First, we removed genes expressed in less than 3 cells and cells expressing less than 10 genes.

Next, cells were normalized by total unique molecular identifiers (UMI) counts and scaled to 20000 reads per cell. Then, we selected a "test set" (approximately 10% of each dataset). Testing samples were randomly chosen considering cell ratios in each cluster ("balanced split").

6.4.3. POST-PROCESSING

After generating a count matrix with a generative model (*e.g.* ACTIVA or scGAN), we add the gene names (from the real data) and save them as a Scanpy/Seurat object. We then use Seurat to identify 3000 highly variable genes through the use of variance-stabilization transformation (VST)[307], which applies a negative binomial regression to identify outlier genes. The shared highly variable genes are then used for integration[222], allowing biological feature overlap between different datasets to perform the downstream analyses presented in this work. Next, we perform a gene-level scaling, *i.e.* centering each feature's mean to zero and scaling by the standard deviation. The feature space is then reduced to 50 principal components, followed by Uniform Manifold Approximation and Projection (UMAP)[241] and t-distributed Stochastic Neighbor Embedding (t-SNE)[308]. As noted by Marouf *et al.*[135], analysis with lower-dimensional representations has two main advantages: (i) most biologically relevant information is captured while noise is reduced, and (ii) statistically, it is more acceptable to use lower-dimensional embeddings in classification tasks when samples and features are of the same order of magnitude, which is often the case with scRNAseq datasets (such as the ones we used). Lastly, we visualize the datasets using Scater[309].

6.5. RESULTS

Assessing generative model quality is notoriously difficult and still remains an open research area[310,208]. Here, we apply some qualitative and quantitative metrics for evaluating synthetic scRNAseq, as used in scGAN[135]. For qualitative metrics, we compare the manifold of generated and real cells using UMAP. For quantitative metrics, we train a classifier to distinguish between real and synthetic cells. We compare our results to Marouf et al. alone to study ACTIVA's performance since their models outperform other state-of-the-art generative models such as Splatter[311] and SUGAR[312]. Training and inference time comparisons are shown in Appendix A. As we show, ACTIVA generates cells that better resemble the real data, and it outperforms competing methods in improving the classification of rare cell populations with data augmentation. ACTIVA is one model that can serve as an alternative to both scGAN and cscGAN, and it trains much faster than both GAN-based models (and it

only needs one training).

6.5.1. ACTIVA TRAINS FASTER THAN GAN-BASED MODELS

To measure the efficiency of ACTIVA in comparison to the state-of-the-art GAN-based models (scGAN and cscGAN), we trained all three models in the exact same computational environment on a single GPU for each dataset (we describe the hardware used in Appendix A). Note that since scGAN and cscGAN train separately, we repeated this process five times to account for any variability and then computed the average training time and standard deviation. As shown in Table 6.1, ACTIVA trains orders of magnitude faster for both datasets (approximately 17 times faster on Brain Small and 6 times faster on 68K PBMC and NeuroCOVID) and only needs one training to produce cells from all populations (scGAN's goal) and specific cell populations (cscGAN's purpose).

6.5.2. ACTIVA GENERATES REALISTIC CELLS

To evaluate the generated cells qualitatively, we analyzed the two-dimensional UMAP representation of the test set (real data) and in-silico generated cells (same size as the test set). We found that the distribution and clusters match closely between ACTIVA-generated cells and real cells (Fig. 6.2A-6.2(B) and Fig. A.11). We also analyzed t-SNE embeddings of the real cells and synthetic cells generated by ACTIVA, which showed similar results. These qualitative assessments demonstrated that ACTIVA learns the underlying manifold of real data, the main goal of generative models. A key feature of ACTIVA is cell-type conditioning, which encourages the network to produce cells from all clusters. This means that generating cells with ACTIVA results in *gaining cells within clusters rather than losing clusters*. Due to this design choice, ACTIVA can generate more cells from the rare populations than scGAN, as shown in Fig. 6.2(A)-6.2(B). ACTIVA's flexible framework allows for adjusting the strength of the cell-type conditioning (which is a parameter in our model) for cases where the exact data representation is more desirable.

Next we quantitatively assessed the quality of the generated cells by training a random forest (RF) classifier (same as in Marouf *et al.* [135]) to distinguish between real and generated cells. The goal is to determine how "realistic" ACTIVA-generated cells are compared to real cells. Ideally, the classifier will not differentiate between the synthetic and real cells, thus resulting in a receiver operating characteristic (ROC) curve that is the same as randomly guessing (0.5 AUC). The RF classifier consists of 1000 trees with the Gini impurity criterion and the square root of the number of genes as

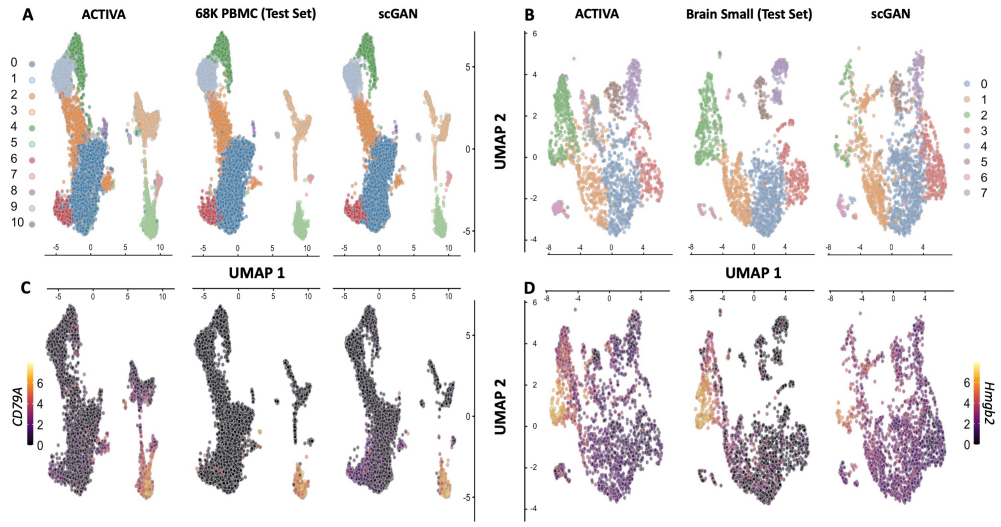


Figure 6.2: **ACTIVA generates high-quality cells that resemble both the cluster and gene expressions present in the training data.** Top row: UMAP plot of ACTIVA generated cells compared with test set and scGAN generated cells, colored by clusters for 68K PBMC (**A**) and Brain Small (**B**). Bottom row: same UMAP plots as top row, colored by selected marker gene expressions. (**C**) corresponds the log expression for CD79A marker gene (for 68K PBMC) and (**D**) illustrates the same for Hmgb2 (for Brain Small). ACTIVA's cell-type conditioning encourages to generate more cells per cluster rather than lose clusters, meaning that ACTIVA will generate more cells from the rare populations (*e.g.* cluster 7 of PBMC and cluster 6 of Brain Small).

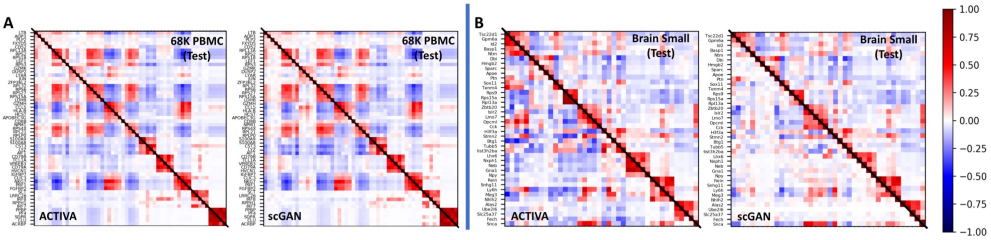


Figure 6.3: **Correlation of top five differentially expressed genes in each cluster for 68K PBMC (A) and Brain Small (B)**. The lower triangular matrices indicate a correlation of generated data, and the upper triangular shows a correlation of real data (same in both plots in panels A and B). For 68K PBMC (A), we investigated pairwise correlation for a total of 55 genes, and for Brain Small (B), we calculated the Pearson correlation for 40 genes. In the ideal case, the correlation plots should be symmetric, and the Correlation Discrepancy (CD), defined in Eq. (6.8), should be zero. The gene correlations in ACTIVA match the real data more closely than scGAN, as shown in the figures and the CD score computed; ACTIVA has a CD score of 1.5816 and 4.6852 for 68K PBMC and Brain Small, respectively, compared to scGAN's 2.2037 and 5.5937.

the maximum number of features used. Maximum depth is set to either all leaves containing less than two samples or until all leaves are pure. We generated cells using ACTIVA and scGAN and performed a five-fold cross-validation on synthetic and real cells (test). ACTIVA performs better than scGAN with AUC scores closer to 0.5 for both datasets (Fig. 6.1). For the Brain Small test set, the mean ACTIVA AUC is 0.62 ± 0.02 compared to scGAN's 0.74 ± 0.01. For 68K PBMC, the mean AUC is 0.68 ± 0.01 for ACTIVA and 0.73 ± 0.01 for scGAN.

Table 6.2: **MMD values for ACTIVA, scGAN and positive control (training set) compared to the test set**. ACTIVA outperforms the scGAN for both datasets, since it has a lower MMD score.

	Brain Small				68K PBMC		
	Training Set	**ACTIVA**	**scGAN**		**Training Set**	**ACTIVA**	**scGAN**
MMD	0.0619	0.7715	0.9686		0.0539	0.7952	0.9047

6.5.3. ACTIVA GENERATES SIMILAR GENE EXPRESSION PROFILES

The marker gene distribution in the generated data should roughly match the gene distribution in real cells to generate cells representing all clusters. We used UMAP representations of ACTIVA, scGAN, and the test set, and colored them based on the expression levels of marker genes. Fig. 6.2(C)-6.2(D) show examples of log-gene expression for marker genes from each dataset. In our qualitative assessment, ACTIVA-generated cells followed the real gene expression closely. For a quantitative assessment, we calculated the Pearson correlation of the top 5 differentially expressed genes from each cluster for both ACTIVA-generated cells and real data. As shown in Fig. 6.3, the pairwise correlation of genes from the ACTIVA-generated cells closely matches those from the real data for both datasets. To quantify the overall gene-gene correlation for synthetic data, we define the following metric: given a correlation matrix of generated samples, G, and a correlation matrix of the real data (test set), R, we compute the 1-norm of the difference in correlations to measure the discrepancy between the correlations

$$CD(G,R) = \max_{1\leq j\leq n} \sum_{i=1}^{n} |G_{i,j} - R_{i,j}|. \tag{6.8}$$

We refer to this metric as *Correlation Discrepancy* (CD) for simplicity. (In the ideal case, $CD(G,R) = 0$, therefore values closer to zero indicate better performance.) Our calculations show that for 68K PBMC, ACTIVA has a CD score of 1.5816, as opposed to scGAN's 2.2037, and for Brain Small ACTIVA outperforms scGAN with a CD score of 4.6852 compared to 5.5937. These values further quantify that generated cells from ACTIVA better preserve the gene-gene correlation present in the real data.

Additionally, we plotted marker gene distribution in all cells against real cells. Fig. A.12 illustrates the distribution of five marker genes from cluster 1 (LTB, LDHB, RPL11, RPL32, RPL13) and cluster 2 (CCL5, NKG7, GZMA, CST7, CTSW). We also investigated known marker genes for specific cell populations, such as for B-cells in PBMC data, finding that ACTIVA generated cells expressed these markers (CD79A, CD19, and MS4A1) in the appropriate clusters (Fig. 6.2C and other figures not shown here) similar to real data. Following Marouf et al., we calculated the maximum mean discrepancy (MMD) between the real data distribution and the generated ones using ACTIVA and scGAN. Simply stated, MMD is a distance metric based on embedding probabilities in a reproducing kernel Hilbert space[313], and since MMD is a distance metric, a lower value of MMD between two distributions indicates the distributions are closer to one another. For consistency, we chose the same kernels as Marouf et al. and calculated MMD on the first 50 principal components. As shown in

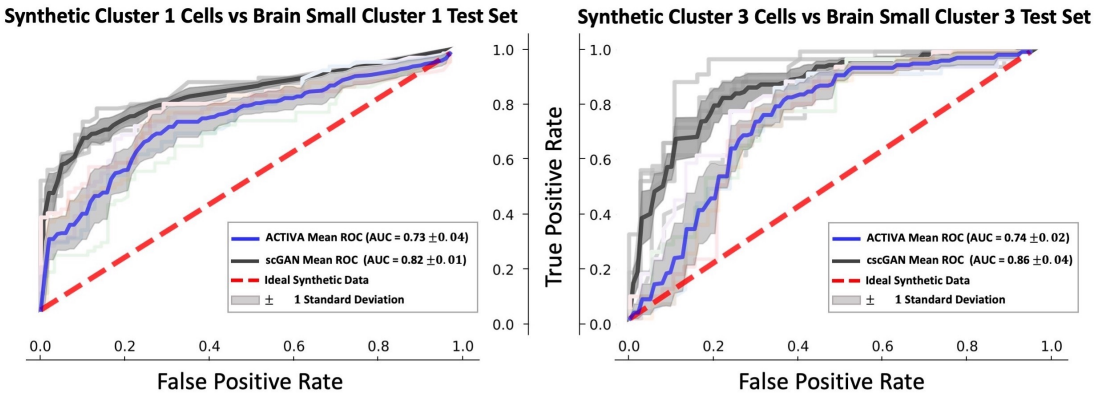


Figure 6.4: **RF classifier distinguishes two real sub-populations from synthetic data for Brain Small**. ACTIVA outperforms cscGAN in producing realistic samples on this dataset since the ROC curve is closer to chance (red dashed line).

Table 6.2, ACTIVA had a lower MMD score than scGAN, demonstrating an improvement in the quality of the generated cells compared to scGAN.

Based on qualitative and quantitative evaluations of our model, we conclude that ACTIVA has learned the underlying marker gene distribution of real data, as desired. However, we suspect that assuming a different prior in model formulation (*e.g.* , Zero-Inflated Negative Binomial) could further improve our model's learning of real data.

6.5.4. ACTIVA GENERATES SPECIFIC CELL-TYPES ON DEMAND

Since we minimize a cell-type identification loss in the training objective, ACTIVA is encouraged to produce correctly classified cells. Therefore, the accuracy of the generated cell types depends on the classifier selected. In Tables A.7 and A.7, we show that ACTIVA's classifier accurately distinguishes rare cell types, achieving an F1 score of 0.89 when trained with only 1% sub-population in the training cells. ACTIVA generates specific cell types from the manifold it has learned, which then filters through the identifier network to produce specific sub-populations. To quantify the quality of the generated samples, we trained an RF classifier (as in §6.5.2) to distinguish between generated and real sub-populations in the data. Fig. 6.4 illustrates ACTIVA's performance against cscGAN for the Brain Small dataset, with ACTIVA achieving better AUC scores. Similar results were obtained for 68K PBMC sub-populations, although the AUC gap between cscGAN and ACTIVA was narrower.

6.5.5. *ACTIVA IMPROVES CLASSIFICATION OF RARE CELLS*

The main goal of designing generative models is to augment sparse datasets with additional data to improve downstream analyses. Given the performance of our model and conditioner, we hypothesized that classifying rare cells in a dataset can be improved through augmentation with ACTIVA, *i.e.* using synthetic rare cells alongside real data. We next directly compared against cscGAN to demonstrate the feasibility of augmenting rare populations to improve classification. We utilized the data-augmentation experiment presented by Marouf *et al.*[135]. That is, we chose the cells in cluster 2 of 68K PBMC and downsampled those cells to 10%, 5%, 1%, and 0.5% of the actual cluster size while keeping the populations fixed. The workflow of the downsampling and exact sizes is shown in Fig. A.10 (Appendix A). We then trained ACTIVA on the downsampled subsets and generated 1500 synthetic cluster 2 cells to augment the data (Marouf *et al.* generated 5000 cells). After that, we used an RF classifier to identify cluster 2 cells versus all other cells. This classification was done on (i) downsampled cells without augmentation and (ii) downsampled cells with ACTIVA augmentation. F1 scores are measured on a held-out test set (10% of the total real cluster 2 cells), shown in Fig. 6.5. The classifier is identical to the one described in §6.5.2 with the addition of accounting cluster-size imbalance, as it was done by Marouf *et al.* , since RF classifiers are sensitive to unbalanced classes[314]. Most notably, our results show an improvement of 0.4526 in F1 score (from 0.4736 to 0.9262) when augmenting 0.5% of real cells and an improvement of 0.2568 (from 0.6829 to 0.9397) on the 1% dataset. ACTIVA also outperforms augmentation with cscGAN for the rarest case since cscGAN achieves an F1 score of 0.8774 as opposed to ACTIVA's 0.9262. These results indicate ACTIVA's promising and powerful application in rare cell-type identification and classification.

6.6. DISCUSSION AND CONCLUSION

In this chapter, we propose a deep generative model for generating realistic scRNAseq data. Our model, ACTIVA, consists of an automatic cell-type identification network coupled with IntroVAE that aims to learn the distribution of the original data and the existing sub-populations. Due to the architectural choices and single-stream training, ACTIVA trains orders of magnitude faster than the state-of-the-art GAN-based model, producing comparably high-quality samples. ACTIVA can be easily trained on different datasets to either enlarge the entire dataset (generate samples from all clusters) or augment specific rare populations.

ACTIVA can generate hundreds of thousands of cells in only a few seconds (on

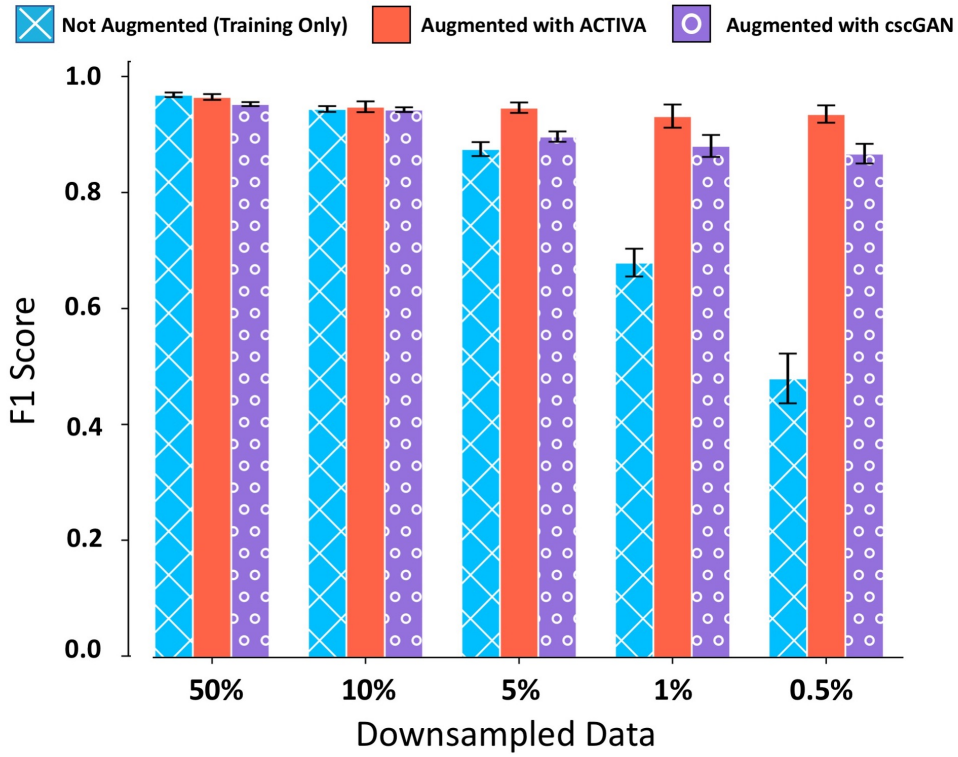


Figure 6.5: **Augmentation with ACTIVA improves classification of rare populations**. Mean F1 scores of the RF classifier for training data (with no augmentation), shown in blue, and training data with augmentation in red and purple. Error bars indicate the range for five different random seeds for sub-sampling cluster 2 cells.

a GPU), which enables benchmarking of new scRNAseq tools' accuracy and scalability. We showed that, for these datasets, using ACTIVA to augment rare populations improves downstream classification by more than 40% in the rarest case of real cells used (0.5% of the training samples). We believe that ACTIVA learns the underlying higher dimensional manifold of the scRNAseq data, even where few cells are available. The deliberate architectural choices of ACTIVA provide insights as to why this learning occurs. As Marouf *et al.* [135] also noted, the fully connected layers of our three networks share information learned from all other populations. In fact, the only cluster-specific parameters are the ones learned in the batch normalization layer. This is also shown by the accuracy of the conditioner network when trained on rare populations. However, if the type-identifying network does not classify sub-populations accurately, this can directly affect the performance of the generator and inference model due to the conditioning. We keep this fact in mind and, therefore, allow for the flexibility of adding any classifier to our existing architecture. Given our architecture, we hypothesize that the conditioner network could be used directly as the encoder or its learned parameters could be transferred to the encoder network, which we plan to explore in the future.

Lopez *et al.* [302] demonstrate that the latent manifold of VAEs can also be useful for analyses such as clustering or denoising. A deep investigation of the learned manifold of ACTIVA can further improve the interpretability of our model or yield new research questions to explore. We also hypothesize that assuming a different prior, such as a Zero Inflated Negative Binomial or a Poisson distribution, could further improve the quality of generated samples given that scRNAseq follow such distributions. Our experiments show that ACTIVA learns to generate high-quality samples on complex datasets from different species. ACTIVA potentially reduces the need for human and animal sample sizes and sequencing depth in studies, saving costs and time and improving the robustness of scRNAseq research with smaller datasets. Furthermore, ACTIVA would benefit studies where large or diverse patient sample sizes are not available, such as rare and emerging diseases.

The more I learn, the more I realize how much I don't know.

Albert Einstein

7 Conclusion

This dissertation, along with my broader PhD research, has aimed to introduce innovative mathematical methodologies essential for effectively modeling complex biological systems. When delving into mathematical modeling for biological phenomena, we can broadly categorize the approaches into three key areas. The first involves modeling under the assumption of known governing equations, providing a foundational framework for understanding system dynamics. The second category encompasses algorithms that leverage large-scale observations, operating on the assumption of complete data to extract meaningful insights from biological processes. Lastly, the third category focuses on modeling in scenarios where assumptions or observations are incomplete, challenging researchers to generalize learnings to different systems or improve initial assumptions. Throughout my research journey, my overarching aim has been to contribute to each of these modeling paradigms by proposing innovative numerical and deep learning techniques. By developing novel mathematical tools and frameworks, my work strives to enhance our modeling arsenal and ability to analyze complex biological processes with diverse characteristics and data availability, irrespective of the challenges posed by governing equations, data completeness, or inherent assumptions.

Given the different modeling paradigms, my dissertation naturally consists of three parts: In the first part, I propose a novel formulation of the finite volumes method on

deforming geometries that preserves mass. In Chapter 2, co-authors and I propose a new numerical method for solving reaction-diffusion systems on a dividing yeast cell, a single-cell organism that divides asymmetrically. Through this work, I identified that the traditional finite volumes fail to preserve mass when solving the reaction-diffusion PDEs on a deforming geometry. Through our analytical analysis, I found that this issue can be attributed to the classical finite volume methods integrating the time-discretized equation in space. To mitigate this, we proposed to use spatio-temporal control volumes that "move" and deform with the geometry through time. Our theoretical and numerical results showed that our formulation of the finite volume preserved mass, a key advancement for modeling reaction-diffusion systems.

We applied our approach to the budding process of yeast and were able to computationally explain the asymmetric distribution of cell content between mother and daughter cells. The proposed technique has the potential to significantly advance mathematical biology by providing a reliable tool for simulating reaction-diffusion systems on complex geometries. Our approach can be used to study a wide range of intricate biological processes, including protein aggregation, cell division, and wound healing. While this work holds promise, biological systems often lack well-defined mathematical formulations, given their inherent complexity and variability. Therefore, assuming governing equations in biological systems may be infeasible. As a result, the subsequent chapters of my dissertation focused on data-driven modeling of biological processes, particularly single-cell RNA sequencing studies.

The second part of my dissertation starts by providing the mathematical and biological background for the subsequent chapters. In the first portion of Chapter 3, I present an overview of single-cell RNA sequencing and spatial transcriptomic technologies, which generate a wealth of biological data with unprecedented resolution. These technologies are instrumental in delineating cellular heterogeneity found in many tissues (*e.g.* tumors). However, traditional statistical methods cannot adequately illuminate the true underlying biology due to the high dimensionality and intricacies of datasets generated by single-cell technologies. As a result, many have turned to developing and using machine learning algorithms for analyzing transcriptomic data. In the latter part of Chapter 3, I provide an overview of machine learning and specific deep learning architectures often used for analyzing and modeling single-cell datasets. In the same chapter, I provide a mathematical review of key deep learning architectures, including feedforward neural networks, convolutional neural networks, recurrent neural networks, residual networks, autoencoders, variational autoencoders (VAEs), generative adversarial networks (GANs), and graph neural networks (GNNs).

Chapter 4 presents a novel *interpretable* deep learning architecture that learns the importance of each gene based on their contribution to the training task. Our model

called *scANNA*, is able to learn the importance of each gene through our use of *neural attention* and our proposed *deep projection blocks*. We trained scANNA on large-scale scRNAseq datasets on an auxiliary task to extract the attention weights (importance scoring). We used these weights without any retraining to perform various downstream tasks, such as global marker selection, cell-type classification, and differential expression analysis. Notably, our approach performs better or comparably to methods that have been designed for each specific downstream task, while scANNA is only trained once on an auxiliary task. Another promising application of scANNA is in transfer learning, where we demonstrated scANNA's effectiveness in leveraging learnings from a large atlas to improve the robustness and accuracy of small-scale scRNAseq studies (similar to the atlas) with minimal fine-tuning.

Our findings from scANNA highlight the potential of attention-based deep learning techniques to substantially improve traditional deep learning methods when applied to scRNAseq data. ScANNA has shown the potential to complement or potentially replace task-specific deep learning pipelines. Given the limited adoption of attention-based deep learning methods in scRNAseq analyses, our work suggests valuable insights may still be hidden within the wealth of publicly available scRNAseq datasets. Furthermore, it is worth noting that scANNA's architectural framework can be adapted to create additional interpretable models for scRNAseq, including applications in spatial transcriptomics. I envision integrating attention layers in deep learning models will facilitate a transition from specialized single-purpose models designed for specific tasks to versatile interpretable tools, empowering researchers without extensive computational backgrounds to expedite their discovery processes. Despite scANNA's potential, one limiting factor is the need for large-scale datasets for scANNA's training, which is often unavailable in pilot or small-scale studies.

The last part of my dissertation proposes methods for enhancing the modeling of data-limited biological systems, the third and last mathematical modeling paradigm we discussed. In this part, my goal is to propose methods that exploit the underlying structure of the data and data manifold to improve the analysis of such datasets. In Chapter 5, the main idea is to discretize PDEs on graphs and to embed those graphs in infinite dimensional manifolds, thereby constructing networks that follow the dynamics of the chosen PDEs. This process allows us to design networks that adhere to our chosen dynamics based on the specific application needs. Moreover, using the rich analysis tools in PDEs, we can prove mathematical conditions on the network's stability, architecture, etc. In this chapter, I applied this idea to construct a novel graph network using the drift-diffusion PDEs, proving theorems on the architecture choices that guarantee model stability. As an initial validation, I applied our model for semi-supervised learning of standard benchmarking datasets, which showed our model,

DDGNN, outperforming all graph networks we tested. Moreover, I demonstrated that, in contrast to most GNNs, our model's performance improves with the addition of more layers. This positions DDGNN as a unique GNN in capturing more intricate and complex relationships within the data, which are often crucial for making accurate predictions. Our last experiment was to test DDGNN's ability to learn biological representations from the scRNAseq dataset, with an emphasis on data-limited studies. Our results indicated that DDGNN can learn better biological representations, outperforming state-of-the-art methods on low-data experiments. These results signify our approach's potential to accurately and robustly identify cell types, even when the number of total observations is less than 1000 cells. The framework presented in this chapter, particularly the innovative approach of constructing graph networks based on discretized PDEs, holds promise for significantly enhancing the analysis of data-limited scRNAseq studies, offering new avenues for improving the accuracy and robustness of cell type identification, even in scenarios with limited observations.

In some cases, models designed for data-limited studies can face challenges in capturing the full complexity of the underlying biological mechanism. Therefore, it is imperative to develop frameworks to augment existing data with realistic synthetic samples, enhancing the robustness and generalizability of the analysis. For this reason, the last chapter of my dissertation focuses on developing a novel deep-learning approach for generating realistic synthetic scRNAseq data. Chapter 6 proposes *ACTIVA*, a VAE-based method that learns the latent distribution of the real scRNAseq data, thus being able to generate realistic cells. ACTIVA uses an innovative adversarial VAE that addresses the main challenges of traditional VAEs, thus outperforming state-of-the-art scRNAseq generators, including GAN-based techniques. Once trained, ACTIVA can generate realistic scRNAseq data from all populations or specific cell types for data augmentation. Our results demonstrated that ACTIVA can generate realistic rare cell types, which, when used to augment the real data, can significantly improve the identification of those rare populations. Our results indicate that ACTIVA is a powerful tool for enhancing the analysis of data-limited scRNAseq experiments. The development of synthetic data generators, like ACTIVA, can be instrumental in enhancing the robustness and generalizability of analyses, highlighting their importance in advancing the modeling of data-limited biological systems.

Each chapter of my dissertation has introduced mathematical frameworks powered by cutting-edge techniques, aiming to expand our toolkit for mathematical biology research. The culminating work of my PhD research offers novel approaches and insights for the three modeling regimes that govern our understanding of complex biological mechanisms. These methods offer practical solutions to bridge theoretical assumptions, empirical data, and biological complexity. The presented models

and methodologies have the potential to advance research in diverse fields, providing robust foundations for a better understanding of biological systems. By addressing these three regimes, I hope my research moves us closer toward a unified framework in modeling biological phenomena, where mathematics and computation synergize to unravel the intricate secrets of life through the combined power of biology and mathematics.

References

[1] W. Commons, "Saccharomyces cerevisiae cells in dic microscopy," 2010. File: S_cerevisiae_under_DIC_microscopy.jpg.

[2] A. Amoussouvi, L. Teufel, M. Reis, M. Seeger, J. K. Schlichting, G. Schreiber, A. Herrmann, and E. Klipp, "Transcriptional timing and noise of yeast cell cycle regulators—a single cell and single molecule approach," *npj Systems Biology and Applications*, vol. 4, no. 1, p. 17, 2018.

[3] J. Pejlare and K. Bråting, *Writing the History of Mathematics: Interpretations of the Mathematics of the Past and Its Relation to the Mathematics of Today*, pp. 1–26. Cham: Springer International Publishing, 2019.

[4] K. Lerner, *History of Mathematics: Renaissance Advancements in Notation Enhance the Translation and Precision of Mathematics.* Thomson Gale, 2001.

[5] D. M. Burton, "The history of mathematics: An introduction," *Group*, vol. 3, no. 3, p. 35, 1985.

[6] N. Erfanian, A. A. Heydari, A. M. Feriz, P. Iañez, A. Derakhshani, M. Ghasemigol, M. Farahpour, S. M. Razavi, S. Nasseri, H. Safarpour, and A. Sahebkar, "Deep learning applications in single-cell genomics and transcriptomics data analysis," *Biomedicine and Pharmacotherapy*, vol. 165, p. 115077, 2023.

[7] Y. Louzoun, "The evolution of mathematical immunology," *Immunological Reviews*, vol. 216, no. 1, pp. 9–20, 2007.

[8] J. Adams, "Sequencing Human Genome: the Contributions of Francis Collins and Craig Venter | Learn Science at Scitable — nature.com." https://www.nature.com/scitable/topicpage/sequencing-human-genome-the-contributions-of-francis-686/, 2008.

[9] F. S. Collins and L. Fink, "The human genome project.," *Alcohol Health Res World*, vol. 19, no. 3, pp. 190–195, 1995.

[10] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology.," *Mol Syst Biol*, vol. 12, p. 878, Jul 2016.

[11] B. Tang, Z. Pan, K. Yin, and A. Khateeb, "Recent advances of deep learning in bioinformatics and computational biology.," *Front Genet*, vol. 10, p. 214, 2019.

[12] C. Xu and S. A. Jackson, "Machine learning and complex biological data," *Genome Biology*, vol. 20, no. 1, p. 76, 2019.

[13] A. L. Fradkov, "Early history of machine learning," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1385–1390, 2020. 21st IFAC World Congress.

[14] A. A. Heydari, J. Rezaei, Naghmehand Prieto, and S. Patel, "Novel Representation Learning Improves Personalizing Blood Test Ranges and Disease Risk Prediction," *ResearchSquare*, 2023.

[15] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *Advances in Neural Information Processing Systems* (Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, eds.), vol. 22, Curran Associates, Inc., 2009.

[16] C. Cleret de Langavant, A. Guittet, M. Theillard, F. Temprano-Coleto, and F. Gibou, "Level-set simulations of soluble surfactant driven flows," *Journal of Computational Physics*, vol. 348, pp. 271–297, 2017.

[17] A. A. Heydari, S. S. Sindi, and M. Theillard, "Conservative finite volume method on deforming geometries: The case of protein aggregation in dividing yeast cells," *Journal of Computational Physics*, vol. 448, p. 110755, 2022.

[18] M. J. Beira and P. J. Sebastião, "A differential equations model-fitting analysis of covid-19 epidemiological data to explain multi-wave dynamics," *Scientific Reports*, vol. 11, no. 1, p. 16312, 2021.

[19] L. Zhao, F. Santiago, E. M. Rutter, S. Khatri, and S. S. Sindi, "Modeling and global sensitivity analysis of strategies to mitigate covid-19 transmission on a structured college campus," *Bulletin of Mathematical Biology*, vol. 85, no. 2, p. 13, 2023.

[20] X. Yang, Y. Liu, and G.-K. Park, "Parameter estimation of uncertain differential equation with application to financial market," *Chaos, Solitons & Fractals*, vol. 139, p. 110026, 2020.

[21] A. Fabretti, "A dynamical model for financial market: Among common market strategies who and how moves the price to fluctuate, inflate, and burst?," *Mathematics*, vol. 10, no. 5, 2022.

[22] V. D'Argenio, "The high-throughput analyses era: Are we ready for the data struggle?," *High Throughput*, vol. 7, Mar 2018.

[23] A. A. Heydari, O. A. Davalos, L. Zhao, K. K. Hoyer, and S. S. Sindi, "Activa: realistic single-cell rna-seq generation with automatic cell-type identification using introspective variational autoencoders," *Bioinformatics*, vol. 38, no. 8, pp. 2194–2201, 2022.

[24] A. A. Heydari and S. S. Sindi, "Deep learning in spatial transcriptomics: Learning from the next next-generation sequencing," *Biophysics Reviews*, vol. 4, no. 1, p. 011306, 2023.

[25] A. A. Heydari, O. A. Davalos, K. K. Hoyer, and S. S. Sindi, "N-ACT: An interpretable deep learning model for automatic cell type and salient gene identification," *Proceedings of the 2022 International Conference on Machine Learning Workshop on Computational Biology*, pp. 2022–05, 2022.

[26] J. D. Marth, "A unified vision of the building blocks of life," *Nature cell biology*, vol. 10, no. 9, pp. 1015–1015, 2008.

[27] J. D. Murray, *Mathematical biology II: spatial models and biomedical applications*, vol. 3. Springer-Verlag, 2001.

[28] J. D. Murray, *Mathematical biology: I. An introduction*, vol. 17. Springer Science & Business Media, 2007.

[29] J. Albert, "A hybrid of the chemical master equation and the gillespie algorithm for efficient stochastic simulations of sub-networks," *PloS one*, vol. 11, no. 3, p. e0149909, 2016.

[30] M. Tanaka, S. R. Collins, B. H. Toyama, and J. S. Weissman, "The physical basis of how prion conformations determine strain phenotypes," *Nature*, vol. 442, no. 7102, pp. 585–589, 2006.

[31] D. A. Charlebois and G. Balázsi, "Modeling cell population dynamics," *In silico biology*, vol. 13, no. 1-2, pp. 21–39, 2019.

[32] A. Kinkhabwala, A. Khmelinskii, and M. Knop, "Analytical model for macromolecular partitioning during yeast cell division," *BMC Biophysics*, vol. 7, p. 10, Sep 2014.

[33] N. A. Cookson, S. W. Cookson, L. S. Tsimring, and J. Hasty, "Cell cycle-dependent variations in protein concentration," *Nucleic acids research*, vol. 38, no. 8, pp. 2676–2681, 2010.

[34] S. Sindi, F. Santiago, and K. Flores, "Numerical approaches to division and label structured population models," *Letters in Biomathematics*, vol. 7, no. 1, pp. 153–170, 2020.

[35] S. Hross and J. Hasenauer, "Analysis of cfse time-series data using division-, age-and label-structured population models," *Bioinformatics*, vol. 32, no. 15, pp. 2321–2329, 2016.

[36] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, 2002. New York, NY.

[37] J. A. Sethian, *Level set methods and fast marching methods*. Cambridge University Press, 1999. Cambridge.

[38] M. Theillard, F. Gibou, and D. Saintillan, "Sharp numerical simulation of incompressible two-phase flows," *Journal of Computational Physics*, vol. 391, pp. 91 – 118, 2019.

[39] M. Mirzadeh, M. Theillard, A. Helgadöttir, D. Boy, and F. Gibou, "An adaptive, finite difference solver for the nonlinear poisson-boltzmann equation with applications to biomolecular computations," *Communications in Computational Physics*, vol. 13, no. 1, p. 150173, 2013.

[40] M. Theillard, L. F. Djodom, J.-L. Vié, and F. Gibou, "A second-order sharp numerical method for solving the linear elasticity equations on irregular domains and adaptive grids–application to shape optimization," *Journal of Computational Physics*, vol. 233, pp. 430–448, 2013.

[41] F. Vermolen, M. W. G. van Rossum, E. J. Perez, and J. Adam, *Modeling of Self Healing of Skin Tissue*, pp. 337–363. Dordrecht: Springer Netherlands, 2007.

[42] M. A. Alias and P. R. Buenzli, "A level-set method for the evolution of cells and tissue during curvature-controlled growth," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 36, no. 1, p. e3279, 2020. e3279 cnm.3279.

[43] Y. Guyot, F. Luyten, J. Schrooten, I. Papantoniou, and L. Geris, "A three-dimensional computational fluid dynamics model of shear stress distribution during neotissue growth in a perfusion bioreactor," *Biotechnology and Bioengineering*, vol. 112, no. 12, pp. 2591–2600, 2015.

[44] Y. Guyot, I. Papantoniou, F. P. Luyten, and L. Geris, "Coupling curvature-dependent and shear stress-stimulated neotissue growth in dynamic bioreactor cultures: a 3d computational model of a complete scaffold," *Biomechanics and Modeling in Mechanobiology*, vol. 15, pp. 169–180, Feb 2016.

[45] P. Macklin and J. Lowengrub, "An improved geometry-aware curvature discretization for level set methods: Application to tumor growth," *Journal of Computational Physics*, vol. 215, no. 2, pp. 392 – 401, 2006.

[46] S. Wise, J. Lowengrub, H. Frieboes, and V. Cristini, "Three-dimensional multispecies nonlinear tumor growthi: Model and numerical method," *Journal of Theoretical Biology*, vol. 253, no. 3, pp. 524 – 543, 2008.

[47] E. M. Rutter, T. L. Stepien, B. J. Anderies, J. D. Plasencia, E. C. Woolf, A. C. Scheck, G. H. Turner, Q. Liu, D. Frakes, V. Kodibagkar, *et al.*, "Mathematical analysis of glioma growth in a murine model," *Scientific Reports*, vol. 7, no. 1, p. 1–16, 2017.

[48] E. Javierre, F. J. Vermolen, C. Vuik, and S. van der Zwaag, "A mathematical analysis of physiological and morphological aspects of wound closure," *Journal of Mathematical Biology*, vol. 59, pp. 605–630, Nov 2009.

[49] L. Yang, J. C. Effler, B. L. Kutscher, S. E. Sullivan, D. N. Robinson, and P. A. Iglesias, "Modeling cellular deformations using the level set formalism," *BMC systems biology*, vol. 2, pp. 68–68, Jul 2008. 18652669[pmid].

[50] J. U. Adams, *Essentials of Cell Biology*. Cambridge, MA: NPG Education, 2010.

[51] A. Ciechanover and Y. T. Kwon, "Protein quality control by molecular chaperones in neurodegeneration," *Frontiers in neuroscience*, vol. 11, p. 185, 2017.

[52] M. F. Tuite and T. R. Serio, "The prion hypothesis: from biological anomaly to basic regulatory mechanism," *Nature reviews Molecular cell biology*, vol. 11, no. 12, pp. 823–833, 2010.

[53] P. Lantos, "From slow virus to prion: a review of transmissible spongiform encephalopathies," *Histopathology*, vol. 20, no. 1, pp. 1–11, 1992.

[54] S. S. Sindi, *Mathematical Modeling of Prion Disease*. InTechOpen, 03 2017.

[55] J. K. Davis and S. S. Sindi, "A study in nucleated polymerization models of protein aggregation," *Applied Mathematics Letters*, vol. 40, pp. 97 – 101, 2015.

[56] N. I. of Health, "Creutzfeldt-jakob disease fact sheet," 2018.

[57] A. S. Association, "Fatal familial insomnia," 2019.

[58] D. Hall and H. Edskes, "Computational modeling of the relationship between amyloid and disease," *Biophysical Reviews*, vol. 4, no. 3, pp. 205–222, 2012.

[59] D. Botstein, S. A. Chervitz, and M. Cherry, "Yeast as a model organism," *Science*, vol. 277, no. 5330, pp. 1259–1260, 1997.

[60] B. Sampaio-Marques, W. C. Burhans, and P. Ludovico, "Yeast at the forefront of research on ageing and age-related diseases," *Yeasts in Biotechnology and Human Health*, pp. 217–242, 2019.

[61] A. Derdowski, S. Sindi, C. Klaips, S. DiSalvo, and T. Serio, "A size threshold limits prion transmission and establishes phenotypic diversity," *Science*, vol. 330, no. 6004, pp. 680–683, 2010.

[62] P. Satpute-Krishnan, S. X. Langseth, and T. R. Serio, "Hsp104-Dependent Remodeling of Prion Complexes Mediates Protein-Only Inheritance," *PLoS Biology*, vol. 5, no. 2, p. e24, 2007.

[63] B. Cox, F. Ness, and M. Tuite, "Analysis of the generation and segregation of propagons: Entities that propagate the [PSI^+] prion in yeast," *Genetics*, vol. 165, no. 1, pp. 23–33, 2003.

[64] B. Cox and M. Tuite, "The life of [psi]," *Current Genetics*, pp. 1–8, 2017.

[65] G. A. Newby and S. Lindquist, "Blessings in disguise: biological benefits of prion-like mechanisms," *Trends in cell biology*, vol. 23, no. 6, pp. 251–259, 2013.

[66] C. Hatzis and D. Porro, "Morphologically-structured models of growing budding yeast populations," *Journal of biotechnology*, vol. 124, no. 2, pp. 420–438, 2006.

[67] J. Yang, M. A. McCormick, J. Zheng, Z. Xie, M. Tsuchiya, S. Tsuchiyama, H. El-Samad, Q. Ouyang, M. Kaeberlein, B. K. Kennedy, *et al.*, "Systematic analysis of asymmetric partitioning of yeast proteome between mother and daughter cells reveals aging factors and mechanism of lifespan asymmetry," *Proceedings of the National Academy of Sciences*, vol. 112, no. 38, pp. 11977–11982, 2015.

[68] D. Kaganovich, R. R. Kopito, and J. Frydman, "Misfolded proteins partition between two distinct quality control compartments," *Nature*, vol. 454, pp. 1088–1095, 2008.

[69] C. Min and F. Gibou, "A second order accurate level set method on non-graded adaptive cartesian grids," *Journal of Computational Physics*, vol. 225, p. 300321, 2007.

[70] S. Mazumder, *Numerical methods for partial differential equations: finite difference and finite volume methods*. Academic Press, 2015.

[71] F. Losasso, F. Gibou, and R. Fedkiw, "Simulating water and smoke with an octree data structure," *ACM Trans. Graph. (SIGGRAPH Proc.)*, pp. 457–462, 2004.

[72] A. Guittet, M. Theillard, and F. Gibou, "A stable projection method for the incompressible navierstokes equations on arbitrary geometries and adaptive quad/octrees," *Journal of Computational Physics*, vol. 292, pp. 215 – 238, 2015.

[73] M. Theillard and D. Saintillan, "Computational mean-field modeling of confined active fluids," *Journal of Computational Physics*, vol. submitted, 2018.

[74] T. Bellotti and M. Theillard, "A coupled level-set and reference map method for interface representation with applications to two-phase flows simulation," *Journal of Computational Physics*, vol. 392, pp. 266–290, 2019.

[75] M. Theillard, "A volume-preserving reference map method for the level set representation," *Journal of Computational Physics*, vol. 442, p. 110478, 2021.

[76] C. Min and F. Gibou, "Geometric integration over irregular domains with application to level-set methods," *J. Comput. Phys.*, vol. 226, p. 14321443, Oct. 2007.

[77] M. Theillard, C. Rycroft, and F. Gibou, "A multigrid method on non-graded adaptive Octree and Quadtree Cartesian grids," *J. Sci. Comput.*, vol. 55, pp. 1–15, 2013.

[78] M. Sussman, K. M. Smith, M. Hussaini, M. Ohta, and Z.-W. Rong, "A sharp interface method for incompressible two-phase flows," *J. Comput. Phys.*, vol. 221, pp. 469–505, 2007.

[79] T. D. Aslam, "A partial differential equation approach to multidimensional extrapolation," *Journal of Computational Physics*, vol. 193, no. 1, pp. 349–355, 2004.

[80] M. Theillard, F. Gibou, and T. Pollock, "A sharp computational method for the simulation of the solidification of binary alloys," *Journal of scientific computing*, vol. 63, no. 2, pp. 330–354, 2015.

[81] E. Gaburro, M. Dumbser, and M. J. Castro, "Direct arbitrary-lagrangian-eulerian finite volume schemes on moving nonconforming unstructured meshes," *Computers & Fluids*, vol. 159, pp. 254–275, 2017.

[82] I. Babushka, J. Chandra, and J. E. Flaherty, *Adaptive computational methods for partial differential equations*, vol. 16. SIAM, 1983.

[83] L. J. Byrne, D. J. Cole, B. S. Cox, M. S. Ridout, B. J. T. Morgan, and M. F. Tuite, "The number and transmission of [psi+] prion seeds (propagons) in the yeast saccharomyces cerevisiae," *PLOS ONE*, vol. 4, pp. 1–10, 03 2009.

[84] C. B. Tyson, P. G. Lord, and A. E. Wheals, "Dependency of size of saccharomyces cerevisiae cells on growth rate.," *Journal of Bacteriology*, vol. 138, no. 1, pp. 92–98, 1979.

[85] R. Milo, R. Phillips, and N. Orme, *Cell Biology by The Numbers*. Garland Science, 2015.

[86] R. Wang, A. Kamgoue, C. Normand, I. Léger-Silvestre, T. Mangeat, and O. Gadal, "High resolution microscopy reveals the nuclear shape of budding yeast during cell cycle and in various biological states," *Journal of Cell Science*, vol. 129, no. 24, pp. 4480–4495, 2016.

[87] M. Ogrodnik, H. Salmonowicz, R. Brown, J. Turkowska, W. Średniawa, S. Pattabiraman, T. Amen, A.-c. Abraham, N. Eichler, R. Lyakhovetsky, and D. Kaganovich, "Dynamic junq inclusion bodies are asymmetrically inherited in mammalian cell lines through the asymmetric partitioning of vimentin," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 111, pp. 8049–8054, Jun 2014. 24843142[pmid].

[88] T. S. G. Database, "Sup35 / ydr172w overview." https://www.yeastgenome.org/locus/S000002579.

[89] J. Villali, J. Dark, T. M. Brechtel, F. Pei, S. S. Sindi, and T. R. Serio, "Nucleation seed size determines amyloid clearance and establishes a barrier to prion appearance in yeast," *Nature structural & molecular biology*, vol. 27, no. 6, pp. 540–549, 2020.

[90] L. Larsson, J. Frisén, and J. Lundeberg, "Spatially resolved transcriptomics adds a new dimension to genomics," *Nature Methods*, vol. 18, no. 1, pp. 15–18, 2021.

[91] J. S. Packer, Q. Zhu, C. Huynh, P. Sivaramakrishnan, E. Preston, H. Dueck, D. Stefanik, K. Tan, C. Trapnell, J. Kim, R. H. Waterston, and J. I. Murray, "A lineage-resolved molecular atlas of c. elegans embryogenesis at single-cell resolution," *Science*, vol. 365, no. 6459, p. eaax1971, 2019.

[92] X. Han, R. Wang, Y. Zhou, L. Fei, H. Sun, S. Lai, A. Saadatpour, Z. Zhou, H. Chen, F. Ye, D. Huang, Y. Xu, W. Huang, M. Jiang, X. Jiang, J. Mao, Y. Chen, C. Lu, J. Xie, Q. Fang, Y. Wang, R. Yue, T. Li, H. Huang, S. H. Orkin, G.-C. Yuan, M. Chen, and G. Guo, "Mapping the mouse cell atlas by microwell-seq," *Cell*, vol. 172, no. 5, pp. 1091–1107.e17, 2018.

[93] N. Schaum, J. Karkanias, N. F. Neff, A. P. May, S. R. Quake, T. Wyss-Coray, S. Darmanis, J. Batson, O. Botvinnik, M. B. Chen, S. Chen, F. Green, R. C. Jones, A. Maynard, L. Penland, A. O. Pisco, R. V. Sit, G. M. Stanley, J. T. Webber, F. Zanini, A. S. Baghel, I. Bakerman, I. Bansal, D. Berdnik, B. Bilen, D. Brownfield, C. Cain, M. B. Chen, M. Cho, G. Cirolia, S. D. Conley, A. Demers, K. Demir, A. de Morree, T. Divita, H. du Bois, L. B. T. Dulgeroff, H. Ebadi, F. H. Espinoza, M. Fish, Q. Gan, B. M. George, A. Gillich, G. Genetiano, X. Gu, G. S. Gulati, Y. Hang, S. Hosseinzadeh, A. Huang, T. Iram, T. Isobe, F. Ives, R. C. Jones, K. S. Kao, G. Karnam, A. M. Kershner, B. M. Kiss, W. Kong, M. E. Kumar, J. Y. Lam, D. P. Lee, S. E. Lee, G. Li, Q. Li, L. Liu, A. Lo, W.-J. Lu, A. Manjunath, A. P. May, K. L. May, O. L. May, M. McKay, R. J. Metzger, M. Mignardi, D. Min, A. N. Nabhan, N. F. Neff, K. M. Ng, J. Noh, R. Patkar, W. C. Peng, R. Puccinelli, E. J. Rulifson, S. S. Sikandar, R. Sinha, R. V. Sit, K. Szade, W. Tan, C. Tato, K. Tellez, K. J. Travaglini, C. Tropini, L. Waldburger, L. J. van Weele, M. N. Wosczyna, J. Xiang, S. Xue, J. Youngyunpipatkul, M. E. Zardeneta, F. Zhang, L. Zhou, A. P. May, N. F. Neff, R. V. Sit, P. Castro, D. Croote, J. L. DeRisi, G. M. Stanley, J. T. Webber, A. S. Baghel, M. B. Chen, F. H. Espinoza, B. M. George, G. S. Gulati, A. M. Kershner, B. M. Kiss, C. S. Kuo, J. Y. Lam, B. Lehallier, A. N. Nabhan, K. M. Ng, P. K. Nguyen, E. J. Rulifson, S. S. Sikandar, S. Y. Tan, K. J. Travaglini, L. J. van Weele, B. M. Wang, M. N. Wosczyna, H. Yousef, A. P. May, S. R. Quake, G. M. Stanley, J. T. Webber, P. A. Beachy, C. K. F. Chan, B. M. George, G. S. Gulati, K. C. Huang, A. M. Kershner, B. M. Kiss, A. N. Nabhan, K. M. Ng, P. K. Nguyen, E. J. Rulifson, S. S. Sikandar, K. J. Travaglini, B. M. Wang, K. Weinberg, M. N. Wosczyna, S. M. Wu, B. A. Barres, P. A. Beachy, C. K. F. Chan, M. F. Clarke, S. K. Kim, M. A. Krasnow, M. E. Kumar, C. S. Kuo, A. P. May, R. J. Metzger, N. F. Neff, R. Nusse, P. K. Nguyen, T. A. Rando, J. Sonnenburg, B. M. Wang, I. L. Weissman, S. M. Wu, S. R. Quake, T. T. M. Consortium, O. coordination, L. coordination, O. collection, processing, L. preparation, sequencing, C. data analysis, C. type annotation, W. group, S. text writing group, and P. investigators, "Single-cell transcriptomics of 20 mouse organs creates a tabula muris," *Nature*, vol. 562, no. 7727, pp. 367–372, 2018.

[94] A. Regev, S. A. Teichmann, E. S. Lander, I. Amit, C. Benoist, E. Birney, B. Bodenmiller, P. Campbell, P. Carninci, M. Clatworthy, H. Clevers, B. Deplancke, I. Dunham, J. Eberwine, R. Eils, W. Enard, A. Farmer, L. Fugger, B. Gottgens, N. Hacohen, M. Haniffa, M. Hemberg, S. Kim, P. Klenerman, A. Kriegstein, E. Lein, S. Linnarsson, E. Lundberg, J. Lundeberg, P. Majumder, J. C. Marioni,

M. Merad, M. Mhlanga, M. Nawijn, M. Netea, G. Nolan, D. Pe'er, A. Phillipakis, C. P. Ponting, S. Quake, W. Reik, O. Rozenblatt-Rosen, J. Sanes, R. Satija, T. N. Schumacher, A. Shalek, E. Shapiro, P. Sharma, J. W. Shin, O. Stegle, M. Stratton, M. J. T. Stubbington, F. J. Theis, M. Uhlen, A. van Oudenaarden, A. Wagner, F. Watt, J. Weissman, B. Wold, R. Xavier, and N. Yosef, "The human cell atlas.," *Elife*, vol. 6, Dec 2017.

[95] K. Davie, J. Janssens, D. Koldere, M. De Waegeneer, U. Pech, Ł. Kreft, S. Aibar, S. Makhzami, V. Christiaens, C. Bravo González-Blas, S. Poovathingal, G. Hulselmans, K. I. Spanier, T. Moerman, B. Vanspauwen, S. Geurs, T. Voet, J. Lammertyn, B. Thienpont, S. Liu, N. Konstantinides, M. Fiers, P. Verstreken, and S. Aerts, "A single-cell transcriptome atlas of the aging drosophila brain.," *Cell*, vol. 174, pp. 982–998, Aug 2018.

[96] Y. Zhang, D. Wang, M. Peng, L. Tang, J. Ouyang, F. Xiong, C. Guo, Y. Tang, Y. Zhou, Q. Liao, X. Wu, H. Wang, J. Yu, Y. Li, X. Li, G. Li, Z. Zeng, Y. Tan, and W. Xiong, "Singlecell RNA sequencing in cancer research," *Journal of Experimental & Clinical Cancer Research*, vol. 40, no. 1, p. 81, 2021.

[97] A. Derakhshani, Z. Asadzadeh, H. Safarpour, P. Leone, M. A. Shadbad, A. Heydari, B. Baradaran, and V. Racanelli, "Regulation of ctla-4 and pd-l1 expression in relapsing-remitting multiple sclerosis patients after treatment with fingolimod, ifnb-1a, glatiramer acetate, and dimethyl fumarate drugs," *Journal of Personalized Medicine*, vol. 11, no. 8, 2021.

[98] K. Anthony, R.-S. Ciro, F. Vicente, C. S. J., M. B. J., S. Hayley, P. Emanuela, S. Grégory, A. Ferhat, V. Pandurangan, and O. C. H., "Severely ill patients with covid-19 display impaired exhaustion features in sars-cov-2–reactive cd8+ t cells," *Science Immunology*, vol. 6, p. eabe4782, 2022/01/30 2021.

[99] S. R. Park, S. Namkoong, L. Friesen, C.-S. Cho, Z. Z. Zhang, Y.-C. Chen, E. Yoon, C. H. Kim, H. Kwak, H. M. Kang, and J. H. Lee, "Single-cell transcriptome analysis of colon cancer cell response to 5-fluorouracil-induced dna damage," *Cell Reports*, vol. 32, no. 8, p. 108077, 2020.

[100] Y. Su, D. Chen, D. Yuan, C. Lausted, J. Choi, C. L. Dai, V. Voillet, V. R. Duvvuri, K. Scherler, P. Troisch, P. Baloni, G. Qin, B. Smith, S. A. Kornilov, C. Rostomily, A. Xu, J. Li, S. Dong, A. Rothchild, J. Zhou, K. Murray, R. Edmark, S. Hong, J. E. Heath, J. Earls, R. Zhang, J. Xie, S. Li, R. Roper, L. Jones, Y. Zhou, L. Rowen, R. Liu, S. Mackay, D. S. O'Mahony, C. R. Dale, J. A. Wallick, H. A. Algren, M. A. Zager, W. Wei, N. D. Price, S. Huang, N. Subramanian, K. Wang, A. T. Magis, J. J. Hadlock, L. Hood, A. Aderem, J. A. Bluestone, L. L. Lanier, P. D. Greenberg, R. Gottardo, M. M. Davis, J. D. Goldman,

and J. R. Heath, "Multi-omics resolves a sharp disease-state shift between mild and moderate covid-19," *Cell*, vol. 183, no. 6, pp. 1479–1495.e20, 2020.

[101] F. Iqbal, A. Lupieri, M. Aikawa, and E. Aikawa, "Harnessing single-cell RNA sequencing to better understand how diseased cells behave the way they do in cardiovascular disease," *Arteriosclerosis, Thrombosis, and Vascular Biology*, vol. 41, pp. 585–600, 2022/01/30 2021.

[102] M. Heming, X. Li, S. Rauber, A. K. Mausberg, A.-L. Borsch, M. Hartlehnert, A. Singhal, I.-N. Lu, M. Fleischer, F. Szepanowski, O. Witzke, T. Brenner, U. Dittmer, N. Yosef, C. Kleinschnitz, H. Wiendl, M. Stettner, and G. Meyer Zu Horste, "Neurological manifestations of covid-19 feature t cell exhaustion and dedifferentiated monocytes in cerebrospinal fluid," *Immunity*, vol. 54, pp. 164–175.e6, 01 2021.

[103] A. A. Pollen, T. J. Nowakowski, J. Shuga, X. Wang, A. A. Leyrat, J. H. Lui, N. Li, L. Szpankowski, B. Fowler, P. Chen, N. Ramalingam, G. Sun, M. Thu, M. Norris, R. Lebofsky, D. Toppani, D. W. Kemp, M. Wong, B. Clerkson, B. N. Jones, S. Wu, L. Knutsson, B. Alvarado, J. Wang, L. S. Weaver, A. P. May, R. C. Jones, M. A. Unger, A. R. Kriegstein, and J. A. A. West, "Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex," *Nature Biotechnology*, vol. 32, no. 10, pp. 1053–1058, 2014.

[104] B. Treutlein, D. G. Brownfield, A. R. Wu, N. F. Neff, G. L. Mantalas, F. H. Espinoza, T. J. Desai, M. A. Krasnow, and S. R. Quake, "Reconstructing lineage hierarchies of the distal lung epithelium using single-cell rna-seq," *Nature*, vol. 509, no. 7500, pp. 371–375, 2014.

[105] M. J. Barresi and S. F. Gilbert, *Developmental Biology*. Oxford University Press, 2019.

[106] R. Dries, J. Chen, N. Del Rossi, M. M. Khan, A. Sistig, and G.-C. Yuan, "Advances in spatial transcriptomic data analysis," *Genome Research*, vol. 31, no. 10, pp. 1706–1718, 2021.

[107] T. Noel, Q. S. Wang, A. Greka, and J. L. Marshall, "Principles of spatial transcriptomics analysis: A practical walk-through in kidney tissue," *Frontiers in Physiology*, vol. 12, 2022.

[108] A. Rao, D. Barkley, G. S. França, and I. Yanai, "Exploring tissue architecture using spatial transcriptomics," *Nature*, vol. 596, no. 7871, pp. 211–220, 2021.

[109] R. Ke, M. Mignardi, A. Pacureanu, J. Svedlund, J. Botling, C. Wahlby, and

M. Nilsson, "In situ sequencing for RNA analysis in preserved tissue and cells," *Nature Methods*, vol. 10, no. 9, pp. 857–860, 2013.

[110] S. Codeluppi, L. E. Borm, A. Zeisel, G. L. Manno, J. A. van Lunteren, C. I. Svensson, and S. Linnarsson, "Spatial organization of the somatosensory cortex revealed by cyclic smfish," *bioRxiv*, 2018.

[111] A. Raj, P. van den Bogaard, S. A. Rifkin, A. van Oudenaarden, and S. Tyagi, "Imaging individual mRNA molecules using multiple singly labeled probes," *Nature Methods*, vol. 5, no. 10, pp. 877–879, 2008.

[112] A. M. Femino, F. S. Fay, K. Fogarty, and R. H. Singer, "Visualization of single RNA transcripts in situ," *Science*, vol. 280, no. 5363, pp. 585–590, 1998.

[113] S. Alon, D. R. Goodwin, A. Sinha, A. T. Wassie, F. Chen, E. R. Daugharthy, Y. Bando, A. Kajita, A. G. Xue, K. Marrett, R. Prior, Y. Cui, A. C. Payne, C.-C. Yao, H.-J. Suk, R. Wang, C.-C. J. Yu, P. Tillberg, P. Reginato, N. Pak, S. Liu, S. Punthambaker, E. P. R. Iyer, R. E. Kohman, J. A. Miller, E. S. Lein, A. Lako, N. Cullen, S. Rodig, K. Helvie, D. L. Abravanel, N. Wagle, B. E. Johnson, J. Klughammer, M. Slyper, J. Waldman, J. Jané-Valbuena, O. Rozenblatt-Rosen, A. Regev, null null, G. M. Church, A. H. Marblestone, E. S. Boyden, H. R. Ali, M. A. Sa'd, S. Alon, S. Aparicio, G. Battistoni, S. Balasubramanian, R. Becker, B. Bodenmiller, E. S. Boyden, D. Bressan, A. Bruna, M. Burger, C. Caldas, M. Callari, I. G. Cannell, H. Casbolt, N. Chornay, Y. Cui, A. Dariush, K. Dinh, A. Emenari, Y. Eyal-Lubling, J. Fan, A. Fatemi, E. Fisher, E. A. González-Solares, C. González-Fernández, D. Goodwin, W. Greenwood, F. Grimaldi, G. J. Hannon, O. Harris, S. Harris, C. Jauset, J. A. Joyce, E. D. Karagiannis, T. Kovačević, L. Kuett, R. Kunes, A. K. Yoldaş, D. Lai, E. Laks, H. Lee, M. Lee, G. Lerda, Y. Li, A. McPherson, N. Millar, C. M. Mulvey, F. Nugent, C. H. O'Flanagan, M. Paez-Ribes, I. Pearsall, F. Qosaj, A. J. Roth, O. M. Rueda, T. Ruiz, K. Sawicka, L. A. Sepúlveda, S. P. Shah, A. Shea, A. Sinha, A. Smith, S. Tavaré, S. Tietscher, I. Vázquez-García, S. L. Vogl, N. A. Walton, A. T. Wassie, S. S. Watson, J. Weselak, S. A. Wild, E. Williams, J. Windhager, T. Whitmarsh, C. Xia, P. Zheng, and X. Zhuang, "Expansion sequencing: Spatially precise in situ transcriptomics in intact biological systems," *Science*, vol. 371, no. 6528, p. eaax2656, 2021.

[114] S. Codeluppi, L. E. Borm, A. Zeisel, G. La Manno, J. A. van Lunteren, C. I. Svensson, and S. Linnarsson, "Spatial organization of the somatosensory cortex revealed by osmfish," *Nature Methods*, vol. 15, no. 11, pp. 932–935, 2018.

[115] K. H. Chen, A. N. Boettiger, J. R. Moffitt, S. Wang, and X. Zhuang, "Spatially

resolved, highly multiplexed RNA profiling in single cells," *Science*, vol. 348, no. 6233, p. aaa6090, 2015.

[116] C.-H. L. Eng, M. Lawson, Q. Zhu, R. Dries, N. Koulena, Y. Takei, J. Yun, C. Cronin, C. Karp, G.-C. Yuan, and L. Cai, "Transcriptome-scale super-resolved imaging in tissues by RNA seqfish+," *Nature*, vol. 568, no. 7751, pp. 235–239, 2019.

[117] X. Wang, W. E. Allen, M. A. Wright, E. L. Sylwestrak, N. Samusik, S. Vesuna, K. Evans, C. Liu, C. Ramakrishnan, J. Liu, G. P. Nolan, F.-A. Bava, and K. Deisseroth, "Three-dimensional intact-tissue sequencing of single-cell transcriptional states," *Science*, vol. 361, no. 6400, p. eaat5691, 2018.

[118] 10x Genomics, "Spatial transcriptomics." https://www.10xgenomics.com/spatial-transcriptomics, December 2021.

[119] P. L. Ståhl, F. Salmén, S. Vickovic, A. Lundmark, J. F. Navarro, J. Magnusson, S. Giacomello, M. Asp, J. O. Westholm, M. Huss, A. Mollbrink, S. Linnarsson, S. Codeluppi, Å. Borg, F. Pontén, P. I. Costea, P. Sahlén, J. Mulder, O. Bergmann, J. Lundeberg, and J. Frisén, "Visualization and analysis of gene expression in tissue sections by spatial transcriptomics," *Science*, vol. 353, no. 6294, pp. 78–82, 2016.

[120] S. G. Rodriques, R. R. Stickels, A. Goeva, C. A. Martin, E. Murray, C. R. Vanderburg, J. Welch, L. M. Chen, F. Chen, and E. Z. Macosko, "Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution," *Science*, vol. 363, no. 6434, pp. 1463–1467, 2019.

[121] S. Vickovic, G. Eraslan, F. Salmén, J. Klughammer, L. Stenbeck, D. Schapiro, T. Aijo, R. Bonneau, L. Bergenstråhle, J. Navarro, J. Gould, G. K. Griffin, Å. Borg, M. Ronaghi, J. Frisén, J. Lundeberg, A. Regev, and P. L. Ståhl, "High-definition spatial transcriptomics for in situ tissue profiling," *Nature Methods*, vol. 16, no. 10, pp. 987–990, 2019.

[122] T. Biancalani, G. Scalia, L. Buffoni, R. Avasthi, Z. Lu, A. Sanger, N. Tokcan, C. R. Vanderburg, Å. Segerstolpe, M. Zhang, I. Avraham-Davidi, S. Vickovic, M. Nitzan, S. Ma, A. Subramanian, M. Lipinski, J. Buenrostro, N. B. Brown, D. Fanelli, X. Zhuang, E. Z. Macosko, and A. Regev, "Deep learning and alignment of spatially resolved single-cell transcriptomes with tangram," *Nature Methods*, vol. 18, no. 11, pp. 1352–1362, 2021.

[123] F. Salmén, P. L. Ståhl, A. Mollbrink, J. Navarro, S. Vickovic, J. Frisén, and J. Lundeberg, "Barcoded solid-phase RNA capture for spatial transcriptomics

profiling in mammalian tissue sections," *Nature Protocols*, vol. 13, no. 11, pp. 2501–2534, 2018.

[124] A. Jemt, F. Salmén, A. Lundmark, A. Mollbrink, J. Fernández Navarro, P. L. Ståhl, T. Yucel-Lindberg, and J. Lundeberg, "An automated approach to prepare tissue-derived spatially barcoded rna-sequencing libraries," *Scientific Reports*, vol. 6, no. 1, p. 37137, 2016.

[125] N. J. Tustison, P. A. Cook, A. Klein, G. Song, S. R. Das, J. T. Duda, B. M. Kandel, N. van Strien, J. R. Stone, J. C. Gee, and B. B. Avants, "Large-scale evaluation of ants and freesurfer cortical thickness measurements," *NeuroImage*, vol. 99, pp. 166–179, 2014.

[126] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "Voxelmorph: A learning framework for deformable medical image registration," *IEEE Transactions on Medical Imaging*, vol. 38, no. 8, pp. 1788–1800, 2019.

[127] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[128] G. Eraslan, L. M. Simon, M. Mircea, N. S. Mueller, and F. J. Theis, "Single-cell rna-seq denoising using a deep count autoencoder," *Nature Communications*, vol. 10, no. 1, p. 390, 2019.

[129] M. B. Badsha, R. Li, B. Liu, Y. I. Li, M. Xian, N. E. Banovich, and A. Q. Fu, "Imputation of single-cell gene expression with an autoencoder neural network," *Quantitative Biology*, vol. 8, no. 1, pp. 78–94, 2020.

[130] X. Li, K. Wang, Y. Lyu, H. Pan, J. Zhang, D. Stambolian, K. Susztak, M. P. Reilly, G. Hu, and M. Li, "Deep learning enables accurate clustering with batch effect removal in single-cell rna-seq analysis," *Nature Communications*, vol. 11, no. 1, p. 2338, 2020.

[131] J. Hu, X. Li, G. Hu, Y. Lyu, K. Susztak, and M. Li, "Iterative transfer learning with neural network for clustering and cell type classification in single-cell rna-seq analysis," *Nature Machine Intelligence*, vol. 2, no. 10, pp. 607–618, 2020.

[132] X. Shao, H. Yang, X. Zhuang, J. Liao, P. Yang, J. Cheng, X. Lu, H. Chen, and X. Fan, "scdeepsort: a pre-trained cell-type annotation method for single-cell transcriptomics using deep learning with a weighted graph neural network," *Nucleic Acids Research*, vol. 49, pp. e122–e122, 1/31/2022 2021.

[133] F. Ma and M. Pellegrini, "Actinn: automated identification of cell types in single cell RNA sequencing," *Bioinformatics*, vol. 36, pp. 533–538, 1/31/2022 2020.

[134] O. A. Davalos, A. A. Heydari, E. J. Fertig, S. S. Sindi, and K. K. Hoyer, "Boosting single-cell rna sequencing analysis with simple neural attention," *bioRxiv*, 2023.

[135] M. Marouf, P. Machart, V. Bansal, C. Kilian, D. S. Magruder, C. F. Krebs, and S. Bonn, "Realistic in silico generation and augmentation of single-cell rna-seq data using generative adversarial networks," *Nature Communications*, vol. 11, no. 1, p. 166, 2020.

[136] K. Bayoudh, R. Knani, F. Hamdaoui, and A. Mtibaa, "A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets," *The Visual Computer*, 2021.

[137] R. K. Gupta and J. Kuznicki, "Biological and medical importance of cellular heterogeneity deciphered by single-cell RNA sequencing," *Cells*, vol. 9, p. 1751, 07 2020.

[138] P. van Galen, V. Hovestadt, M. H. Wadsworth II, T. K. Hughes, G. K. Griffin, S. Battaglia, J. A. Verga, J. Stephansky, T. J. Pastika, J. Lombardi Story, G. S. Pinkus, O. Pozdnyakova, I. Galinsky, R. M. Stone, T. A. Graubert, A. K. Shalek, J. C. Aster, A. A. Lane, and B. E. Bernstein, "Single-cell rna-seq reveals aml hierarchies relevant to disease progression and immunity," *Cell*, vol. 176, no. 6, pp. 1265–1281.e24, 2019.

[139] T. Masuda, R. Sankowski, O. Staszewski, C. Bottcher, L. Amann, Sagar, C. Scheiwe, S. Nessler, P. Kunz, G. van Loo, V. A. Coenen, P. C. Reinacher, A. Michel, U. Sure, R. Gold, D. Grun, J. Priller, C. Stadelmann, and M. Prinz, "Spatial and temporal heterogeneity of mouse and human microglia at single-cell resolution," *Nature*, vol. 566, no. 7744, pp. 388–392, 2019.

[140] J. M. Churko, P. Garg, B. Treutlein, M. Venkatasubramanian, H. Wu, J. Lee, Q. N. Wessells, S.-Y. Chen, W.-Y. Chen, K. Chetal, G. Mantalas, N. Neff, E. Jabart, A. Sharma, G. P. Nolan, N. Salomonis, and J. C. Wu, "Defining human cardiac transcription factor hierarchies using integrated single-cell heterogeneity analysis," *Nature Communications*, vol. 9, no. 1, p. 4906, 2018.

[141] A. Gross, J. Schoendube, S. Zimmermann, M. Steeb, R. Zengerle, and P. Koltay, "Technologies for single-cell isolation," *International Journal of Molecular Sciences*, vol. 16, no. 8, pp. 16897–16919, 2015.

[142] S. Ma, T. W. Murphy, and C. Lu, "Microfluidics for genome-wide studies involving next generation sequencing," *Biomicrofluidics*, vol. 11, pp. 021501–021501, 03 2017.

[143] B. Hwang, J. H. Lee, and D. Bang, "Single-cell RNA sequencing technologies and bioinformaticspipelines," *Experimental & Molecular Medicine*, vol. 50, no. 8, pp. 1–14, 2018.

[144] A. Haque, J. Engel, S. A. Teichmann, and T. Lonnberg, "A practical guide to single-cell rna-sequencing for biomedical research and clinical applications," *Genome Medicine*, vol. 9, no. 1, p. 75, 2017.

[145] R. Stark, M. Grzelak, and J. Hadfield, "RNA sequencing: the teenage years," *Nature Reviews Genetics*, vol. 20, no. 11, pp. 631–656, 2019.

[146] E. L. van Dijk, H. Auger, Y. Jaszczyszyn, and C. Thermes, "Ten years of next-generation sequencing technology," *Trends in Genetics*, vol. 30, no. 9, pp. 418–426, 2014.

[147] X. Zhang, T. Li, F. Liu, Y. Chen, J. Yao, Z. Li, Y. Huang, and J. Wang, "Comparative analysis of droplet-based ultra-high-throughput single-cell rna-seq systems," *Molecular Cell*, vol. 73, no. 1, pp. 130–142.e5, 2019.

[148] A. A. Kolodziejczyk, J. K. Kim, V. Svensson, J. C. Marioni, and S. A. Teichmann, "The technology and biology of single-cell RNA sequencing," *Molecular Cell*, vol. 58, no. 4, pp. 610–620, 2015.

[149] X.-t. Huang, X. Li, P.-z. Qin, Y. Zhu, S.-n. Xu, and J.-p. Chen, "Technical advances in single-cell RNAsequencing and applications in normal and malignant hematopoiesis," *Frontiers in Oncology*, vol. 8, 2018.

[150] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[151] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[152] L. Prechelt, "Early stopping-but when?," in *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, (Berlin, Heidelberg), pp. 55–69, Springer-Verlag, 1998.

[153] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[154] F. Scarselli and A. Chung Tsoi, "Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results," *Neural Networks*, vol. 11, no. 1, pp. 15–37, 1998.

[155] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical

systems," *IEEE Transactions on Neural Networks*, vol. 6, no. 4, pp. 911–917, 1995.

[156] M. Uzair and N. Jamil, "Effects of hidden layers on the efficiency of neural networks," in *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pp. 1–6, 2020.

[157] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), vol. 25, Curran Associates, Inc., 2012.

[158] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios," 2021.

[159] M. Tan and Q. V. Le, "Efficientnetv2: Smaller models and faster training," 2021.

[160] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[161] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[162] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), pp. 4171–4186, Association for Computational Linguistics, 2019.

[163] R. Poplin, P.-C. Chang, D. Alexander, S. Schwartz, T. Colthurst, A. Ku, D. Newburger, J. Dijamco, N. Nguyen, P. T. Afshar, S. S. Gross, L. Dorfman, C. Y. McLean, and M. A. DePristo, "A universal snp and small-indel variant

caller using deep neural networks," *Nature Biotechnology*, vol. 36, no. 10, pp. 983–987, 2018.

[164] H. Li, U. Shaham, K. P. Stanton, Y. Yao, R. R. Montgomery, and Y. Kluger, "Gating mass cytometry data by deep learning," *Bioinformatics*, vol. 33, pp. 3423–3430, 07 2017.

[165] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[166] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017.

[167] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[168] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, (Madison, WI, USA), p. 807814, Omnipress, 2010.

[169] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[170] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Y. W. Teh and M. Titterington, eds.), vol. 9 of *Proceedings of Machine Learning Research*, (Chia Laguna Resort, Sardinia, Italy), pp. 249–256, PMLR, 13–15 May 2010.

[171] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imageNet classification," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[172] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[173] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, pp. 187–217, 2022/02/02 1980.

[174] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[175] Y. LeCun and Y. Bengio, *Convolutional Networks for Images, Speech, and Time Series*, pp. 255–258. Cambridge, MA, USA: MIT Press, 1998.

[176] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[177] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[178] J. F. Kolen and S. C. Kremer, *Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies*, pp. 237–243. IEEE, 2001.

[179] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, nov 1997.

[180] J. C. Kimmel, A. S. Brack, and W. F. Marshall, "Deep convolutional and recurrent neural networks for cell motility discrimination and prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 2, pp. 562–574, 2021.

[181] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning* (F. Bach and D. Blei, eds.), vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 448–456, PMLR, 07–09 Jul 2015.

[182] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[183] A. Shah, E. Kadam, H. Shah, S. Shinde, and S. Shingade, "Deep residual networks with exponential linear unit," in *Proceedings of the Third International Symposium on Computer Vision and the Internet*, pp. 59–65, 2016.

[184] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[185] D. H. Ballard, "Modular learning in neural networks.," in *Aaai*, vol. 647, pp. 279–284, 1987.

[186] G. E. Hinton, *20 - CONNECTIONIST LEARNING PROCEDURES11This chapter appeared in Volume 40 of Artificial Intelligence in 1989, reprinted with permission of North-Holland Publishing. It is a revised version of Technical Report CMU-CS-87-115, which has the same title and was prepared in June 1987 while the author was at Carnegie Mellon University. The research was supported by contract N00014-86-K-00167 from the Office of Naval Research and by grant IST-8520359 from the National Science Foundation.*, pp. 555–610. San Francisco (CA): Morgan Kaufmann, 1990.

[187] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.

[188] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *CoRR*, vol. abs/1312.6114, 2013.

[189] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014.

[190] J. He, D. Spokoyny, G. Neubig, and T. Berg-Kirkpatrick, "Lagging inference networks and posterior collapse in variational autoencoders," in *International Conference on Learning Representations*, 2019.

[191] Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick, "Improved variational autoencoders for text modeling using dilated convolutions," in *International conference on machine learning*, pp. 3881–3890, PMLR, 2017.

[192] A. A. Heydari and A. Mehmood, "SRVAE: super resolution using variational autoencoders," in *Proc.SPIE*, vol. 11400, 4 2020.

[193] S. Semeniuta, A. Severyn, and E. Barth, "A hybrid convolutional variational autoencoder for text generation," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, (Copenhagen, Denmark), pp. 627–637, Association for Computational Linguistics, Sept. 2017.

[194] A. A. Heydari, C. A. Thompson, and A. Mehmood, "SoftAdapt: Techniques for adaptive loss weighting of neural networks with multi-part loss functions," *arXiv preprint arXiv:1912.12355*, 2019.

[195] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf, "Wasserstein auto-encoders," in *International Conference on Learning Representations*, 2018.

[196] T. Daniel and A. Tamar, "SoftIntroVAE: Analyzing and improving the introspective variational autoencoder," 2021.

[197] I. Goodfellow *et al.*, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[198] G. K. Dziugaite *et al.*, "Training generative neural networks via maximum mean discrepancy optimization," *arXiv:1505.03906*, 2015.

[199] J.-Y. Zhu *et al.*, "Generative visual manipulation on the natural image manifold," in *European Conference on Computer Vision*, pp. 597–613, Springer, 2016.

[200] W. Fedus *et al.*, "MaskGAN: better text generation via filling in the_," *arXiv:1801.07736*, 2018.

[201] J. Engel *et al.*, "Gansynth: Adversarial neural audio synthesis," *arXiv:1902.08710*, 2019.

[202] Q. Liu *et al.*, "hicGAN infers super resolution Hi-C data with generative adversarial networks," *Bioinformatics*, vol. 35, pp. i99–i107, 07 2019.

[203] Z. Wang *et al.*, "Generative adversarial networks in computer vision: A survey and taxonomy," *arXiv:1906.01529*, 2019.

[204] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv:1701.04862*, 2017.

[205] A. B. L. Larsen *et al.*, "Autoencoding beyond pixels using a learned similarity metric," vol. 48 of *Proceedings of Machine Learning Research*, (New York, New York, USA), pp. 1558–1566, PMLR, 20–22 Jun 2016.

[206] M. Arjovsky *et al.*, "Wasserstein generative adversarial networks," vol. 70 of *Proceedings of Machine Learning Research*, (International Convention Centre, Sydney, Australia), pp. 214–223, PMLR, 06–11 Aug 2017.

[207] L. Metz *et al.*, "Unrolled generative adversarial networks," *CoRR*, vol. abs/1611.02163, 2016.

[208] M. Lucic *et al.*, "Are GANs created equal? A large-scale study," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, (Red Hook, NY, USA), p. 698707, Curran Associates Inc., 2018.

[209] P. A. Szabo, H. M. Levitin, M. Miron, M. E. Snyder, T. Senda, J. Yuan, Y. L. Cheng, E. C. Bush, P. Dogra, P. Thapa, D. L. Farber, and P. A. Sims, "Single-cell transcriptomics of human t cells reveals tissue and activation signatures in health and disease," *Nat Commun*, vol. 10, no. 1, p. 4706, 2019.

[210] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2015 2015.

[211] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, vol. 3, pp. 111–132, 2021.

[212] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Comput. Surv.*, vol. 54, no. 10s, p. Article 200, 2022.

[213] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, 2021.

[214] F. Yang, W. Wang, F. Wang, Y. Fang, D. Tang, J. Huang, H. Lu, and J. Yao, "scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data," *Nature Machine Intelligence*, vol. 4, no. 10, pp. 852–866, 2022.

[215] D. Buterez, I. Bica, I. Tariq, H. Andrés-Terré, and P. Liò, "Cellvgae: an unsupervised scrna-seq analysis workflow with graph attention networks," *Bioinformatics*, vol. 38, no. 5, pp. 1277–1286, 2021.

[216] C. Xu, L. Cai, and J. Gao, "An efficient scrna-seq dropout imputation method using graph attention network," *BMC Bioinformatics*, vol. 22, no. 1, p. 582, 2021.

[217] H. Zhang, J. Shao, and R. Salakhutdinov, "Deep neural networks with multi-branch architectures are intrinsically less non-convex," in *International Conference on Artificial Intelligence and Statistics*.

[218] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," vol. Preprint, p. Available at https://doi.org/10.48550/arXiv.1607.06450, 2016.

[219] M. E. Nelson, S. G. Riva, and A. Cvejic, "Smash: A scalable, general marker gene identification framework for single-cell RNA sequencing and spatial transcriptomics," *bioRxiv*, 2021.

[220] B. Dumitrascu, S. Villar, D. G. Mixon, and B. E. Engelhardt, "Optimal marker gene selection for cell type discrimination in single cell analyses," *Nat Commun*, vol. 12, no. 1, p. 1186, 2021.

[221] M. A. A, O. Ibanez-Sole, I. Inza, A. Izeta, and M. J. Arauzo-Bravo, "Triku: a feature selection method based on nearest neighbors for single-cell data," *Gigascience*, vol. 11, 2022.

[222] T. Stuart *et al.*, "Comprehensive integration of single-cell data," *Cell*, vol. 177, no. 7, pp. 1888 – 1902.e21, 2019.

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

[224] T. Stuart, A. Butler, P. Hoffman, C. Hafemeister, E. Papalexi, W. M. Mauck, Y. Hao, M. Stoeckius, P. Smibert, and R. Satija, "Comprehensive integration of single-cell data," *Cell*, vol. 177, no. 7, pp. 1888–1902.e21, 2019.

[225] M. E. Nelson, S. G. Riva, and A. Cvejic, "Smash: a scalable, general marker gene identification framework for single-cell rna-sequencing," *BMC Bioinformatics*, vol. 23, no. 1, p. 328, 2022.

[226] F. A. Wolf, P. Angerer, and F. J. Theis, "Scanpy: large-scale single-cell gene expression data analysis," *Genome Biology*, vol. 19, no. 1, p. 15, 2018.

[227] R. Satija, J. A. Farrell, D. Gennert, A. F. Schier, and A. Regev, "Spatial reconstruction of single-cell gene expression data," *Nat Biotechnol*, vol. 33, no. 5, pp. 495–502, 2015.

[228] F. Ma and M. Pellegrini, "Actinn: automated identification of cell types in single cell rna sequencing," *Bioinformatics*, vol. 36, no. 2, pp. 533–538, 2019.

[229] Y. Lin, Y. Cao, H. J. Kim, A. Salim, T. P. Speed, D. M. Lin, P. Yang, and J. Y. H. Yang, "scclassify: sample size estimation and multiscale classification of cells using single and multiple reference," *Mol Syst Biol*, vol. 16, no. 6, p. e9389, 2020.

[230] Y. Tan and P. Cahan, "Singlecellnet: A computational tool to classify single cell rna-seq data across platforms and across species," *Cell Syst*, vol. 9, no. 2, pp. 207–213 e2, 2019.

[231] J. K. de Kanter, P. Lijnzaad, T. Candelli, T. Margaritis, and F. C. P. Holstege, "Chetah: a selective, hierarchical cell type identification method for single-cell rna sequencing," *Nucleic Acids Res*, vol. 47, no. 16, p. e95, 2019.

[232] J. Alquicira-Hernandez, A. Sathe, H. P. Ji, Q. Nguyen, and J. E. Powell, "scpred: accurate supervised method for cell-type classification from single-cell rna-seq data," *Genome Biol*, vol. 20, no. 1, p. 264, 2019.

[233] W. Lin, P. Noel, E. H. Borazanci, J. Lee, A. Amini, I. W. Han, J. S. Heo, G. S. Jameson, C. Fraser, M. Steinbach, Y. Woo, Y. Fong, D. Cridebring, D. D. Von Hoff, J. O. Park, and H. Han, "Single-cell transcriptome analysis of tumor and stromal compartments of pancreatic ductal adenocarcinoma primary tumors and metastatic lesions," *Genome Medicine*, vol. 12, no. 1, p. 80, 2020.

[234] M. Lotfollahi, M. Naghipourfar, M. D. Luecken, M. Khajavi, M. Büttner, M. Wagenstetter, . Avsec, A. Gayoso, N. Yosef, M. Interlandi, S. Rybakov, A. V. Misharin, and F. J. Theis, "Mapping single-cell data to reference atlases by transfer learning," *Nature Biotechnology*, vol. 40, no. 1, pp. 121–130, 2022.

[235] C. Xu, R. Lopez, E. Mehlman, J. Regier, M. I. Jordan, and N. Yosef, "Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models," *Molecular Systems Biology*, vol. 17, no. 1, p. e9620, 2021.

[236] M. Lotfollahi, F. A. Wolf, and F. J. Theis, "scgen predicts single-cell perturbation responses," *Nature Methods*, vol. 16, no. 8, pp. 715–721, 2019.

[237] J. C. Kimmel and D. R. Kelley, "Semisupervised adversarial neural networks for single-cell classification," *Genome Res*, vol. 31, no. 10, pp. 1781–1793, 2021.

[238] D. Berthelot, N. Carlini, I. J. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *ArXiv*, vol. Preprint, p. Available at https://doi.org/10.48550/arXiv.1905.02249, 2019.

[239] Y. Hao, S. Hao, E. Andersen-Nissen, I. I. I. W. M. Mauck, S. Zheng, A. Butler, M. J. Lee, A. J. Wilk, C. Darby, M. Zager, P. Hoffman, M. Stoeckius, E. Papalexi, E. P. Mimitou, J. Jain, A. Srivastava, T. Stuart, L. M. Fleming, B. Yeung, A. J. Rogers, J. M. McElrath, C. A. Blish, R. Gottardo, P. Smibert, and R. Satija, "Integrated analysis of multimodal single-cell data," *Cell*, vol. 184, no. 13, pp. 3573–3587.e29, 2022.

[240] I. Korsunsky, N. Millard, J. Fan, K. Slowikowski, F. Zhang, K. Wei, Y. Baglaenko, M. Brenner, P.-R. Loh, and S. Raychaudhuri, "Fast, sensitive and accurate integration of single-cell data with harmony," *Nature Methods*, vol. 16, no. 12, pp. 1289–1296, 2019.

[241] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2018.

[242] H. Kan, K. Zhang, A. Mao, L. Geng, M. Gao, L. Feng, Q. You, and X. Ma, "Single-cell transcriptome analysis reveals cellular heterogeneity in the ascending aortas of normal and high-fat diet-fed mice," *Exp Mol Med*, vol. 53, no. 9, pp. 1379–1389, 2021.

[243] M. Heming, X. Li, S. Räuber, A. K. Mausberg, A.-L. Börsch, M. Hartlehnert, A. Singhal, I. N. Lu, M. Fleischer, F. Szepanowski, O. Witzke, T. Brenner, U. Dittmer, N. Yosef, C. Kleinschnitz, H. Wiendl, M. Stettner, and G. Meyer Zu Hörste, "Neurological manifestations of covid-19 feature t cell exhaustion and dedifferentiated monocytes in cerebrospinal fluid," *Immunity*, vol. 54, no. 1, pp. 164–175, 2021.

[244] C. Yao, S. A. Bora, T. Parimon, T. Zaman, O. A. Friedman, J. A. Palatinus, N. S. Surapaneni, Y. P. Matusov, G. Cerro Chiang, A. G. Kassar, N. Patel, C. E. R. Green, A. W. Aziz, H. Suri, J. Suda, A. A. Lopez, G. A. Martins, B. R. Stripp, S. A. Gharib, H. S. Goodridge, and P. Chen, "Cell-type-specific immune dysregulation in severely ill covid-19 patients," *Cell Rep*, vol. 34, no. 1, p. 108590, 2021.

[245] A. L. Ji, A. J. Rubin, K. Thrane, S. Jiang, D. L. Reynolds, R. M. Meyers, M. G. Guo, B. M. George, A. Mollbrink, J. Bergenstråhle, L. Larsson, Y. Bai, B. Zhu, A. Bhaduri, J. M. Meyers, X. Rovira-Clavé, S. T. Hollmig, S. Z. Aasi, G. P. Nolan, J. Lundeberg, and P. A. Khavari, "Multimodal analysis of composition and spatial architecture in human squamous cell carcinoma," *Cell*, vol. 182, no. 2, pp. 497–514, 2020.

[246] S. Lukassen, R. L. Chua, T. Trefzer, N. C. Kahn, M. A. Schneider, T. Muley, H. Winter, M. Meister, C. Veith, A. W. Boots, B. P. Hennig, M. Kreuter, C. Conrad, and R. Eils, "Sars-cov-2 receptor ace2 and tmprss2 are primarily expressed in bronchial transient secretory cells," *EMBO J*, vol. 39, no. 10, p. e105114, 2020.

[247] N. G. Steele, E. S. Carpenter, S. B. Kemp, V. R. Sirihorachai, S. The, L. Delrosario, J. Lazarus, E.-a. D. Amir, V. Gunchick, C. Espinoza, S. Bell, L. Harris, F. Lima, V. Irizarry-Negron, D. Paglia, J. Macchia, A. K. Y. Chu, H. Schofield, E.-J. Wamsteker, R. Kwon, A. Schulman, A. Prabhu, R. Law, A. Sondhi, J. Yu, A. Patel, K. Donahue, H. Nathan, C. Cho, M. A. Anderson, V. Sahai, C. A. Lyssiotis, W. Zou, B. L. Allen, A. Rao, H. C. Crawford, F. Bednar, T. L. Frankel, and M. Pasca di Magliano, "Multimodal mapping of the tumor and peripheral blood immune landscape in human pancreatic cancer," *Nature Cancer*, vol. 1, no. 11, pp. 1097–1112, 2020.

[248] E. Elyada, M. Bolisetty, P. Laise, W. F. Flynn, E. T. Courtois, R. A. Burkhart, J. A. Teinor, P. Belleau, G. Biffi, M. S. Lucito, S. Sivajothi, T. D. Armstrong, D. D. Engle, K. H. Yu, Y. Hao, C. L. Wolfgang, Y. Park, J. Preall, E. M. Jaffee, A. Califano, P. Robson, and D. A. Tuveson, "Cross-species single-cell analysis of pancreatic ductal adenocarcinoma reveals antigen-presenting cancer-associated fibroblasts," *Cancer Discovery*, vol. 9, no. 8, pp. 1102–1123, 2019.

[249] R. Moncada, D. Barkley, F. Wagner, M. Chiodin, J. C. Devlin, M. Baron, C. H. Hajdu, D. M. Simeone, and I. Yanai, "Integrating microarray-based spatial transcriptomics and single-cell rna-seq reveals tissue architecture in pancreatic ductal adenocarcinomas," *Nature Biotechnology*, vol. 38, no. 3, pp. 333–342, 2020.

[250] V. Bernard, A. Semaan, J. Huang, F. A. San Lucas, F. C. Mulu, B. M. Stephens, P. A. Guerrero, Y. Huang, J. Zhao, N. Kamyabi, S. Sen, P. A. Scheet, C. M. Taniguchi, M. P. Kim, C.-W. Tzeng, M. H. Katz, A. D. Singhi, A. Maitra, and H. A. Alvarez, "Single-cell transcriptomics of pancreatic cancer precursors demonstrates epithelial and microenvironmental heterogeneity as an early event in neoplastic progression," *Clinical Cancer Research*, vol. 25, no. 7, pp. 2194–2205, 2019.

[251] J. Peng, B.-F. Sun, C.-Y. Chen, J.-Y. Zhou, Y.-S. Chen, H. Chen, L. Liu, D. Huang, J. Jiang, G.-S. Cui, Y. Yang, W. Wang, D. Guo, M. Dai, J. Guo, T. Zhang, Q. Liao, Y. Liu, Y.-L. Zhao, D.-L. Han, Y. Zhao, Y.-G. Yang, and W. Wu, "Single-cell rna-seq highlights intra-tumoral heterogeneity and malignant progression in pancreatic ductal adenocarcinoma," *Cell Research*, vol. 29, no. 9, pp. 725–738, 2019.

[252] B. Kinny-Köster, S. Guinn, J. A. Tandurella, J. T. Mitchell, D. N. Sidiropoulos, M. Loth, M. R. Lyman, A. B. Pucsek, T. T. Seppälä, C. Cherry, R. Suri, H. Zlomke, J. He, C. L. Wolfgang, J. Yu, L. Zheng, D. P. Ryan, D. T. Ting, A. Kimmelman, A. Gupta, L. Danilova, J. H. Elisseeff, L. D. Wood, G. Stein-OBrien, L. T. Kagohara, E. M. Jaffee, R. A. Burkhart, E. J. Fertig, and J. W. Zimmerman, "Inflammatory signaling in pancreatic cancer transfers between a single-cell rna sequencing atlas and co-culture," *bioRxiv*, p. 2022.07.14.500096, 2022.

[253] B. Daniel, K. E. Yost, S. Hsiung, K. Sandor, Y. Xia, Y. Qi, K. J. Hiam-Galvez, M. Black, C. J. Raposo, Q. Shi, S. L. Meier, J. A. Belk, J. R. Giles, E. J. Wherry, H. Y. Chang, T. Egawa, and A. T. Satpathy, "Divergent clonal differentiation trajectories of t cell exhaustion," *Nature Immunology*, vol. 23, no. 11, pp. 1614–1627, 2022.

[254] J. R. Giles, S. F. Ngiow, S. Manne, A. E. Baxter, O. Khan, P. Wang, R. Staupe, M. S. Abdel-Hakeem, H. Huang, D. Mathew, M. M. Painter, J. E. Wu, Y. J. Huang, R. R. Goel, P. K. Yan, G. C. Karakousis, X. Xu, T. C. Mitchell, A. C. Huang, and E. J. Wherry, "Shared and distinct biological circuits in effector, memory and exhausted cd8(+) t cells revealed by temporal single-cell transcriptomics and epigenetics," *Nat Immunol*, vol. 23, no. 11, pp. 1600–1613, 2022.

[255] M. D. Luecken and F. J. Theis, "Current best practices in single-cell rna-seq analysis: a tutorial," *Molecular Systems Biology*, vol. 15, no. 6, p. e8746, 2019.

[256] G. Pasquini, J. E. Rojo Arias, P. Schäfer, and V. Busskamp, "Automated methods for cell type annotation on scrna-seq data," *Computational and Structural Biotechnology Journal*, vol. 19, pp. 961–969, 2021.

[257] S. Fischer and J. Gillis, "How many markers are needed to robustly determine a cell's type?," *iScience*, vol. 24, no. 11, p. 103292, 2021.

[258] G. L. Stein-O'Brien, B. S. Clark, T. Sherman, C. Zibetti, Q. Hu, R. Sealfon, S. Liu, J. Qian, C. Colantuoni, S. Blackshaw, L. A. Goff, and E. J. Fertig, "Decomposing cell identity for transfer learning across cellular measurements, platforms, tissues, and species," *Cell Systems*, vol. 8, pp. 395–411.e8, 2023/05/26 2019.

[259] E. Y. Chen, C. M. Tan, Y. Kou, Q. Duan, Z. Wang, G. V. Meirelles, N. R. Clark, and A. Ma'ayan, "Enrichr: interactive and collaborative html5 gene list enrichment analysis tool," *BMC Bioinformatics*, vol. 14, p. 128, 2013.

[260] M. V. Kuleshov, M. R. Jones, A. D. Rouillard, N. F. Fernandez, Q. Duan, Z. Wang, S. Koplev, S. L. Jenkins, K. M. Jagodnik, A. Lachmann, M. G. McDermott, C. D. Monteiro, G. W. Gundersen, and A. Ma'ayan, "Enrichr: a comprehensive gene set enrichment analysis web server 2016 update," *Nucleic Acids Res*, vol. 44, no. W1, pp. W90–7, 2016.

[261] Z. Xie, A. Bailey, M. V. Kuleshov, D. J. B. Clarke, J. E. Evangelista, S. L. Jenkins, A. Lachmann, M. L. Wojciechowicz, E. Kropiwnicki, K. M. Jagodnik, M. Jeon, and A. Ma'ayan, "Gene set knowledge discovery with enrichr," *Current Protocols*, vol. 1, no. 3, p. e90, 2021.

[262] G. W. Brodland, "How computational models can help unlock biological systems," *Seminars in Cell and Developmental Biology*, vol. 47-48, pp. 62–73, 2015. Coding and non-coding RNAs & Mammalian development.

[263] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature Medicine*, vol. 25, pp. 24–29, Jan 2019.

[264] W. Huber, V. J. Carey, L. Long, S. Falcon, and R. Gentleman, "Graphs in molecular biology," *BMC Bioinformatics*, vol. 8, no. 6, p. S8, 2007.

[265] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2, pp. 729–734 vol. 2, 2005.

[266] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[267] P. Velikovi, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2018.

[268] E. Rocheteau, C. Tong, P. Velickovic, N. D. Lane, and P. Liò, "Predicting patient outcomes with graph representation learning," *CoRR*, vol. abs/2101.03940, 2021.

[269] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, Curran Associates, Inc., 2018.

[270] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[271] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, A. Stuart, K. Bhattacharya, and A. Anandkumar, "Multipole graph neural operator for parametric partial differential equations," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 6755–6766, Curran Associates, Inc., 2020.

[272] H. Gao, M. J. Zahr, and J.-X. Wang, "Physics-informed graph neural galerkin networks: A unified framework for solving pde-governed forward and inverse problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 390, p. 114502, 2022.

[273] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, (Red Hook, NY, USA), p. 38443852, Curran Associates Inc., 2016.

[274] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014.

[275] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 29–38, IEEE Computer Society, jul 2017.

[276] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst, "Geodesic convolutional neural networks on riemannian manifolds," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pp. 832–840, 2015.

[277] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, p. 12631272, JMLR.org, 2017.

[278] M. Eliasof, E. Haber, and E. Treister, "Pde-gcn: Novel architectures for graph neural networks motivated by partial differential equations," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 3836–3849, Curran Associates, Inc., 2021.

[279] L. Zhao and L. Akoglu, "Pairnorm: Tackling oversmoothing in gnns," in *International Conference on Learning Representations*, 2020.

[280] M. Balcilar, G. Renton, P. Héroux, B. Gaüzère, S. Adam, and P. Honeine, "Analyzing the expressive power of graph neural networks in a spectral perspective," in *International Conference on Learning Representations*, 2021.

[281] B. Chamberlain, J. Rowbottom, M. I. Gorinova, M. Bronstein, S. Webb, and E. Rossi, "Grand: Graph neural diffusion," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 1407–1418, PMLR, 18–24 Jul 2021.

[282] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," in *International Conference on Learning Representations*, 2021.

[283] J. Wang, A. Ma, Y. Chang, J. Gong, Y. Jiang, R. Qi, C. Wang, H. Fu, Q. Ma, and D. Xu, "scgnn is a novel graph neural network framework for single-cell rna-seq analyses," *Nature Communications*, vol. 12, no. 1, p. 1882, 2021.

[284] L. Ruthotto and E. Haber, "Deep neural networks motivated by partial differential equations," *Journal of Mathematical Imaging and Vision*, vol. 62, pp. 352–364, 2018.

[285] D. Zhou and C. J. Burges, "High-order regularization on graphs," in *MLG-2008: 6th International Workshop on Mining and Learning with Graphs, Helsinki, Finland*, July 2008.

[286] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[287] W. Chung, H. H. Eum, H.-O. Lee, K.-M. Lee, H.-B. Lee, K.-T. Kim, H. S. Ryu, S. Kim, J. E. Lee, Y. H. Park, Z. Kan, W. Han, and W.-Y. Park, "Single-cell rna-seq enables comprehensive tumour and immune cell profiling in primary breast cancer," *Nature Communications*, vol. 8, no. 1, p. 15081, 2017.

[288] A. M. Klein, L. Mazutis, I. Akartuna, N. Tallapragada, A. Veres, V. Li, L. Peshkin, D. A. Weitz, and M. W. Kirschner, "Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells," *Cell*, vol. 161, no. 5, pp. 1187–1201, 2015.

[289] A. Zeisel, A. B. Muñoz-Manchado, S. Codeluppi, P. Lönnerberg, G. L. Manno, A. Juréus, S. Marques, H. Munguba, L. He, C. Betsholtz, C. Rolny, G. Castelo-Branco, J. Hjerling-Leffler, and S. Linnarsson, "Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq," *Science*, vol. 347, no. 6226, pp. 1138–1142, 2015.

[290] X. Tang *et al.*, "The single-cell sequencing: new developments and medical applications," *Cell & Bioscience*, vol. 9, no. 1, p. 53, 2019.

[291] F. Tang *et al.*, "mRNA-seq whole-transcriptome analysis of a single cell," *Nature methods*, vol. 6, no. 5, pp. 377–382, 2009.

[292] A. Regev *et al.*, "The human cell atlas.," *Elife*, vol. 6, Dec 2017.

[293] K. S. Button *et al.*, "Power failure: why small sample size undermines the reliability of neuroscience," *Nature reviews neuroscience*, vol. 14, no. 5, pp. 365–376, 2013.

[294] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

[295] S. Benidt and D. Nettleton, "SimSeq: a nonparametric approach to simulation of RNA-sequence datasets," *Bioinformatics*, vol. 31, no. 13, pp. 2131–2140, 2015.

[296] A. C. Frazee *et al.*, "Polyester: simulating RNA-seq datasets with differential transcript expression," *Bioinformatics*, vol. 31, no. 17, pp. 2778–2784, 2015.

[297] X. Zhang *et al.*, "Simulating multiple faceted variability in single cell rna sequencing," *Nature Communications*, vol. 10, no. 1, p. 2611, 2019.

[298] A. T. Assefa *et al.*, "SPsimSeq: semi-parametric simulation of bulk and single-cell RNA-sequencing data," *Bioinformatics*, vol. 36, no. 10, pp. 3276–3278, 2020.

[299] D. Gerard, "Data-based rna-seq simulations by binomial thinning," *BMC Bioinformatics*, vol. 21, no. 1, p. 206, 2020.

[300] T. Miyato and M. Koyama, "cGANs with projection discriminator," in *International Conference on Learning Representations*, 2018.

[301] H. Huang *et al.*, "IntroVAE: Introspective variational autoencoders for photographic image synthesis," in *Advances in Neural Information Processing Systems 31* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), pp. 52–63, Curran Associates, Inc., 2018.

[302] R. Lopez, J. Regier, M. B. Cole, M. I. Jordan, and N. Yosef, "Deep generative modeling for single-cell transcriptomics," *Nature Methods*, vol. 15, no. 12, pp. 1053–1058, 2018.

[303] K. Zheng *et al.*, "Conditional introspective variational autoencoder for image synthesis," *IEEE Access*, vol. 8, pp. 153905–153913, 2020.

[304] T. Abdelaal *et al.*, "A comparison of automatic cell identification methods for single-cell RNA sequencing data," *Genome Biology*, vol. 20, no. 1, p. 194, 2019.

[305] J. Lucas, G. Tucker, R. B. Grosse, and M. Norouzi, "Don't blame the ELBO! A linear VAE perspective on posterior collapse," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, pp. 9408–9418, Curran Associates, Inc., 2019.

[306] G. X. Y. Zheng *et al.*, "Massively parallel digital transcriptional profiling of single cells," *Nature Communications*, vol. 8, no. 1, p. 14049, 2017.

[307] C. Hafemeister and R. Satija, "Normalization and variance stabilization of single-cell RNA-seq data using regularized negative binomial regression," *Genome Biology*, vol. 20, no. 1, p. 296, 2019.

[308] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[309] D. J. McCarthy *et al.*, "Scater: pre-processing, quality control, normalization and visualization of single-cell RNA-seq data in R," *Bioinformatics*, vol. 33, pp. 1179–1186, 01 2017.

[310] L. Theis *et al.*, "A note on the evaluation of generative models," in *International Conference on Learning Representations*, Apr 2016.

[311] L. Zappia *et al.*, "Splatter: simulation of single-cell RNA sequencing data," *Genome Biology*, vol. 18, no. 1, p. 174, 2017.

[312] O. Lindenbaum *et al.*, "Geometry based data generation," in *Advances in Neural Information Processing Systems* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, pp. 1400–1411, Curran Associates, Inc., 2018.

[313] A. Gretton *et al.*, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. 25, pp. 723–773, 2012.

[314] B. Zadrozny *et al.*, "Cost-sensitive learning by cost-proportionate example weighting," in *Third IEEE International Conference on Data Mining*, pp. 435–442, 2003.

[315] J. Zhao *et al.*, "Energy-based generative adversarial network," *arXiv:1609.03126*, 2016.

[316] T. S. Andrews and M. Hemberg, "M3Drop: dropout-based feature selection for scRNASeq," *Bioinformatics*, vol. 35, pp. 2865–2867, 12 2018.

[317] H. L. Crowell, C. Soneson, P.-L. Germain, D. Calini, L. Collin, C. Raposo, D. Malhotra, and M. D. Robinson, "muscat detects subpopulation-specific state transitions from multi-sample multi-condition single-cell transcriptomics data," *Nature Communications*, vol. 11, no. 1, p. 6077, 2020.

[318] H. L. Crowell, C. Soneson, P.-L. Germain, D. Calini, L. Collin, C. Raposo, D. Malhotra, and M. D. Robinson, "On the discovery of population-specific state transitions from multi-sample multi-condition single-cell rna sequencing data," *bioRxiv*, 2019.

[319] T. Daniel and A. Tamar, "Soft-introvae: Analyzing and improving the introspective variational autoencoder," 2021.

A

Supplemental Material for Chapter 6

DATA POST-PROCESSING

After generating a count matrix with a generative model (e.g., ACTIVA or scGAN), we add the gene names (from the real data) and save them as a Scanpy/Seurat object. We then use Seurat to identify 3000 highly variable genes through the use of variance-stabilization transformation (VST) [307], which applies a negative binomial regression to identify outlier genes. The shared highly variable genes are then used for integration [222], allowing biological feature overlap between different datasets to perform the downstream analyses presented in this work. Next, we perform a gene-level scaling, i.e., centering the mean of each feature to zero and scaling by the standard deviation. The feature space is then reduced to 50 principal components, followed by the Uniform Manifold Approximation and Projection (UMAP)[241] and t-distributed Stochastic Neighbor Embedding (t-SNE)[308]. As noted by [135], analysis with lower-dimensional representations has two main advantages: (i) most biologically relevant information is captured while noise is reduced, and (ii) statistically, it is more acceptable to use lower-dimensional embeddings in classification tasks when samples and features are of the same order of magnitude, which is often the case with scRNAseq datasets (such as the ones we used). We visualize the datasets using Scater [309].

COMPLETE COMPUTATIONAL ENVIRONMENT

Development and testing were done on Accelerated Computing EC2 instances (`p3.2xlarge` and `p3.8xlarge`) of Amazon Web Services. Our package automatically installs all requirements and dependencies. They are listed in a requirements file, but for the sake of completeness, they are as follows: Python v3.7.6, PyTorch v1.5.1, NumPy v1.18.5, SciPy v1.4.1, Pandas v1.2.0, Scanpy v1.6.0, AnnData v0.7.5, and Scikit-learn v0.24.0. For data pre- and post-processing, we used LoomPy v3.0.6, SeuratDisk v0.0.0.9013, Seurat v3.2.3, scatter v1.16.2, and R v4.0.3. To evaluate dropout rates, we used M3Drop v1.18.0 and ggplot2 v3.3.2. Differential state analysis was performed with Muscat v1.6.0. The scGAN package was run in a Docker container (using the provided dockerfile at `https://github.com/imsb-uke/scGAN/tree/master/dockerfile`). Reported training times for ACTIVA/scGAN were averages of 5 times on a single NVIDIA-Tesla V100 GPU. Inference times were averages of 5 measurements on (i) V100 GPU (GPU time) and (ii) 2.3 GHz Quad-Core Intel Core i7 (on a 2020 MacBook Pro).

GPU TRAINING TIMES

Table A.1: **Training time (in seconds) on 68K PBMC for ACTIVA and scGAN on 1 NVIDIA Tesla V100 GPU**. We trained each model 5 times under the same conditions to find an average training time. We have not included cscGAN times since training that model took longer than scGAN. We can see that ACTIVA trains much faster (6.3 times faster on average). scGAN and cscGAN were run in a Docker container (by Marouf et al.) using the Dockerfile located at `https://github.com/imsb-uke/scGAN/tree/master/dockerfile`).

Iteration	ACTIVA	scGAN	cscGAN
1	25968.2447	164225.5618	175978.2588
2	26218.7306	165308.7142	176041.7922
3	25935.2738	164391.8113	176318.0701
4	26091.8337	165281.5137	175941.7097
5	25915.6733	164988.1371	175792.6410
Average	26025.95 (≈7.2 hours)	164839.14 (≈45.7 hours)	176014.49 (≈48.8 hours)

Table A.2: **Training time (in seconds) on Brain Small for ACTIVA and scGAN on 1 NVIDIA Tesla V100 GPU**. We trained each model 5 times under the same conditions to find an average training time. ACTIVA trains approximately 17 times faster than scGAN. scGAN and cscGAN were run in a Docker container (using the Dockerfile located at https://github.com/imsb-uke/scGAN/tree/master/dockerfile).

Iteration	ACTIVA	scGAN	cscGAN
1	8277.4867	142371.9793	145980.3154
2	7922.1005	141103.8196	145952.2580
3	8107.3591	143008.7401	145261.1851
4	7983.4031	142532.3804	146076.8253
5	8084.2473	142173.5900	146009.3626
Average	8074.91 (≈2.2 hours)	142238.10 (≈39.5 hours)	145855.98 (≈40.5 hours)

Table A.3: **Training time (in seconds) on NeuroCOVID for ACTIVA and scGAN on 1 NVIDIA Tesla V100 GPU**. We trained each model 5 times under the same conditions to find an average training time. ACTIVA trains approximately six times faster than scGAN. scGAN and cscGAN were run in a Docker container (using the Dockerfile located at https://github.com/imsb-uke/scGAN/tree/master/dockerfile).

Iteration	ACTIVA	scGAN	cscGAN
1	29604.6613	184421.5509	187618.5119
2	29351.4168	183863.5722	187440.2821
3	29719.3984	184249.4072	188414.8095
4	29492.1603	183120.8326	187922.5381
5	29575.2929	183814.1538	187924.3118
Average	29548.58 (≈ 8.2 hours)	183749.10 (≈ 51.0 hours)	187864.09 (≈ 52.1 hours)

RUNTIME ANALYSIS

As mentioned in Chapter 6, ACTIVA trains much faster than the GAN-based models due to its architecture. ACTIVA's runtime depends on several factors, such as the complexity of the dataset, number of genes, sparsity, and the hardware used for

training −even using different GPUs will result in training time differences. However, since ACTIVA uses batch training (as done in almost all deep learning models), out-of-memory (OOM) errors are more controlled, and the runtime scales linearly (at worst). Here, we aim to demonstrate this fact with so-called "corner" cases. Due to the nature of scRNAseq experiments, we believe that it is unlikely that ACTIVA will be trained with 10^5 cells or more, but we provide the runtimes for 10^5, 2×10^5, and 5×10^5 for reference. Due to the computational costs, we did not replicate the runtime experiment for scGAN and cscGAN. However, the runtimes can be extrapolated from our previous results and Marouf *et al.*[135].
To construct the mentioned corner cases, we generated random data $X_{runtime} \in \mathbb{R}^{n \times d}$, where d is the number of genes in 68K PBMC, and n is the number of cells ($n \in \{100000, 200000, 500000\}$). Next, we measured the percentage of non-zero entries in the 68K PBMC data (often called density). We found that the 68K PBMC had a density of about 3%. However, we set the $X_{runtime}$ density to be 30% to create a "worst-case" for training. Since we were only interested in the runtime of ACTIVA for $X_{runtime}$ and not the data generation quality for this random data, we used all of $X_{runtime}$ for training. The training data dimensions were 100000×17789, 200000×17789, and 500000×17789. We repeated training on each dataset five times and presented the runtimes in Table A.4.

Table A.4: **Runtime analysis for ACTIVA with various numbers of cells**. To simulate the "worst cases", we chose the same number of genes as in 68K PBMC (17789), and we took the density of the count matrix to be 30% (compared to about 3% for 68K PBMC). We then trained ACTIVA for various numbers of cells for five iterations, which we present in this table. The reported times are measured in seconds.

Iteration	100K	200K	500K
1	33143.3375	61270.0890	129722.8039
2	33053.8834	61255.4751	126041.6814
3	32931.2144	61183.5369	124377.9299
4	32995.6469	61230.1366	130018.3193
5	33181.0439	61091.5801	128588.0704
Average	33061.0252 ($\approx$ 9.1 hours)	61206.1635 ($\approx$ 17.0 hours)	127749.7609 ($\approx$35.48 hours)

RESULTS ON NEURO-COVID DATA

This section presents our qualitative and quantitative results on our third dataset: NeuroCOVID.

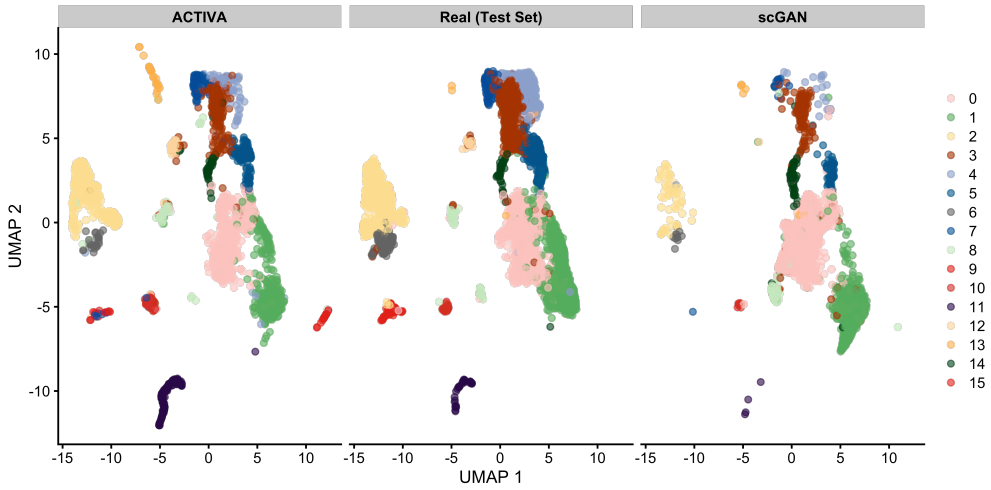


Figure A.1: **UMAP plot of ACTIVA generated cells compared with test set and scGAN generated cells, colored by clusters for NeuroCOVID**. ACTIVA's cell-type conditioning encourages the model to generate more cells per cluster, meaning that ACTIVA will generate more cells from the rare populations, as shown in the above visualization.

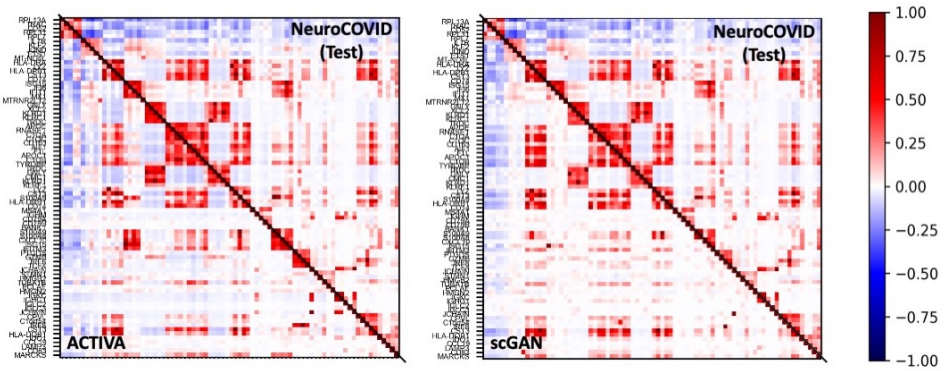


Figure A.2: **Results of correlation analysis for NeuroCOVID**. We performed the correlation analysis described in Section 5.3, where we looked at the correlation of the top genes five genes from each cluster and measured the pair-wise correlation of those genes in each dataset (test set, ACTIVA, and scGAN). As presented in this figure, ACTIVA has a closer relationship to the real data than scGAN, achieving a correlation discrepancy (CD) score of 4.19 compared to scGAN's 7.31 (lower CD is better).

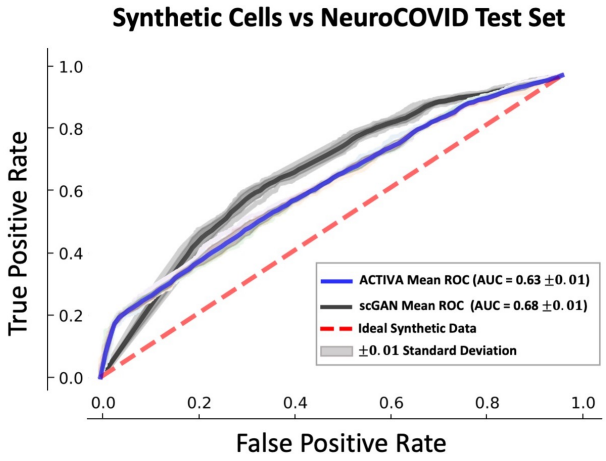


Figure A.3: **Random Forest identification of synthetic data (generated from ACTIVA and scGAN) against the real data (test set)**. As described in Chapter 6, AUCs closer to 0.5 indicate better performance for the generative model since that means that the classifier could not properly distinguish the difference between real cells and generated cells.

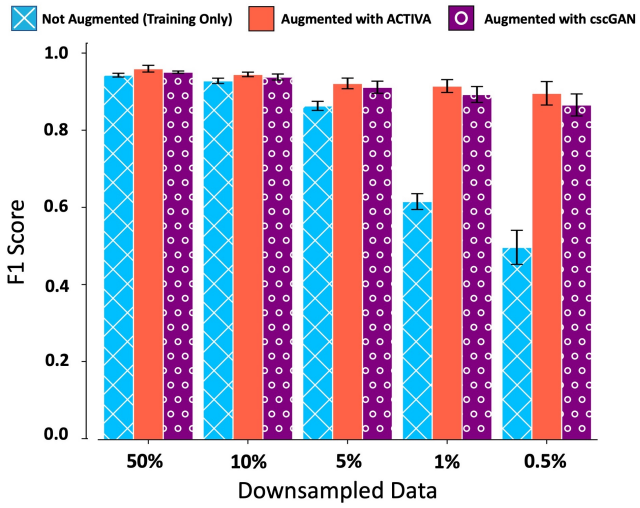


Figure A.4: **Mean F1 scores of Random Forest (RF) classifier for NeuroCOVID**. Training data (with no augmentation), shown in blue, and training data augmented with ACTIVA (shown in red) and scGAN (shown in purple), respectively. Error bars indicate the range for five different random seeds for sub-sampling cluster 7 cells in NeuroCOVID data.

CHOOSING ADVERSARIAL CONSTANT m

Ensuring the numerical balance between the KL divergence regularization of real and fake samples is crucial to ACTIVA's sample quality. Therefore, an m that is extremely large or small could affect the quality of the generated cells. The adversarial training in IntroVAE is similar to Energy-based GANs[315]. The following are two strategies we followed for choosing an appropriate value of m, based on[315]:

1. An effective strategy is to train the model as VAE for n epochs and track the minimized $\mathbb{KL}$ divergence. This will estimate the capacity of *Gen*'s reconstruction of single cells without a critic's input. Then, set m to be close to minimized $\mathbb{KL}$ divergence after n epochs. Our package already implements this feature, with a default VAE-only training of 10 epochs. In practice, we found that values of m roughly close to this minimized divergence performed well.
2. Another strategy for choosing m can be a rough grid-search starting from large values of m (which could be the upper bound of $\mathbb{KL}$ values) and gradually going to 0.

NETWORK ARCHITECTURE

In this section, we present the architecture of our model with the input $x \in \mathbb{R}^M$. Latent vectors of our model live in a 128-dimensional space, but for the sake of generality, we assume $z \in \mathbb{R}^D$. In the encoder and generator networks, Adam[286] optimizer is used with a learning rate $lr = 0.0002$, and moving averages decay rates $\beta_1 = 0.9$, $\beta_2 = 0.999$. Gradients are calculated on mini-batches of size 128 with the adversarial constant $m = 110$. We describe each component in more detail in A.

TRAINING AND INFERENCE PROCEDURE

Unlike GANs, our model does not require a training schedule. The cell-type classifier in ACTIVA can be pre-trained or trained simultaneously with *Enc* and *Gen*. To start the adversarial training and pick an appropriate m, we first introduce the model as a VAE for 10 epochs while training ACTINN in parallel (these options are readily available and adjustable in our package). After the initial warm-up, we train the IntroVAE component for 600 epochs with $\alpha_1 = 1$ and $\alpha_2 = 0.5$. We also provide an option for dynamic weight-balancing using SoftAdapt that would be useful in the case of a posterior collapse (which did not occur in our experiments).

For inference, we input a random noise tensor sampled from a multi-variate Gaussian to the trained generator. For cell-specific generation, outputs are automatically

filtered through the trained classifier to produce the desired sub-populations on demand.

ENCODER NETWORK

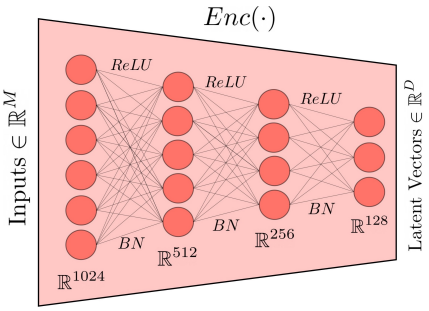


Figure A.5: **The encoder network of ACTIVA**.

Enc consists of fully connected layers, with Rectified Linear Units (*ReLU*)[168] as the activation between two layers, where we also perform batch normalization operation in[181] (denoted as *BN*) after *ReLU*. The input to the network is $x \in \mathbb{R}^M$, which goes through the network with layers $\{1024, 512, 256, 128\}$, as shown in Fig. A.5. Adam optimizer is used with a learning rate $lr = 0.0002$, and moving average decay rates $\beta_1 = 0.9$, $\beta_2 = 0.999$. Gradients are calculated on mini-batches of size 128, with the adversarial constant $m = 110$.

GENERATOR NETWORK

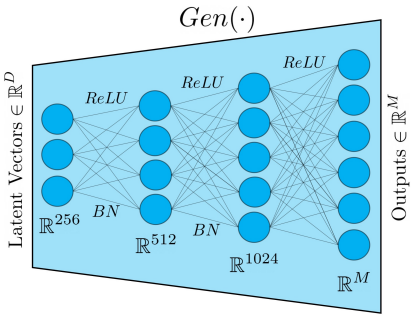


Figure A.6: **Architecture of ACTIVA's generator network**.

The generator network mirrors the encoder network, consisting of an input latent vector $z \in \mathbb{R}^D$ going through the layers $\{256, 512, 1024, M\}$ as shown in Fig. A.6. Note that in the last layer of the generator (mapping from 1024 to M), we use *ReLU* without *BN*. This is because we want to ensure that all generated values are non-negative. Similar to the encoder, we optimize the network using Adam with a learning rate $lr = 0.0002$, $\beta_1 = 0.9$, $\beta_2 = 0.999$. Gradients are calculated on 128-cell mini-batches.

AUTOMATED CELL TYPE NETWORK (ACTINN)

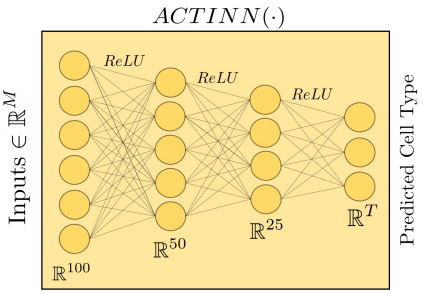


Figure A.7: **Automatic cell-type identification network of ACTIVA, which is ACTINN**.

ACTINN[133] uses a fully connected neural network architecture for the supervised classification of cell types. scRNAseq data is usually high dimensional and often sparse, making neural networks a promising method for analyzing such data. We implemented ACTINN in PyTorch to identify cell types and conditioning ACTIVA. Our implementation has the same architecture as in[133], which was implemented in TensorFlow (A.7). Cross entropy measures the loss between the predicted classes and the actual cell types. Optimization is done with Adam, and an exponential "staircase" decay is used, with the initial learning rate being $lr = 0.0001$ and a decay rate of 0.95 applied after every 1000 optimization steps. Our implementation of ACTINN in PyTorch differs from the original implementation in two ways: (1) we do not use a SoftMax layer between the last hidden layer and the output layer (due to the implementation of cross-entropy in PyTorch), and (2) we train for fewer number of epochs (between 5-10 as opposed to 50 epochs in the original implementation). Although we train for fewer epochs than[133], we did not notice a drop in the accuracy of our model; that is, our results on 68K PBMC closely match the results found by[304] (results are shown in Tables A.6-A.7). Gradients are calculated on mini-batches of size 128.

ADDITIONAL RESULTS

DROPOUT RATE

A common feature in scRNAseq data is technical zero counts (referred to as “dropouts"), which can arise from low RNA capture. To analyze the dropout rates as a function of mean gene expression for our generated data, we use functions from M3Drop[316] to extract the observed dropout rates for ACTIVA and scGAN and compare those with the raw data for 68K PBMC, Brain Small and NeuroCOVID. Our analysis showed that our model produced a lower dropout rate for higher mean expressions, which we can attribute to the prior distribution assumption. Our dropouts were very close to scGAN’s for 68K PBMC, while for NeuroCOVID, our data resembled the real data more closely. In the Brain Small dataset, we observed a steeper dropout rate for ACTIVA in relation to the real dataset. We hypothesize that the steeper dropout rate is due to fewer samples in the training data, about four times fewer than 68K PBMC and NeuroCOVID.

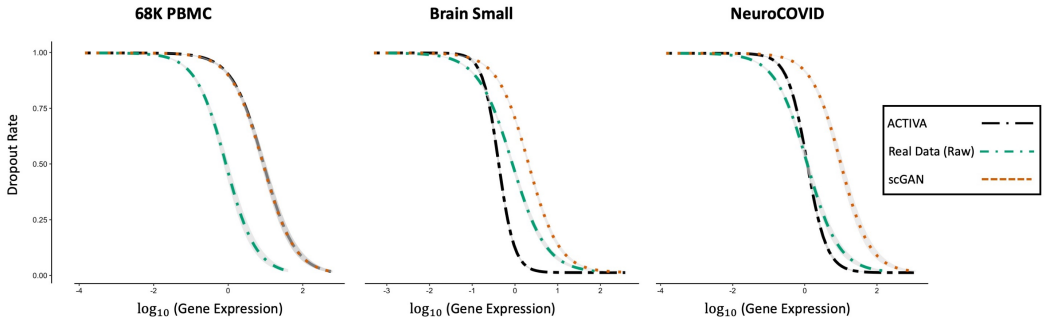


Figure A.8: **Observed dropout rates as a function of** $\log_{10}$**(Gene Expression) for all three datasets**. The dropout rate for ACTIVA is shown in Black (two-dashed line), Real in green (dot-dashed line), and scGAN in orange (dotted line) with the 95% confidence interval for each line shown in grey. Data was fit using a binomial generalized linear model.

DIFFERENTIAL STATES IN CLUSTERS

As another evaluation metric for ACTIVA’s generative quality, we perform a differential state (DS) analysis using muscat[317]. DS accounts for sample-to-sample as well as cell-to-cell variability, allowing us to conclude extrapolated to the samples rather than cells[318]. In our case, we measure the sample-to-sample variability between generated and real cells (test data). The idea is that if the generative models generate

realistic samples, then there should be *fewer DS genes* about the real data. To perform the differential state (DS) analysis, we sample each data (ACTIVA, real test set, and scGAN) without replacement to create five pseudo-replicates for each dataset, which formed the “pseudobulk data". From this pseudobulk, ACTIVA was directly compared with the real (referred to as ACTIVA-Real in Fig. A.9), and scGAN was compared with the real (scGAN-REAL). For all three datasets, we find that the generated-real cluster pairs contain only a few DS genes compared to the real data, indicating a high quality of synthetic data. However, ACTIVA has fewer DS genes for NeruoCOVID and Brain Small than scGAN, while scGAN has fewer DS genes in the 68K PBMC clusters.

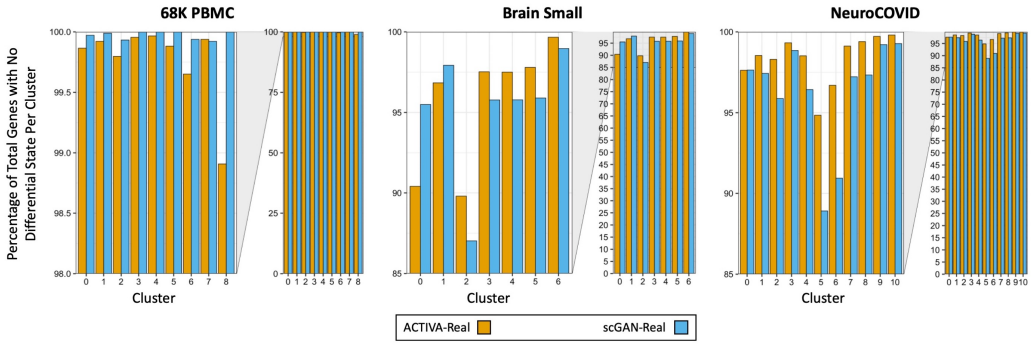


Figure A.9: **Barplots displaying the percent of genes with no differential state per cluster for all three datasets**. Values closer to 100% indicate a better match with the real data. In orange, ACTIVA is compared with the real data (test set), and in blue, scGAN is compared with the real data (test set). Our results show a high null differential state per cluster for all three datasets. The statistical analyses were performed in clusters with a sufficient number of cells (clusters with too few cells are not shown here).

DOWNSAMPLING AND DATA AUGMENTATION

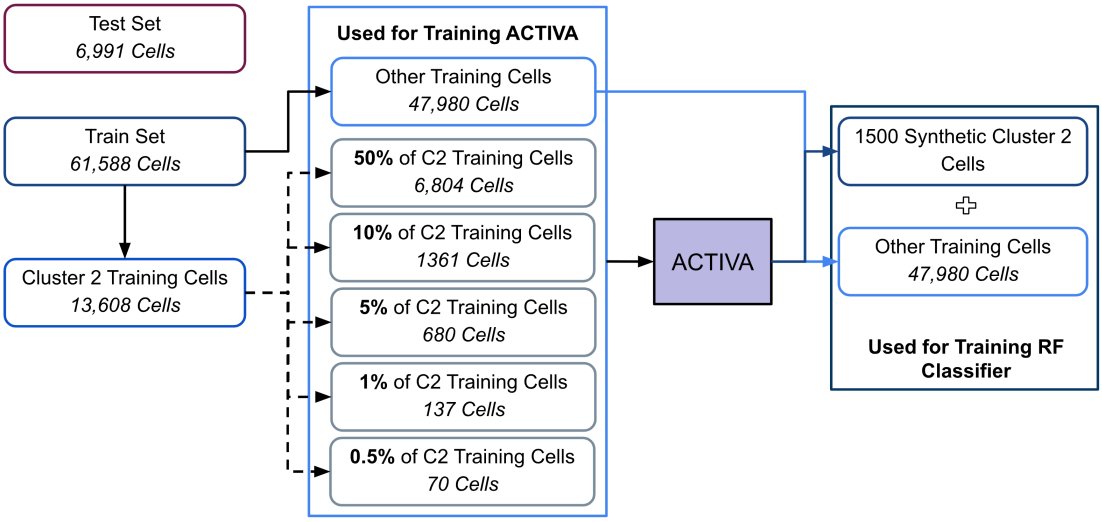


Figure A.10: **Downsampling process for evaluating the impact of data augmentation with ACTIVA**. Test splits are shown in red frames, and Training splits are in blue frames, with ACTIVA generated in purple frames. First, we separate the test set from the training set and then label cluster 2 cells to differentiate them from all other clusters. Cells from all other clusters (besides cluster 2) are used in all training and testing modes. For cluster 2 cells, we randomly subsample a fraction of the cells (10%, 5%, 1% or 0.5%) and use this subset in addition to all other cells to train ACTIVA. In other words, ACTIVA and the RF will include (i) training data from all other clusters and (ii) one of the downsampled versions of cluster 2 cells (highlighted in light blue). For the performance evaluation of RF without data augmentation ("no-augmentation"), we only use the desired cluster 2 subset and all other training cells to train the classifier. For training mode ACTIVA augmentation, we generate 1500 cells for data augmentation and add to the training cells we used in "no-augmentation" mode. So for training the RF in augmentation mode, we use (i) training data from all other clusters, (ii) one of the downsampled version of cluster 2 cells (highlighted in light blue) and (iii) 1500 ACTIVA generated cells [trained on the same downsampled data in (ii)].

ACCURACY OF THE CLASSIFIER ON EACH DATASET

As mentioned in Chapter 6, the sub-population cell generation depends on correctly classifying the cell types. We show that ACTIVA's classifier, ACTINN, can

classify rare-cell populations with training cells as few as 71 cells (see Table A.7). We did observe that in some cases, if the number of training samples for a specific cluster is extremely low, then ACTIVA does not learn that cell type (as is expected from any machine learning model); for example, the classifier does not learn the 10th cell-types of the PBMC data, since there were only 19 training cells available (out of 61K training cells). However, for the cluster 2 cell population, having 50 cells resulted in an F1 score of 0.74. This was useful for studying the impact of data augmentation with ACTIVA, as described in Chapter 6.

Table A.5: **Accuracy of ACTIVA's classifier network (ACTINN) on the test sets of Brain Small, 68K PBMC, and NeuroCOVID**. Three metrics were used: (1) Accuracy-number of correct predictions overall predictions; (2) *F1 Score (Non-Weighted)*: unweighted mean of per-label accuracy (not counting for cell-type imbalance); and (3) *Weighted F1 Score*: per-type accuracy but weighted by the number of cells for each cell-type.

Test Set	Accuracy	F1 Score (Non-Weighted)	Weighted F1 Score
Brain Small	0.9649	0.9674	0.9654
68K PBMC	0.9223	0.7448	0.9216
NeuroCOVID	0.9790	0.9697	0.9790

Table A.6: **Accuracy of ACTIVA's classifier on Brain Small**. Here we present the accuracy of ACTIVA's classifier network on the test and training set for the Brain Small dataset (see main text Section 4.3 and Fig. 3).

Cluster	Testing Cells	Test Precision	Test Recall	Test F1-Score		Training Cells	Train Precision	Train Recall	Train F1-Score
0	978	0.97	0.97	0.97		8808	0.99	1.00	0.99
1	304	0.98	0.99	0.98		2738	0.99	1.00	0.99
2	271	0.90	0.93	0.92		2439	0.98	0.99	0.98
3	182	0.99	0.96	0.97		1646	1.00	0.98	0.99
4	141	0.99	0.94	0.97		1271	1.00	0.99	0.99
5	59	0.98	0.98	0.98		535	1.00	0.99	1.00
6	35	1.00	0.94	0.97		323	1.00	0.97	0.98
7	27	1.00	0.93	0.96		243	0.99	0.99	0.99

Table A.7: **Accuracy of ACTIVA's classifier on 68K PBMC**. Here we present the accuracy of ACTIVA's classifier network on the test and training set for 68K PBMC dataset(see main text Section 4.3 and Fig. 3).

Cluster	Testing Cells	Test Precision	Test Recall	Test F1-Score		Training Cells	Train Precision	Train Recall	Train F1-Score
0	1791	0.89	0.90	0.90		15768	0.99	1.00	1.00
1	1545	0.88	0.87	0.88		13608	1.00	0.99	1.00
2	1515	0.93	0.96	0.95		13344	1.00	1.00	1.00
3	697	0.91	0.87	0.89		6145	1.00	0.99	1.00
4	483	0.99	0.98	0.99		4258	1.00	0.99	1.00
5	466	0.97	0.97	0.97		4105	1.00	1.00	1.00
6	413	1.00	1.00	1.00		3644	1.00	1.00	1.00
7	71	0.98	0.82	0.89		626	1.00	0.95	0.97
8	8	0.00	0.00	0.00		71	1.00	0.68	0.81
9	2	0.00	0.00	0.00		19	0.00	0.00	0.00

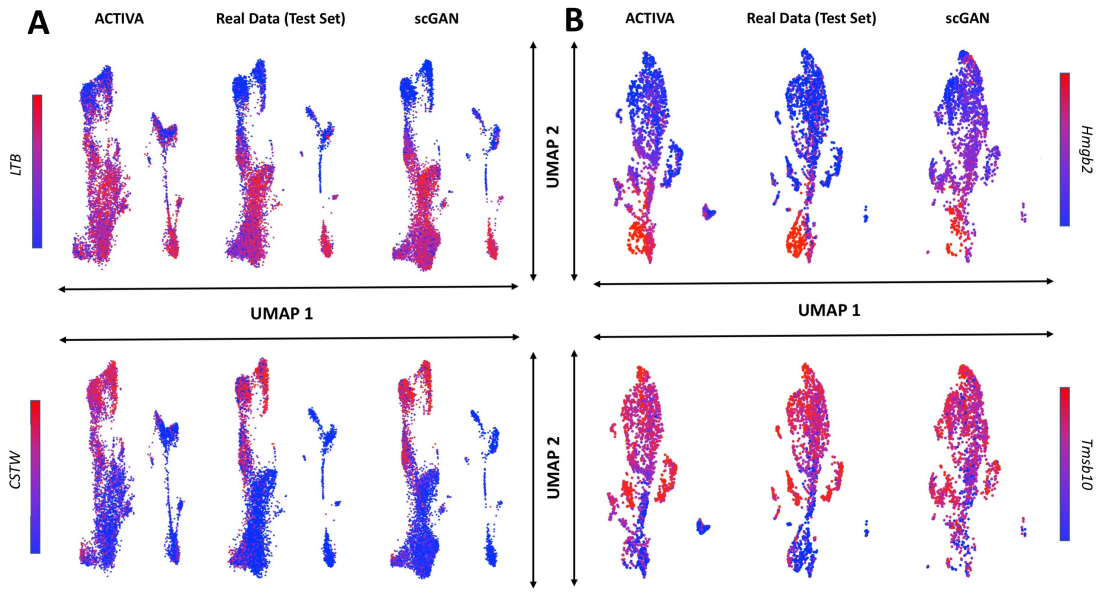


Figure A.11: **UMAP of Synthetic Cells (generated by ACTIVA and scGAN with a subset of real data as training data) compared to real data (not used in training) colored by gene expression**. Column (A). **Column A**: here, we present results from 68K PBMC data with a UMAP of 6991 cells generated by ACTIVA and scGAN, respectively, along with the test set, colored by the gene expression of two markers genes. **Column B**: we show the results on the Brain Small test set and 1997 ACTIVA and scGAN generated cells, colored by the gene expression of two marker genes. For both datasets, we see that cells generated by ACTIVA resemble the real data well, while more diversity is present among the generated cells.

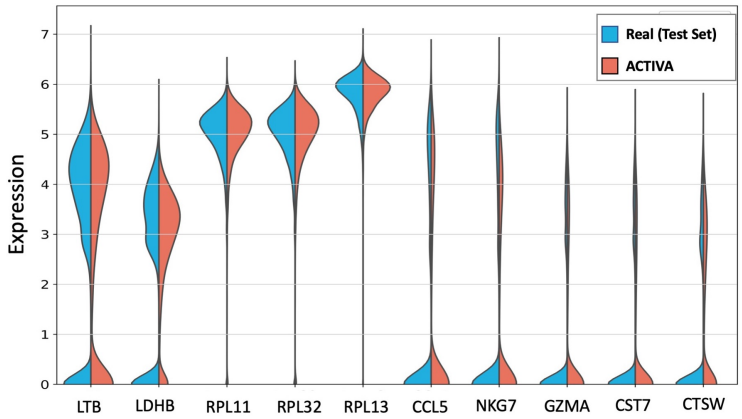


Figure A.12: **Logarithmic expression of the top marker genes**. The first five are for cluster 1, and the last 5 for cluster 2 in 68K PBMC test set (in blue) and ACTIVA generated cells (in red). The similarity between the distributions indicates that ACTIVA has learned the underlying marker gene expression.

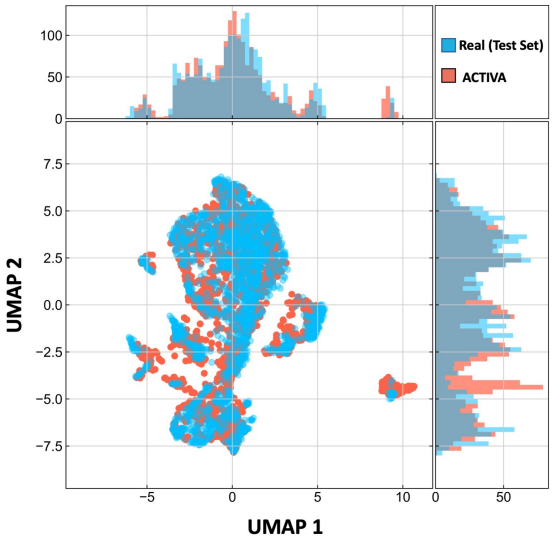


Figure A.13: **UMAP of cells generated by ACTIVA compared to real test cells from 20K Brain Small**. The histograms on the top and right of the UMAP plot display the counts of cells on the horizontal and vertical axis, respectively. This figure shows that ACTIVA has learned the underlying distribution of the real data while having some diversity in the generated samples (which is desired for generative models).

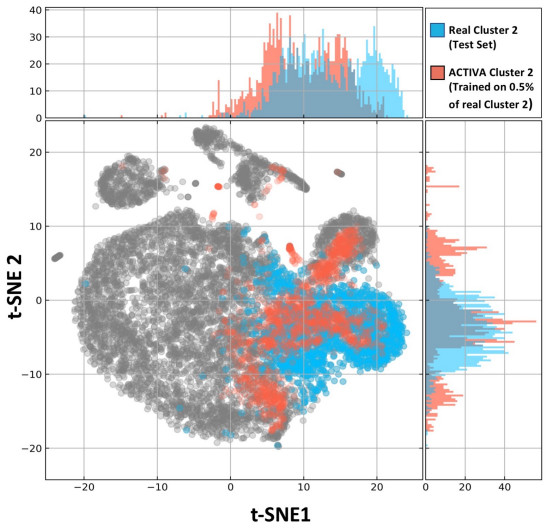


Figure A.14: **Qualitative comparison of ACTIVA-generated cells with real test set data**. *t*-SNE plot of all the cells in the test set (in grey), real cluster 2 cells (in blue), and ACTIVA generated cells when trained with only 0.5% of cluster 2 cells (in red). This figure illustrates that ACTIVA learns to generate a sub-population even when trained on 239 cells (out of 7915 training cells). This qualitative evaluation, combined with the results in Chapter 6, shows the promising application of ACTIVA for improving the downstream classification of rare populations.

B

Reconstruction Loss for VAEs with Gaussian Assumptions

The expected negative reconstruction error L_{AE} is given by:

$$L_{AE} = \mathbb{E}_{z\sim q(z|x)} \left[\log p_\theta(x|z)\right].$$

In many cases, this term is “chosen" to be the mean squared error (MSE) between the training data and reconstructed samples. However, it is important to formally derive the reconstruction loss since the true reconstruction loss is not MSE alone, as shown below. For simplicity, we normalize the expected reconstruction error by the number of samples n as

$$L_N = \frac{1}{n}\mathbb{E}_{z\sim q(z|x)} \left[-\log p_\theta(x|z)\right],$$

and let $d(x,y) = \|x-y\|_2^2$. Moreover, for simplicity, we assume that $\sigma_\theta^2(z) = \sigma_\theta^2 I$. Now given the Gaussian assumption on the likelihood $P_\theta(x|z)$, we have the normalized reconstruction loss as:

$$L_N = \frac{1}{n}\mathbb{E}_{z\sim q(z|x)} \left[\frac{1}{2}\left(\log(2\pi\sigma_\theta^2) + \frac{d\left(x,\mu_\theta(z)\right)}{\sigma_\theta^2}\right)\right].$$

Defining $MSE = \frac{1}{n}\mathbb{E}_{z\sim q(z|x)}d(x,\mu_\theta(z))$ and substituting in the above expression yields:

$$L_N = \frac{1}{2}\left(\log(2\pi\sigma_\theta^2) + \frac{MSE}{\sigma_\theta^2}\right) = \alpha + \beta MSE, \quad \text{for } \alpha,\beta \in \mathbb{R}.$$

Now we can explicitly assuming that $\sigma_\theta^2 = \frac{1}{2}$ will give us:

$$L_N = \frac{\log(\pi)}{2} + MSE.$$

We can see that choosing MSE as the reconstruction loss simplifies the true reconstruction loss, which could be sub-optimal in some cases. However, it is common to take MSE as the reconstruction loss when training a VAE model with a Gaussian Likelihood function [*e.g.* as done in Daniel *et al.*[319]].