Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів аналізу даних scRNA-seq за допомогою глибинних нейронних мереж
(тема)

Виконав:
студент 2 курсу, групи СШІм-19-2
Баранов Є. О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник доц. Шергін В. Л.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Баранову Євгенію Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів аналізу даних scRNA-seq за допомогою глибинних нейронних мереж
затверджена наказом університету від 29 березня 20 21 р. № 390 Ст
2. Термін подання студентом роботи до екзаменаційної комісії _____ травня 20 21 р.
3. Вихідні дані до роботи Науково-технічні публікації, відкриті джерела в мережі Інтернет, відкриті набори даних, документація мови програмування Python, документація фреймворків для побудови нейронних мереж Tensorflow та PyTorch.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Огляд методів аналізу даних scRNA-seq на основі глибинних нейронних мереж
3) Програмна реалізація та порівняльний аналіз мереж

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	доц. Шергін В. Л.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на дипломну роботу	29.03.2021	Виконано
2	Аналіз предметної області та постановка задачі	1.04.2021-7.01.2021	Виконано
3	Огляд методів аналізу даних scRNA-seq	8.04.2021-12.04.2021	Виконано
4	Програмна реалізація та порівняльний аналіз	13.04.2021-19.04.2021	Виконано
5	Обробка і оформлення результатів	20.04.2021-21.04.2021	Виконано
6	Оформлення графічних матеріалів	22.04.2021	Виконано
7	Оформлення пояснювальної записки	26.04.2021-28.04.2021	Виконано
8	Попередній захист	14.05.2021	Виконано
9	Захист перед ЕК		

Дата видачі завдання 29 березня 2021 р.

Студент _____
(підпис)

Керівник роботи _____ _____доц. Шергін В. Л._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 94 с., 1 табл., 21 рис., 2 дод., 24 джерела.

АВТОКОДУВАЛЬНИК, ГЕНЕРАТИВНО-ЗМАГАЛЬНА МЕРЕЖА, ГЛИБИННА НЕЙРОННА МЕРЕЖА, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, РИБОНУКЛЕЇНОВА КИСЛОТА, СЕКВЕНУВАННЯ ПООДИНОКИХ КЛІТИН, СТАТИСТИЧНИЙ РОЗПОДІЛ.

Об'єкт дослідження – процес секвенування РНК методом секвенування поодиноких клітин та інтелектуальний аналіз отриманих результатів.

Предмет дослідження – методи аналізу даних секвенованої РНК за допомогою глибинних нейронних мереж.

Мета роботи – побудування оптимальної штучної нейронної мережі або набору мереж, яка б забезпечила очистку даних scRNA-seq, видалення шуму та зменшення розмірності даних для їх візуалізації для спрощення подальшої обробки інформації дослідниками.

Методи дослідження – порівняльний аналіз глибинних нейронних мереж та аналіз можливості їх застосування для вирішення різних задач.

В результаті кваліфікаційної роботи виявлено, що використання глибинної нейронної мережі sciGAN в поєданні з алгоритмічними методами для для видалення шуму та кластеризації даних дозволяє максимізувати ефективність обробки даних.

Значимість роботи оцінюється як висока, що обумовлено поточним станом в світовій системі охорони здоров'я і поточною пандемією короновірусної хвороби.

РЕФЕРАТ

Пояснительная записка: 94 с., 1 табл., 21 рис., 2 прил., 24 источника.

АВТОКОДИРОВЩИК, ГЕНЕРАТИВНО-СОСТЯЗАТЕЛЬНАЯ СЕТЬ, ГЛУБИННАЯ НЕЙРОННАЯ СЕТЬ, ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ, РИБОНУКЛЕИНОВАЯ КИСЛОТА, СЕКВЕНИРОВАНИЕ ЕДИНИЧНЫХ КЛЕТОК, СТАТИСТИЧЕСКОЕ РАСПРЕДЕЛЕНИЕ.

Объект исследования – процесс секвенирования РНК методом секвенирования единичных клеток и интеллектуальный анализ полученных результатов.

Предмет исследования – методы анализа данных секвенированной РНК с помощью глубинных нейронных сетей.

Цель работы – построение оптимальной искусственной нейронной сети или набора сетей, которая бы обеспечила очистку данных scRNA-seq, удаление шума и уменьшение размерности данных для их визуализации для упрощения дальнейшей обработки информации исследователями.

Методы исследования – сравнительный анализ глубинных нейронных сетей и анализ возможности их применения для решения различных задач.

В результате квалификационной работы выявлено, что использование глубинной нейронной сети sciGAN в сочетании с алгоритмическими методами для для удаления шума и кластеризации данных позволяет максимизировать эффективность обработки данных.

Значимость работы оценивается как высокая, что обусловлено текущим состоянием мировой системы здравоохранения и мировой пандемией коронавирусной болезни.

ABSTRACT

Explanatory note: 94 p., 1 tabl., 21 fig., 2 ann., 24 sources.

AUTOENCODER, GENERATIVE ADVERSARIAL NETWORK, DATA ANALYSIS, DEEP NEURAL NETWORK, RIBONUCLEIC ACID, SINGLE CELL SEQUENCING, STATISTICAL DISTRIBUTION.

The object of research is the process of the RNA sequencing using the single-cell RNA sequencing method and data analysis of its results.

The subjects of research are scRNA data analysis methods using deep neural networks.

The objective of this research is assembling an optimal artificial neural network or a set of networks which would clean the input scRNA-seq data, perform denoising and dimensionality reduction for data visualization in order to simplify the data processing for researchers.

The research methods are comparative analysis of deep neural networks and analysis of the possibilities of their usage for various applications.

As a result of this work it is revealed that the use of deep neural network sciGAN in combination with algorithmic methods for noise removal and data clustering allows to maximize the efficiency of data processing.

The importance of this work is due to current state of the world healthcare system and the ongoing pandemic of the coronavirus disease.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

Експресія генів – процес, при якому спадкова інформація генів (нуклеотидна послідовність) використовується для синтезу функціонального продукту: білка або РНК;

Секвенування ДНК — набір біохімічних методів встановлення послідовності нуклеотидних основ ДНК: аденіну, гуаніну, цитозину і тиміну;

Транскрипт – молекула РНК, що з’являється в результаті експресії відповідного гена або ділянки ДНК;

In situ – аналіз будь-якого явища в тому місці, де воно відбувається, без необхідності переміщення до спеіального середовища;

MCMC – Markov Chain Monte Carlo – методи Монте-Карло марковських ланцюгів;

MMD – Maximum Mean Disrepancy – максимальна середня невідповідність;

MNN – Mutual Nearest Neighbour – алгоритм взаємних найближчих сусідів;

PCA – Principal Component Analysis – аналіз первинних компонентів;

PCR – polymerase chain reaction, полімеразна ланцюгова реакція – метод розмноження молекул ДНК за допомогою розриву подвійної спіралі та доповнення одинарних ланцюгів за допомогою полімерази;

RNA-seq – RNA sequencing – технологія секвенування РНК;

scRNA-seq – Single-cell RNA sequencing – секвенування РНК одинарних клітин;

t-SNE – t-distributed Stochastic Neighbor Embedding – T-розподілене вкладення стохастичної близькості;

UMI – Unique Molecule Identifier – унікальний модифікатор молекули.

ВСТУП

Сучасне суспільство у 21 столітті досягло значних успіхів в технологічному розвитку, однак і досі з'являються проблеми, які людство не може вирішити одразу швидко та ефективно. Однією з таких проблем є виникнення нових заразних та небезпечних для людини захворювань.

Завдяки розвитку науки та техніки людство навчилося використовувати різноманітні заходи для боротьби з новими зараженнями, створювати вакцини для вироблення імунітету від захворювань та змогло майже повністю знищити такі хвороби, як віспа, холера, чума та інші.

Розвиток науки та охорони здоров'я привів до суттєвого покращення рівня та довготривалості життя майже в усіх країнах світу, але незважаючи на це з'явилися нові проблеми, що несуть загрозу суспільству. Таким прикладом є епідемія ВІЛ/СНІД, яка забирає життя мільйонів людей щороку, і наразі немає жодного абсолютно ефективного методу лікування, в наявності лише методи зупинити прогрес хвороби, а ліки є лише експериментальні, без перевірок на ефективність та небезпечність. Також слід відмітити захворювання на рак, яке на більш пізніх стадіях невиліковне, а для більш ранніх стадій потребує радіаційної терапії, що може само завдати невиліковної шкоди організму хворого.

Ще однією загрозою виявилися епідемії атипічної пневмонії (SARS-CoV, 2002-2004 роки) та пандемія COVID-19 (SARS-CoV-2, 2019-наш час). Основною проблемою стала відсутність ефективного методу лікування, через що основними методами захисту населення стали ізоляція хворих та симптоматичне лікування.

Для тестування нових препаратів та методів лікування від існуючих хвороб необхідні методи відстежування змін в організмі людини. Незважаючи

на те, що при захворюванні на рак в організмі з'являються чітко виражені пухлини, відстежувати динаміку протікання захворювання за допомогою рентгенівських знімків просто так неможливо. Саме для цього і використовується технологія секвенування РНК.

Технології секвенування дозволяють відстежувати зміни в клітинах пацієнтів після лікування та оцінювати ефективність ліків. Крім того, можливість аналізувати процеси в клітинах робить технологію секвенування корисною як для розробки лікування раку [1], так і в процесі розробки та тестування нових вакцин [2].

Використання новітніх методів секвенування дозволяє досягати неймовірних результатів та дає розуміння процесів, що проходять в клітинах організму людини, але перед отриманням якихось висновків дані після секвенування треба обробити. Стохастична природа даних та мінливість процесів синтезу РНК вносить в дані такі проблеми як пакетний ефект (batch effect), події відсіву даних, проблеми наявності технічного шуму, тощо [3]. Детальніше ці проблеми будуть розглянуті в наступному розділі.

Для подолання таких проблем та отримання чистих даних наразі використовуються різноманітні методи – як ручні, так і з використанням різноманітних алгоритмів. Для пришвидшення процесів розробки та отримання нових ліків необхідним стає автоматичний процес аналізу даних секвенування РНК за допомогою нейронних мереж.

Дане дослідження в силу вищезазначених фактів і поточних обставин можна визначити як високо актуальне.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Історія технологій секвенування

Дезоксирибонуклеїнова кислота (ДНК) була вперше відкрита та виділена Фрідріхом Мішером у 1869 році, але вона залишалася недостатньо вивченою протягом багатьох десятиліть, оскільки, як вважалося, білки, а не ДНК, зберігали генетичний план життя. Ця ситуація змінилася після 1944 року в результаті деяких експериментів Освальда Евері, Коліна Маклауда та Макліна Маккарті, які показали, що очищена ДНК може змінити один штам бактерій на інший. Вперше було показано, що ДНК здатна трансформувати властивості клітин.

У 1953 році Джеймс Уотсон та Френсіс Крик висунули свою модель подвійної спіралі ДНК, засновану на кристалізованих рентгенівських структурах, які вивчала Розалінд Франклін. Згідно з моделлю, ДНК складається з двох ланцюжків нуклеотидів, згорнутих навколо один одного, з’єднаних водневими зв’язками і протікаючих в протилежних напрямках. Кожна ланцюг складається з чотирьох взаємодоповнюючих нуклеотидів – аденіну (A), цитозіну (C), гуаніну (G) і тиміну (T) – A на одній нитці завжди в парі з T на іншій, а C завжди в парі з G. Вони припустили, що така структура дозволяє кожному ланцюжку використовуватись для реконструкції іншого, ідея центральна для передачі спадкової інформації між поколіннями.

Фредерік Сангер – один з небагатьох вчених, якому було вручено дві Нобелівські премії – одну за секвенування білків, а іншу – за секвенування ДНК. Фундамент для секвенування білків був вперше закладений роботою Фредеріка Сангера, який до 1955 року завершив послідовність всіх амінокислот в інсуліні, невеликому білку, що виділяється підшлунковою

залозою. Це забезпечило перші незаперечні докази того, що білки були хімічними утвореннями зі специфічним молекулярним малюнком, а не випадковою сумішшю матеріалу, суспендованого в рідині.

Успіх Сенгера в секвенуванні інсуліну вплинув на розвиток рентгенівської кристалографії, зокрема на роботи Уотсона і Крика, які до цього часу намагалися зрозуміти, як ДНК керувала утворенням білків у клітині. Незабаром після відвідування ряду лекцій, проведених Фредеріком Сангером у жовтні 1954 року, Крик розпочав розробку теорії, яка стверджувала, що розташування нуклеотидів у ДНК визначає послідовність амінокислот у білках, що, у свою чергу, допомагає визначити функцію білка. Цю теорію він опублікував у 1958 р. Ілюстрація технології Сенгера зображена на рисунку 1.1.

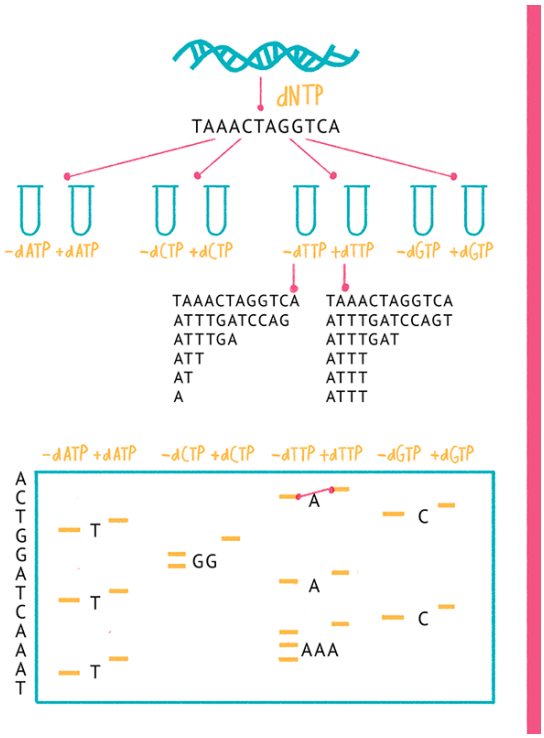


Рисунок 1.1 – Секвенування послідовності ДНК за допомогою методу Сенгера

Секвенування РНК було однією з найбільш ранніх форм секвенування нуклеотидів. Головною віхою секвенування РНК є знаходження послідовності першого повного гена та повного геному бактеріофага MS2, це було зроблено та опубліковано Уолтером Фієрсом та його колегами з Університету Гента (Гент, Бельгія), в 1972 р. та 1976 р.

Першим повноцінним геномом ДНК, який було секвеновано, був геном бактеріофага φX174 у 1977 р. Вчені Ради медичних досліджень Великобританії розшифрували повну послідовність ДНК вірусу Епштейна-Барра в 1984 році, виявивши, що він містив 172 282 нуклеотиди. Завершення послідовності ознаменувало важливий поворотний момент у секвенуванні ДНК, оскільки це було досягнуто без попереднього знання генетичного профілю вірусу.

Кілька нових методів секвенування ДНК були розроблені в середині та наприкінці 1990-х років і були впроваджені в комерційних секвенаторах ДНК до 2000 року. Разом вони були названі методами секвенування «наступного покоління» або «другого покоління» (NGS) для того, щоб щоб відрізнити їх від попередніх методів, включаючи секвенування Сенгера. На відміну від першого покоління секвенування, технологія NGS, як правило, характеризується високою масштабованістю, що дозволяє секвенувати весь геном одночасно. Зазвичай це досягається шляхом фрагментації геному на невеликі шматочки, довільної вибірки для фрагмента та його секвенуванням з використанням однієї з різноманітних технологій, таких як описані нижче. Обробка цілого геному можлива через те, що обробляється відразу декілька фрагментів (через що технологію NGS називають «масово паралельним» секвенуваням) в автоматизованому процесі.

1.2 Технологія RNA-seq

1.2.1 Загальний опис технології

RNA-Seq (названа абревіатурою «РНК-секвенування») – це техніка секвенування, яка використовує секвенування наступного покоління (NGS) для виявлення присутності та кількості РНК у біологічному зразку в даний момент, аналізуючи постійно мінливий клітинний транскриптом. Зокрема, RNA-Seq полегшує здатність розглядати альтернативні генно-сплайсинговані транскрипти, посттранскрипційні модифікації, злиття генів, мутації та зміни в експресії генів з плином часу, або відмінності в експресії генів у різних групах чи методах лікування. RNA -Seq також може бути використана для визначення меж екзону/інтрону. Останні досягнення RNA-Seq включають секвенування одиничних клітин та секвенування фіксованої тканини in situ – без необхідності використання спеціальних умов.

До створення RNA-Seq дослідження експресії генів проводили з мікрочипами на основі гібридизації. Проблеми з мікрочипами включають артефакти перехресної гібридизації, погану кількісну оцінку низько і сильно експресованих генів і необхідність знати послідовність ще до проведення експерименту. Через ці технічні проблеми транскриптомія перейшла до методів, заснованих на послідовності. Ці методи розвивалися від секвенування Сангера до хімічних методів (наприклад, серійний аналіз експресії генів) і, нарешті, до сучасної технології, секвенування кДНК наступного покоління (особливо RNA-Seq).

Технологічна платформа для швидкого широкомасштабного секвенування була створена в 2005 році фірмами 454 Life Sciences і Illumina (раніше Solexa), і спочатку використовувалася для секвенування

геномів. Перші роботи з секвенування транскріптомов з'явилися в 2008 році. У числі перших були секвенувати транскриптом дріжджів, арабидопсиса і миші.

В даний час РНК-секвенування здійснюється в основному з використанням трьох інструментальних платформ широкомасштабного секвенування: Illumina, 454 Life Sciences і SOLiD.

У 2019 вдалося секвенувати РНК зі шкіри, хрящів, печінки і скелетних м'язів цуценя Тумата вовка або собаки віком 14300 років.

1.2.2 Опис процесу секвенування

В залежності від інструментів та підходів процес секвенування відрізняється, але в ньому можна виділити деякі загальні моменти. На рисунку 1.2 зображений типовий процес секвенування.

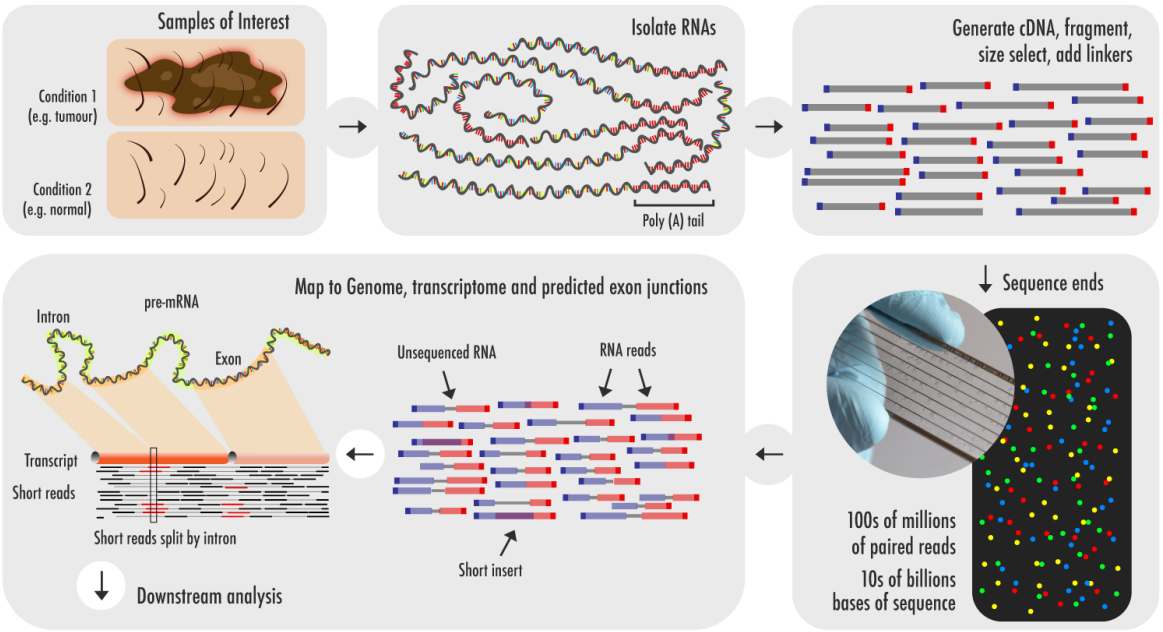


Рисунок 1.2 – Загальна схема секвенування РНК.

Першим кроком під час секвенування є вибір даних для секвенування. Наприклад, для секвенування можуть бути використані ділянки органу взяті у пацієнта хворого раком, одна ділянка з раковою пухлиною, а інша без. Іншим прикладом може бути забір тканини з легенів людини хворої на COVID-19 для оцінки пошкоджень та їх причин. Ці зразки будуть поміщені в спеціальний апарат, який буде проводити аналіз, але перед цим потрібно провести додаткову обробку.

Наступним кроком є підготовка бібліотеки (library preparation), під час цього кроку відбувається виділення тієї РНК, яка буде аналізуватися.

Деякі види РНК, такі як рибосомна РНК (рРНК), можуть становити до 80% загальної клітинної РНК. Частіше за все, цей тип РНК не потрібно аналізувати і він не цікавий для дослідження, тож його треба прибрати. Це допомагає забезпечити відстеження рідкісних транскриптів які представляють цінність. рРНК можна прибрати за допомогою різних хімічних методів, таких як ензимна дегідратація, тощо.

Апарати для проведення секвенування мають свої обмеження і можуть правильно обробляти лише послідовності обмеженої довжини. Один з типів РНК, мікро РНК, досить короткий (становить лише 18-25 нуклеотидів), тож може бути опрацьований повністю, для всіх інших видів потрібно провести фрагментацію, щоб отримані фрагменти були менше 200~250 нуклеотидів.

Також слід зазначити, що секвенатори вміють працювати лише з ДНК, тож РНК доведеться перетворити на ДНК. Для цього за допомогою речовини під назвою «зворотня транскриптаза» відбувається процес добудування ланцюгів ДНК, які є комплементарними (доповнювальними) до ланцюга РНК. Отримана ДНК відповідно називається кДНК (комплементарна ДНК).

Наступним кроком до кДНК додаються спеціальні адаптери для визначення напрямку, оскільки ДНК не симетрична і треба знати де її початок,

а де кінець. Для цього використовуються спеціальні сполуки – адаптери, які з'єднуються з обома кінцями молекули кДНК і дозволяють правильно встановити кінцеву послідовність РНК.

Ілюстрація процесів підготовки бібліотеки та інших супутніх процесів зображена на рисунку 1.3.

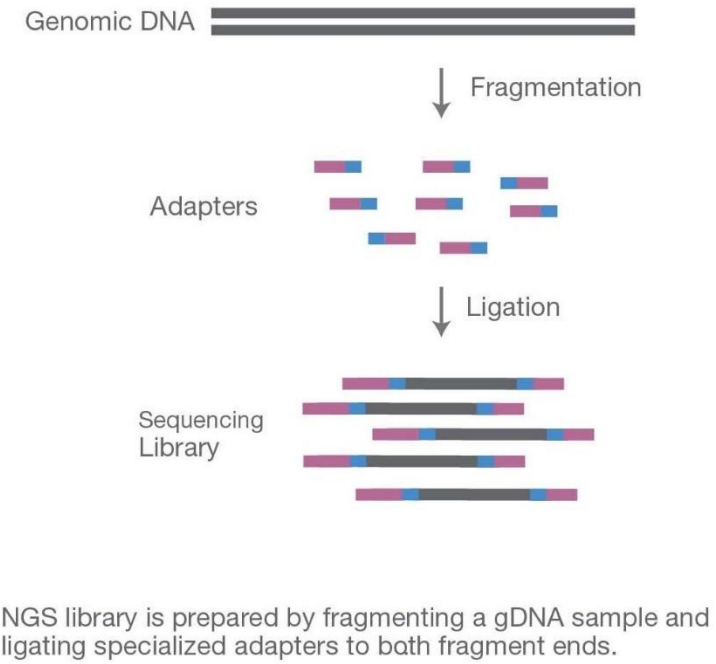


Рисунок 1.3 – Процес підготовки бібліотеки та інших супутніх процесів

Далі фрагменти кДНК закріпляються на матриці та використовується процес ПЛР для того, щоб розмножити ці фрагменти для того, щоб ніякий фрагмент ДНК не зник в процесі секвенування.

Останнім технічним кроком є сам процес секвенування, він виконується спеціальним пристроєм – секвенатором. В залежності від виробника принцип роботи може відрізнятися. Наприклад, в пристроях фірми Illumina використовуються флуоресцентні маркери в хімічних зв'язках з нуклеотидами.

Коли нуклеотид під'єднується до фрагменту кДНК, що аналізується, маркер вивільнюється і подає світловий сигнал, який фіксує секвенатор.

Іллюстрація цього процесу зображена на рисунку 1.4.

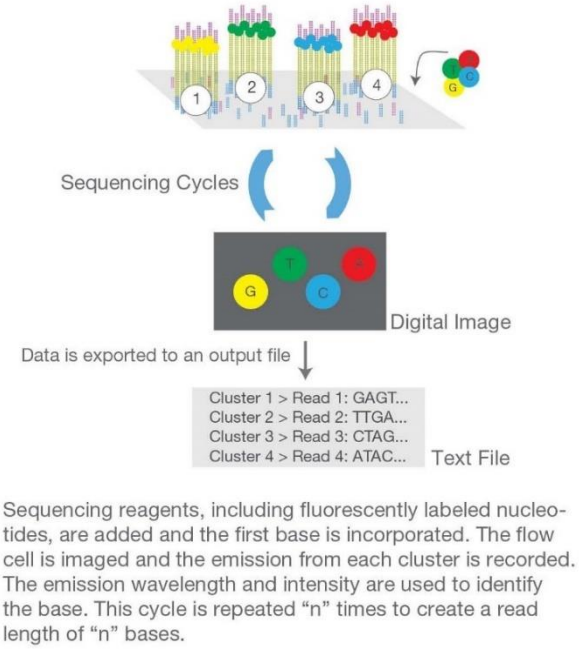


Рисунок 1.4 – Опис процесу секвенування Illumina

Останнім етапом експерименту є аналіз даних та інтерпретація результатів. В сирому вигляді вихідними даними є файл формату FASTQ, який містить дані про кожну прочитану молекулу. Ці файли за розміром досить великі і аналізувати їх вручну неможливо. Наприклад, інструмент Hiseq2000 від компанії Illumina в результаті експерименту може зібрати 200 мільйонів записів 100-нуклеотидних молекул ДНК. Ці дані займають приблизно 50 Гб і їх ручний аналіз не є доцільним через кількість витраченого часу.

Далі в автоматичному процесі файл з записами потрапляє на процедуру вирівнювання даних, фільтрація помилок, а також зіставлення отриманих результатів з таблицею геномів. Отож, фінальним результатом процесу

секвенування РНК є файл FASTQ, а також файл з зіставленими генами та кількістю їх входжень, які знайшов секвенатор.

1.3 Секвенування РНК одинарних клітин

Основною проблемою стандартного секвенування є той факт що весь зразок в процесі секвенування аналізується разом, тож інформація про клітини недоступна. Це становить проблему, тому що часто в процесі аналізу науковців цікавлять саме унікальні клітини, наприклад ракові клітини, які мають деякі мутації. Крім того, окрема інформація про кожну клітину дозволяє проаналізувати структуру та поведінку різних типів клітин.

Для отримання інформації про окремі клітини використовується секвенування одинарних клітин. Ілюстрація технології scRNA-seq надана на рисунку 1.5.

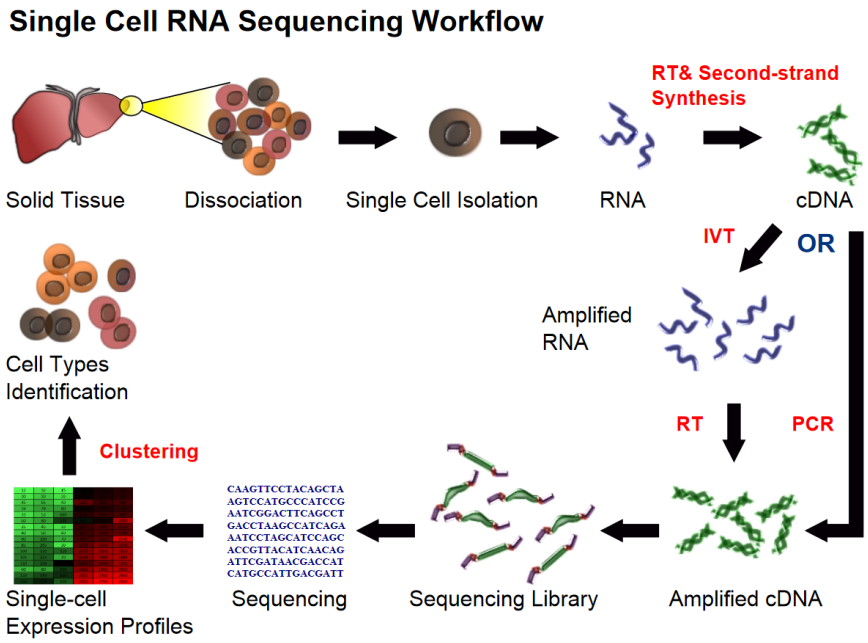


Рисунок 1.5 – Опис процесу секвенування РНК одинарних клітин

Під час цього процесу відбувається ізоляція окремих клітин, потім проводиться процес перетворення РНК на кДНК з використанням зворотної транскриптази, потім ДНК розмножується і аналізується аналогічно з простим RNA-seq.

Технологія scRNA-seq з'явилася в 2009 році і до 2014 року була не поміченою широкою публікою, однак після падіння ціни і підвищення доступності технологій секвенування, scRNA-seq набула стрімкої популярності.

1.4 Основні проблеми аналізу даних scRNA-seq

Для розробки корисних алгоритмів та програмних засобів для аналізу даних scRNA-seq важливо вирішити і зрозуміти обчислювальні завдання, які виникають у завданнях аналізу даних. Хоча для вирішення проблем запропоновано багато підходів, все ще потрібні нові та кращі методи. Більше того, із швидким розвитком технологій одинарних клітин з'являються нові проблеми аналізу даних. Далі наведено перелік основних завдань, які потребують вирішення:

– пакетний ефект (batch effect);

– випадання даних (dropout);

– наявність технічного шуму;

– «прокляття розмірності» (curse of dimensionality);

– проблема масштабування даних.

Детальніше кожна з проблем буде розглянута нижче.

1.4.1 Проблема пакетного ефекту

Дані з високою пропускною здатністю RNA-seq для одинарних клітин часто збираються кількома партіями, з різними умовами, платформами або різними лабораторіями. Неминуче, що різниця між партіями призводить до різних значень експресії генів, які можна сплутати з біологічними варіаціями через міжклітинну гетерогенність. Таке технічне упередження називається пакетним ефектом (batch effect). Якщо його не виправити, пакетний ефект призведе до помилкових структур даних і помилкових висновків при подальшому аналізі. Варто зазначити, що пакетний ефект не є унікальним для аналізу даних одинарних клітин і може бути присутнім і при використанні інших технологій секвенування за умови наявності декількох партій з різних платформ, тощо.

1.4.2 Проблема випадання даних

Для деяких слабо експресованих генів їх невелика кількість молекул РНК та стохастичний характер транскрипційних процесів можуть призвести до помилкових нульових записів у матрицях експресії даних scRNA-seq. Це називається випаданням (dropout). Щоб виправити упередження з появою нульових даних, спричинене відсівом, нещодавно були запропоновані численні статистичні методи (особливо імпутація).

Слід окремо зазначити, що термін dropout також використовується в дослідженнях глибоких нейронних мереж. Українською мовою цей термін зазвичай перекладається як «виключення» та використовується для опису процедури при навчанні глибокої нейронної мережі, коли на кожному кроці навчання частина нейронів випадковим чином вимикається та не бере участь в

навчанні. Таким чином, на кожному кроці тренування глибокої нейронної мережі відбувається навчання різних підмереж, що дозволяє пришвидшити навчання та не дозволяє індивідуальним нейронам «помирати».

1.4.3 Проблема появи технічного шуму

На додаток до пакетного ефекту та випадання, деякі інші технічні фактори можуть також спричинити упередження у даних scRNA-seq, особливо для слабо виражених генів, такі як зміщення ампліфікації кДНК, ефекти клітинного циклу, недостатня довжина послідовності нуклеотидів тощо, і такі упередження називаються технічним шумом. З іншого боку, дані одинарних клітин містять внутрішню біологічну мінливість, яка може дати цінні уявлення про механізми регуляції генів на рівні індивідуальних клітин. Тому надзвичайно важливим та складним є відокремлення технічного шуму від біологічного шуму.

1.4.4 Прокляття розмірності

Поняттям «прокляття розмірності» називають різні явища, що виникають при аналізі та систематизації даних у просторах високої розмірності, які не відбуваються в умовах низьких розмірів, таких як тривимірний фізичний простір.

Сутність проблеми полягає в тому, що коли розмірність збільшується, обсяг простору збільшується настільки швидко, що доступні дані стають розрідженими. Ця розрідженість є нездоланною проблемою для будь-якого методу, який вимагає статистичної значущості. Для отримання статистично обґрунтованого та надійного результату кількість даних, необхідних для

підтвердження результату, часто зростає в геометричній прогресії із розмірністю. Крім того, організація та пошук даних часто спирається на виявлення областей, де об’єкти утворюють групи з подібними властивостями.

Ключовим етапом аналізу даних scRNA-seq є зменшення розмірності. Як правило, набір даних scRNA-seq містить профілі експресії для великої кількості генів, кожен ген відповідає розмірності, а файл експресії кожної клітини відповідає точці даних у просторовому просторі станів клітин. На деяких етапах аналізу даних (наприклад, кластеризація) відстань між точками даних відіграє важливу роль. Однак у просторі високої розмірності, оскільки точки передачі даних стають розрідженими, міри відстані (наприклад, відстань Евкліда, відстань Махаланобіса та Манхеттенська відстань) втрачають свою ефективність, роблячи поняття найближчого сусіда незрозумілим, а проблеми аналізу даних складними. Більше того, це може призвести до проблеми перенавчання, особливо коли кількість точок даних відносно невелика.

Одним із способів полегшити проблеми, спричинені великою розмірністю, є збільшення даних, але в більшості випадків це було б неможливо, оскільки обсяг необхідних даних збільшувався б в геометричній прогресії із збільшенням розмірності. Таким чином, альтернативним і практичним рішенням є зменшення розмірності.

Перевірені на змодельованих та реальних даних, методи зменшення розмірності продемонстрували ефективність при роботі з наборами даних scRNA-seq високої розмірності та допомагають покращити ефективність різних подальших аналізів, наприклад кластеризації, візуалізації даних, виявлення типу клітин, реконструкція траєкторії розвитку, тощо. Однак існуючі методи зменшення розмірності все ще мають деякі обмеження, такі як відсутність стійкості до випадкової вибірки, нездатність виявити глобальні структури при фокусуванні на локальних структурах даних, чутливість до

параметрів та висока обчислювальна вартість.

1.4.5 Проблема масштабування даних

Хоча зменшення розмірності в основному стосується великої кількості генів у даних scRNA-seq, іншим ключовим параметром розміру даних є кількість клітин. Обидва параметри розміру даних ставлять проблему масштабованості. З моменту народження краплинної техніки scRNA-seq (тобто Drop-seq) кількість клітин, профільованих в кожному експерименті, сягала десятків тисяч, а часто і мільйонів. Високопродуктивні одноклітинні проекти, такі як Атлас людських клітин (HCA), генерують дані багатьох клітин, що вимагає більш ефективних та масштабованих алгоритмів для моделювання та аналізу даних. Наприклад, деякі методи зменшення розмірності та кластеризації вимагають множення двох матриць NxN, де N – кількість клітин. Окрім алгоритмічних нововведень, деякі паралельні та високопродуктивні обчислювальні техніки, наприклад, графічні процесори (GPU), також часто використовуються в різних областях біоінформатики.

1.5 Традиційні методи розв'язання задач аналізу даних scRNA-seq

З моменту появи нового методу секвенування для спрощення аналізу даних та вирішення проблем, що постають під час проведення аналізу була запропонована достатня кількість різноманітних рішень, що варіюються від найпростіших маніпуляцій з даними та підрахунків до досить складних алгоритмів, що враховують природу даних.

Зокрема, для вирішення проблеми випадання даних було запропоновано декілька методів, які комбінують існуючі дані за допомогою статистичних

методів.

Наприклад, був запропонований метод для вставляння значень, що випали, під назвою BISCUIT. Цей метод базується на принципі MCMC (Марківські ланцюги Монте-Карло) і будує на основі вхідних даних змішану модель процесів Діріхле. За допомогою цієї моделі відбувається ітеративний процес нормалізації даних та заповнення пропусків наявних в матриці. В якості результату цей алгоритм повертає результуючу матрицю, заповнену необхідними даними.

Іншою подібною моделлю є scUnif – уніфікований статистичний фреймворк, який можна використовувати як для звичайних результатів секвенування, так і для результатів обробки даних одинарних клітин [15]. Цей фреймворк також використовує модель, схожу на алгоритм BISCUIT, але його відмінною рисою є той факт, що цей фреймворк вирішує задачу навчання з вчителем, тож потребує розмічених даних генів на вхід в якості тренувального датасету. Це дозволяє підвищити якість навчання, однак цей же факт і є причиною низької розповсюдженості цього фреймворку – розмічувати гени досить складно, а в даних одинарних клітин в залежності від типу клітини інформація може докорінно відрізнятися.

Алгоритм MAGIC за основу бере марківську матрицю переходів – матрицю з імовірностями переходів з одного стану в інший в марківському процесі. Основним недоліком цього алгоритму є той факт, що окрім додавання відсутніх даних, алгоритм змінює також і існуючі результати секвенування, що вносить додаткові помилки і може не дозволити знайти значущі результати.

Для видалення технічного шуму з вхідних даних використовуються особливі методи, тому що вирішення цієї задачі вирішується не лише алгоритмами, але й за допомогою технічних рішень. Наприклад, для індивідуального врахування молекул використовуються спеціальні

послідовності нуклеотидів під назвою UMI (унікальний ідентифікатор молекули). Ці послідовності є унікальними і приєднуються до кожної молекули, що дозволяє ідентифікувати кожну унікальну молекулу після процесу ПЛР та видалити помилково зчитані молекули. Завдяки можливості відстежувати молекули та ідентифікувати їх, ця технологія стала де-факто стандартом при виконанні експериментів з секвенування РНК, а підтримка UMI кодів включена до складу майже всіх інструментів для декодування генів з послідовності нуклеотидів. Структура молекул з використанням кодів UMI показана на рисунку 1.5.

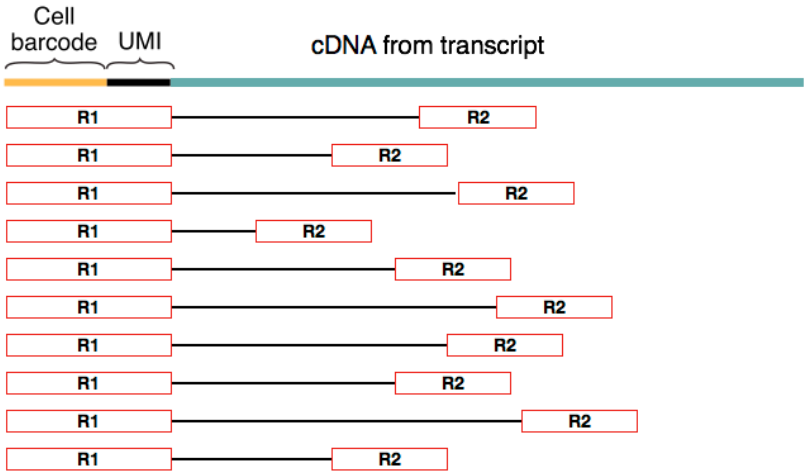


Рисунок 1.5 – Структура молекули кДНК з доданими кодами UMI

Також додатково в матеріал для аналізу окрім UMI також додають спеціальні послідовності РНК під назвою «транскрипт управління РНК», ці послідовності нуклеотидів структурно відомі завчасно, тож їх дуже легко знайти і підрахувати під час аналізу. Їх наявність дозволяє провести калібрування інструментів, тому що завчасно відомий транскрипт матиме приблизно такий же самий розподіл знайдених молекул, що й шукані гени, які

мають біологічно значущу інформацію.

Протягом останніх кількох років спостерігається зростання кількості методів зменшення шуму, наприклад, нормалізації даних, а також статистичних моделей, які б моделювали як біологічний, так і технічний шум.

Багато підходів до зменшення розмірності використовувались або розроблялись для даних scRNA-seq, наприклад, PCA (аналіз основних компонентів), карта дифузії t-SNE (T-розподілене вкладення стохастичної близькості), та ін. Нещодавно були розроблені деякі методи зменшення розмірності, що враховують характерні особливості, унікальні для даних scRNA-seq, такі як ZIFA, що вміють справлятися з випаданням даних.

Алгоритм ZIFA відрізняється тим, що він враховує наявність випадення даних (тобто розуміє, що вхідна матриця може містити велику кількість нулів) та за допомогою факторного аналізу моделює кореляції, на відміну від коваріацій (що використовується, наприклад, в алгоритмі PCA).

Незважаючи на наявність вищезазначених методів, вони вирішують різні задачі і роблять це не ідеально, через що є доцільним дослідження методів аналізу даних за допомогою глибинних нейронних мереж та вироблення універсального набору алгоритмів для повного аналізу даних scRNA-seq.

1.6 Постановка задачі кваліфікаційної роботи

Постановка задачі полягає у аналізі існуючих методів аналізу даних scRNA-seq з ціллю виявлення найкращих з них та формування остаточного процесу обробки даних секвенування.

Виходячи з цього можна сформулювати задачу кваліфікаційної роботи, яка буде складатися з наступних кроків:

– провести аналіз та розглянути наявні методи, що використовуються для аналізу даних scRNA-seq;

– зібрати датасети з даними scRNA-seq з відкритих джерел в мережі Інтернет та використати їх як набори даних для дослідження;

– детально розглянути наявні методи, що використовують глибинні нейронні мережі та порівняти їх ефективність з традиційними методами, за можливості додати до вже існуючих мереж зміни, які дозволять підвищити ефективність;

– запропонувати остаточний варіант глибокої нейронної мережі або низки мереж, які можна буде використовувати для задачі автоматизованого аналізу даних scRNA-seq.

2 ОГЛЯД МЕТОДІВ АНАЛІЗУ ДАНИХ SCRNA-SEQ НА ОСНОВІ НЕЙРОННИХ МЕРЕЖ

2.1 Методи аналізу даних scRNA-seq

Незважаючи на наявність традиційних методів аналізу даних, задачі аналізу даних scRNA-seq не були вирішені в цілому, тож дослідження та розробка нових методів були продовжені. Результатом таких досліджень стали нові методи обробки даних з використанням глибоких нейронних мереж, які досягали результатів в різних задачах [3]. Ці методи будуть розглянуті нижче.

Одним з перших покращень з використанням глибоких нейронних мереж була ідея Шахама [6] використанням залишкових нейронних мереж (residual neural networks) в задачі ідентифікації та видалення пакетного ефекту. Така архітектура нейронних мереж дозволила позбавитися ефекту зникаючих та вибухових градієнтів і дозволила створювати дуже глибокі нейронні мережі, які дозволяли впоратися з пакетним ефектом.

Більше того, для кодування функції втрат у ResNet було використано функцію максимальної середньої невідповідності (MMD, minimum mean disrepancy), міру відстані між двома розподілами ймовірностей. Автори застосували метод MMD-ResNet для усунення пакетних ефектів як у масс-цитометрії, так і в аналізі даних scRNA-seq.

Під час аналізу даних етап видалення пакетного ефекту хронологічно розташований перед кластеризацією клітин. Незважаючи на це, був запропонований новий алгоритм під назвою «deep embedding algoithm for single-cell clustering» («глибокий алгоритм векторного відображення даних для кластеризації одинарних клітин», скорочено «DESC»). Цей метод являється комбінованим і вирішує одразу дві задачі, як задачу видалення пакетного

ефекту, так і задачу кластеризації.

В цьому алгоритмі для задачі зменшення розмірності використовується архітектура автокодувальника в якості кроку попередньої підготовки та ініціалізації параметрів для ітеративної кластеризації.

Крім чисто статистичних методів, для задачі випадання даних був запропонований алгоритм під назвою «AutoImpute» [12], розроблений на основі автокодувальника. В цьому методі нейронна мережа використовується для догенерації матриці експресії з додатковим фокусом на поля з нульовими значеннями. Також у 2020 році був запропонований алгоритм вставки даних, що випали, за допомогою генеративно-змагальних мереж під назвою sclGAN [17].

Для спільного вирішення проблем технічного шуму та випадання даних був додатково запропонований метод DCA (deep counts autoencoder) на основі архітектури автокодувальника. Цей метод в кращу сторону відрізняється в порівнянні з алгоритмами scImpute, drImpute та іншими завдяки тому, що використання глибоких нейронних мереж дозволяє встановлювати нелінійні закономірності між різними генами і завдяки цьому вставляти правильні значення замість нульових, а також робить можливим масштабування до мільйонів клітин завдяки ефективності архітектури, а також можливості використання графічного процесора для обчислень.

Для вирішення задачі зменшення розмірності було запропоновано декілька різних рішень. Зокрема, була запропонована ідея шумознижуючого автокодувальника (denoising autoencoder, DAE) для виконання попереднього тренування. Результати роботи цієї мережі можуть бути використані для подальшого навчання без вчителя. Крім того, дослідження показали, що аналіз отриманої після тренування моделі навіть без подальшої роботи сам дозволяє отримати деяке розуміння біологічних процесів (оскільки під час тренування в

вагах мережі відбувається запам'ятовування шаблонів, притаманних специфічним біологічним процесам).

Крім цього, також була запропонована архітектура варіаційного автокодувальника (variational autoencoder, VAE). В цій архітектурі проходить підрахунок постеріорних імовірностей латентних змінних малої розмірності, за рахунок чого мережа навчається проводити перетворення даних з простору високої розмірності до простору низької розмірності.

Повний список розглянутих алгоритмів наведений в таблиці 2.1.

Таблиця 2.1 – Список розроблених алгоритмів мереж, які використовуються в задачах аналізу даних scRNA-seq.

Назва мережі	Рік створення	Метод в основі	Основна задача мережі
Метод Шахама	2016	Залишкова нейронна мережа	Видалення пакетного ефекту.
Метод Ліна	2017	Зменшення розмірності на основі PCA за допомогою шумознижуючих автокодувальників	Зменшення розмірності, групування клітин, знаходження типу або стану клітини
AutoImpute	2018	Введення даних в матрицю експресії генів на основі автокодувальників	Боротьба з даними, що випадають
scVI	2018	Ієрархічна Баєсівська модель та варіаційний автокодувальник	Боротьба з пакетним ефектом, упередженням через малий розмір бібліотеки, даними, що випадають та допомога для візуалізації даних
VASC	2018	Варіаційний автокодувальник	Моделювання ефектів випадіння даних та знаходження нелінійних відображень початкових даних

Продовження таблиці 2.1

Назва мережі	Рік створення	Метод в основі	Основна задача мережі
scvis	2018	Варіаційний автокодувальник	Моделювання та візуалізація структур в даних scRNA-seq
scScope	2019	Автокодувальник з рекурентною структурою	Видалення пакетного ефекту, введеня даних, що були втрачені через випадіння, ідентифікація субпопуляцій клітин
DCA	2019	Автокодувальник з функцією втрат ZINB	Видалення технічної різниці для боротьби з пакетним ефектом та покращення процедури аналізу
SAVER-X	2019	Баєсівська ієрархічна модель та глибокий автокодувальник з використанням трансферного навчання	Використання існуючих даних для покращення нових датасетів з даними scRNA-seq
sclGAN	2020	Генеративно-змагальна-мережа.	Введення даних, що були втрачені через випадіння.

Як можна побачити із таблиці, майже всі мережі лише використовують архітектуру автокодувальника. Це пов'язано з тим, що в датасетах scRNA-seq дуже важко знайти розмічені дані, тож є можливість використання тільки методів навчання без вчителя. Разом з тим, на вищезазначених задачах архітектура автокодувальника добре себе показує за рахунок того, що вона з початку дозволяє вивчити відображення більшого набору даних в менший набір, що дозволяє одразу вирішити такі задачі як вставлення даних, що випали, пакетного ефекту та технічного шуму.

2.2 Огляд архітектури автокодувальників

2.2.1 Опис загальної структури

Архітектура автокодувальника була запропонована в 2006 році Джеффрі Хінтоном та Русланом Салахутдіновим. [16] Ця мережа використовується для навчання без вчителя і її основною задачею є переведення даних в проміжне відображення (кодування, з англ. encoding), а потім мережа має з кодування як найточніше відновити вхідні дані. Завдяки цьому мережа навчається компактно кодувати та зберігати вхідну інформацію.

Тепер припустимо, що ми маємо лише приклади для тренування без маркування, позначимо цей датасет як $\{x^{(1)}, x^{(2)}, \dots, x^{(n)}\}$, де $x^{(i)} \in R_n$. Нейронна мережа з архітектурою автокодувальника – алгоритм навчання без вчителя, який застосовує алгоритм зворотнього розповсюдження помилки, що встановлюють цільові значення рівними вхідним. Тобто він використовує в основі рівність $y^{(i)} = x^{(i)}$. Архітектура стандартного автокодувальника зображена на рисунку 2.1.

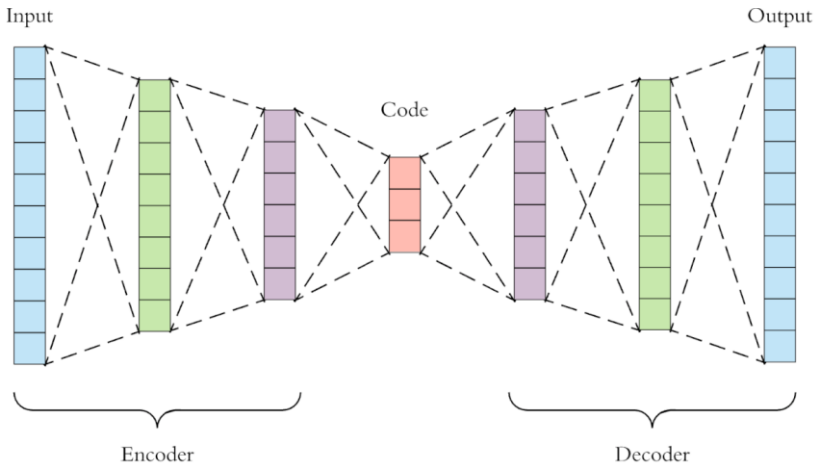


Рисунок 2.1 – Схема архітектури автокодувальника

Незважаючи на те, що з першого погляду можливість відтворити вхідні дані не здається дуже корисною, вся сила автокодувальників проявляється коли на мережу накладають обмеження. Наприклад, на рисунку 2.1 чітко видно, що частина мережі яка формує готове кодування є набагато меншою за розміром порівняно з вхідним шаром мережі. Через те, що мережа не може просто скопіювати дані і досягти цим якогось результату, в процесі навчання мережа навчається зберігати зменшене відображення вхідних даних та запам’ятовувати їх корисні аспекти, які потім допоможуть відтворити оригінальні дані.

Сучасні автокодувальники також пройшли стадію узагальнення і тепер розроблені архітектури мереж, що дозволяють проводити відображення не в якості детермінованої функції, а в якості стохастичного маппінгу. Такий підхід може використовуватися при генерації речень, де з відображення стохастично будуть вибиратися синонімічні терміни для генерації такого ж речення з точки зору сенсового навантаження.

В статті Хінтона [16] була відмічена ідея використання автокодувальників в якості основи для часткового навчання звичайних нейронних мереж. В такій схемі на першому кроці бралися вхідний та перший прихований шар мережі, далі додавався вихідний шар автокодувальника та проводилося тренування прихованого шару. В такій ситуації цей прихований шар навчався особливостям вхідних даних і зберігав деяке його відображення. Потім цей процес ітеративно повторювався для кожного наступного шару, доки всі приховані шари не мали натреновані ваги. Потім ці ваги використовувалися для подальшого тренування. В результаті подальших досліджень виявилося, що просте використання випадково згенерованих значень ваг для шарів нейронної мережі дає порівнянний, а часом й кращий результат ніж затратний та довгий процес попереднього навчання за

допомогою автокодувальника.

Приклад такого використання мережі буде розглянуто в наступному розділі.

2.2.2 Шумознижуючі кодувальники

Шумознижуючі автокодувальники є ще одним кроком розвитку архітектури автокодувальників. Для того, щоб покращити результати та підвищити активність навчання, зазвичай на шари мережі накладають додаткові обмеження, що мають назву «регуляризація». Наприклад, під час навчання можна замість звичайної функції витрат $L(x, g(f(x))$ використовувати спеціальну функцію витрат з додаванням регуляризуючого виразу **Ω**:

$$L(x, g(f(x)) + \Omega(h), \tag{2.1}$$

де L – функція витрат (наприклад, Еквлідова дистанція);

x – вхідні дані;

$g(f(x))$ – результат роботи автокодувальника;

$\Omega(h)$ – функція регуляризації, що залежить від виходу прихованого (кодуючого) шару h.

На відміну від таких архітектур, шумознижуючий автокодувальник не ставить за основу як найточніше відтворення вхідних даних, навпаки, в основі такого автокодувальника стоїть ідея, що вхідні дані несуть в собі непотрібний шум, від якого треба позбавитися. Навчання такої мережі можливе за виконання наступних двох умов:

– незважаючи на наявність шуму, при великій кількості даних є наявна

статистична стабільність;

– під час навчання мережі можна провести узагальнення та знайти в даних корисну інформацію.

В такому випадку для тренування такої мережі вхідні на дані x можна накласти перетворення $C(\bar{x} \mid x)$, яке представляє з себе стохастичне відображення x, з додаванням випадкового шуму. В результаті цього мета навчання змінюється: тепер автокодувальник має вивчити розподіл реконструкції $P_{reconst}(x|\bar{x})$.

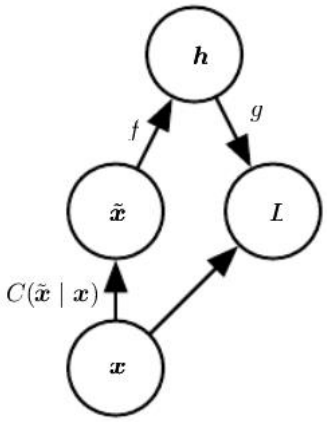


Рисунок 2.2 – Схема навчання шумознижуючого автокодувальника

Схема тренування такої мережі виглядає так, як показано на рисунку 2.2 і складається з наступних кроків:

– взяти нові дані x з тренувального датасету;

– взяти нові дані $\bar{x}$ за допомогою накладання перетворення $C(\bar{x}|\ x = x)$;

– використати пару $(x, \bar{x})$ як вхід мережі для отримання оцінки реконструкції розподілу:

$$P_{reconst}(x|\bar{x}) = P_{decoder}(x \mid h), \quad (2.2)$$

де h – результат роботи кодувальника *f(x̄)*;

$P_{decoder}(x \mid h)$ – розподіл отриманий з кодувальника, який вивчила мережа.

Якщо кодувальник не є стохастичним, така мережа може бути натренована як і будь-яка штучна нейронна мережа прямого розповсюдження. В результаті такого навчання навіть якщо в реальних даних наявний сторонній шум, використання такої мережі допоможе його позбутися.

В рамках даної кваліфікаційної роботи буде розглянуто метод Ліна. В цьому методі для вирішення задачі зниження розмірності тренується глибинна нейронна мережа прямого розповсюдження. Приклад такої мережі показаний на рисунку 2.3.

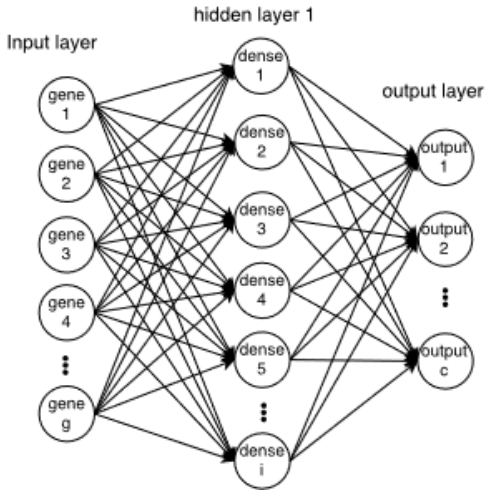


Рисунок 2.3 – Приклад структури мережі прямого розповсюдження для методу Ліна

Через те, що в нас немає жодних ознак для тренування з вчителем, для попереднього тренування шарів мережі (як вказано в розділі 2.2.1) використовується результат роботи автокодувальника. Саме для цього в методі Ліна використовується стандартний шумознижуючий автокодувальник, розглянутий вище.

2.2.3 Варіаційні автокодувальники

Варіаційний автокодувальник – різновид архітектури автокодувальника, яка використовується в якості моделі генерації для створення нової інформації на основі існуючого тренувального датасету. Структура такої мережі показана на рисунку 2.4.

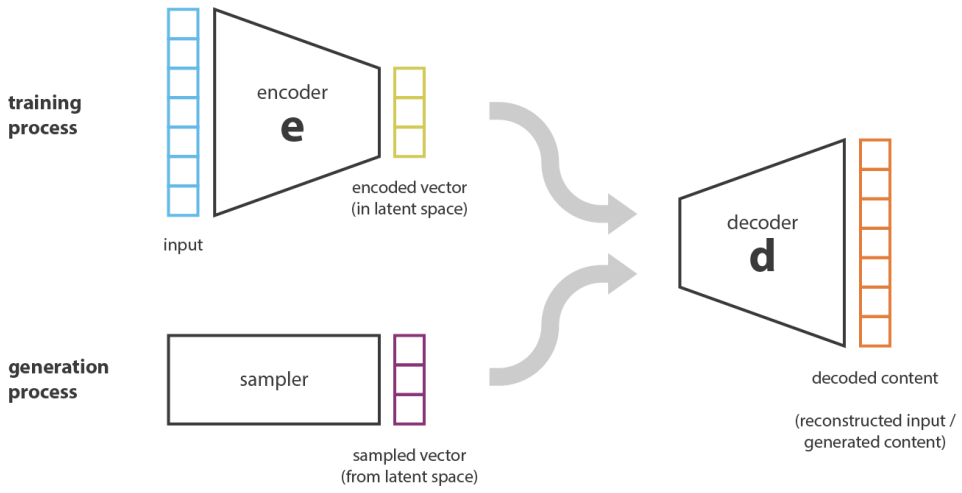


Рисунок 2.4 – Схема архітектури варіаційного автокодувальника

Як видно на рисунку 2.4, мережа ділиться на дві частини – кодувальну та декодувальну, причому кодувальна частина використовується лише під час тренування. Процес навчання майже не відрізняється від тренування

звичайного автокодувальника, але під час генерації процес відрізняється.

На цьому етапі частина, яка займається кодуванням, не використовується. Замість цього відбувається вибірка одного вектору z з латентного простору $p_{model}(z)$. Потім на основі цього вектору відбувається прохід через декодер (генератор) і на виході отримується відображення:

$$p_{model}\big(x; g(z)\big) = p_{model}(x \mid z), \tag{2.3}$$

де z – вектор, вибраний з латентного простору;

$p_{model}(z)$ – розподіл кодування;

$g(z)$ – результат декодування;

x – відповідне декодоване відображення вектору з латентного простору.

Основною можливістю варіаційних автокодувальників, що виділяє їх в порівнянні з іншими автокодувальниками та робить їх ближчими до генеративно-змагальних мереж, є той факт, що латентний простір кодування є неперервним, що дозволяє проводити випадкові операції та інтерполяцію даних. Це досягається за рахунок того, що частина мережі, яка виконує кодування, в якості виходу повертає не просто вектор розміру n, а два вектори – вихідний вектор середніх значень μ та вектор стандартних відхилень σ. Потім на основі цих двох векторів декодувальник бере середні значення та відхилення і генерує новий приклад, який є відновленим відображенням вхідного прикладу x. В випадку, якщо σ у всіх прикладах дорівнює 0, варіаційний автокодувальник веде себе приблизно так само, як і звичайний автокодувальник.

Основною проблемою таких мереж є той факт, що при невеликій кількості вхідних даних при виборі випадкової точки вона може не відповідати жодному прикладу тренувальних даних і буде повертати незрозумілі

результати, що не містять корисної інформації. Для нас ідеальним є варіант коли весь латентний простір так чи інакше зайнятий, а ще краще – якщо він близько до декількох прикладів, що дозволить провести інтерполяцію та в якості результату представити деяку суміш вхідних даних, створивши абсолютно новий результат. На рисунку 2.5 зліва показано типовий розподіл в латентному просторі прикладів, з права показана більш бажана ситуація, в якій простори декількох прикладів перетинаються.

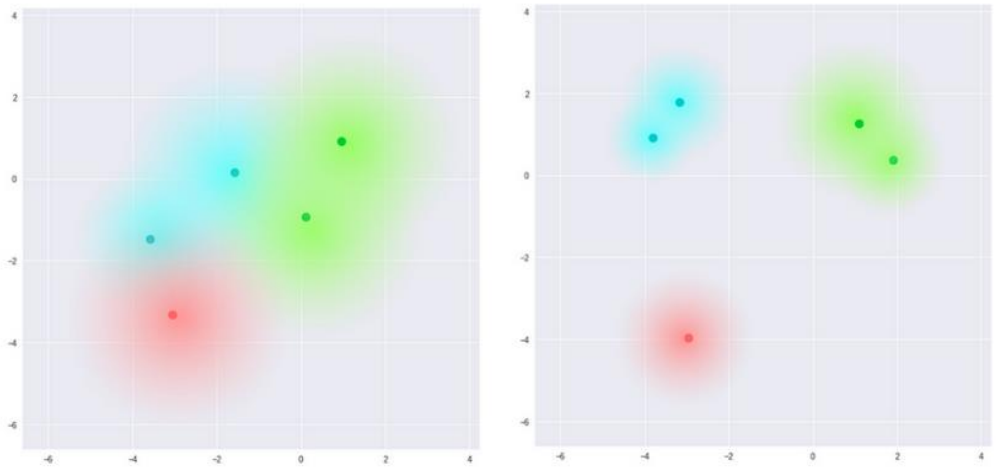


Рисунок 2.5 – Приклади візуалізації різних латентних просторів

Для вирішення цієї проблеми до стандартної функції витрат додається ще один доданок – розходження Кульбака – Лейблера. Через те, що замість звичайного вектору значень використовуються вектори середніх значень та стандартних відхилень, в даній ситуації розходження Кульбака – Лейблера використовується для порівняння та мінімізації різниці між різними розподілами ймовірності. Мінімізація цього розходження означає оптимізацію параметрів розподілу μ та σ таким чином, щоб вони стали близькі до параметрів цільового розподілу:

$$\sum_{i=1}^{n} \sigma_i^2 + \sigma_\text{i}^2 - \log(\sigma_i) - 1. \quad (2.4)$$

Після тренування енкодера та декодера разом, ми отримуємо латентний простір, рівномірно заповнений інформацією, в цьому просторі схожі елементи розташовані недалеко, що дозволяє змішувати існуючи вхідні дані та за допомогою інтерполяції отримувати нові зразки.

В цій кваліфікаційній роботі розглядається декілька архітектур мереж, що мають за основу варіаційний автокодувальник: VASC [20], SAVER-X [22] та scVI .

Архітектура VASC окрім зниження розмірності також використовує метод обробки ситуацій випадання даних.

В порівнянні зі звичайною схемою варіаційного автокодувальника, у VASC відбулися деякі зміни. По-перше, в частині кодувальника першим шаром виступає шар виключення, який займається тим, що змінює вхідні дані і призначає деяким клітинам значення 0 для моделювання випадання даних. Потім процес проходить стандартно для варіаційного автокодувальника, а в кінці останній шар бере готовий результат та на основі розподілу Гумбеля проводить схожу операцію вставки нульових даних відповідно до виходу мережі. Після цього все це оптимізується за допомогою методу зворотного розповсюдження похибки. Завдяки цьому мережа може навчитися залежностям та зв’язкам (в тому числі і нелінійним) між різними генами в даних. На рисунку 2.6 представлена схема архітектури мережі VASC.

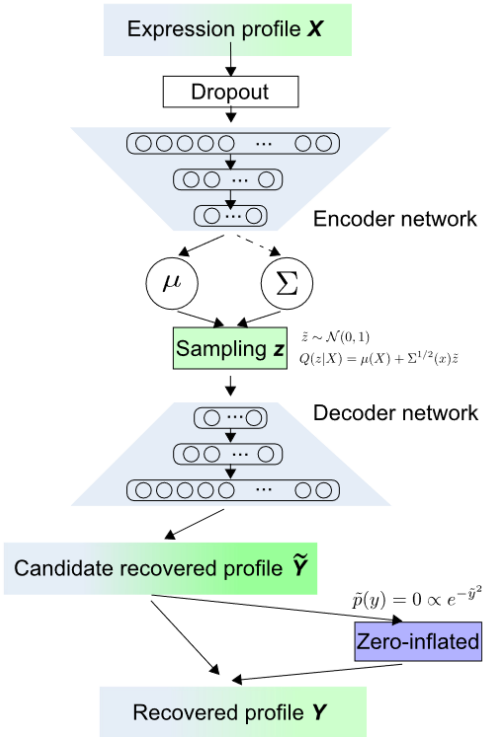


Рисунок 2.6 – Схема архітектури мережі VASC

Алгоритм scVI йде далі та проводить моделювання за допомогою розподілу ZINB – негативною біноміальною регресією з використанням нулів. Такий статистичний розподіл розрахований на той факт, що в матриці експресії генів буде існувати велика кількість нульових значень. В такій мережі також використовуються мітки пакету – вони можуть бути використані для правильного виявлення та видалення пакетного ефекту. Після проходження через кодувальну та декодувальні частини мережі на виході отримуються скориговані значення, що пройшли через нормалізацію, видалення пакетного ефекту та вставку значень, що випали. Таким чином, використання однієї мережі scVI вирішує досить велику кількість проблем, присутніх в вхідних даних. Результати роботи одразу придатні для візуалізації та кластеризації.

Алгоритм SAVER-X досить складний за структурою і використовує як

варіаційний автокодувальник, так і Баєсівську ієрархічну модель. SAVER-X намагається виправити той факт, що майже всі інші алгоритми додають нові кореляції між генами під час процесу роботи, причому ці кореляції ні на чому не базуються та не несуть корисної інформації науковцям.

Крім того, SAVER-X досить добре узагальнює інформацію за рахунок того, що використовується трансферне навчання, тобто процес, при якому мережа, що натренована на одному датасеті та вміє виконувати одну задачу, потім дотреновується на меншому датасеті та вирішує схожу, але відмінну задачу. В статті [22] було відмічено, що проводилося тренування мереж та дослідження на датасетах з тканинами людей та мишей, різних типів клітин тощо, а за допомогою трансферного навчання проводилося донавчання мереж та їх подальша робота вже з іншими видами тварин, клітин, і т. д. Завдяки такому процесу вдалося знайти більш глибокі та систематичні кореляційні зв'язки між різними типами генів.

2.3 Генеративно-змагальні мережі

Генеративно-змагальні мережі – окремий унікальний клас мереж, який за можливостями та структурою схожий на варіаційні автокодувальники, але в той же час істотно від них відрізняється.

В такій архітектурі використовується одразу дві мережі замість однієї: генератор та дискримінатор. Для цих мереж використовується теорія ігор, за її допомогою процес навчання описується як гра з нульовою сумою. Основною задачею генератора під час тренування є створення нових зразків, які будуть максимально схожі на реальні, а задача дискримінатора – визначити, чи був новий поданий на вхід зразок справжнім, чи був створений мережею-генератором.

Якщо використовувати формальну постановку задачі, то в процесі роботи генератор видає зразки $x = g(z;\ \theta^{(g)})$, а задача дискримінатора – для кожного нового прикладу x, що прийшов на вхід мережі розрахувати міру правдоподібності цього зразку $d(x;\ \theta^{(g)})$. Ефект гри з нульовою сумою досягається за рахунок того, що за результатами гри дискримінатор отримує «заохочення», що обчислюється функцією $v(\theta^{(g)}, \theta^{(d)})$, а дискримінатор відповідно отримує $-v(\theta^{(g)}, \theta^{(d)})$. У випадку якщо дискримінатор не впорався з завданням, дискримінатор отримує негативне заохочення, а генератор позитивне. В протилежній ситуації заохочення будуть розподілені відповідно навпаки. Ілюстрація даного процесу зображена на рисунку 2.7.

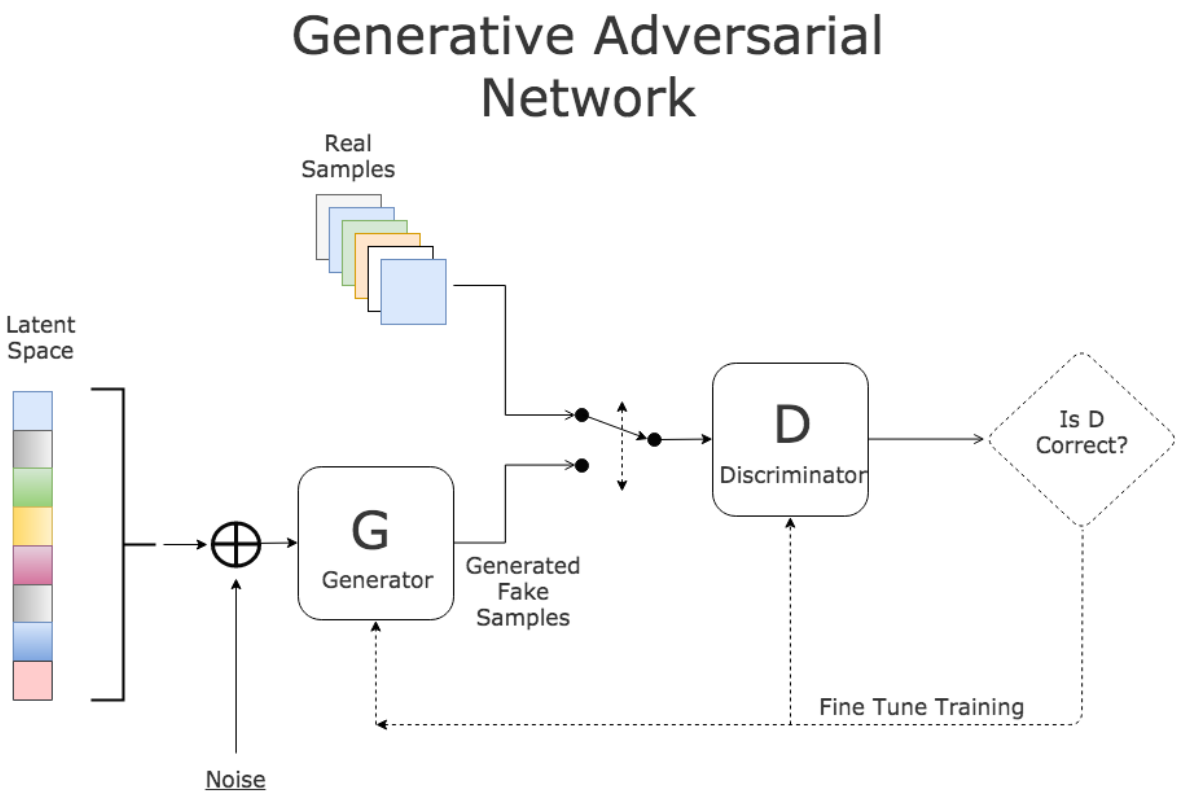


Рисунок 2.7 – Схема навчання генеративно-змагальної мережі

Під час навчання мережі отримують певне заохочення згідно з правилами гри на кожній ітерації, а їх фінальну задачу можна сформулювати наступною формулою:

$$g^* = \arg\min_g \max_d v(g; d)\,, \tag{2.5}$$

де v – функція заохочення;

g – результат роботи генератора;

d – оцінка дискримінатором імовірності того, що вхідні дані несправжні.

В якості оцінки функції витрат для генеративно-змагальних мереж зазвичай використовується функція бінарної крос-ентропії, яка використовується в наступному виді:

$$v(g; d) = E_{x\sim \mathrm{P_{data}}}[\log(d(x)] + E_{z\sim \mathrm{P_z}(z)}[\log\,(1 - d\big(g(z)\big))], \tag{2.6}$$

де d(x) – вихід роботи дискримінатора;

g(z) – вихід роботи генератора;

$E_{x\sim \mathrm{P_{data}}}$ – математичне очікування статистичного розподілу реальних даних;

$E_{z\sim \mathrm{P_z}(z)}$ – математичне очікування статистичного розподілу згенерованих даних.

Завдяки цьому дискримінатор намагається навчитися відрізняти в отриманих даних справжні зразки від згенерованих. Водночас дискримінатор намагається надурити класифікатор та заставити його повірити, що згенерований зразок є справжнім. При достатньому тренуванні, у випадку досягнення сходження (в ідеальному варіанті), результати роботи генератора буде неможливо відрізнити від реальних даних, а дискримінатор при поданні

будь-яких даних буде видавати вірогідність 0.5. Такий процес показаний на рисунку 2.8.

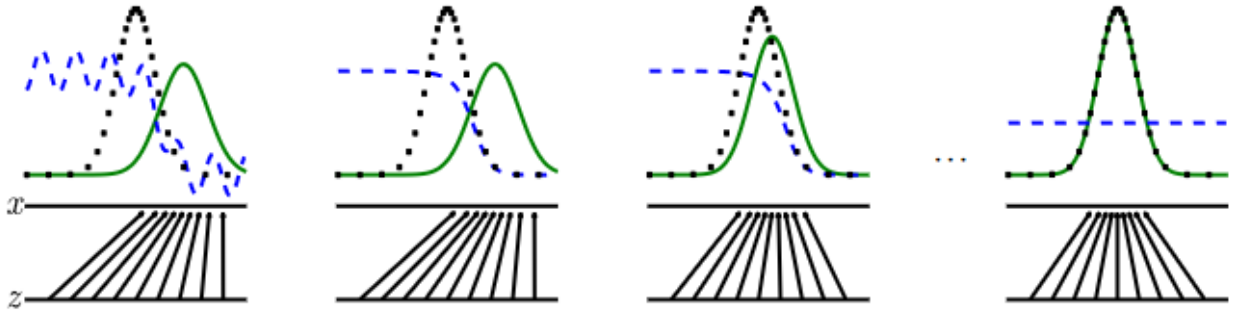


Рисунок 2.8 – Сходження розподілів вхідних даних, генератора та дискримінатора

Незважаючи на теоретичну продуманість мережі, тренування генеративно-змагальних мереж має свої труднощі через те, що функція, яка максимізується під час тренування, має не опуклу форму. Через це існує шанс залишитися в точці локального оптимуму та не дотренувати мережу. В умовах таких обмежень треба проводити дослідження та покращувати використані методи. Наприклад, в роботі Гудфеллоу (в оригінальній статті, в якій пропонувалася ідея використання генеративно-змагальних мереж) було запропоновано використовувати деяку евристичну мотивацію. В ідеалі, треба підібрати цей рівень мотивації так, щоб генератор підвищував логарифмічну вірогідність того, що дискримінатор помилиться замість того, щоб знижувати імовірність того, що дискримінатор зробить правильну оцінку.

В рамках даної кваліфікаційної роботи розглядається конкретна архітектура генеративно-змагальної мережі під назвою scIGAN [17]. Схема його роботи приведена на рисунку 2.9

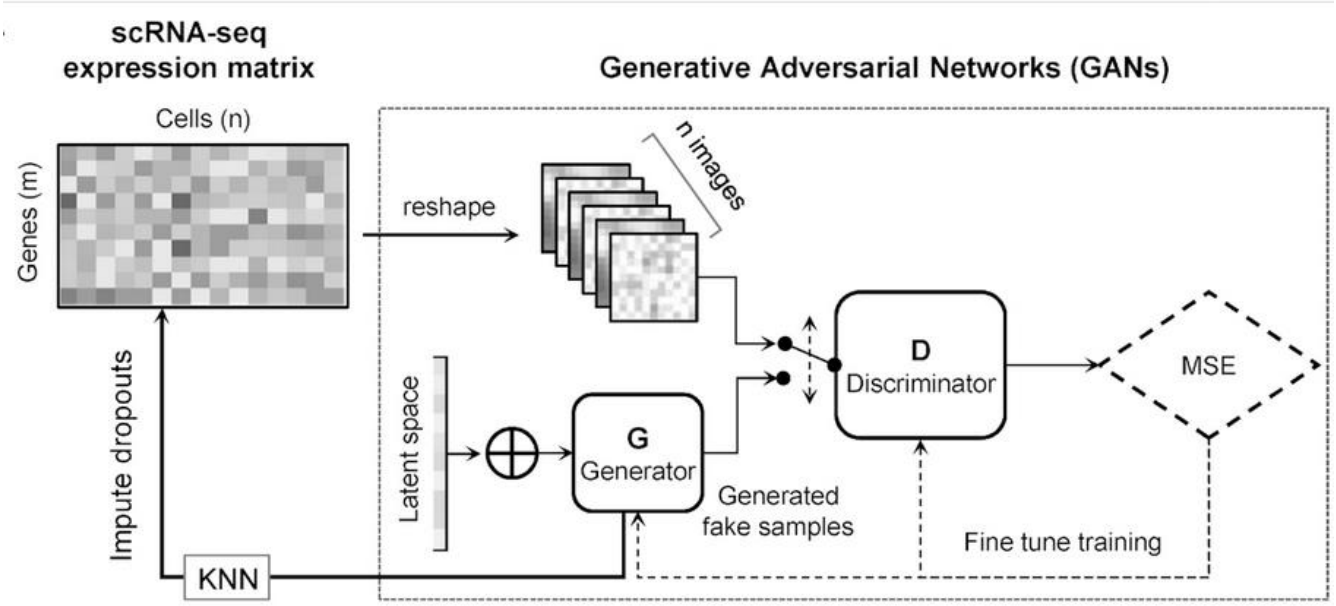


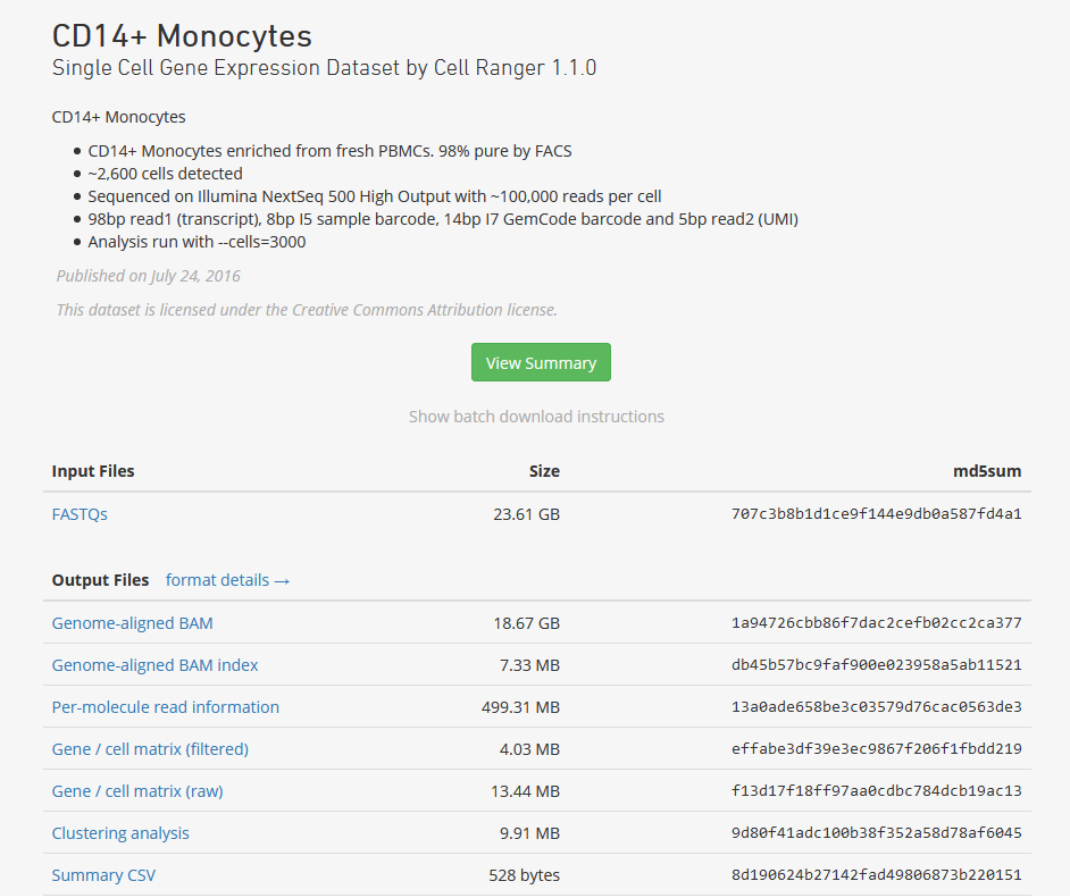
Рисунок 2.9 – схема роботи scIGAN

Особливість цієї мережі полягає в тому, що важко навчити оригінальну модель GAN з даними scRNA-seq, оскільки мінімізація цільової функції відповідає мінімізації дивергенції Йенсена-Шеннона між P_{data} та P_z, яка не є постійною для параметрів генератора. Для вирішення таких труднощів використовується відстань $W(q,p)$ – метрика Вассерштайна.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕРЕЖ

3.1 Вибір тренувального датасету

Для даних scRNA-seq є достатньо велика кількість відкритих даних, які зазвичай використовуються для оцінки швидкості та ефективності нових алгоритмів. В рамках даної кваліфікаційної роботи для оцінки роботи нейронних мереж будуть використані відкриті датасети з дослідження імунітету людини. Всі датасети відкриті та доступні на сайті 10x Genomics, опис датасету та вхідні дані показані на рисунках 3.1 та 3.2.



CD14+ Monocytes

Single Cell Gene Expression Dataset by Cell Ranger 1.1.0

CD14+ Monocytes

- CD14+ Monocytes enriched from fresh PBMCs. 98% pure by FACS
- ~2,600 cells detected
- Sequenced on Illumina NextSeq 500 High Output with ~100,000 reads per cell
- 98bp read1 (transcript), 8bp I5 sample barcode, 14bp I7 GemCode barcode and 5bp read2 (UMI)
- Analysis run with --cells=3000

Published on July 24, 2016

This dataset is licensed under the Creative Commons Attribution license.

View Summary

Show batch download instructions

Input Files	Size	md5sum
FASTQs	23.61 GB	707c3b8b1d1ce9f144e9db0a587fd4a1
Output Files format details →		
Genome-aligned BAM	18.67 GB	1a94726cbb86f7dac2cefb02cc2ca377
Genome-aligned BAM index	7.33 MB	db45b57bc9faf900e023958a5ab11521
Per-molecule read information	499.31 MB	13a0ade658be3c03579d76cac0563de3
Gene / cell matrix (filtered)	4.03 MB	effabe3df39e3ec9867f206f1fbdd219
Gene / cell matrix (raw)	13.44 MB	f13d17f18ff97aa0cdbc784dcb19ac13
Clustering analysis	9.91 MB	9d80f41adc100b38f352a58d78af6045
Summary CSV	528 bytes	8d190624b27142fad49806873b220151

Рисунок 3.1 – Структура датасету CD14+

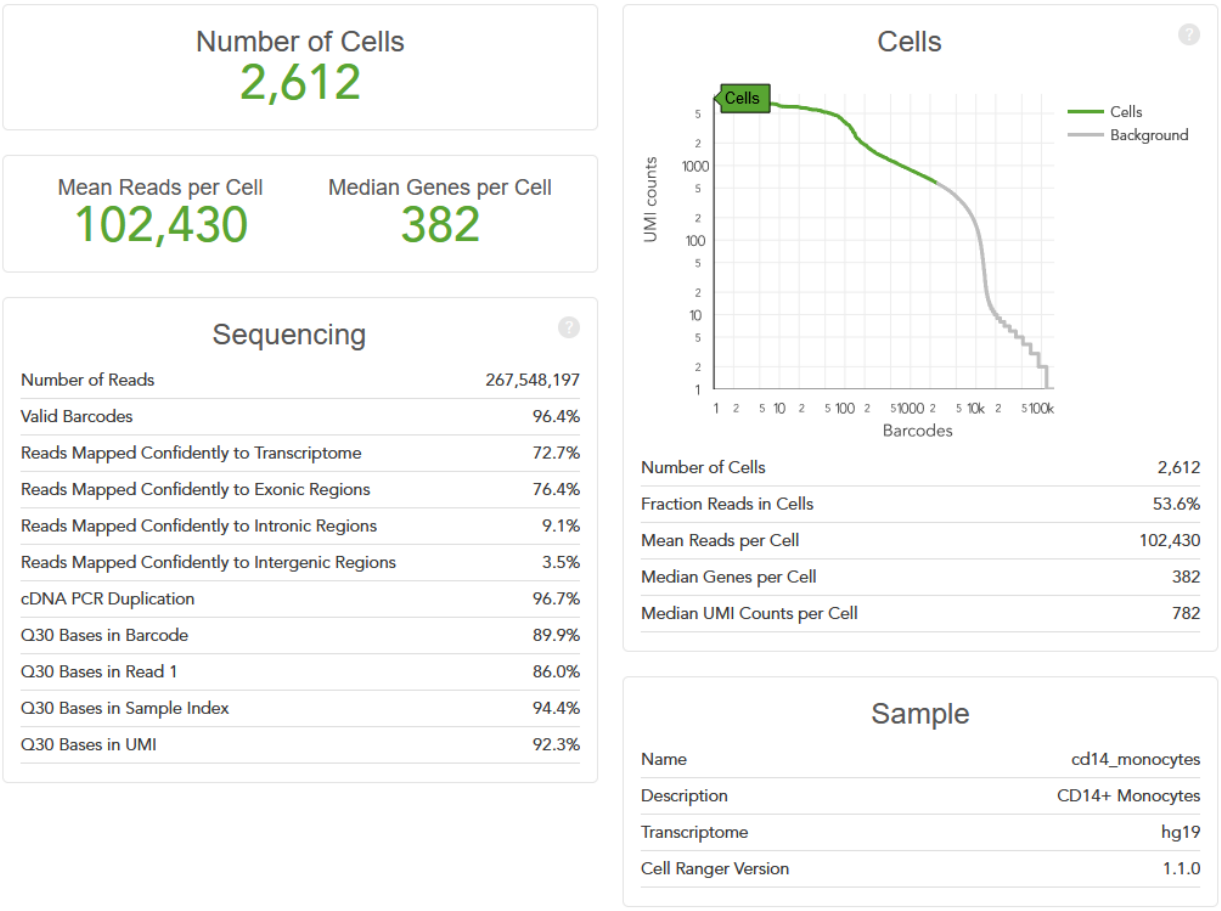


Рисунок 3.2 – Автоматично згенерований опис даних

Як можна побачити з рисунку 3.1, необроблені вхідні дані в форматі FASTQ займають 23 гігабайти дискового простору, що може ускладнювати обробку цього датасету. На щастя, результуючі матриці після обробки займають менший об’єм даних та можуть бути оброблені в цілому.

3.2 Програмна реалізація мереж

Для програмної реалізації представлених нейронних мереж було вирішено використовувати мову програмування Python, тому що вона надає можливість використання великої кількості бібліотек та інструментів як для

аналізу даних за допомогою складних алгоритмів, так і інструментів для побудови глибинних нейронних мереж. Для більш зручної навігації та управління кодом програм, було вирішено скористатися інструментарієм інтегрованого середовища розробки JetBrains PyCharm.

Першою розглянемо мережу scIGAN на основі архітектури генеративно-змагальних мереж. Для побудування цієї мережі буде використовуватися фреймворк PyTorch.

Для опису кожної мережі потрібно побудувати послідовність обчислень, яка складається з послідовних перетворень даних. В такому описі кожне нове перетворення описує новий шар мережі. Крім того, можна згрупувати такі шари та об'єднати їх в «будувальний блок» мережі з можливістю його повторного використання.

Кожен самописний шар в PyTorch має бути класом, що ієрархічно надбудовано над основним класом `nn.Module`. В такому класі має бути реалізований метод `forward(x)`. В цьому методі змінна x позначає вхідні дані, над якими будуть проводитися перетворення. Приклад такого блоку, що може бути повторно використаний, представлено в лістингу 3.1.

Лістинг 3.1 – Код, що реалізує згорточний блок з сигмоїдною функцією активації.

```
self.conv_blocks_1 = nn.Sequential(
    nn.BatchNorm2d(40, 0.8),
    nn.Conv2d(40, self.cn1, 3, stride=1, padding=1),
    nn.BatchNorm2d(self.cn1),
    nn.ReLU(),
    nn.Conv2d(self.cn1, opt.channels, 3, stride=1,
padding=1),
    nn.Sigmoid()
)
```

Фрагмент мережі, що реалізує клас nn.Module показано на лістингу 3.2.

Лістинг 3.2 – Фрагмент опису генератора мережі sciGAN

```
class Discriminator(nn.Module):
  def __init__(self):
      super(Discriminator, self).__init__()

      self.cn1=32
      self.down_size0 = 64
      self.down_size = 32
      self.pre= nn.Sequential(
        nn.Linear(opt.img_size**2,self.down_size0**2),
       )
  def forward(self, img,label_oh):
      out00 = self.pre(img.view((img.size()[0],-
1))).view((img.size()[0],1,self.down_size0,self.down_size0))
        out01 = self.down(out00)#([4, 16, 32, 32])
        label_oh=label_oh.unsqueeze(2)
        label_oh=label_oh.unsqueeze(2)
        out02 = self.conv_blocks02p(label_oh)#([4, 16, 32,
32])
        out1=torch.cat((out01,out02),1)
        out = self.fc(out1.view(out1.size(0), -1))
        out = self.up(out.view(out.size(0), 24,
self.down_size, self.down_size))
        return out
```

Повний код мереж занадто великий для демонстрації тут та представлений в додатку А.

Після побудови мережі її структура зберігається та може бути відображена під час виконання програми за допомогою функції `print`.

Однією з основних переваг фреймворку PyTorch над ручними методами побудови нейронних мереж є той факт, що досить лише описати структуру мережі в методі `forward` і фреймворк сам автоматично на основі цих даних реалізує метод `backward`, який проводить оптимізацію ваг мережі за допомогою методу зворотного розповсюдження похибки.

Як можна побачити з сигнатури методу `forward` на лістингу 3.2, на вхід мережі подаються вхідні дані у вигляді двомірної матриці та позначення типу генів. У випадку, якщо типи генів недоступні (тобто під час використання

мережі за призначенням, коли ми отримали нові дані секвенування), розробники мережі пропонують генерувати позначення за допомогою кластеризації. Генерація позначень організовується за допомогою окремого скрипта на мові програмування R, яку автори мережі надали в якості додаткового матеріалу. В рамках даної роботи цей код приводитися не буде.

Однією з особливостей таких фреймворків як Tensorflow, PyTorch та інших при побудові нейронних мереж є той факт, що код, написаний за допомогою цих інструментів можна прискорити за умови використання графічних процесорів (див. розділ 1.4.5). Графічні процесори створені для обробки та відображення графічної інформації, тому з цією метою вони мають спеціалізовану структуру та можуть проводити паралельну обробку векторизованої інформації. Для генерації коду, що буде виконуватися на графічному процесорі, був розроблений інструмент під назвою CUDA. Наприклад, для фреймворку PyTorch можна підключити функціонал CUDA таким чином, як показано в лістингу 3.3:

Лістинг 3.3 – Використання CUDA в PyTorch

```
cuda = True if torch.cuda.is_available() else False
if cuda:
    generator.cuda()
    discriminator.cuda()
    print("scIGANs is runing on GPUs.")
else:
    print("scIGANs is runing on CPUs.")
```

Таким чином, це дозволяє писати код, який буде доступним як на пристроях з графічним процесором, так і ні. При виклиці функції `toch.cuda.is_available()` фреймворк сам автоматично перевіряє, чи

доступний графічний процесор чи ні. В лістингу 3.1 виклики generator.cuda() та discriminator.cuda() повідомляють, що ваги генератора та дискримінатора треба опрацьовувати на графічному процесорі, тож після запуску всі обчислення будуть працювати на GPU.

Цей підхід є досить ефективним через те, що на системах з графічним процесором обчислення прискорюються приблизно в 10 разів, що стало поштовхом для розвитку великих і обчислювально дорогих глибинних нейронних мереж.

Алгоритм scvis оснований на варіаційному автокодувальнику. Для реалізації цієї мережі використаємо фреймворк Tensorflow. На відміну від PyTorch, цей фреймворк відрізняється тим, що будує граф обчислень статично, побудова шарів відбувається так, що на вхід новим шарам подаються результати роботи минулих шарів, а найпершому шару – змінна `tf.Placeholder()`, яка вказує розмірність вхідних даних, але слугує лише «заглушкою». Насправді весь цей граф обчислень формується без використання вхідних даних і його ініціалізація проводиться після створення об'єкту типу `tf.Session`, який дозволяє викликати мережу та подати їй на вхід дані.

Для запуску мережі треба спочатку її скомпілювати, а потім запустити об'єкт сесії. В реалізації мережі scvis це зроблено так, як показано на лістингу 3.3.

Лістинг 3.3 – Код запуску мережі в Tensorflow

```
self.sess = tf.InteractiveSession()
self.sess.run(tf.global_variables_initializer())
```

Як і з PyTorch, Tensorflow автоматично може знаходити графічний процесор та запускати обчислення на ньому.

Інші нейронні мережі, що розглядалися в цій роботі, будувалися за схожими принципами, тому розглядати їх тут недоцільно. Повний код мереж можна знайти в додатку А.

Тренування мереж проводилося на системі з процесором Intel Core i7-8700, графічним процесором nVidia GeForce GTX 1050 Ti з 4 гігабайтами пам'яті, а також 12 гігабайтами оперативної пам'яті.

3.3 Порівняльний аналіз результатів

В рамках кваліфікаційної роботи розглядалося вирішення декількох різних задач, тож порівняння різних мереж слід робити для кожної задачі окремо.

Слід зазначити, що не кожна мережа робить все, тож для деяких задач ефективність мережі може не оцінюватися, бо

Для оцінки мереж в задачі введення даних, що випали, був взятий тестовий датасет та деякі значення генів були змінені на 0, щоб змоделювати випадіння. Потім для кожної з мереж було проведене вимірювання різниці між отриманим результатом та еталонними вихідними даними. Результати роботи мереж показані на рисунку 3.3.

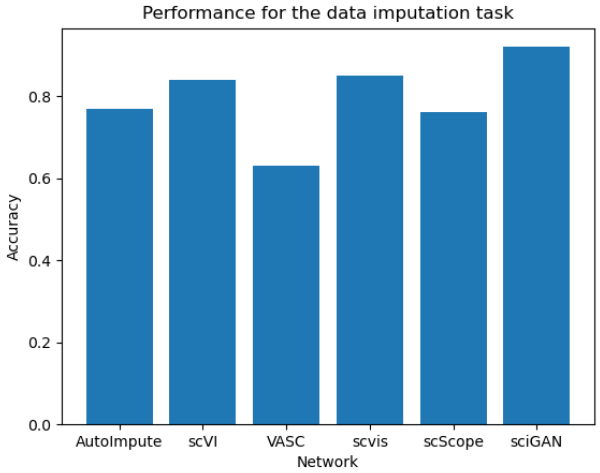


Рисунок 3.3 – Результати роботи мереж для задачі випадіння даних

Для задачі видалення технічного шуму аналогічно був взятий вхідний датасет та до нього був доданий невеликий випадковий шум, щоб змоделювати технічний шум. Потім була підрахована найменша квадратична дистанція для кожного вихідного датасету порівняно з вхідним датасетом. Результати таких вимірювань представлені на рисунку 3.4.

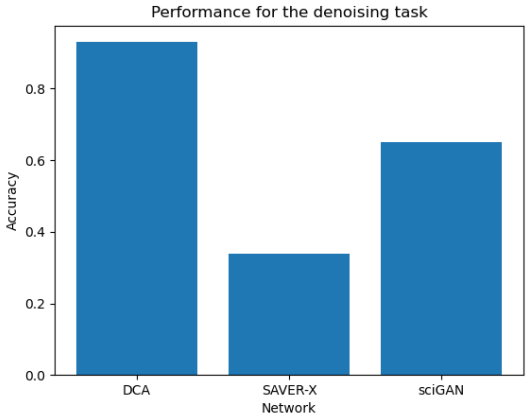


Рисунок 2.3 – Порівняння ефективності мереж в задачі видалення технічного шуму

Для оцінки ефективності роботи з пакетним ефектом був взятий спеціальний датасет з реальними даними, в якому був яскраво виражений пакетний ефект. В якості еталона для порівняння була взята модифікована версія датасета, яка пройшла етап ручної обробки та нормалізації даних. Результати вимірювань ефективності показані на рисунку 3.5.

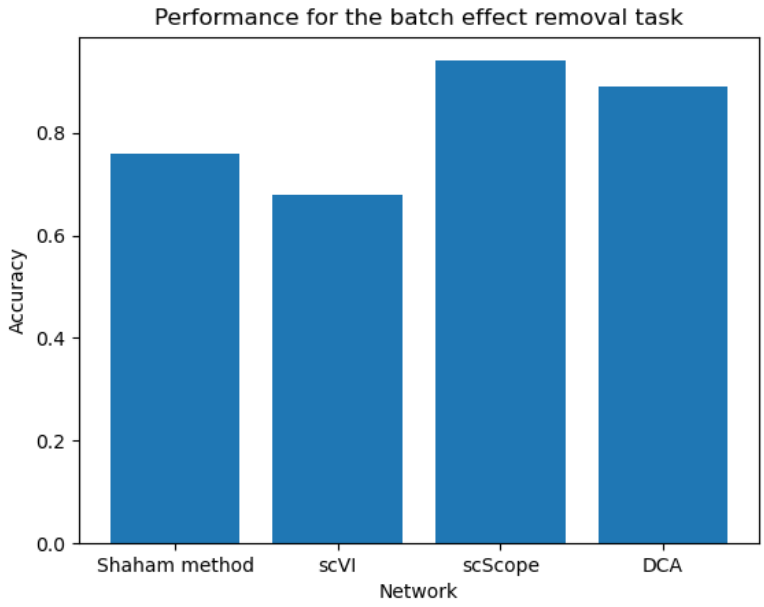


Рисунок 3.5 – Результат оцінки ефективності в задачі видалення пакетного ефекту

Для оцінки задачі зниження розмірності був використаний датасет з назвами груп клітин, які мали під собою біологічне обгрунтування. На вхід мережам для пониження розмірності були подані вхідні дані без міток груп клітин. Потім для цих даних була проведена кластеризація методом k-середніх з кількістю кластерів рівній кількості груп в реальних даних. В якості оцінки використовувалася відповідність кластерів справжнім групам. Результати показані на рисунку 3.6.

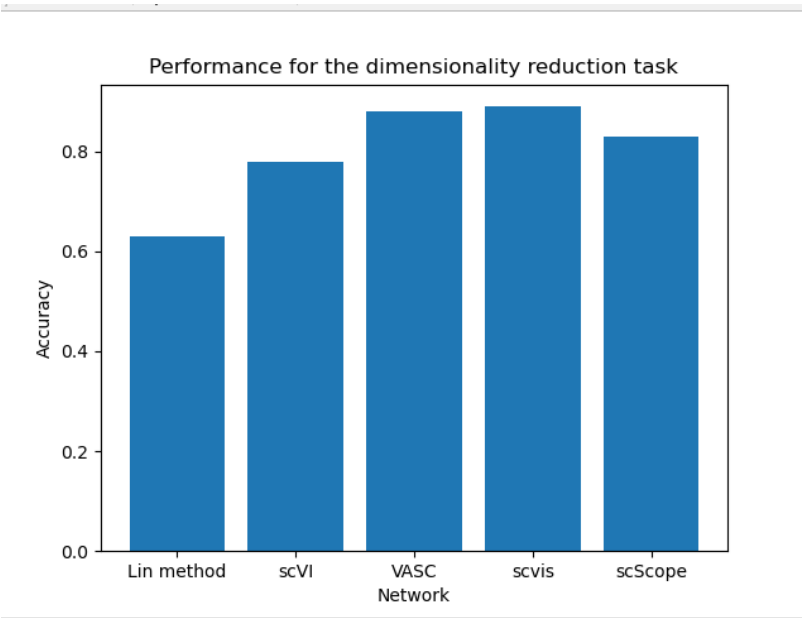


Рисунок 3.6 – Результати роботи мереж на задачі зменшення розмірності

Як можна побачити з наведених графіків, немає мережі, яка б робила все максимально ефективно, кожна мережа має свої переваги та недоліки. Крім того, результати мереж можуть відрізнятися в залежності від датасета, оскільки характер даних в експериментах scRNA-seq може бути досить різним.

3.4. Формування фінального процесу обробки даних RNA-seq

Оскільки жодна з представлених мереж не має значної переваги над іншими, постає питання доцільності комбінування мереж для вирішення більшої кількості завдань. В якості експерименту були взяті мережа VASC для боротьби з випадінням даних та мережу Шахама на залшикових нейронних мережах для видалення пакетного ефекту з даних. При їх сумісному використанні – спочатку обробити дані і видалити пакетний ефект, а потім провести дані через автокодувальник – виявилося, що ефективність мережі

VASC суттєво знизилася в даній ситуації. Результати вимірів показані на рисунку 3.7.

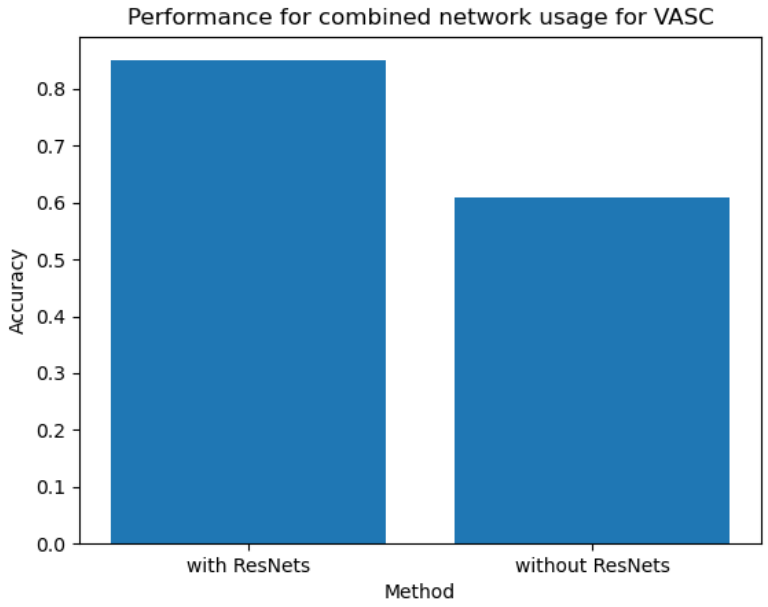


Рисунок 3.7 – Результати роботи мережі VASC після попередньої обробки даних залишковою мережею

Найбільш імовірною причиною того, що результати погіршилися, є той факт, що обробка даних мережею змінює її статистичний розподіл і може вносити нові кореляції, яких до того там не було. Через це, VASC отримує на вхід неправильно статистично розподілені дані, на які мережа не генералізувалася, тож не дивно, що ефективність такої мережі знизилася. Подальші експерименти показали, що інші варіанти комбінованої обробки даних декількома нейронними мережами не є можливою без зниження ефективності.

Як результат, для формулювання фінального вибору оптимальної мережі для аналізу даних scRNA-seq більш доцільно буде використовувати лише одну мережу, яка б була максимально корисною для науковців.

Проблеми технічного шуму та пакетного ефекту є не досить великими при аналізі даних через те, що є алгоритмічні та технічні методи їх вирішення, такі як нормалізація даних та використання контрольних проб в біологічному матеріалі для виявлення пакетного ефекту. В той самий час, задача введення даних є дуже важливою для даних scRNA-seq через природу цих даних. В такому випадку, найбільш оптимальним варіантом для пайплайну обробки даних слід обрати використання sciGAN. Це дозволить якнайкраще змоделювати дані, що відсутні у вхідному датасеті і вставити їх. Звісно, використання цієї мережі слід поєднати з автоматичними алгоритмічними методами типу фільтрації генів, яких зовсім не виявилося у вхідних даних, зменшення розмірності методом t-SNE та інші.

ВИСНОВКИ

В ході кваліфікаційної роботи були розглянуті технології секвенування, в тому числі і технологія секвенування поодиноких клітин. Були розглянуті їх основні принципи роботи, задачі та проблеми, що постають при їх використанні. Для кожної з таких задач були розглянуті алгоритмічні методи їх розв’язання.

В рамках роботи було проведено дослідження існуючих методів аналізу даних scRNA-seq за допомогою глибинних нейронних мереж та проведено аналіз принципів їх роботи та задач, які ці нейронні мережі вирішують. Крім того, був проведений порівняльний аналіз різних архітектур автокодувальників та їх застосування.

Під час виконання роботи кожна з мереж була реалізована та було проведено порівняльне дослідження ефективності мереж в різних задачах (зменшення розмірності, видалення пакетного ефекту, видалення шумів, тощо.). В результаті порівняльного аналізу було виявлено, що найбільш оптимальним є використання мережі sciGAN для введення відсутніх даних до матриці експресії генів.

Виходячи з вищезазначеного, для оптимального аналізу даних scRNA-seq потрібно використовувати нейронну мережу для введення даних і алгоритмічні методи для видалення пакетного ефекту, зменшення розмірності і кластеризації даних та ін. Завдяки цьому дані проходять через максимально можливі трансформації, що очищають дані, максимально зберігають їх структуру та кореляції між окремими генами та знижують розмірність даних. Завдяки цьому їх досить легко візуалізувати та провести статистичний аналіз для пошуку статистично значущої інформації.

Готовий автоматизований процес обробки інформації можна інтегрувати

з різноманітним інструментарієм, який дозволяє обробляти та візуалізувати дані. Такий інструментарій зазвичай поставляється разом з секвенаторами для створення інтегрованого середовища обробки та аналізу даних.

В цілому, використання вищезазначеного процесу має зменшити кількість ручної роботи, яку треба зробити для аналізу матриці експресії генів та частково автоматизувати процес аналізу даних, що дозволить швидше отримувати корисні результати та можливо пришвидшити розробку ліків від таких недугів, як рак, ВІЛ/СНІД та коронавірусна хвороба COVID-19.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Single-cell transcriptomic analysis of primary and metastatic tumor ecosystems in head and neck cancer / S. V. Puram et al. *Cell*. 2017. Vol. 171, no. 7. P. 1611–1624.e24.

2. The application of single-cell RNA sequencing in vaccinology / A. Noé et al. *Journal of immunology research*. 2020. Vol. 2020. P. 1–19.

3. Zheng J., Wang K. Emerging deep learning methods for single-cell RNA-seq data analysis. *Quantitative biology*. 2019. Vol. 7, no. 4. P. 247–254.

4. Normalization and noise reduction for single cell RNA-seq experiments / B. Ding et al. *Bioinformatics*. 2015. Vol. 31, no. 13. P. 2225–2227.

5. Practical impacts of genomic data “cleaning” on biological discovery using surrogate variable analysis / A. E. Jaffe et al. *BMC bioinformatics*. 2015. Vol. 16, no. 1.

6. Removal of batch effects using distribution-matching residual networks / U. Shaham et al. *Bioinformatics*. 2017. Vol. 33, no. 16. P. 2539–2546.

7. Chu Y., Corey D. R. RNA sequencing: platform selection, experimental design, and data interpretation. *Nucleic acid therapeutics*. 2012. Vol. 22, no. 4. P. 271–274.

8. The promise of single-cell sequencing / J. Eberwine et al. *Nature methods*. 2013. Vol. 11, no. 1. P. 25–27.

9. Li W. V., Li J. J. An accurate and robust imputation method scImpute for single-cell RNA-seq data. *Nature communications*. 2018. Vol. 9, no. 1.

10. DrImpute: imputing dropout events in single cell RNA sequencing data / W. Gong et al. *BMC bioinformatics*. 2018. Vol. 19, no. 1.

11. Wang D., Gu J. Vasc: dimension reduction and visualization of single-cell rna-seq data by deep variational autoencoder. *Genomics, proteomics & bioinformatics*. 2018. Vol. 16, no. 5. P. 320–331.

12. AutoImpute: Autoencoder based imputation of single-cell RNA-seq data / D. Talwar et al. *Scientific reports*. 2018. Vol. 8, no. 1.

13. The promise of single-cell sequencing / J. Eberwine et al. *Nature methods*. 2013. Vol. 11, no. 1. P. 25–27.

14. Zhang L., Zhang S. Comparison of computational methods for imputing single-cell RNA-sequencing data. *IEEE/ACM transactions on computational biology and bioinformatics*. 2019. P. 1–1.

15. A unified statistical framework for single cell and bulk RNA sequencing data / L. Zhu et al. *The annals of applied statistics*. 2018. Vol. 12, no. 1.

16. Hinton G. E. Reducing the dimensionality of data with neural networks. *Science*. 2006. Vol. 313, no. 5786. P. 504–507.

17. ScIGANs: single-cell RNA-seq imputation using generative adversarial networks / Y. Xu et al. *Nucleic acids research*. 2020. Vol. 48, no. 15. e85-e85.

18. Bengio Y., Courville A., Goodfellow I. Deep learning. MIT Press, 2016. 800 p.

19. Using neural networks for reducing the dimensions of single-cell RNA-Seq data / C. Lin et al. *Nucleic acids research*. 2017. Vol. 45, no. 17. e156-e156.

20. Wang D., Gu J. VASC: dimension reduction and visualization of single-cell RNA-seq data by deep variational autoencoder. *Genomics, proteomics & bioinformatics*. 2018. Vol. 16, no. 5. P. 320–331.

21. Deep generative modeling for single-cell transcriptomics / R. Lopez et al. *Nature methods*. 2018. Vol. 15, no. 12. P. 1053–1058.

22. Data denoising with transfer learning in single-cell transcriptomics / J. Wang et al. Nature methods. 2019. Vol. 16, no. 9. P. 875–878.

23. Adaptive multistep self-learning procedure for solving principal component analysis task / E. V. Bodyanskiy et al. *Electrical and computer systems*. 2017. Vol. 24, no. 100. P. 118–123.

24. Бодянский Е. В. Искусственные нейронные сети : підручник. Харьков : СМИТ, 2005. 408 с.