Transfer learning and consensus clustering-based analysis of single-cell data

by
Dinesh Joshi

Under the supervision of
Dr. Debarka Sengupta

Submitted in partial fulfillment of the requirements for the degree of Master of Technology, Computational Biology

Center for Computational Biology Indraprastha Institute of Information Technology - Delhi
June, 2022

Certificate

This is to certify that the thesis titled *"Transfer learning and consensus clustering-based analysis of single-cell data"* being submitted by **Dinesh Joshi** to the Indraprastha Institute of Information Technology Delhi, for the award of the Master of Technology, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

June,2022

Dr Debarka Sengupta

Department of Computational Biology
Indraprastha Institute of Information Technology Delhi
New Delhi 110 020

Acknowledgements

I would like to thank Prof. Debarka Sengupta for his continuous guidance and mentoring, without him this work would not have been possible. From day one of starting my thesis, he constantly pushed my limits to aim for higher goals in life and to always do better at my work. Whenever I faced any issue or need some help in clarifying my doubts, he used to give his time even on days when he is busy. At times, when I was struck at my work and I approached him for some suggestions, he used to give some insights and would refine our approach to doing the work, and that has surely helped me to keep track and complete my work timely.

I thank my mother, brother and sister, who have helped me indirectly in innumerable ways during my work so that I can focus on my thesis completely. They have continuously supported me in my studies right from my under-graduation to graduation. I owe my deepest gratitude to them for this everlasting support.

During my thesis work, for every small or trivial doubt regarding my thesis work or career advice, I used to go to my PhD mentor Namrata Bhattacharya and I thank her a lot for this. She always tried to come up with new suggestions or ideas for our work, and that has really helped me shape my thesis work nicely. I want to also thank my teammates Rishabh and Kiran, for supporting me through the ups and downs that came during our web server work. My fellow batch mates were always there to help me and clarify any doubts related to biology and have always encouraged and supported me in my academic as well as personal life.I thank all of them.

I also thank all the faculty members and staff of the Department of Computational Biology and IIIT Delhi for always helping us throughout our college journey. My special thanks to Mr.Adarsh from the IT department, for his continuous support and help in providing access to the college IT infrastructure and helping us with the webserver deployment work.

Abstract

With the rapid development of sequencing technologies, single-cell genomics has become a primary focus of researchers. Through single-cell RNA-sequencing data analysis, it has now become possible to study some fundamental questions in biology at the cell-level resolution like which cell types are present in a tissue, what function these cells carry and how are these functions different from healthy tissues etc. However, utilizing publicly available large scale cell atlases for analysis is not so easy and has its own challenges. Since single-cell data analysis workflow comprises many steps like data harmonizing, batch effect correction, normalization, visualization, clustering, cell-type classification and differential gene expression analysis, so different methods have to be used to carry out these tasks. But with advancements in computational power and techniques, utilising large scale reference datasets to gain knowledge and then transfer it to smaller query datasets has become common. So to facilitate the users with these recent transfer learning (TL) techniques to do single-cell analysis, we have come up with Transcend, a webserver which hosts a number of such pretrained TL methods. We hope that our webserver will be widely adopted as a single source for exploring transfer learning methods for single-cell analysis.

Moreover, clustering single-cell RNA-seq data to uncover patterns, can lead to the identification of new or rare cell types or subtypes but clustering single-cell data is also a challenge due to the highly sparse read count data and due to presence of technical variation in the data it becomes difficult to capture biological variation in data, which can lead to poor clustering of cells. So to get better clustering results, we explored the concept of consensus clustering. After trying out multiple consensus clustering algorithms, we were able to get improved clustering results which are stable and can be further used to improvise the clustering results of other methods.

Contents

List of Figures

Chapter 1

Introduction

1.1 Single Cell Genomics

Genomics is an interdisciplinary field of biology, where the study of the complete genome is carried out, in order to advance the understanding of diseases and diagnostic measures at a very fundamental level. Although there are various types of sequencing like single-cell DNA methylome sequencing, single-cell assay for transposase-accessible chromatin sequencing etc, we are focusing on the RNA-sequencing profiles. Traditionally, RNA-seq profiling used to be bulk-RNA seq, where it is used with samples composed of a mixture of different cells. The bulk-RNA seq has many applications like finding out the signature biomarkers between tissues of healthy/diseased samples or control/treated samples, differentiating between tissues by comparing the respective transcriptomes and in addition, it can be used to find and label the new genes etc. But bulk RNA-seq gives an estimate of the average expression level of each gene across the population of cells without considering the heterogeneity among the cells. So, it doesn't give a clear idea about the individual cells of a sample and can not be used for studying the heterogenous systems like early development studies etc.

With the advent of next-generation sequencing technologies, transcriptome profiling became less cost-intensive and less time taking, which led to single-cell RNA sequencing. The single-cell RNA sequencing (scRNA-seq) perfectly overcame the limitations of bulk-RNA seq and made it possible to estimate the distribution of expression levels of each gene across the population of cells. Hence, now it became possible to answer some fundamental biological questions like what type of cells are present in the tissue, which functions these cells carry, and how are these functions different from the healthy tissues. Cell-type-specific information can be comprehended, which can help in discovering the new or rare cell types, understanding the cell differentiation during development and finding out cell composition between healthy and diseased tissues.

1.2 Challenges with scRNA-seq data

scRNA-seq data comes with many problems since the starting material per cell is very less. Hence it leads to very sparse data, having a large number of zeros in the data. The zero in the data can be real or false. When the gene was not expressed in the cell, then it is counted as a 'real' zero whereas when the gene was expressed in the cell but we are not able to detect it due to technical limitations it is counted as a 'false' zero or it is called as a 'dropout'. This leads to unnecessary variation between the cells which has not emerged due to biological variation but due to the technical issues like the gene might not have been PCR amplified to a sufficient level. However, this problem can be adjusted by doing the normalization. Another issue, with the single-cell RNA seq data, is the problem of batch effects. Data integration or data harmonization is one of the most crucial stages in single-cell data analysis. The batch effect is a problem which arises when multiple datasets from different laboratories, sequenced with different technologies and machines are integrated to form a single large reference dataset. This leads to technical noise in the data, and it becomes difficult to find out the actual biological variation present in the data.

1.2.1 Transfer Learning

In classic supervised machine learning, the model is trained on data of a specific task and domain having labelled examples and then this model is used to predict unseen samples from the same task and same domain. When some other task or other domain is used, then a new model is again trained which requires again the labelled examples. This approach fails when we do not have enough labelled examples to train a model which is reliable, which leads to the use of a transfer learning strategy. The transfer learning methods try to leverage the data of some source task or source domain, to extract and store knowledge of this source task which is then applied to the target task or target domain [4].

There are different types of transfer learning (TL) depending upon the task, domain and availability of labelled/ unlabeled examples. When transfer learning is applied for the same source and target tasks like classification etc, and labels are only present in the source domain, we call it Transductive TL. Inside this, if we have different domains, it is called Domain adaptation and if different languages are used in the source and target domain, then it is classified as Cross-lingual learning. When the transfer learning is applied to different source and target tasks and labelled data is present in the target domain, we call it Inductive TL. Inside this, if the tasks are being learned simultaneously, then it is called Multi-task learning and if the task is being learned sequentially, then it is termed Sequential transfer learning. For our case, we will be using sequential TL, as we have huge source domain datasets and small target domain datasets. The model is trained with the source dataset and then the knowledge is transferred to the target task and domain. It is very efficient to adapt to multiple target tasks at a time, with this approach[4].

1.2.2 Sequential Transfer Learning for single-cell analysis

Sequential TL is done in two-stage namely pre-training and adaptation. In the pre-training step, a reference model is trained on the source dataset which is usually large in size. Then knowledge from this already trained model is used, and with the help of the target dataset, the model is finetuned to adapt to the target task or domain. The complete flow of the sequential transfer learning can be found in 1.1

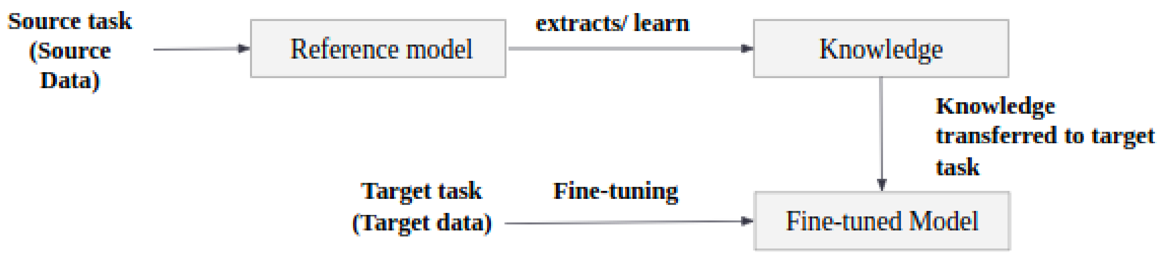

Source task (Source Data)
Reference model
extracts/ learn
Knowledge
Knowledge transferred to target task
Target task (Target data)
Fine-tuning
Fine-tuned Model

Figure 1.1: Flow Diagram of Sequential TL

This type of TL will be very helpful in the case of single-cell analysis as many times source and target datasets are not present simultaneously. Also with the advent of next-generation sequencing technology, large scale cell atlases are available, which can be readily used for doing single-cell analysis. These cell atlases act as reference datasets for the source task and are utilized to gain knowledge, which is then transferred to target tasks like finding cellular heterogeneity, cell type classification etc.

However, utilizing these large reference datasets has always been challenging, due to multiple problems like batch effect, dropouts etc. Transfer learning had been applied in the past for denoising the single-cell data, variance decomposition, dimensionality reduction and cell-type classification. Also, single-cell data analysis is usually performed in many steps using different tools and packages, which makes it even more difficult to do in a time constraint scenario. Here, these large reference datasets will be used to uncover the cellular heterogeneity of the target dataset, which otherwise would have been difficult to find.

1.3 Clustering single-cell RNA-seq data

Analysing any data specifically unlabelled data through clustering is a crucial part of any domain. For a given set of objects, clustering is defined as placing objects in a group similar (or related) to one another and different (or unrelated) to objects in other groups. Clustering is used to find hidden patterns from the data and uncover some findings. For single-cell RNA seq data, clustering is very helpful in uncovering patterns which lead to the finding of new or rare cell types or subtypes present in the dataset[5]. Various methods for clustering of single-cell data are present like non-negative matrix factorization based clustering, graph partitioning algorithms, Louvain algorithm and Leiden algorithm etc. With advancements in the computational power with respect to the hardware, different deep learning methods have been employed on the single-cell data before doing clustering on top of the low-dimensional data given by the models. This approach has also significantly improved the clustering results. Finally, some clustering methods have combined other different clustering results to obtain one single stable clustering, which is better than the individual clusterings.

1.4 Thesis outline

In chapter 2, we will introduce Transcend, a webserver for hosting pretrained transfer learning models for single-cell analysis. In this section, we will go through all the dataset details that have been used in the webserver. We will explore some of the transfer learning models that are used in the webserver to gain deeper insights into their workings. We will also discuss the implementation of the webserver which will include the application architecture, system design and deployment procedure. At last, all the features of the web server will be explored in-depth, to get an overall idea of the working of the webserver.

In chapter 3, we will see how some of the transfer learning-based methods are used for clustering the single-cell data by comparing each other's clustering results using visualizations and different clustering metrics. We will also see, how clustering is impacted when using the concept of consensus clustering and will explore different consensus algorithms which are used for the same.

chapter 4, will be a concluding part of the thesis, where the results and overall analysis of the methods used will be summarized followed by the future scope of work for the web server and consensus clustering.

Chapter 2

Transcend - Webserver

2.1 Introduction

With the advancements in sequencing technologies, large scale cell atlases are readily available for analysis of cellular heterogeneity, developmental studies, cell type identification, and finding new or rare cell types. But it is always a challenge to make use of such large atlases because reference datasets come from different labs being sequenced with different technologies, which leads to batch effects in the data. This batch effect causes technical variation in the data, which masks the biological variation and hence leads to irrelevant findings.

Most of the biological questions can be cast into either cell level or gene-level algorithmic queries. There are different tasks in the whole process of single-cell analysis. i) Stratification via embedding is done by projecting the cells for identification from high-dimensionality to low-dimensionality, these embeddings can then be used to visualize the clustered data ii) Data Harmonization is a crucial part of the analysis, where batch-effect free embeddings are evaluated to compare data across conditions iii) Annotation, is a step in the analysis where the actual transfer of cell types from one dataset to another takes place. iv) Normalizing/ Imputation deals with computing average expression levels while removing the technical artefacts v) Differential expression is done to find gene expression discrepancies between cell types.

All these stages in single-cell analysis require different tools and methods which are available in the form of different packages, tools and software at different places. For example, various data harmonization methods are used like Harmony, Scanorama, mnnCorrect, Seurat, LIGER, CONOS and cell-type classification methods like Seurat, SVM, and Logistic regression have been used [6].

Utilizing large reference datasets has become very prominent in the computer vision and natural language processing domains. These domains have successfully utilized these large datasets for solving the target task and target domains. ImageNet[7] and BERT [8] like models have been able to use the prior knowledge by training themselves on huge unlabeled data, to solve problems for target specific tasks. This strategy of transfer learning (TL) has also gained popularity in the single-cell RNA-seq data analysis, due to the presence of large scale reference cell atlases which can be utilized in order to solve target data specific tasks, where target data has relatively very few data samples.

With TL showing improved results for tasks related to scRNA-seq data like clustering and annotation / cell-type classification etc, different methods have been introduced. Methods like single-cell Variational Inference(scVI)[1], single-cell ANnotation using Variational Inference (scANVI) [2], transfer Variational Autoencoder (trVAE) [9], total Variational Inference (TotalVI) [10], and single-cell Embedded topic modelling (scETM) [3] etc, have employed transfer learning strategies very successfully, and each method covers almost all the stages of single-cell data analysis.

But all these methods are present in various places, and each method covers a subset of tasks from clustering, annotation / cell-type classification, dimensionality reduction ad imputation etc. So an end-user, who is doing analysis for their dataset has to go through a lot of effort in exploring all these methods along with looking for data being used, which is generally available at different places with different formats. People who want to utilize these TL based methods require technical expertise to make use of these tools for their own tasks, so there is no end-to-end code available for them to use. A big disadvantage for all these methods is the requirement of computational power in terms of hardware resources and the time taken to train these models for large data and then utilise them for small datasets.

To tackle all these problems, we come up with Transcend, a web server, hosting pre-trained transfer learning models for single-cell analysis. This web server acts as a model repository where models trained on large reference datasets are present and they can be fine-tuned for respective query datasets with minimal effort. The whole code for finetuning of these models along with reference to the datasets used is provided with an end-to-end python notebook.

2.2 Data Collection and Pre-procesing

2.2.1 Data Collection

Data has been collected from various known sources. Different species and their organs/tissue data have been referred. Primarily, human and mouse species have been used and data of organs like kidney, pancreas, blood, spleen, lung epithelial, prostate and colon has been used.

Following are the sources of data collection:

- **Covid 19 cell atlas (https://www.covid19cellatlas.org/)**

  This website contains data with visualizations for almost all the organs for healthy donors as well as patient donors. We have used data from healthy donors for our use case. Also, the data is available with the site of origin containing the research paper link, so it is easier to cite the data source. Data is present in h5ad format, making it easy to use for data reading and pre-processing.

- **Gene Expression Omnibus (https://www.ncbi.nlm.nih.gov/geo/)**

  This is a well-known database in genomics having nearly all the datasets related to all the species and organs. The dataset is present here in raw as well as pre-processed form and all the information about the contributor of the dataset is provided.

- **10X Genomics (https://www.10xgenomics.com/)**

  It is a widely used sequencing technology capable of sequencing millions of cells with Chromium technology. This technology covers different modalities of single-cell data be it scRNA-seq, scATAC-seq, immune profiling or spatial gene expression and target gene expression data. The data format 10x genomics is also compatible with packages like scanpy (single-cell analysis in python), which make it more efficient to use for data pre-processing.

2.2.2 Data Pre-processing

Almost all the datasets used in the web server for different methods follow a common procedure for preprocessing. Following are the data preprocessing steps applied to both reference and query datasets:

- **Reading Data in annotated data format object**

  The most compatible data object to work with single-cell data for the Scanpy python package is annotated data object (or ann data object). This data object contains a mix of pandas dataframes and numpy arrays. Mainly the matrix is stored in a numpy ndarray format, where the dimension is n(no. of observations)

* d (no. of variables). Observations refer to the cells (cell barcodes) and variables correspond to the genes (gene symbols/ gene ids). Also, annotations for the observations and variables are stoked in the form of pandas series objects. Annotations usually include information like cell-type information, batch/study information, metrics like mean and dispersion score of genes, information about highly variables genes etc. Usually, the barcode file, genes file and count matrix are read separately and then combined with all three in a single annotated data object.

- **Normalizing and Log transforming the data**

  Since the count data, has a different range of values for the features due to the sequencing depth, it is necessary to normalize the data before going for any type of analysis. We have applied the CPM (counts per million) normalization for our data and then log-transformed the data so that the data actually gives equal importance to the values of highly expressed and lowly expressed genes.

- **Selecting highly variables genes**

  The data is having a large number of genes or features. These genes contain both types of genes like housekeeping / ribosomal genes and immune response elements. The housekeeping genes are the least variable and immune response elements are the most variable, and we take into account the highly variable genes for our analysis. For most of the methods, we have taken the highly number of genes to be between 1000-2000. This reduces the matrix size drastically and improves the computational efficiency for the training of our models.

2.3 Methods

A large number of different methods are available for doing single-cell analysis which uses transfer learning techniques. Transcend, hosts many pretrained models covering major species and organs. Currently, seven models are available in the webserver, which are used to do either clustering, cell-type annotation, dimensionality reduction and imputation etc. We will cover all the models briefly. Following are the transfer learning methods used in the webserver:

- **Single-cell Variational Inference (scVI)**[1]

  It is a fully probabilistic approach for the analysis of scRNA-seq data. It is a hierarchical bayesian model, with conditional distributions specified by deep neural networks, which can be efficiently trained on large datasets.

Generally, scRNA-seq workflows combine many standard machine learning methods. The workflow includes the normalization of the data, then reducing the dimensions of the data for visualization, applying an ad-hoc algorithm for correcting batch effects, then clustering the data to identify cell states from the corrected latent space and finally performing differential expression to match the clusters to known cell types. Since all the steps in the workflow include certain assumptions, so scVI led to the unified model assumptions for the whole pipeline.

Hence, a deep generative model that addresses all these tasks and easily scales by leveraging the stochastic optimization is introduced which is known as scVI (single-cell variational inference).

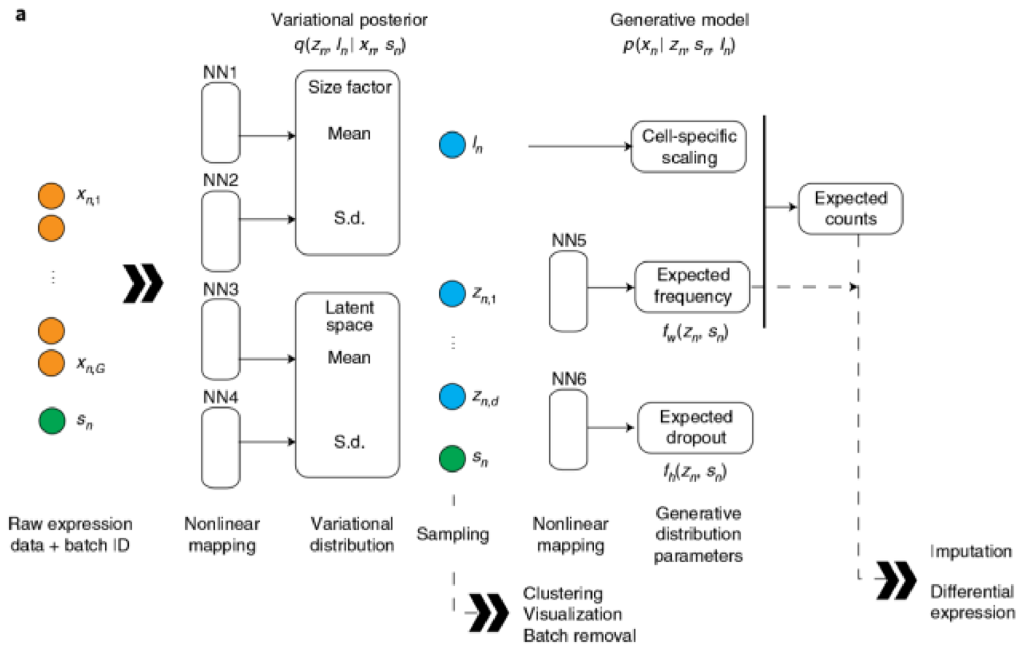


Figure 2.1: scVI Model Architecture [1]

The expression profile of each cell is encoded through a nonlinear transformation (encoder neural network) to a low-dimensional latent vector (here it is set to 10 dimensions) of random variables. The latent representation is then decoded by another nonlinear transformation (decoder neural network) to generate a posterior estimate of the distributional parameters of each gene in each cell. The model maps the latent space to the parameters of a ZINB (zero-inflated negative binomial distribution).

Each point in a low-dimensional latent vector is a cell, that can be used for visualization and clustering of the data. And this low-dimensional vector contains the possible biological variation in the data which can be further analysed through clustering.

- **Single-cell ANnotation using Variational Inference (scANVI)** [2]

  scANVI is an extension of scVI (single-cell Variational Inference), which is a semi-supervised method for automating the cell type annotation process. It leverages the existing cell annotations, to assign cell types to unseen data. Refer to Fig. to know how the scVI method is extended for the annotation problem.

  From Fig. , it can be inferred that cell type annotations are also given as input to the encoder network to come up with a single joint latent space which gives an embedding for the cells. The scANVI model uses a bayesian approach in a semi-supervised fashion to annotate cells. Once fitted, the model is able to provide posterior estimates for the unobserved cell state c_n, which can be particularly useful when labels cannot be entirely trusted. Because the marginal distribution $p(x_{ng}, c_n|s_n)$ if c_n observed is not amenable to exact Bayesian computation, the posterior inference is intractable. Consequently, we use variational inference parameterized by neural networks to approximate the posterior distribution[1, 11].

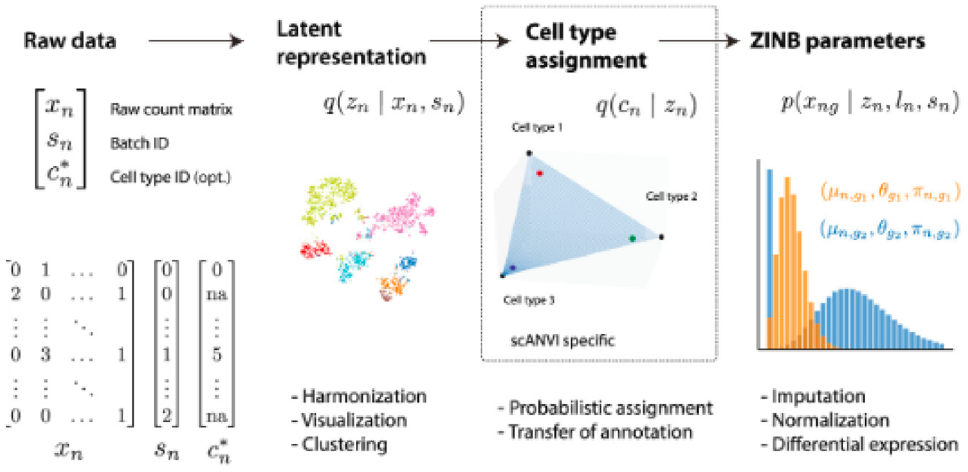


Figure 2.2: scANVI algorithmic overview[2]

- **single-cell Embedded Topic Modelling (scETM)**[3] single-cell Embedded Topic Model (scETM), is a generative topic model that facilitates integrative analysis of large-scale single-cell transcriptomic data.

Embedded Topic model:

Topics models are statistical tools for discovering the hidden semantic structure in a collection of documents. ETM enjoys the good properties of topic models and the good properties of word embeddings[12]. The concept of embedded topic modelling originated from modelling the text data, where documents and words were considered as samples and features respectively and the matrices values correspond to the word frequency.

ETM for scRNA-seq data:

Inheriting the idea of ETM applied to text data, for single-cell RNA-seq data, the documents are replaced with the samples and the words are replaced with the genes and at last, the matrices value corresponds to the read count (UMI) in place of the word frequency.

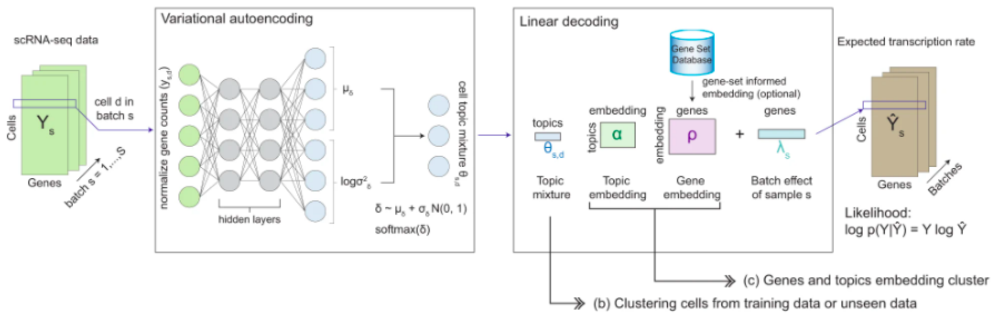


Figure 2.3: scETM Model Architecture[3]

In the context of scRNA-seq data analysis, each sampled single-cell transcriptome is provided as a vector of normalized gene counts to a 2-layer fully connected neural network (i.e encoder) which infers the topic mixing proportions of the cell. The trained encoder on a reference scRNA-seq data can be used to infer the topic mixture of unseen scRNA-seq data collected from different tissues or species[3].

scETM model training architecture is shown in Fig. above, given as input the scRNA-seq data matrices across multiple experiments or studies (i.e., batches), scETM models the single-cell transcriptomes using an embedded topic-modelling approach. Each scRNA-seq profile serves as an input to a variational autoencoder (VAE) as the normalized gene counts. The encoder network produces a stochastic sample of the latent topic mixture, which can be used for clustering cells[3].

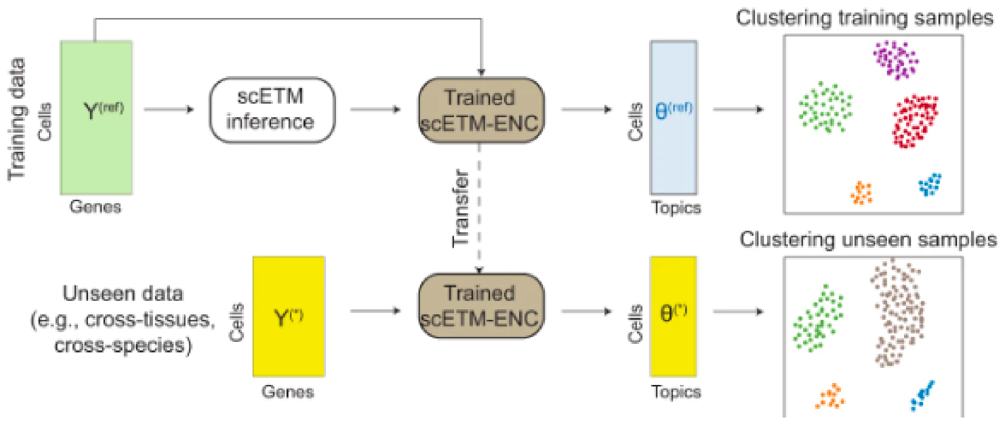


Figure 2.4: scETM Workflow[3]

The workflow in Fig 2.4 is used to perform zero-shot transfer learning. The trained scETM-encoder on a reference scRNA-seq dataset is used to infer the cell topic mixture * from an unseen scRNA-seq dataset without training them. The resulting cell mixtures are then visualized via UMAP visualization and evaluated by standard unsupervised clustering metrics using the ground-truth cell types. Same way, the final model can be fine-tuned to the target dataset to adapt to the nuances of the target query data to infer similar cell relations more accurately.

For interpretability, a linear decoder with the gene and topic embeddings as the learnable parameters is used. Specifically, the cells-by-genes count matrix is factorized into a cells-by-topics matrix (inferred by the encoder), topics-by-embedding , and embedding-by-genes matrices. This tri-factorization design allows for exploring the relations among cells, genes, and topics in a highly interpretable way[3].

- **Using transfer learning from prior reference knowledge to improve the clustering of single-cell RNA-Seq data (scRNA)**[13]

  This method attempts to get key information from large reference/source datasets and use it to cluster the target dataset using the non-negative matrix factorization method. This method works best when there is complete overlap between the cell types present in the source and target query dataset. When there is a partial overlap in the cell type, this method still performs better than other state-of-art methods but with a little margin. However, if there is no overlap between the cell types, then the method doesn't perform reasonably well as compared to other state of the art methods.

Other models:

More transfer learning methods have been implemented like **transformer Variational Auto Encoder (trVAE)**, which imposes strict regularization than conditional VAEs to reconstruct samples within the same conditions and across conditions[9].

Total Variational Inference (TotalVI) is a model that has been used for the imputation of the single-cell RNA-seq data. It combines the paired measurement of RNA and surface protein from the same cell, and it learns a joint low-dimensional probabilistic representation of RNA and protein measurements. Through TotalVI, for a reference RNA and protein data, a joint latent space is learned and then for only RNA data as query dataset, its protein data is imputed with the help of learned reference latent space[10].

Other methods like **Learning with AuToEncoder (LATE)** are also used for imputation and have proved to outperform state of the art methods for the same. The LATE method trains an autoencoder with random initial values of the parameters, whereas the TRANSLATE (TRANSfer learning with LATE) method further allows for the use of a reference gene expression data set to provide LATE with an initial set of parameter estimates[14].

2.4 Implementation of the Webserver

2.4.1 Application Architecture

Transcend, the webserver is built using state of the art technologies for easy scalability. To keep the application architecture suitable for all the devices like mobile and desktop etc, we have followed the API (Application Programming Interface) approach, which makes it suitable to work across different platforms without working on everything from scratch. Application frontend and backend are kept separated so that there is not much dependency of frontend with backend.

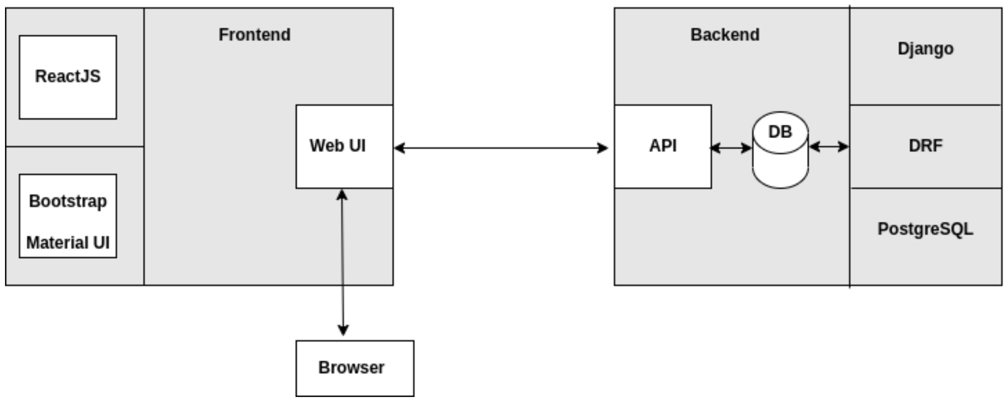


Figure 2.5: Application Design

An abstract view of the application is shown in the Fig 2.5 , where we are using React JS for the frontend along with responsive elements like bootstrap and easy to use wrappers like Material UI for building the dynamic and interactive frontend with all sorts of functionalities for searching, sorting and downloading files.

The backend of the application is built using Django (a python-based framework) and the API development is done using Django REST framework (a Django bases framework). The database used is PostgreSQL, which offers better security and access to multiple users to interact with the database at a single point in time.

2.4.2 System Design

Transcend webserver is made using three components namely frontend, backend and database. Each of the components is described below.

Database

The database of the application is the most crucial part of Transcend because it contains the complete details about every model. First database table includes everything about the model like:

- **ModelName:** model name for the specific methods used
- **Contributor:** the name of lab, person or organization who is responsible for the hosting of these models

- **DataName:** the dataset name on which the model is trained
- **Species:** the species of the dataset like human, mouse etc
- **Organ/ Tissue:** the organ/ tissue name on which model is trained like blood, liver, kidney and pancreas etc
- **Action:** This provides clickable buttons for downloading the model file and notebook file
- **Rating:** it provides the statistics for the number of times a model is downloaded and liked by the user
- **Number of cells:** number of cells used while training the model
- **Number of genes:** number of genes: number of genes used while training the model
- **Model reference/source:** the research paper / GitHub link referred for the paper, so that the model can easily be cited
- **Dataset reference/source:** dataset source paper or website for citation
- **Dataset download link:** For downloading the dataset
- **File upload(single zip file containing model and notebook):** This is a folder upload tab where a single zip file is uploaded containing the pretrained model file and associated code notebook used for training/ finetuning the code.

The second database table contains basic user information who has uploaded/ hosted the model like:

- **Name:** Name of the person having the issue/query/message
- **User Name:** name of the lab or organization the person is associated with
- **Email:** mailing address of the person
- **Occupation:** occupation of the person like student, faculty, researcher etc

The database management system used is PostgreSQL, which is good at securing the databases and allows efficient retrieval of data for multi-user access.

Backend Design

Transcend's backend is built using Django and Django REST framework for the APIs. Django provides an easy way to query the database with ORM (Object Relational Mapper) which helps in avoiding the raw SQL queries for doing the database CRUD operations. The REST (Representational State Transfer) APIs provide the way to access the API endpoints and serve the data to the UI. This REST API architecture has multiple benefits to a web application such as:

- Supports all standard HTTP methods
- Support multiple formats for data transfer
- Easy to implement
- Easy to scale

Frontend Design

Transcend's frontend is developed using React JS - a library designed by Facebook, for developing web applications. The best thing about React JS is its component-based architecture where different web pages are composed of different small components which can be reused efficiently. For making the pages responsive Bootstrap is used and for easy interactivity, Material UI is used which offer a range of functionalities with wrapper functions.

2.4.3 Deployment

Deployment of the website is done at the institute server (192.168.17.155). The frontend is provided as a build folder to the Django backend, so that frontend and backend can be hosted at a single server.

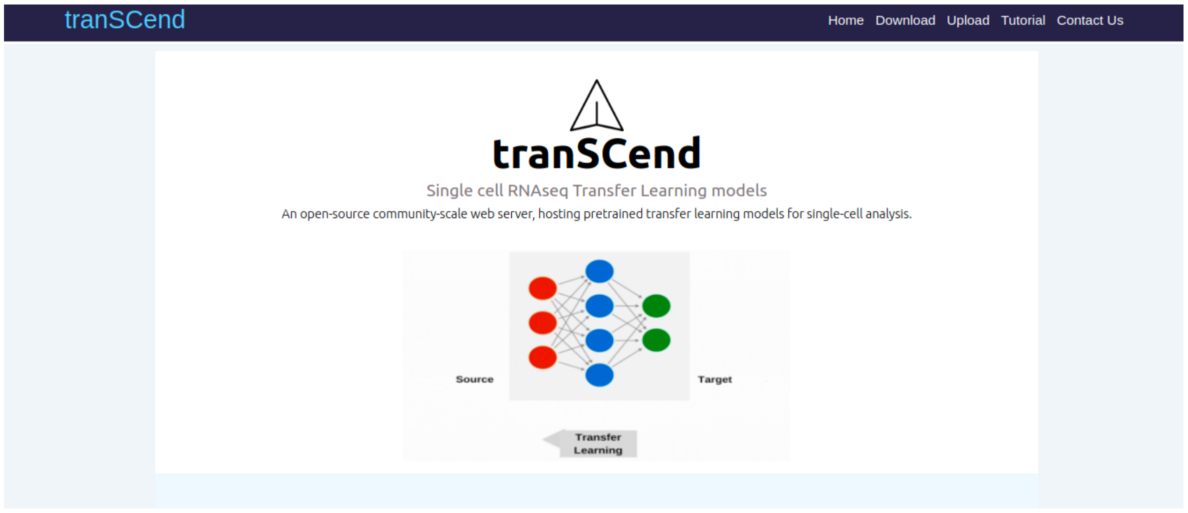
Gunicorn and Ngnix are used for the deployment of the application. Gunicorn interacts closely with the Django backend and prevents direct interaction of the client request to the server code. Nginx server is used to pass the traffic for the process and collects all request and forwards it to Gunicorn to fetch the required details.

Finally, the server is mapped to a domain name and secured using LetsEncrpyt, so that the user details are safe and nothing is exposed to third party websites.
The web server can be accessed at https://transcend.senguptalab.iiitd.edu.in/.

2.5 Transcend Webserver Features

2.5.1 Transcend Homepage

Fig 2.6 and Fig 2.7 show the homepage of the Transcend. There are two sections featuring models and a model hub on the homepage. Featured models, present the top four models available in the webserver based on the model popularity by likes and downloads. The model hub provides a clickable button to navigate to the model browser/download page and model download page.



An open-source community-scale web server, hosting pretrained transfer learning models for single-cell analysis.

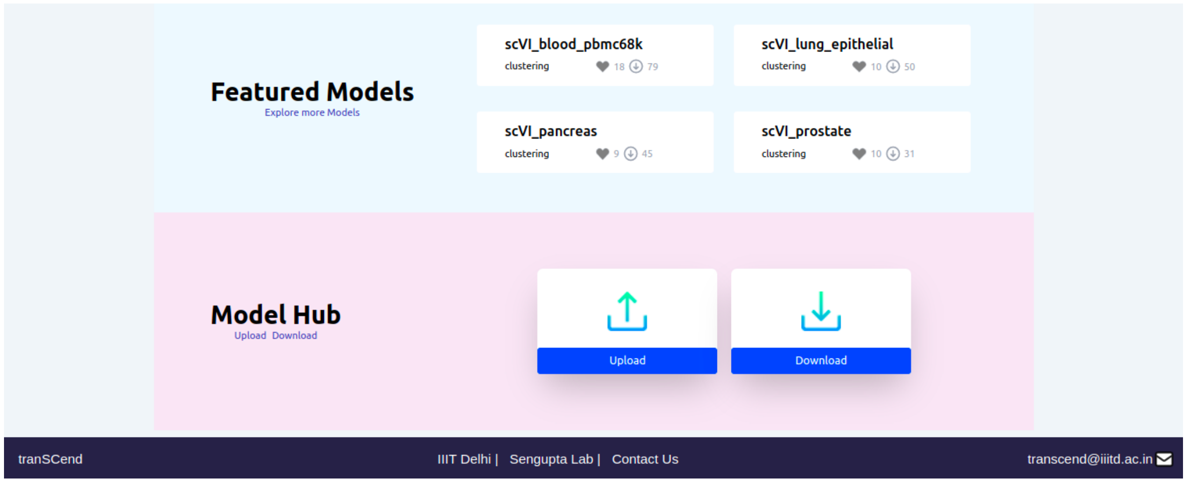
Figure 2.6: Homepage Top

2.5.2 Model Browser Page

Fig 2.8 shows the model browser page. This page contains a model table which contains every detail about the model with an action tab from where these models and notebook files can be directly downloaded.

Fig 2.10 and 2.11 shows the search bar functionality for the model browser table and column filter functionality respectively. It makes it efficient to use the table and search the required models.

Fig 2.12 shows how different columns can be grouped to search for specific features. Here it is shown how the table is grouped for the types of model.



Featured Models

Explore more Models

scVI_blood_pbmc68k
clustering 18 79

scVI_lung_epithelial
clustering 10 50

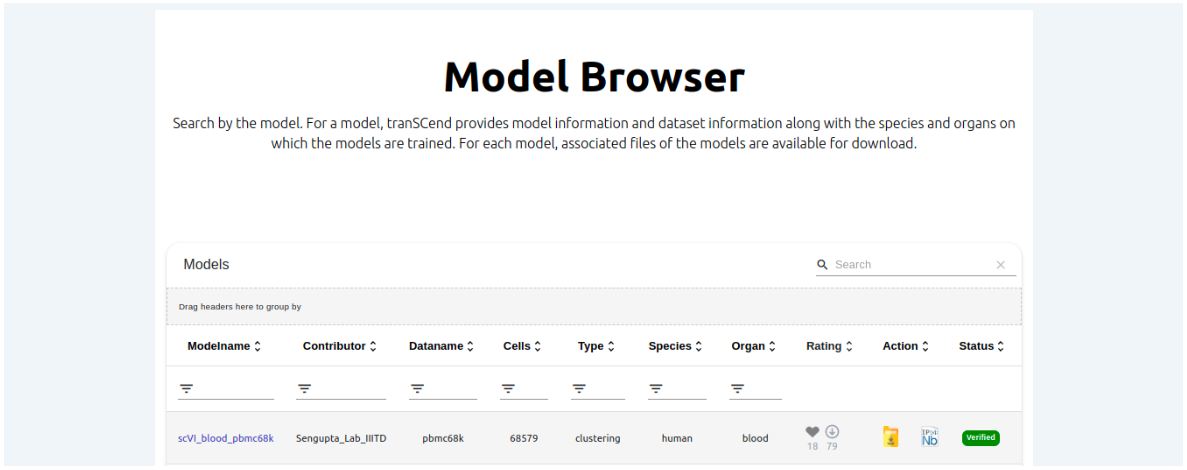
scVI_pancreas
clustering 9 45

scVI_prostate
clustering 10 31

Model Hub

Upload Download

Figure 2.7: Homepage Bottom



Model Browser

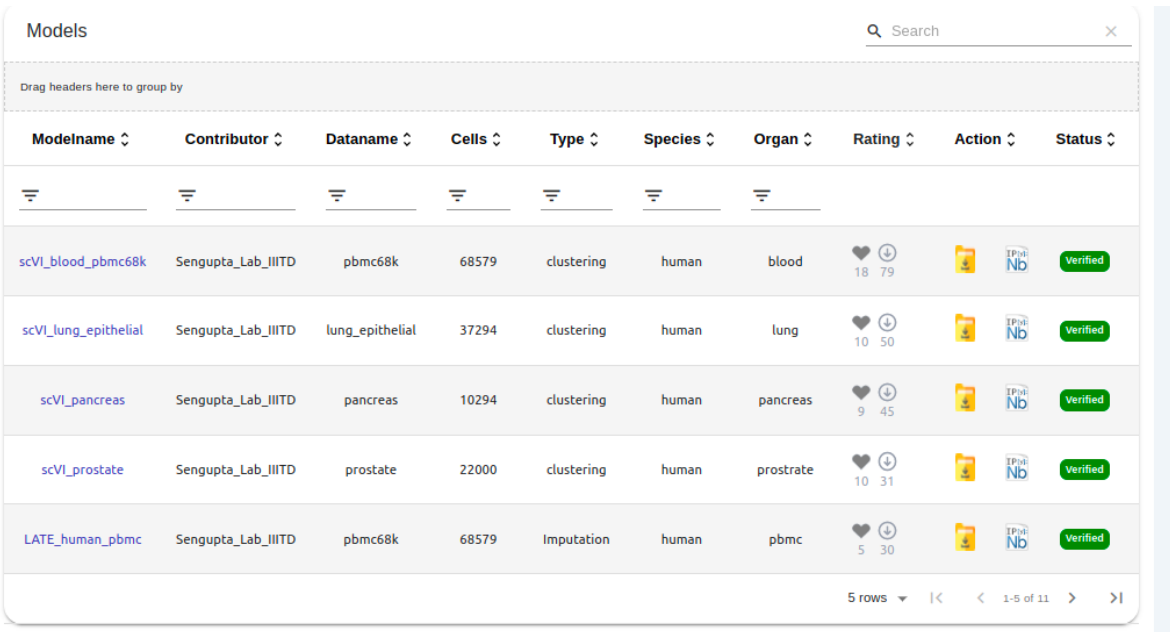
Search by the model. For a model, tranSCend provides model information and dataset information along with the species and organs on which the models are trained. For each model, associated files of the models are available for download.

Modelname	Contributor	Dataname	Cells	Type	Species	Organ	Rating	Action	Status
scVI_blood_pbmc68k	Sengupta_Lab_IIITD	pbmc68k	68579	clustering	human	blood	18 79		Verified

Figure 2.8: Model Browser Page

2.5.3 Model Card Page

Fig 2.13 and 2.14 is the model card page, which is specific to each model and contains all the details about the model and also contains the citation information for model as well as the dataset used.


Models

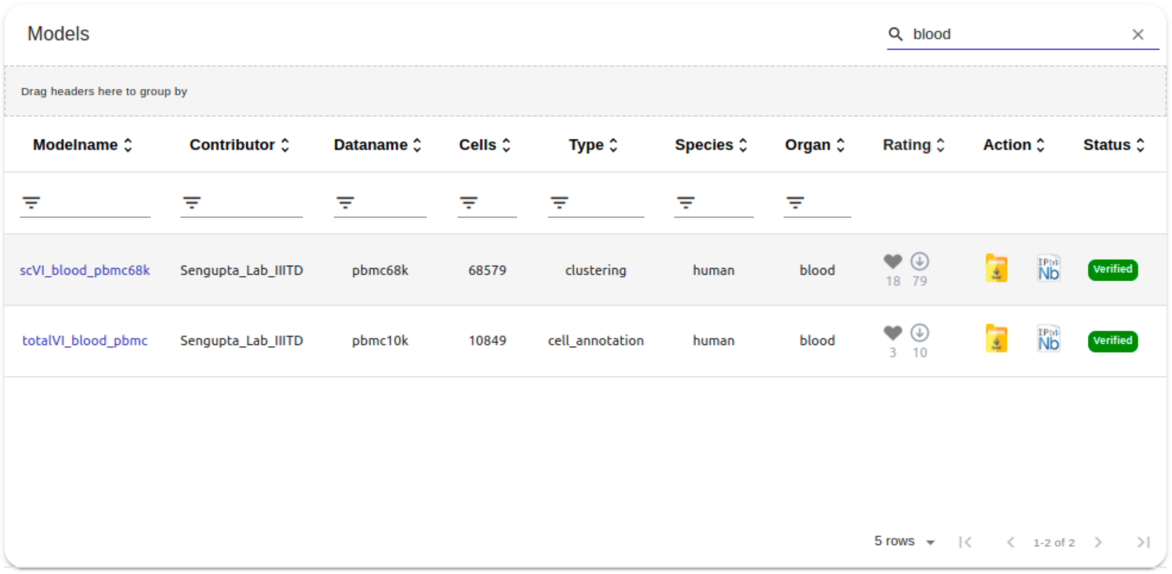
Search

Drag headers here to group by

Modelname	Contributor	Dataname	Cells	Type	Species	Organ	Rating	Action	Status
scVI_blood_pbmc68k	Sengupta_Lab_IIITD	pbmc68k	68579	clustering	human	blood	18 79		Verified
scVI_lung_epithelial	Sengupta_Lab_IIITD	lung_epithelial	37294	clustering	human	lung	10 50		Verified
scVI_pancreas	Sengupta_Lab_IIITD	pancreas	10294	clustering	human	pancreas	9 45		Verified
scVI_prostate	Sengupta_Lab_IIITD	prostate	22000	clustering	human	prostrate	10 31		Verified
LATE_human_pbmc	Sengupta_Lab_IIITD	pbmc68k	68579	Imputation	human	pbmc	5 30		Verified

5 rows 1-5 of 11

Figure 2.9: Full Model Browser Table


Models

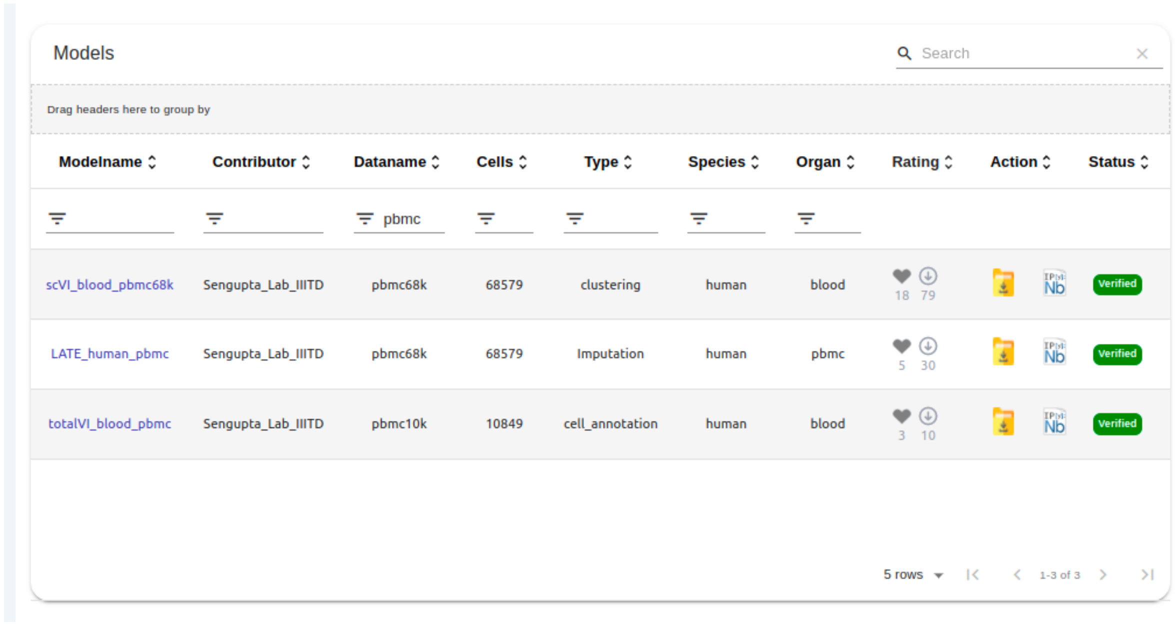
blood

Drag headers here to group by

Modelname	Contributor	Dataname	Cells	Type	Species	Organ	Rating	Action	Status
scVI_blood_pbmc68k	Sengupta_Lab_IIITD	pbmc68k	68579	clustering	human	blood	18 79		Verified
totalVI_blood_pbmc	Sengupta_Lab_IIITD	pbmc10k	10849	cell_annotation	human	blood	3 10		Verified

5 rows 1-2 of 2

Figure 2.10: Model Browser Table Search Bar



Models

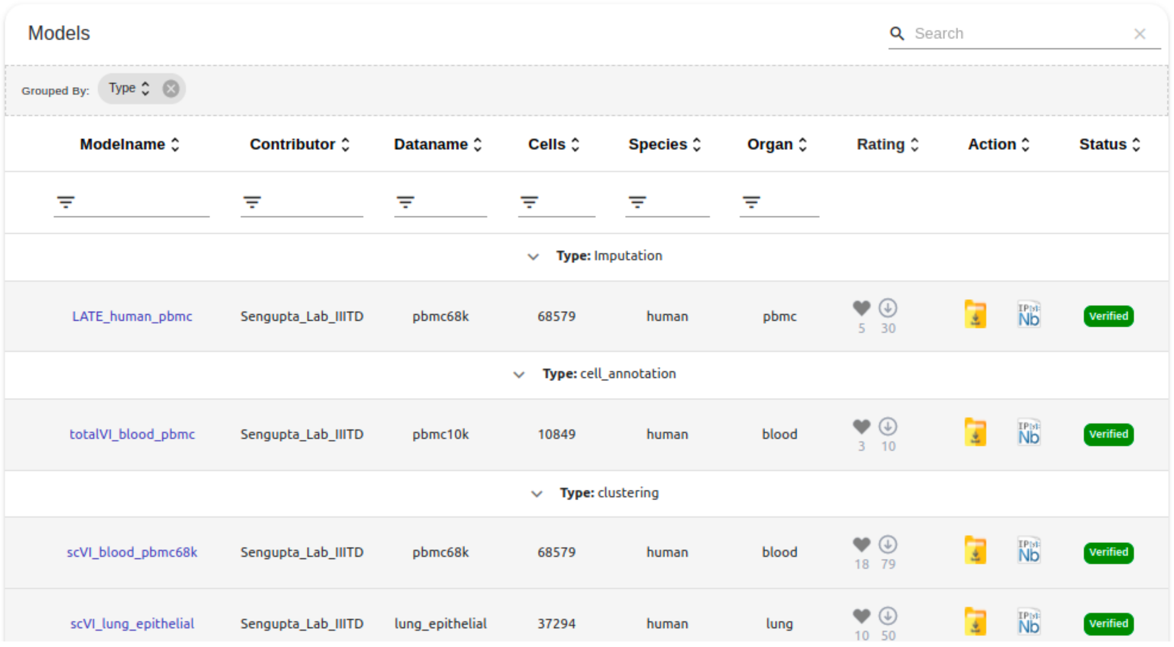
Search

Drag headers here to group by

Modelname	Contributor	Dataname	Cells	Type	Species	Organ	Rating	Action	Status
		pbmc							
scVI_blood_pbmc68k	Sengupta_Lab_IIITD	pbmc68k	68579	clustering	human	blood	18 79		Verified
LATE_human_pbmc	Sengupta_Lab_IIITD	pbmc68k	68579	Imputation	human	pbmc	5 30		Verified
totalVI_blood_pbmc	Sengupta_Lab_IIITD	pbmc10k	10849	cell_annotation	human	blood	3 10		Verified

5 rows 1-3 of 3

Figure 2.11: Model Browser Table Column Filter



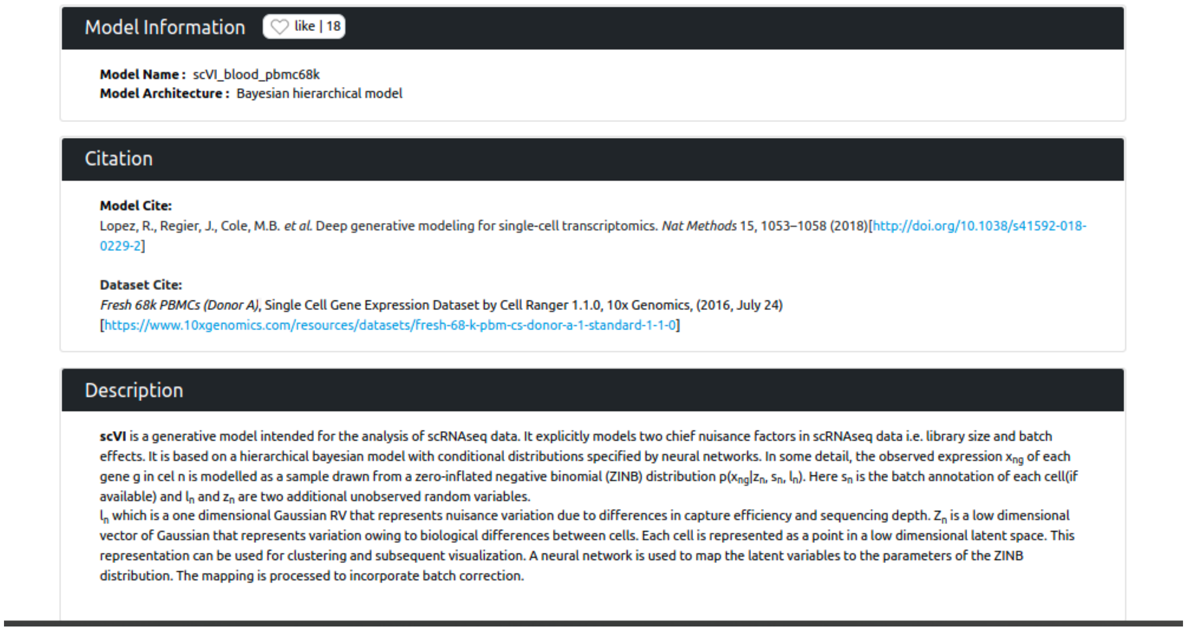
Models

Search

Grouped By: Type

Modelname	Contributor	Dataname	Cells	Species	Organ	Rating	Action	Status
Type: Imputation								
LATE_human_pbmc	Sengupta_Lab_IIITD	pbmc68k	68579	human	pbmc	5 30		Verified
Type: cell_annotation								
totalVI_blood_pbmc	Sengupta_Lab_IIITD	pbmc10k	10849	human	blood	3 10		Verified
Type: clustering								
scVI_blood_pbmc68k	Sengupta_Lab_IIITD	pbmc68k	68579	human	blood	18 79		Verified
scVI_lung_epithelial	Sengupta_Lab_IIITD	lung_epithelial	37294	human	lung	10 50		Verified

Figure 2.12: Model Browser Table Column Header Grouping



Model Information like | 18

Model Name : scVI_blood_pbmc68k
Model Architecture : Bayesian hierarchical model

Citation

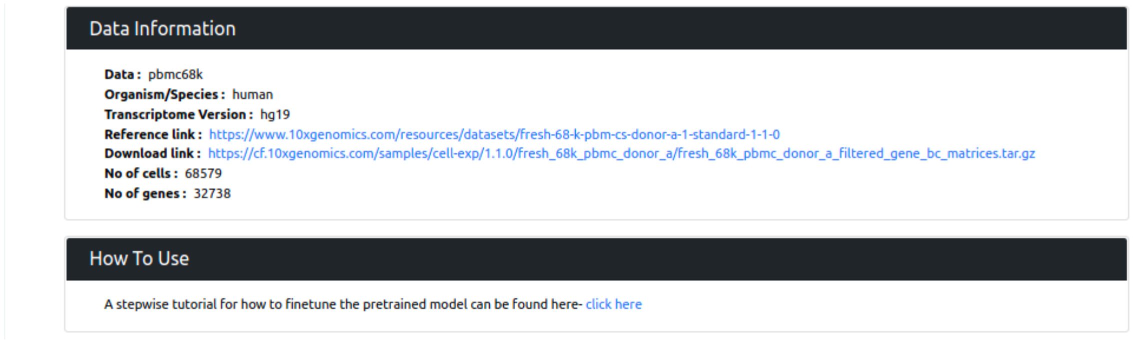
Model Cite:
Lopez, R., Regier, J., Cole, M.B. *et al.* Deep generative modeling for single-cell transcriptomics. *Nat Methods* 15, 1053–1058 (2018)[http://doi.org/10.1038/s41592-018-0229-2]

Dataset Cite:
Fresh 68k PBMCs (Donor A), Single Cell Gene Expression Dataset by Cell Ranger 1.1.0, 10x Genomics, (2016, July 24)
[https://www.10xgenomics.com/resources/datasets/fresh-68-k-pbm-cs-donor-a-1-standard-1-1-0]

Description

scVI is a generative model intended for the analysis of scRNAseq data. It explicitly models two chief nuisance factors in scRNAseq data i.e. library size and batch effects. It is based on a hierarchical bayesian model with conditional distributions specified by neural networks. In some detail, the observed expression x_{ng} of each gene g in cel n is modelled as a sample drawn from a zero-inflated negative binomial (ZINB) distribution $p(x_{ng}|z_n, s_n, l_n)$. Here s_n is the batch annotation of each cell(if available) and l_n and z_n are two additional unobserved random variables.
l_n which is a one dimensional Gaussian RV that represents nuisance variation due to differences in capture efficiency and sequencing depth. Z_n is a low dimensional vector of Gaussian that represents variation owing to biological differences between cells. Each cell is represented as a point in a low dimensional latent space. This representation can be used for clustering and subsequent visualization. A neural network is used to map the latent variables to the parameters of the ZINB distribution. The mapping is processed to incorporate batch correction.

Figure 2.13: Model Card Page Top



Data Information

Data : pbmc68k
Organism/Species : human
Transcriptome Version : hg19
Reference link : https://www.10xgenomics.com/resources/datasets/fresh-68-k-pbm-cs-donor-a-1-standard-1-1-0
Download link : https://cf.10xgenomics.com/samples/cell-exp/1.1.0/fresh_68k_pbmc_donor_a/fresh_68k_pbmc_donor_a_filtered_gene_bc_matrices.tar.gz
No of cells : 68579
No of genes : 32738

How To Use

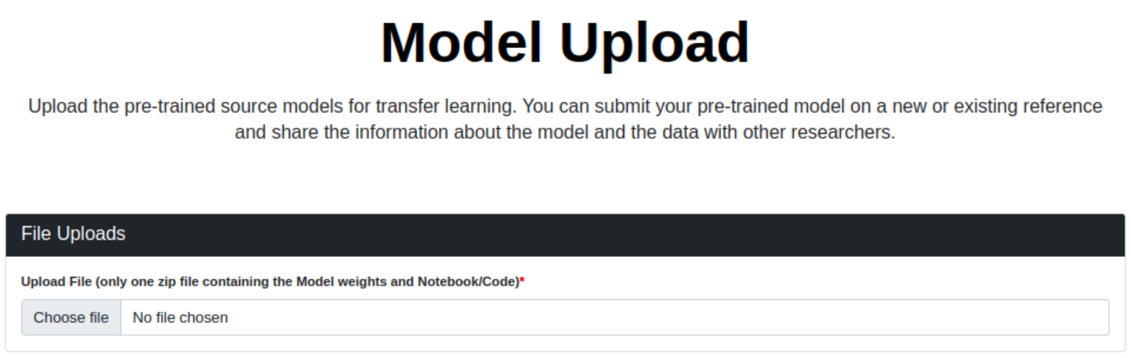
A stepwise tutorial for how to finetune the pretrained model can be found here- click here

Figure 2.14: Model Card Page Bottom

2.5.4 Model Upload Page

Fig 2.15, 2.16, 2.17, and 2.18 contain the model upload page with specific sections like file uploads section, user information section, model information section and dataset information section.

Transcend will be helpful to all researchers in the computational biology domain if they are looking for a single platform with all the transfer methods available for single-cell analysis. It is an open-source community platform which will grow with the



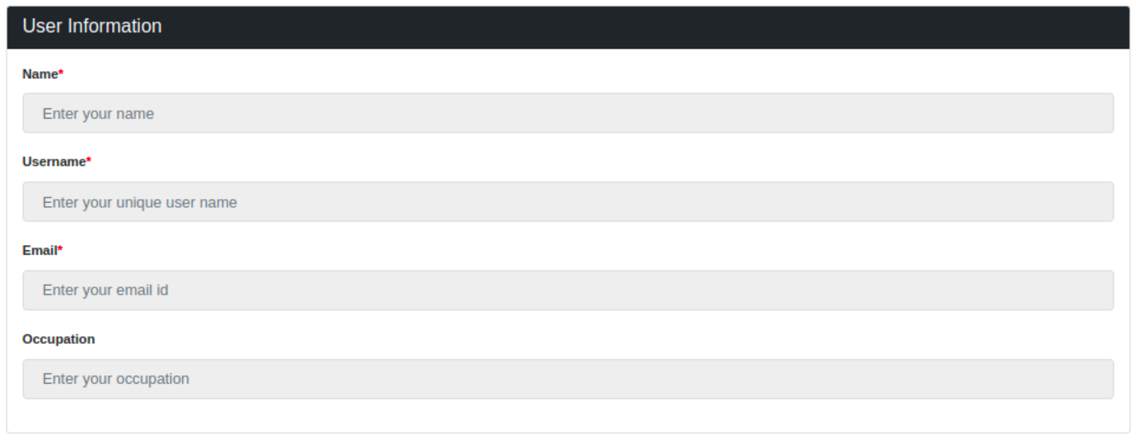
Model Upload

Upload the pre-trained source models for transfer learning. You can submit your pre-trained model on a new or existing reference and share the information about the model and the data with other researchers.

File Uploads

Upload File (only one zip file containing the Model weights and Notebook/Code)*

Figure 2.15: Model Upload File Upload Field



User Information

Name*

Enter your name

Username*

Enter your unique user name

Email*

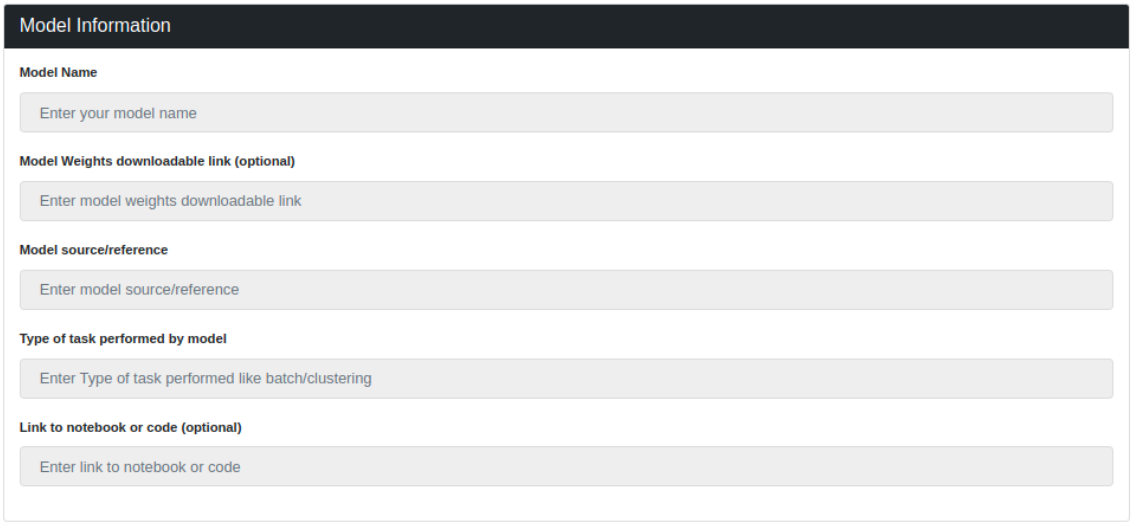
Enter your email id

Occupation

Enter your occupation

Figure 2.16: Model Upload User Information Field

community and will be a huge benefit for the biologist doing single-cell genomics.



Model Information

Model Name

Enter your model name

Model Weights downloadable link (optional)

Enter model weights downloadable link

Model source/reference

Enter model source/reference

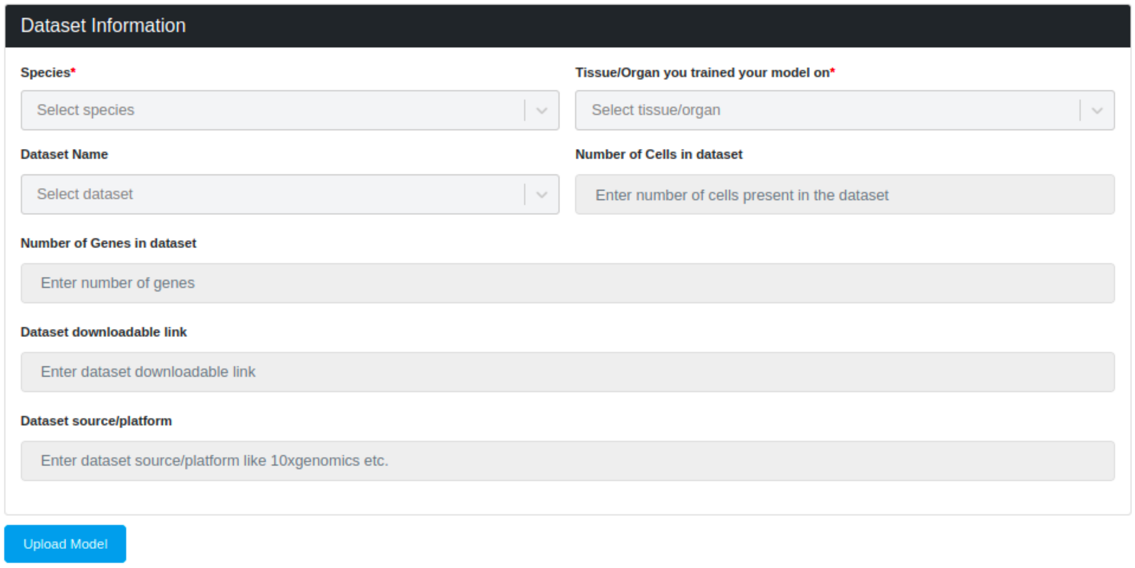
Type of task performed by model

Enter Type of task performed like batch/clustering

Link to notebook or code (optional)

Enter link to notebook or code

Figure 2.17: Model Upload Model Information Field



Dataset Information

Species*

Select species

Tissue/Organ you trained your model on*

Select tissue/organ

Dataset Name

Select dataset

Number of Cells in dataset

Enter number of cells present in the dataset

Number of Genes in dataset

Enter number of genes

Dataset downloadable link

Enter dataset downloadable link

Dataset source/platform

Enter dataset source/platform like 10xgenomics etc.

Figure 2.18: Model Upload Dataset Information Field

Chapter 3

Clustering single-cell RNA-seq data using Consensus Clustering

3.1 Consensus Clustering

Consensus clustering is also called an aggregation of clustering, which refers to the task of combining many different clusterings of a particular dataset into a single consensus clustering without using the original features of the data, and it gives a generalized representative of all the individual clusterings. Individual clustering obtained is usually unstable in nature due to random initialization, so consensus clustering gives a stable clustering output[15].

The cluster ensemble problem is more challenging than the classifier ensemble, as with the clusters, the labels are just symbolic and with that one must solve the correspondence problem. In addition, we know the clustering result from individual methods can vary in shape and number of clusters, also individual clustering methods share a particular view of data. Moreover, the exact number of right clusters is not known in advance. The right number of clusters often depends on the resolution at which data is inspected and generally more than one clustering result is correct with respect to that data[15].

3.2 Why consensus clustering?

Following are the reasons for using consensus clustering:

- **Knowledge reuse** Prior Individual clustering results can be effectively used for getting one single combined clustering, which is often an improved version of the individual clusterings and provides a generalized clustering result for multiple clusterings[15].

- **Distributed computing** Often the data is present in various geographic locations due to the organizational needs and constraints. So consensus clustering can really push the concept of distributed computing by leveraging only the base clustering labels to come up with a single generalized result. All the data need not be present in a single place and it maximizes the efficiency of computing the results[15].

- **Privacy-preserving** A cluster ensemble can be helpful in 'privacy-preserving' scenarios where it is not possible to gather data objects but different computing entities present at distributed locations can share very high-level abstract information, and that can be utilized to do feature-distributed clustering. Since different computing entities have access to different views of the data resulting in different features of the data, can be disjointed or they can partially overlap[15].

- **Improved quality and robustness of results**

  Cluster ensembles can also be compared with classifier ensembles. For regression or classification tasks, the classifier ensembles have shown to perform better with the involvement of strong learners and specifically when these base learners are powerful and have different inductive biases then cluster ensembles provide a better-generalized result. Also, ensembles are known to stabilize the individual models' results which are quite unstable at times[15].

3.3 How does Consensus clustering work?

Cluster ensembles generate a single consensus clustering label by using base labels obtained from multiple clustering algorithms. The consensus clustering label achieves a high clustering performance and improves the stability of the results.

In the Fig 3.1, we can see the consensus clustering workflow.

Objective function for the consensus clustering: Averaged Normalized Mutual Information score (ANMI)

The ANMI is a measure where different individual labellings $\lambda(q)$ are taken with the single optimal label λ in a pair and the normalized information score is averaged taking into account every individual label.

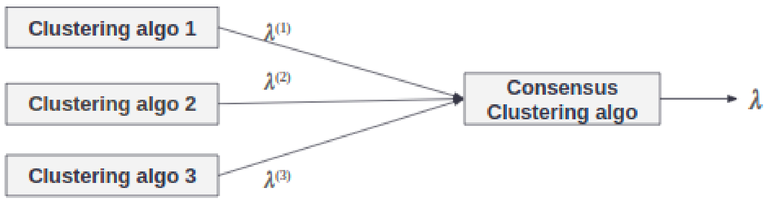


Figure 3.1: Consensus Clustering Workflow

Based on this pairwise measure of mutual information, a measure is defined between a set of **r** labellings, λ, and a single labelling λ as the average normalized mutual information (ANMI):

$$\phi^{ANMI}(\Lambda, \lambda) = \frac{1}{r} \sum_{q=1}^{r} \phi^{(NMI)}(\lambda, \lambda^{(q)})$$

The optimal clustering is the one having maximum average mutual information with all the individual labellings λ(q) in Λ given that the desired number of consensus clustered is k[15].

3.4 Data Collection

The data which is referred to in chapter 2 has been used for the methods used in the current topic of consensus clustering.

3.5 Methods

- **Cluster-based Similarity Partitioning Algorithm (CSPA)**[15]

  A clustering signifies a relationship between objects in the same cluster and can thus be used to establish a measure of pairwise similarity. This induced similarity measure is then used to recluster the objects, yielding a combined clustering. In short, It induces a similarity measure from the partitionings and then reclusters the objects.

- **Hyper-Graph Partition Algorithm (HGPA)**[15]

The cluster ensemble problem is formulated as partitioning the hypergraph by cutting a minimal number of hyperedges. All hyperedges are considered to have the same weight. Also, all vertices are equally weighted. This includes n-way relationship information, while CSPA only considers pairwise relationships. Now, we look for a hyperedge separator that partitions the hypergraph into k unconnected components of approximately the same size.

Note: Regular graph edge: An edge that connects exactly two vertices. **Hyper-Graph edge:** A hyperedge is a generalization of an edge in that it can connect any set of vertices.

- **Hybrid Bipartite Graph Formulation (HBGF)**[16]

  HBGF is a graph formulation that reduces a cluster ensemble problem to a bipartite graph partitioning problem. HBGF simultaneously models the instances and clusters of a given ensemble as vertices of a bipartite graph. It achieves lossless reduction and allows the similarity among instances and the similarity among clusters to be considered simultaneously in the final clustering[16].

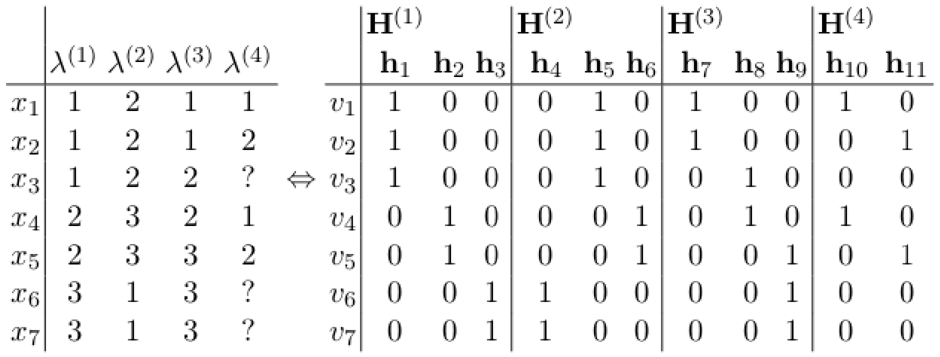
- **Non-Negative Matrix Factorization based Consensus Clustering(NMF)**[17]

  NMF is the problem of factorizing a given nonnegative data matrix X into two matrix factors, i.e., X AB, while requiring A and B to be nonnegative. It has been shown that the consensus clustering problem is equivalent to a symmetric nonnegative matrix factorization problem. Here different clustering labels generated from different methods are treated as partitions, and the objective is to reduce the distance between these partitions. The consensus partition is found, which leads to the desired optimal clustering labels[17].

- **Meta-Clustering Algorithm (MCLA)**[15]

  It is clustering of clusters. Each cluster is represented by a hyperedge. The idea in MCLA is to group and collapse related hyperedges and assign each object to the collapsed hyperedge in which it participates most strongly. The hyperedges that are considered related for the purpose of collapsing are determined by a graph-based clustering of hyperedges.

  The concatenated block matrix $H = H^{(1\cdots r)}(H^{1}...H^{(r)})$ defines the adjacency matrix of a hypergraph with n vertices and $\sum_{q=1}^{r} k^{(q)}$ hyperedges. Each column vector ha specifies a hyperedge ha , where 1 indicates that the vertex corresponding to the row is part of that hyperedge and 0 indicates that it is not. Thus, we have mapped each cluster to a hyperedge and the set of clusterings to a hypergraph.



	$\lambda^{(1)}$	$\lambda^{(2)}$	$\lambda^{(3)}$	$\lambda^{(4)}$
x_1	1	2	1	1
x_2	1	2	1	2
x_3	1	2	2	?
x_4	2	3	2	1
x_5	2	3	3	2
x_6	3	1	3	?
x_7	3	1	3	?

$\Leftrightarrow$

	$\mathbf{H}^{(1)}$			$\mathbf{H}^{(2)}$			$\mathbf{H}^{(3)}$			$\mathbf{H}^{(4)}$	
	$\mathbf{h}_1$	$\mathbf{h}_2$	$\mathbf{h}_3$	$\mathbf{h}_4$	$\mathbf{h}_5$	$\mathbf{h}_6$	$\mathbf{h}_7$	$\mathbf{h}_8$	$\mathbf{h}_9$	$\mathbf{h}_{10}$	$\mathbf{h}_{11}$
v_1	1	0	0	0	1	0	1	0	0	1	0
v_2	1	0	0	0	1	0	1	0	0	0	1
v_3	1	0	0	0	1	0	0	1	0	0	0
v_4	0	1	0	0	0	1	0	1	0	1	0
v_5	0	1	0	0	0	1	0	0	1	0	1
v_6	0	0	1	1	0	0	0	0	1	0	0
v_7	0	0	1	1	0	0	0	0	1	0	0

Figure 3.2: HyperGraph Representation
[15]

Steps for Meta clustering:

- **Construct Meta-graph** Vertices: h (hyperedges of H) Edges: $\sum_{q=1}^{r} k^{(q)}$
  The edge weights are proportional to the similarity between vertices. Formally, edge weight $w_{a,b}$ between two vertices h_a and h_b is defined by a Binary Jaccard measure of the corresponding indicator vectors h_a and h_b is

$$w_{a,b} = \frac{h_a^{+} h_b}{||h_a^2|| + ||h_b^2|| - h_a^{+} h_b}$$

  Since the clusters are non-overlapping (hard), there are no edges among vertices of the same clustering H (q) and, thus, the meta-graph is r-partite, as shown in Figure 3.3 4-partite graph.

- **Cluster hyperedges** Find matching labels by partitioning the meta-graph into k balanced meta-clusters. Graph partitioning package METIS is used in this step. This results in a clustering of the h vectors. Each meta-cluster has approximately r vertices. Since each vertex in the meta-graph represents a distinct cluster label, a meta-cluster represents a group of corresponding labels.

- **Collapse meta-clusters** For each of the k meta-clusters, we collapse the hyperedges into a single meta-hyperedge. Each meta-hyperedge has an association vector which contains an entry for each object describing its level of association with the corresponding meta-cluster. The level is computed by averaging all indicator vectors h of a particular meta-cluster. An entry of 0 or 1 indicates the weakest or strongest association respectively.

- **Compete for objects** Compete for objects,in this step, each object is assigned to its most associated meta cluster: Specifically, an object is assigned to the meta-cluster with the highest entry in the association vector. Ties are broken randomly. Note that not every meta-cluster can be guaranteed to win at least one object. Thus, there are at most k labels in the final combined clustering λ.

  Figure 3.3 shows 4-partite graph is shown. Dark edges shows edge weights where darker edges have more weightage. The three meta-clusters are indicated by symbols o, $\times$, and $+$.

3.6 Result

Consensus clustering has been done for two datasets namely Pancreas data and PBMC data.

3.6.1 Results of Individual Clusterings for Pancreas Data

Pancreas data:

- Source Dataset: Human and Mouse Pancreas data, Baron et al (GSE84133)
- Target dataset: Human Pancreas Data, Segerstolpe et al (E-MTAB-5061)

Clustering algorithm 1: single-cell Variational Inference (scVI)

This method has already been referred to in chapter 2, under Methods section.
Pre-training:

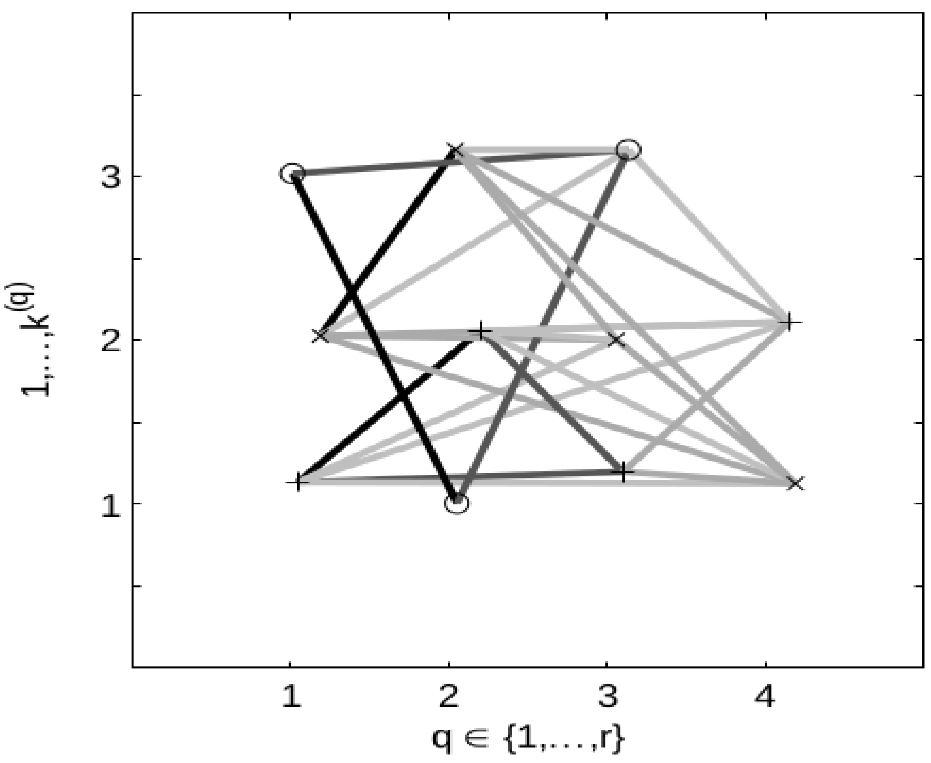


Figure 3.3: Mcla 4-Partite Graph
[15]

- Output: Clustering
- Architecture: Bayesian Hierarchical Model
- Type: Unsupervised learning
- Source Dataset: Human and Mouse Pancreas data, Baron et al (GSE84133)
- Cell types present: Yes
- observations (cells) * variables (#genes): 8569 * 20125
- Highly Variables Genes selected: 2000

The Source data can be visualized using UMAP and it is shown in Fig 3.4.

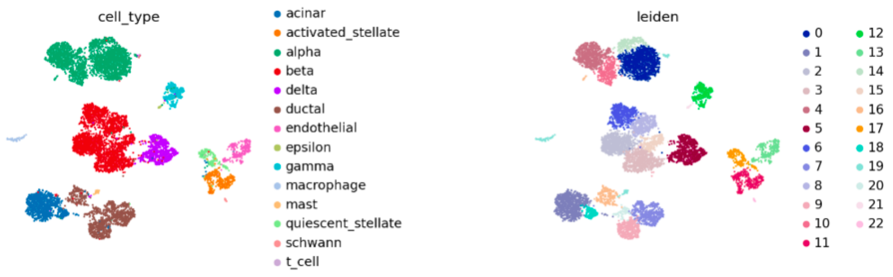


Figure 3.4: Source Pancreas UMAP

Fine-tuning:

- Target dataset: Human Pancreas Data, Segerstolpe et al (E-MTAB-5061)
- Cell types (Ground Truth) present: Yes
- observations (cells) * variables (#genes): 2394 * 34363
- Highly Variables Genes selected: 2000

The target data clustering result after fine-tuning the model is shown in Fig 3.5.

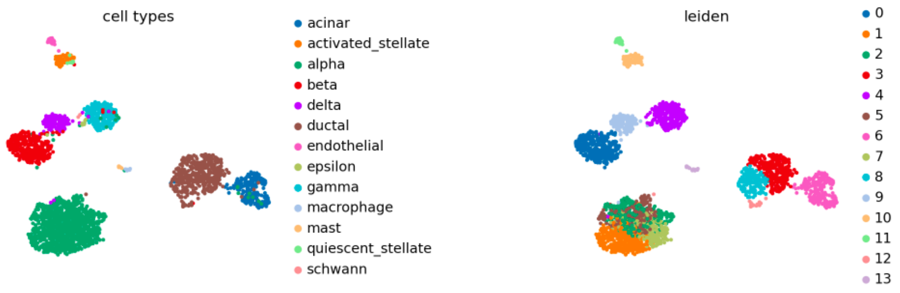


Figure 3.5: Target Data Clustering Result

Clustering Metrics: Adjusted Rand Index (ARI): 0.46, Normalized Mutual Information (NMI): 0.76, Silhouette score: 0.36

Clustering algorithm 2: single-cell ANnotation using Variational Inference (scANVI)

This method has already been referred to in chapter 2, under Methods section.
Pre-training:

- Output: Clustering and Classification
- Architecture: Bayesian Hierarchical Model
- Type: Semi-supervised learning
- Source Dataset: Human and Mouse Pancreas data, Baron et al (GSE84133)
- Cell types present: Yes
- observations (cells) * variables (genes): 8569 * 20125
- Highly Variables Genes selected: 2000

The Source data can be visualized using UMAP and it is shown in Fig 3.6.

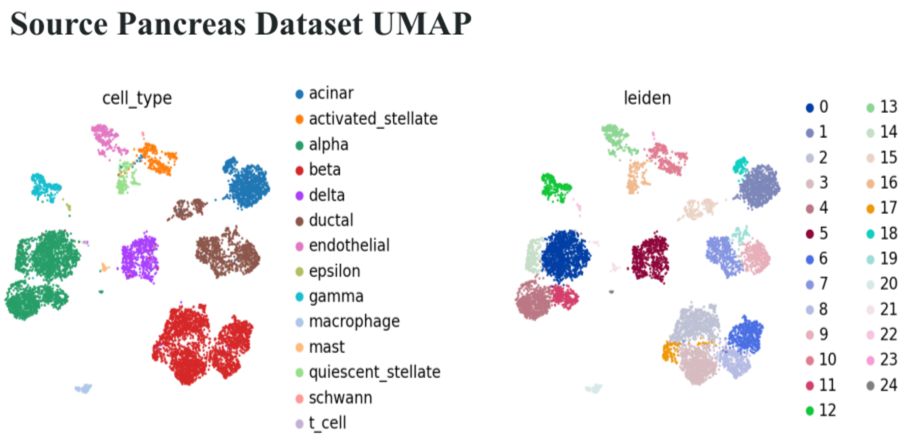


Figure 3.6: scANVI Source Pancreas UMAP

Fine-tuning:

- Target dataset: Human Pancreas Data, Segerstolpe et al (E-MTAB-5061)
- Cell types (Ground truth) present: Yes
- observations (cells) * variables (genes): 2394 * 34363
- Highly Variables Genes selected: 2000

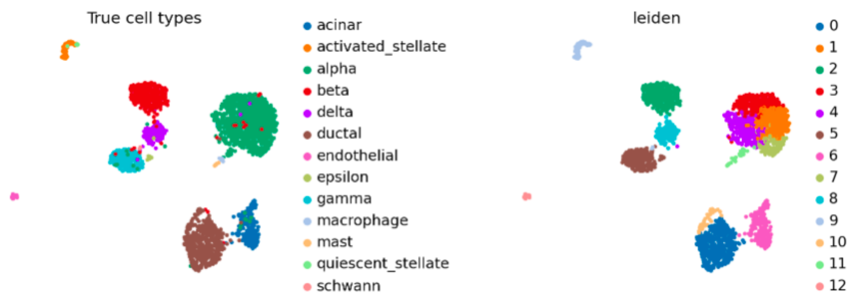


Figure 3.7: scANVI Target Clustering Result

Target data clustering result after fine-tuning the model is shown in Fig 3.23.

Clustering Metrics: Adjusted Rand Index (ARI): 0.51, Normalized Mutual Information (NMI): 0.77, Silhouette score: 0.43

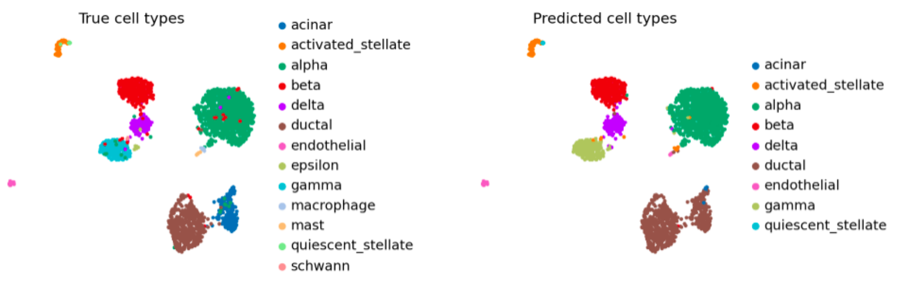


Figure 3.8: scANVI Target Classification Result

The target data classification result is shown in Fig 3.8 and also it can be visualized through a heatmap for observed vs predicted cell-types , as shown in Fig 3.9.

Classification results: Accuracy: 0.89, Weighted F1-score: 0.92

Clustering algorithm 3: single-cell Embedded Topic Modelling (scETM)

This method has already been referred to in chapter 2, under Methods section.

Pre-training:

- Output: Clustering
- Architecture: Generative Embedded Topic Model

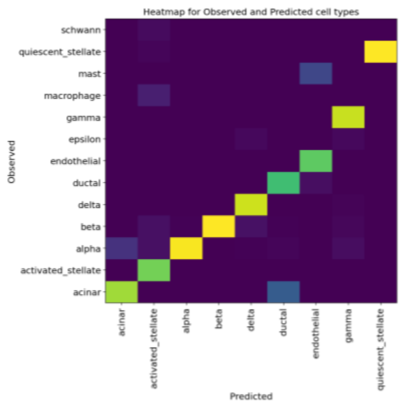


Figure 3.9: scANVI Target Classification Heatmap

- Type: supervised learning
- Source Dataset: Human and Mouse Pancreas data, Baron et al (GSE84133)
- Cell types present: Yes
- observations (cells) * variables (genes): 8569 * 20125

The source data can be visualized through UMAP as shown in Fig 3.10.

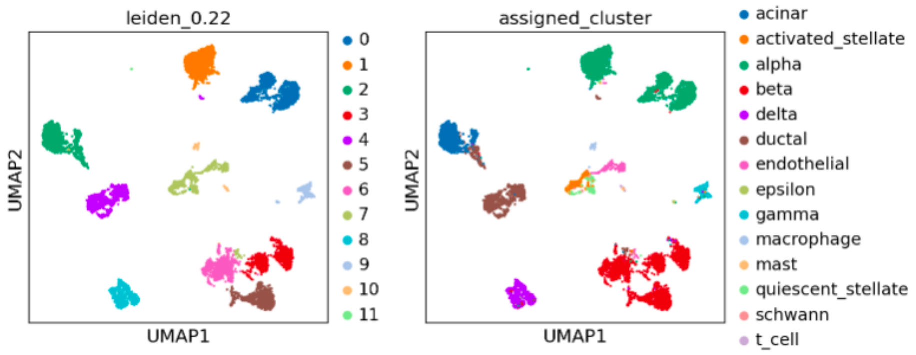


Figure 3.10: scETM Source Pancreas UMAP

Fine-tuning:

- Target dataset: Human Pancreas Data, Segerstolpe et al (E-MTAB-5061)
- Cell types (Ground truth) present: Yes
- Number of genes intersecting: 19112
- observations (cells) * variables (genes): 2394 * 19112

The target data clustering result after fine-tuning the scETM model is shown in Fig 3.11.

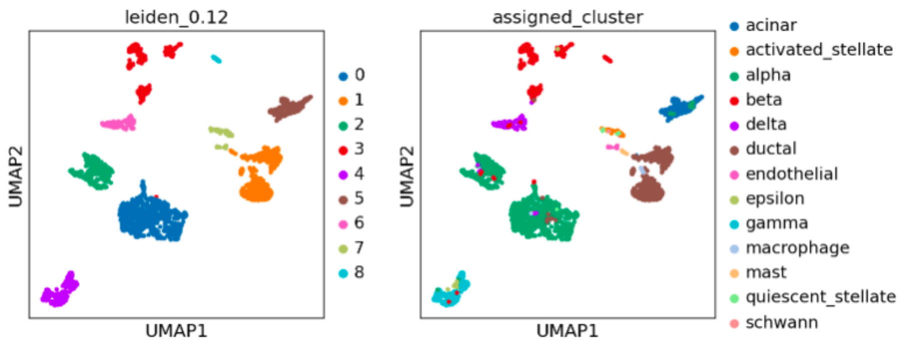


Figure 3.11: scETM Target Data Clustering Result

Clustering Metrics: Adjusted Rand Index (ARI): 0.73, Normalized Mutual Information (NMI): 0.83, Average Silhouette width (ASW) : 0.46

Common parameters for Consensus clustering:

- Number of output clusters (N) : 7 [Other: 14,12,10,9,8,6]
- Individual Clustering Methods Used: scANVI, scVI, scETM

Ground truth used:

- Human Pancreas Dataset cell type annotations
- Frozen PBMCs 3k Dataset cell type annotations

3.6.2 Consensus Clustering Results for Pancreas Data

Different results are obtained used different number of expected output clusters. Fig 3.12, 3.13, 3.14, 3.15, 3.16, 3.17, 3.18 show different results obtained with setting number of output clusters as 14,12,10,9,8,7,6 respectively.

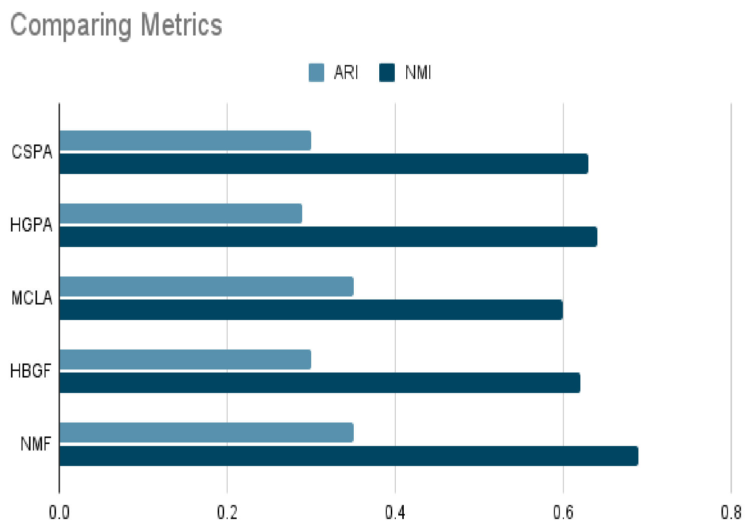


Figure 3.12: Number of Output Clusters(N)=14

Consensus clustering best results for N=7

Best results were obtained when number of output clusters (N) = 7, and then we visualized the consensus clustering results through UMAP as shown in Fig 3.19 and 3.20. Also we compared different clustering metrics for different concensus algorithms, as shown in Fig. 3.21.

3.6.3 Results of Individual Clusterings for PBMC data

PBMC data:

- Source Dataset: Fresh 68k PBMCs (Donor A) with cell-type annotation
- Target dataset: Frozen 3k PBMCs (Donor A)

Clustering algorithm 1: single-cell Variational Inference (scVI)

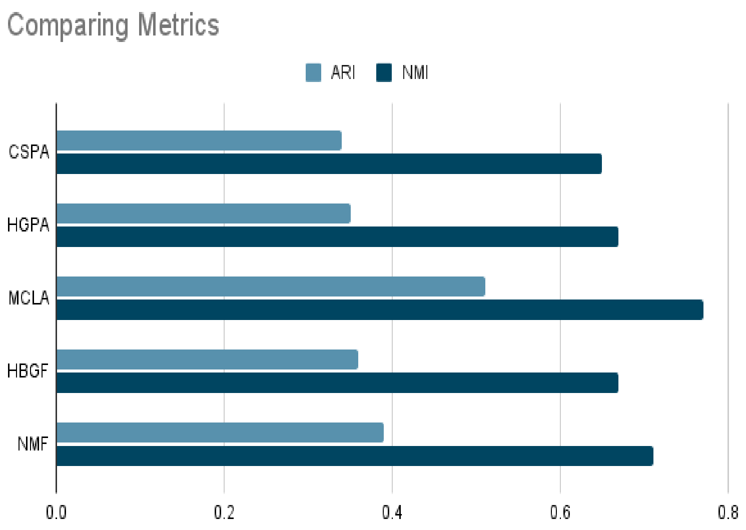


Figure 3.13: Number of Output Clusters(N)=12

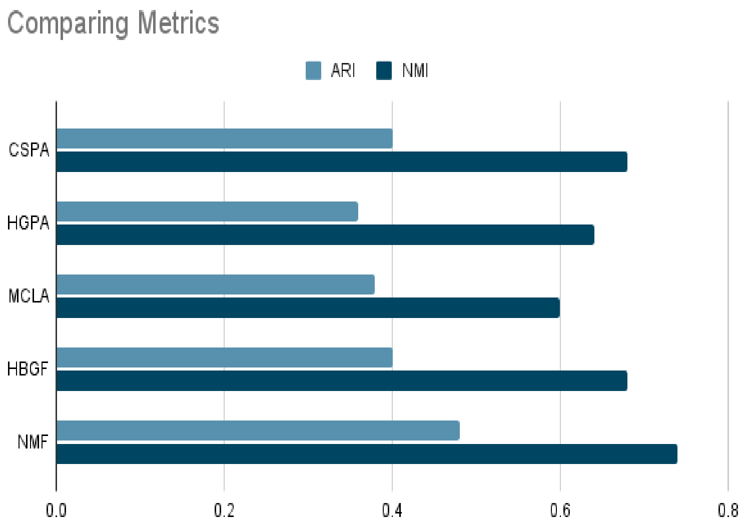


Figure 3.14: Number of Output Clusters(N)=10

This method has already been referred to in chapter 2, under Methods section.

Pre-training:

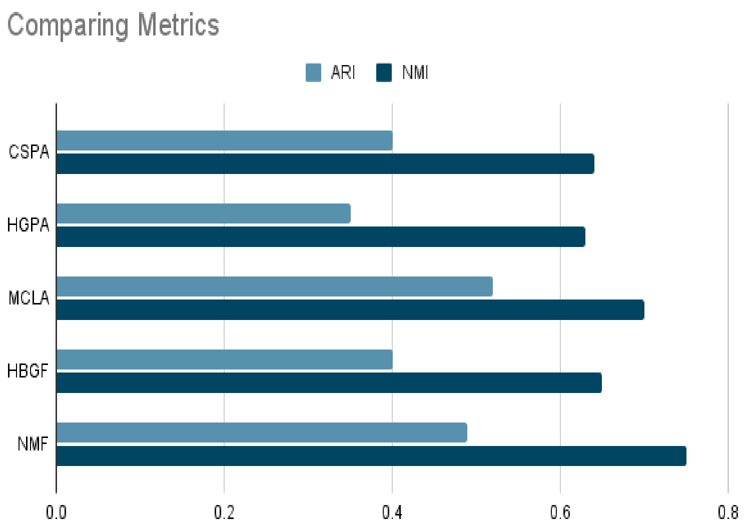


Figure 3.15: Number of Output Clusters(N)=9

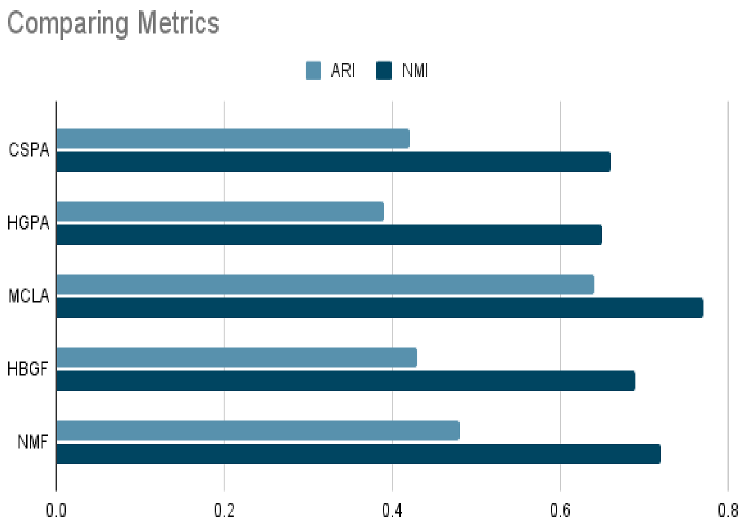


Figure 3.16: Number of Output Clusters(N)=8

- Output: Clustering
- Architecture: Bayesian Hierarchical Model

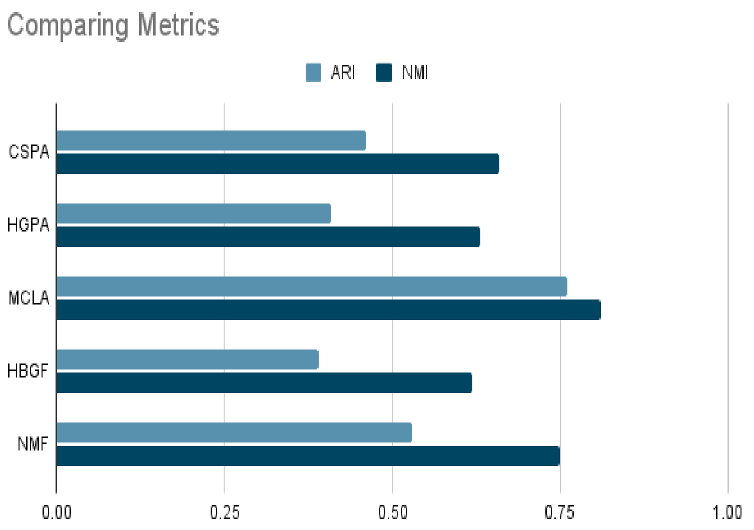


Figure 3.17: Number of Output Clusters(N)=7

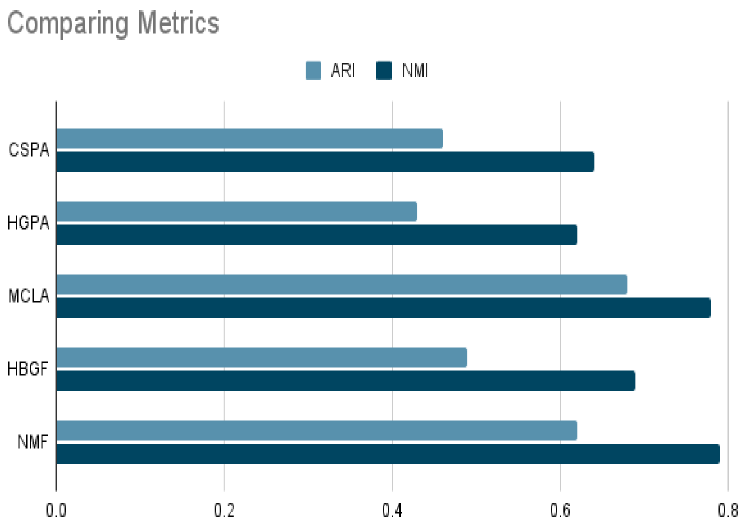


Figure 3.18: Number of Output Clusters(N)=6

- Type: Unsupervised learning
- Source Dataset: Fresh 68k PBMCs (Donor A) with cell-type annotation

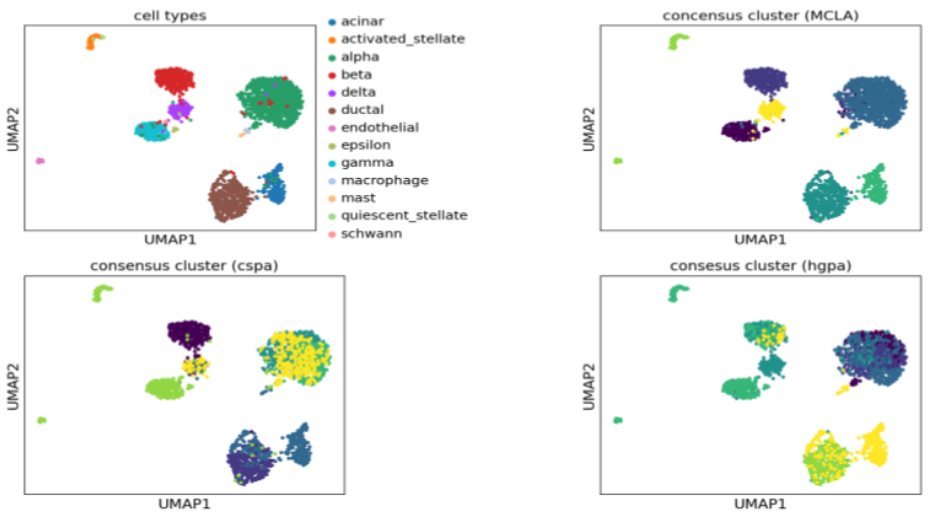


Figure 3.19: Clustering using mcla,cspa, hgpa

- Cell types present: Yes
- observations (cells) * variables (genes): 68579 * 32738
- Highly Variables Genes selected: 2000

Fine-tuning:

- Target dataset: Frozen 3k PBMCs (Donor A)
- Cell types (Ground truth) present: Yes
- observations (cells) * variables (genes): 2700 * 32738
- Highly Variables Genes selected: 2000

The Target data clustering result is obtained after fine-tuning the scVI model , as shown in Fig. 3.22.

Clustering Metrics: Adjusted Rand Index (ARI): 0.36, Normalized Mutual Information (NMI): 0.40, Silhouette score: 0.33

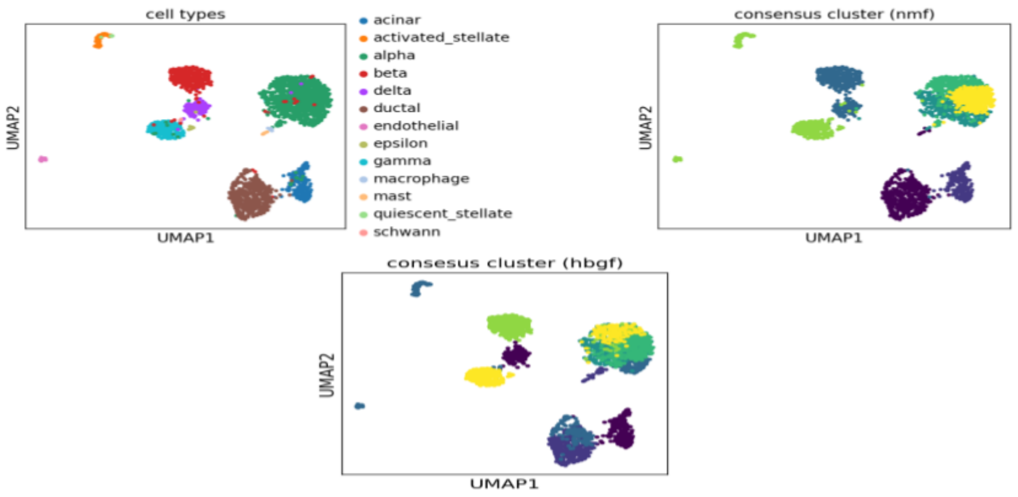


Figure 3.20: Clustering using nmf and hbgf

Clustering algorithm 2: single-cell ANnotation using Variational Inference (scANVI) This method has already been referred to in Chapter 2, under section 3 Methods.

Pre-training:

- Output: Clustering and Classification
- Architecture: Bayesian Hierarchical Model
- Type: Semi-supervised learning
- Source Dataset: Fresh 68k PBMCs (Donor A) with cell-type annotation
- Cell types present: Yes
- observations (cells) * variables (genes): 68579 * 32738
- Highly Variables Genes selected: 2000

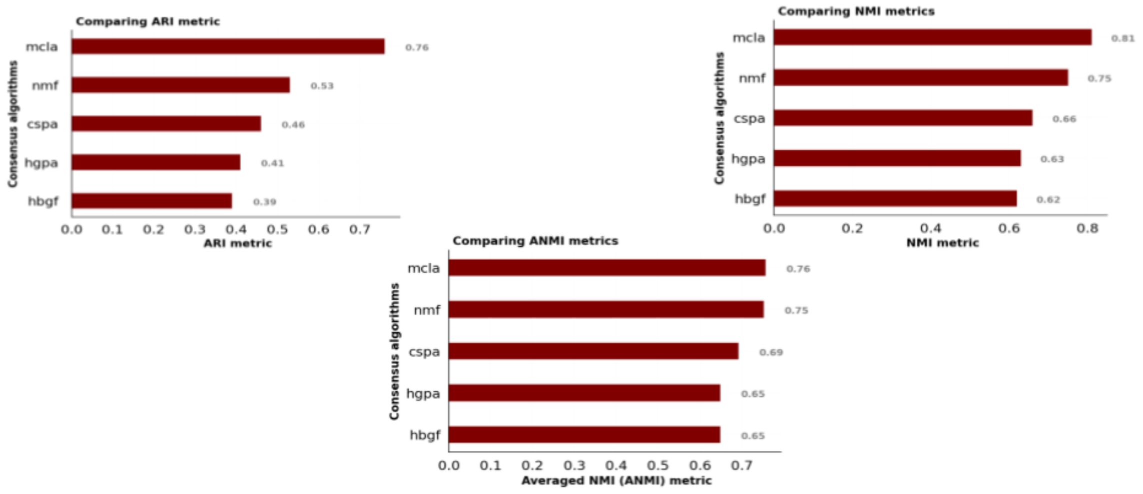


Figure 3.21: Comparison plots for clustering metrics

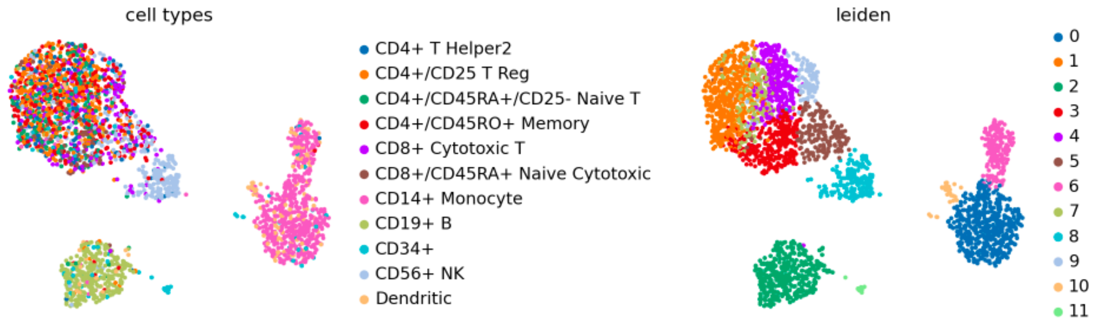


Figure 3.22: scVI PBMC Target Data Clustering Result

Fine-tuning:

- Target dataset: Frozen 3k PBMCs (Donor A)
- Cell types (Ground truth) present: Yes
- observations (cells) * variables (genes): 2700 * 32738
- Highly Variables Genes selected: 2000

The target data clustering is obtained after fine-tuning the scANVI model, is shown in Fig. 3.23.

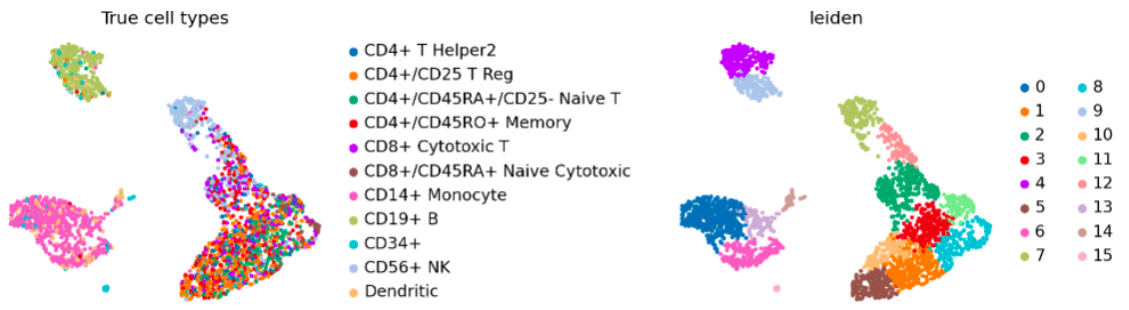


Figure 3.23: scANVI PBMC Target Data Clustering Result

Clustering Metrics: Adjusted Rand Index (ARI): 0.38, Normalized Mutual Information (NMI): 0.45, Silhouette score: 0.39

The target classification result using scANVI fine-tuned is shown in Fig. 3.24.

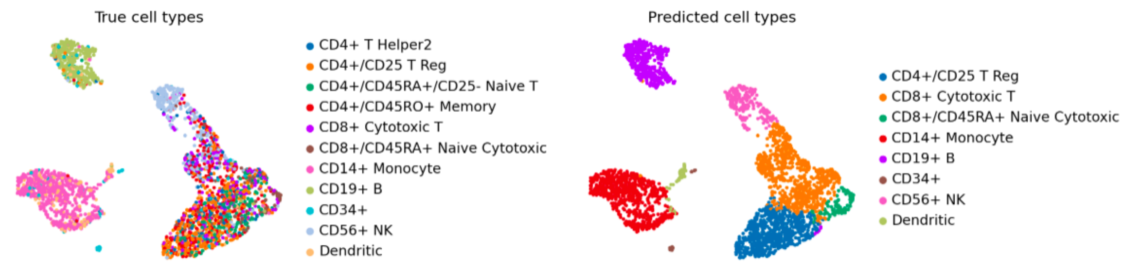


Figure 3.24: scANVI PBMC Target Data Classification Result

Classification result: Accuracy: 52.22, Weighted F1-score: 0.58

Clustering algorithm 3: single-cell Embedded Topic Modelling (scETM)
This method has already been referred to in Chapter 2, under section 3 Methods.

Pre-training:

- Output: Clustering
- Architecture: Generative Embedded Topic Model

- Type: supervised learning
- Source Dataset: Fresh 68k PBMCs (Donor A) with cell-type annotation
- Cell types present: Yes
- observations (cells) * variables (genes): 68579 * 32738

The source data can be visualized using UMAP , as shown in Fig. 3.25.

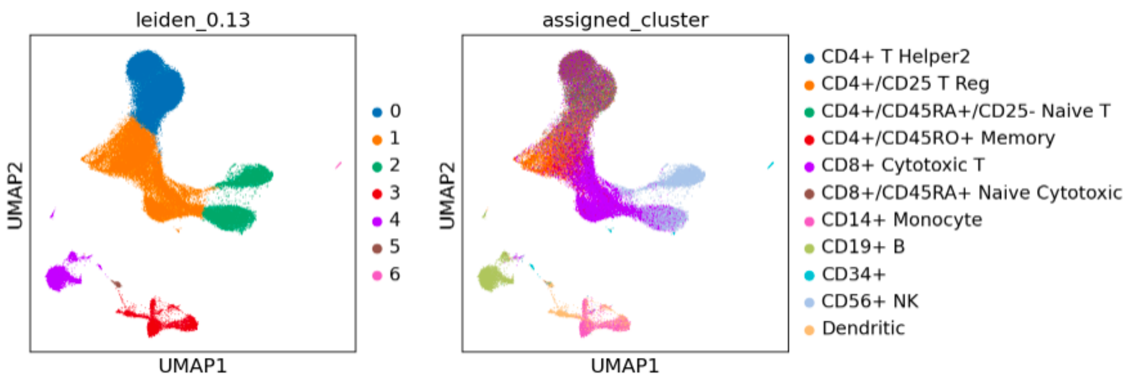


Figure 3.25: scETM PBMC Source UMAP

Fine-tuning:

- Target dataset: Frozen 3k PBMCs (Donor A)
- Cell types (Ground truth) present: Yes
- Number of genes intersecting: 32021
- observations (cells) * variables (genes): 2700 * 32021

The pbmc target data clustering result is obtained after fine-tuning the model , as shown in Fig. 3.26.

Metrics: Adjusted Rand Index (ARI): 0.35, Normalized Mutual Information (NMI): 0.49, Average Silhouette width (ASW) : 0.036

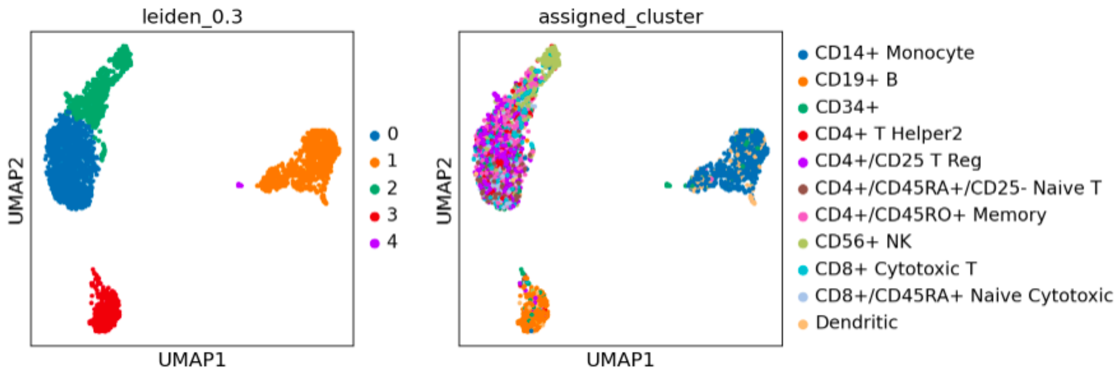


Figure 3.26: scETM PBMC Target Data Clustering Result

3.6.4 Consensus clustering Results for PBMC data

Different consensus clustering results are obtained by using different number of expected output clusters. Fig 3.27, 3.28, 3.29, 3.30, 3.31, 3.32 show the clustering metrics comparison for ARI and NMI by setting number of output clusters as 14,12,10,8,7,6.

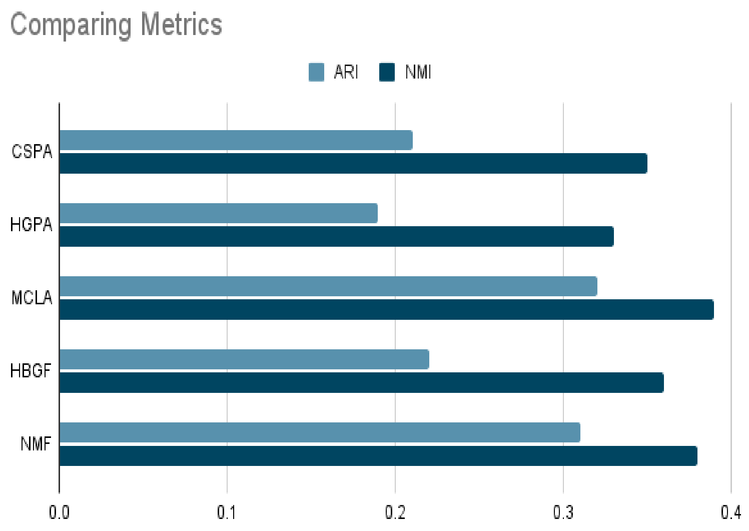


Figure 3.27: Number of Output Clusters(N)=14

Consensus clustering best results for N=7

Best consensus results were obtained when number of output clusters (N) = 7. The

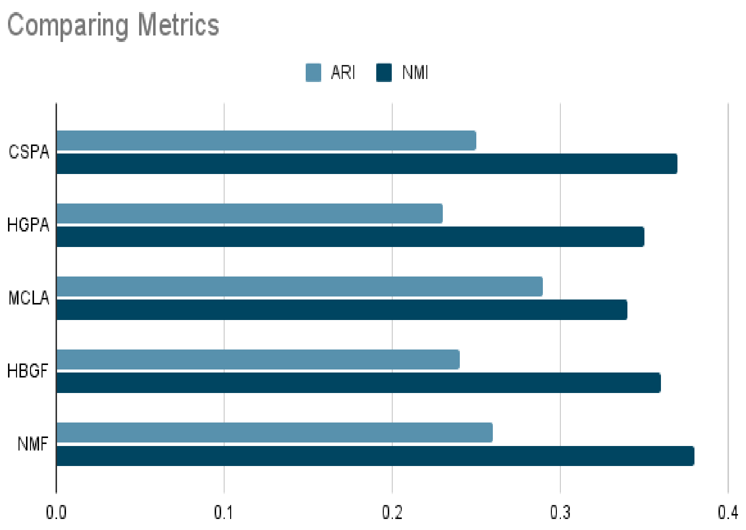


Figure 3.28: Number of Output Clusters(N)=12

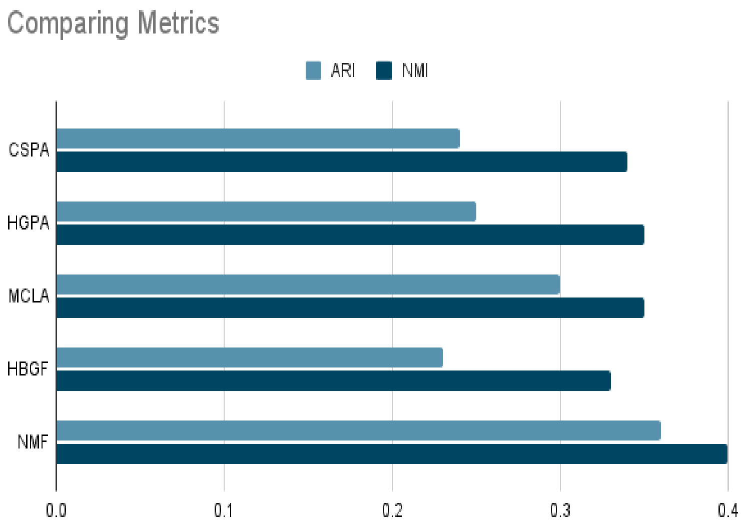


Figure 3.29: Number of Output Clusters(N)=10

clustering results obtained by each consensus algorithm is shown in Fig 3.33 and 3.34. Also all the clustering metric were compared for N=7, as shown in Fig. 3.35.

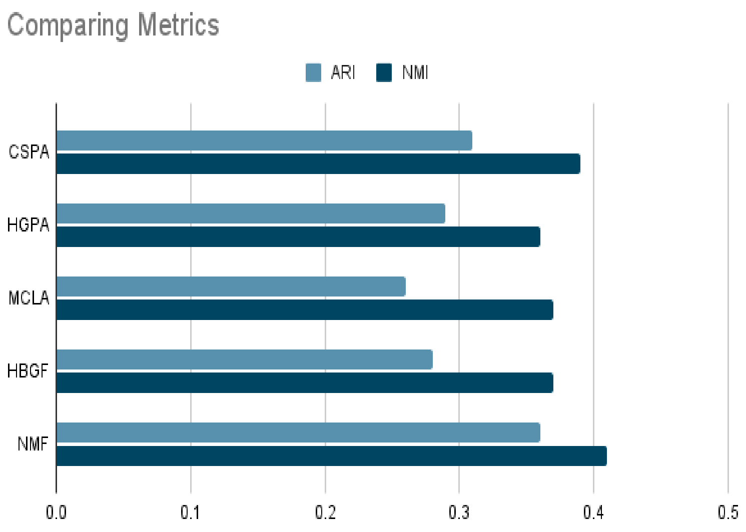


Figure 3.30: Number of Output Clusters(N)=8

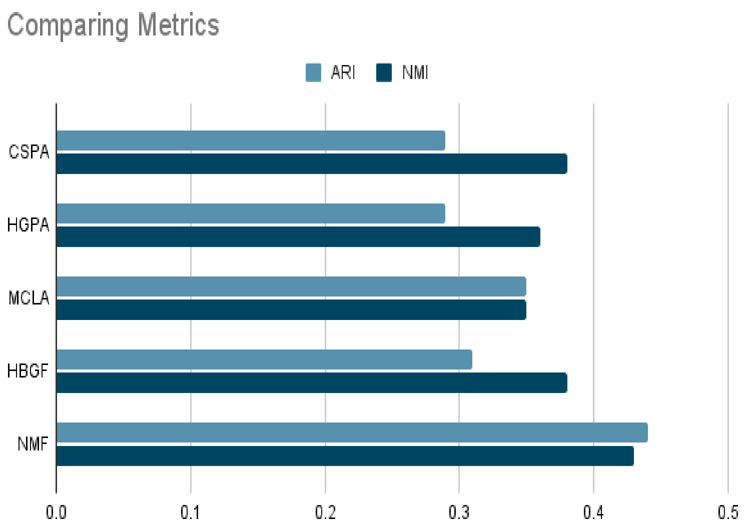


Figure 3.31: Number of Output Clusters(N)=7

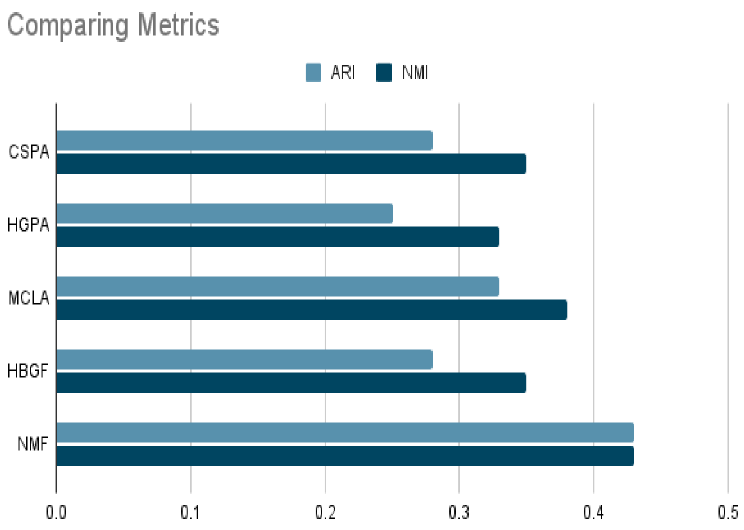


Figure 3.32: Number of Output Clusters(N)=6

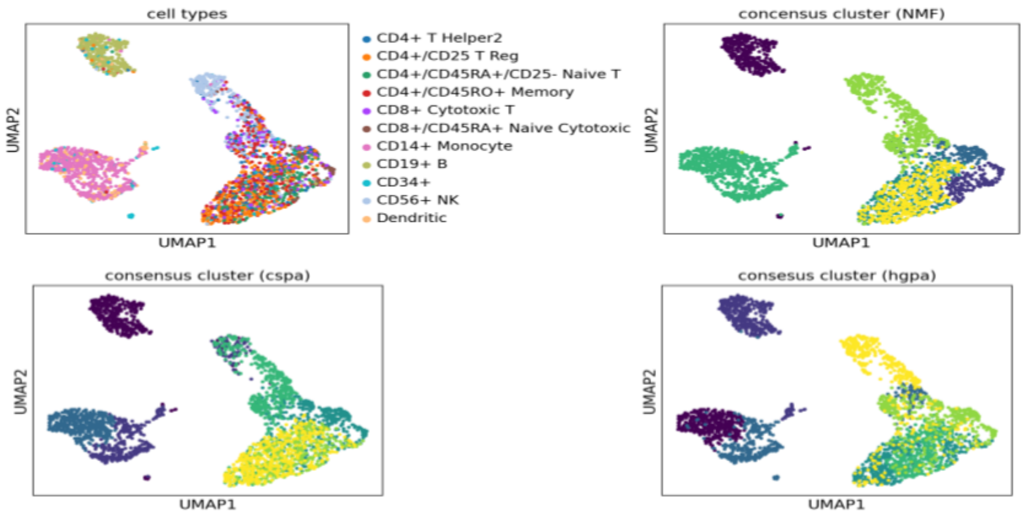
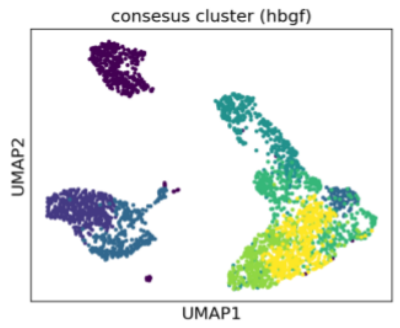


Figure 3.33: Consensus clustering results for nmf, cspa, hgpa



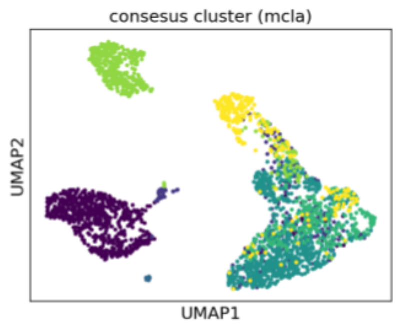


Figure 3.34: Clustering using mcla,hbgf

3.7 Conclusion

From the results of applying the consensus algorithm to Pancreas data, we can infer that the meta clustering algorithm (mcla) outperformed all other consensus algorithms as well as performed better than all the individual algorithms. NMF based consensus algorithm performed the second-best, doing better than all other algorithms as well as individual clustering results.

From the results of applying the consensus algorithm to PBMC data, we can infer that the Non-negative matrix factorization (NMF) based consensus algorithm outperformed all other consensus algorithms as well as performed better than all the individual algorithms.

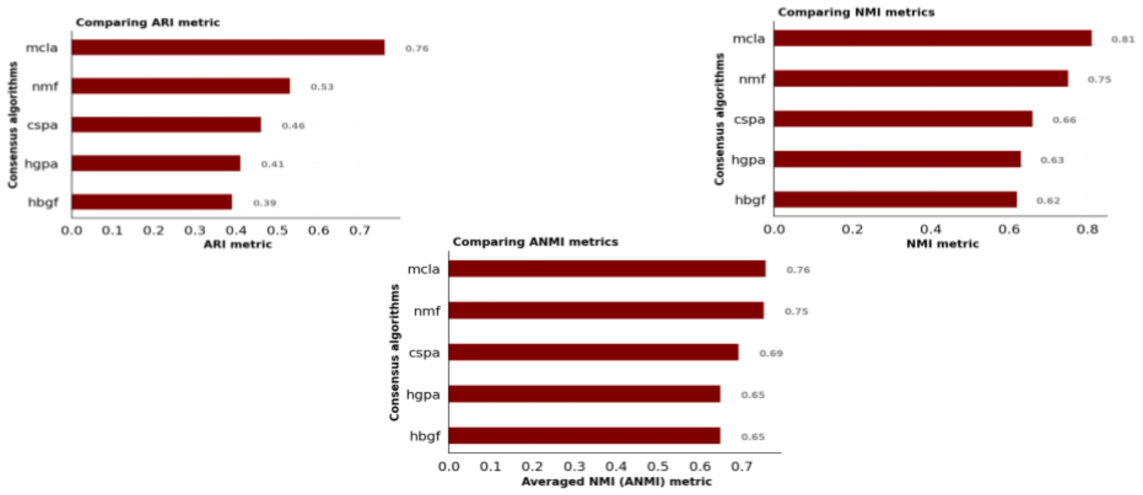


Figure 3.35: Comparison plots for clustering metrics

Chapter 4

Conclusion & Future Scope

4.1 Conclusion

The thesis achieves two goals- first, it provides Transcend, a webserver which hosts a number of pretrained transfer learning methods for single-cell analysis. This is an open-source community-scale webserver and we hope that it will be widely adopted as a single source of a point where everything related to the work of transfer learning in single-cell genomics can be found. This webserver would prove to be beneficial to all the researchers, bioinformaticians or non-bioinformaticians, computational biologists and students who are willing to do analysis on single-cell data and seek some fundamental questions of biology which can help in disease biomarkers, finding new cell types, rare cell types and understanding developmental studies etc.

Secondly, we implemented various consensus algorithms to come up with a single generalized cluster ensemble. We tried with different single-cell data sets and applied individual clustering methods to get individual clustering labels and then we used these individual labels as base labels for our consensus algorithms. We tried different consensus algorithms and found that for the pancreas dataset, the meta-clustering algorithm (mcla) outperformed every other algorithm and similarly for pbmc blood data, the non-negative matrix factorization (nmf) based consensus algorithm outperformed all other algorithms. So we were able to come up with single clustering results that performed better than individual clustering algorithms and the results are stable and can be reproduced.

4.2 Future Scope

For Transcend webserver, only a limited number of transfer learning methods have been implemented, so all other transfer learning methods which are available for single-cell analysis can be explored and their models can be pretrained and hosted on the webserver. In the current version of Transcend, only six to seven organs/tissues and two species are considered, so we can extend this to all the organs and important species so that the webserver can cover a large space of the single-cell data. In addition to this, a model can be trained on large scale cell atlases, which can be adapted for many smaller target tasks. For the second part of the thesis, only three individual clustering method results were used as base labels for the consensus clustering, so more than three methods can be explored to see the impact on final single consensus clustering. Moreover, these consensus clustering results can be validated on other publicly available single-cell datasets to gain more confidence in the already achieved results.

Bibliography

[1] 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.

[2] 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.

[3] Y. Zhao, H. Cai, Z. Zhang, J. Tang, and Y. Li, "Learning interpretable cellular and gene signature embeddings from single-cell transcriptomic data," *Nature communications*, vol. 12, no. 1, pp. 1–15, 2021.

[4] S. Ruder, "Neural transfer learning for natural language processing," Ph.D. dissertation, National University of Ireland, Galway, 2019.

[5] X. Lin, H. Liu, Z. Wei, S. B. Roy, and N. Gao, "An active learning approach for clustering single-cell rna-seq data," *Laboratory Investigation*, vol. 102, no. 3, pp. 227–235, 2022.

[6] M. Lotfollahi, M. Naghipourfar, M. D. Luecken, M. Khajavi, M. Büttner, M. Wagenstetter, Ž. Avsec, A. Gayoso, N. Yosef, M. Interlandi *et al.*, "Mapping single-cell data to reference atlases by transfer learning," *Nature Biotechnology*, vol. 40, no. 1, pp. 121–130, 2022.

[7] 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.* Ieee, 2009, pp. 248–255.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[9] M. Lotfollahi, M. Naghipourfar, F. J. Theis, and F. A. Wolf, "Conditional out-of-sample generation for unpaired data using trvae," *arXiv preprint arXiv:1910.01791*, 2019.

[10] A. Gayoso, Z. Steier, R. Lopez, J. Regier, K. L. Nazor, A. Streets, and N. Yosef, "Joint probabilistic modeling of single-cell multi-omic data with totalvi," *Nature methods*, vol. 18, no. 3, pp. 272–282, 2021.

[11] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[12] A. B. Dieng, F. J. Ruiz, and D. M. Blei, "Topic modeling in embedding spaces," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 439–453, 2020.

[13] B. Mieth, J. R. Hockley, N. Görnitz, M. M.-C. Vidovic, K.-R. Müller, A. Gutteridge, and D. Ziemek, "Using transfer learning from prior reference knowledge to improve the clustering of single-cell rna-seq data," *Scientific reports*, vol. 9, no. 1, pp. 1–14, 2019.

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

[15] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *Journal of machine learning research*, vol. 3, no. Dec, pp. 583–617, 2002.

[16] X. Z. Fern and C. E. Brodley, "Solving cluster ensemble problems by bipartite graph partitioning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 36.

[17] T. Li, C. Ding, and M. I. Jordan, "Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization," in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, 2007, pp. 577–582.