Improving Cell Classification Schemes in scRNAseq Data to Support COVID-19 Diagnostics

Team: Anusha Phadnis (CIS519), Amelia Schroeder (CIS519), Aoife O'Farrell (CIS519)
Project Mentor TA: Kelly

1) Abstract

The COVID-19 pandemic has severely impacted our medical care system. Sequencing of mRNA on a single cell level (scRNAseq) can provide insight into COVID-19 patient health - however, labeling of cell clusters is typically done manually in a time-consuming and error-prone process. Here, we train two models on single cell transcriptional data from 90 COVID-19 patients and 23 healthy controls across 624,325 cells on the basis of 4,209 highly variable genes - the first a modification of an automatic cell labeling classifier (CellO) and the second a novel gradient boosting model. Both models were evaluated on an external test dataset from a different publication of 33 COVID-19 patients and 13 healthy controls across 37,208 cells, and we found the CellO model to be a more accurate classifier (71.3% vs 21.4%) in predicting cell-type across 12 different levels. From here, a novel Adaboost model was created taking these predicted cell labels as input to predict whether samples came from COVID- patients or healthy controls, and a test accuracy of 89% was achieved. Overall, these improvements demonstrate that machine learning prediction of cell labels from transcriptional data can improve accuracy and speed of downstream data applications without sacrificing quality.

2) Introduction

Single cell transcriptomics data, although useful to scientists (see Appendix I for information), can be extremely difficult to parse. First, mRNA transcript reads must be filtered (to ensure high quality), normalized, and quantified by cell barcode. Unsupervised machine learning through principal component analysis (PCA) and unified manifold approximation and projection (UMAP) is used to cluster similar cells based on gene expression. These clusters are then labeled manually by the researcher, in a difficult process that is time consuming and prone to error. Once cells are clustered, they can be compared to identify differentially expressed genes and improve understanding of the patient's overall health. Although some algorithms exist to automatically label cells based on gene expression, they are prone to errors. In this paper, we will focus on CellO (cell ontology-based classification), a Python package that makes hierarchical predictions through cascaded logistic regression to classify cell types based on the Cell Ontology - a structured "vocabulary" of cell types based on several bioinformatics databases. However, there exist several similar algorithms, including Azimuth and scPred.

In this system, we will input single cell RNA sequencing data from both COVID-19 patients or healthy controls. The labels of each cell type will be predicted using the CellO model, which will feed into our Adaboost classifier to predict whether the sample came from an infected patient or a healthy control. As such, the intermediate output of the system will be a predicted cell label for each individual cell (454,571 in the training set, 169,754 in the validation set, and 37,208 in the test dataset), and the final analysis output will be a binary classification prediction of COVID status by patient (79 in the training set, 34 in the validation set, and 46 in the test set).

3) Background

Cell type prediction software is an incredibly useful tool for bioinformaticians. However, many of the tools available currently are difficult to use. For instance, the classifier CellAssign requires input of defined cell type markers, making the use of the algorithm little different from manual labeling [Zhang, 2019]. scPred, a commonly used Python application, requires the input of a prelabeled training dataset, which can potentially bias results [Alquicira-Hernandez, 2019]. Finally, the majority of classifiers fail to take into account cell hierarchies - the relationship between progenitor cells and their more differentiated progeny (an example of hematopoietic cell hierarchies is pictured to the right). The algorithm CellO avoids these common pitfalls by basing its predictions on the Cell Ontology, thus avoiding the need for prelabeled training sets and allowing for hierarchical predictions. This can be especially beneficial in the prediction of rare cell types and in minimizing errors, as unknown cell types can be predicted as their earlier progenitor (using an example from the image above, classifying a T lymphocyte as a common lymphoid progenitor, an earlier progenitor in the same lineage). However, this can still lead to some challenges in the use of this model, as it may define cells using labels so high up in the hierarchy as to be functionally meaningless. We aim to improve this function as part of our pipeline.

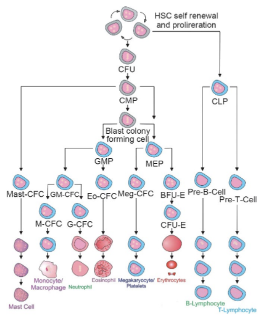


Figure 1: Hematopoietic cell hierarchy relationships. *(Image from Pulmonary Vascular Research Institute, 2015)*

After an extensive literature search, we could not find any ML model using scRNAseq information to classify patient disease status that did not involve smoothing the scRNA-seq expression data into bulk level expression for each patient. As such, we create such a classifier with the aim of improving understanding of key physiological differences in patients with disease. The LazyPredict Python package allows for the rapid creation of several models and selection of high-performing options for further analysis [Pandala, 2020]. As this is a novel analysis technique, we will utilize this package to determine the best model to reach this goal.

Most Relevant Prior Work, and Location of Project Datasets:

A. Stephenson et. al. “The cellular immune response to COVID-19 deciphered by single cell multi-omics across three UK centers,” *MedRxiv*. (2021). This group’s single cell RNAseq dataset from a cohort of 130 patients was used as our training set. The paper can be found here: https://www.medrxiv.org/content/10.1101/2021.01.13.21249725v1.full.pdf
B. Liu, C., et al. “Time-resolved systems immunology reveals a late juncture linked to fatal COVID-19”. This paper contains scRNA-seq data for 46 COVID-19 and healthy patients across 37,208 annotated cells which will serve as the external test data set for evaluations of our methods. Their paper can be found here: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7874909/
C. Bernstein et. al. “CellO: comprehensive and hierarchical cell type classification of human cells with the Cell Ontology” *iScience* (2020). This paper describes the algorithm we have modified as part of this project. Their paper can be found here: https://www.sciencedirect.com/science/article/pii/S258900422031110X

4) Summary of Our Contributions

1. **Contribution(s) in Code:** We have modified the existing Python package CellO to improve cell type classification from scRNAseq data by adding weights to genes of interest and by tuning hyperparameters. CellO originally weights all genes as equally important to predicting cell type - based on immunology and sequencing literature, we have identified genes of interest that play a more important role in cell type classification and upweighted their importance as predictors. We have also written the code to develop a gradient boosting algorithm to predict cell labels from scRNAseq data.

2. **Contribution(s) in Analysis:** We have developed a novel two-step ML analysis program, in which an algorithm is first used to predict cell labels from highly variable gene expression, then an Adaboost algorithm is employed to predict patient diagnosis. Single cell RNA sequencing provides extremely high resolution information of a patient's overall health - as such, the processing of this data and classification using a ML algorithm is a novel analysis technique that may benefit physicians, scientists, and patients.

5) Detailed Description of Contributions

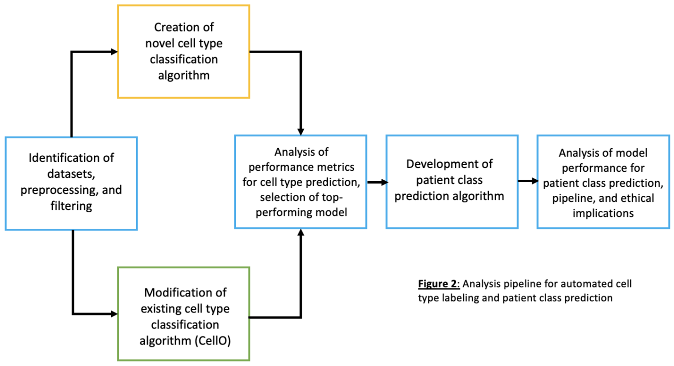

Creation of novel cell type classification algorithm
Identification of datasets, preprocessing, and filtering
Analysis of performance metrics for cell type prediction, selection of top-performing model
Development of patient class prediction algorithm
Analysis of model performance for patient class prediction, pipeline, and ethical implications
Modification of existing cell type classification algorithm (CellO)

Figure 2: Analysis pipeline for automated cell type labeling and patient class prediction

6.1 Methods

Analysis Pipeline: The COVID-19 pandemic is an ongoing health crisis. To develop a ML algorithm capable of processing sequencing data from COVID-19 patients and provide meaningful results for physicians and scientists, the following analysis pipeline was developed (see image, left).

Data Preprocessing, Cleaning and Filtering: To begin, the training and test datasets were cleaned for use in later steps. Unnecessary information for our overall aim (including patient age, disease severity, ethnicity, etc) was removed from the dataframe. Cell labels used by the researchers in their initial clustering were examined and syntax was altered to match expected outcomes from CellO prediction software (for example, the label "DCs" was replaced with "myeloid dendritic cell, human." See Appendix I for more information). From the training dataset, two control populations (those who were hospitalized for non-COVID disease, and those injected with IV-LPS to mimic acute inflammatory response) were removed to simplify the model. Next, the two datasets were analyzed using the scanpy preprocessing (scanpy.pp) package in Python to identify the top 5,000 highly variable genes (HVGs - a measure of cell-cell variation within the pool) present in the training dataset [Wolf, 2021]. These HVGs were compared to the list of genes present in the test dataset, and the intersecting training HVGs and test gene set were used. The final datasets contained information regarding ID of patient, individual cell barcode, patient status (i.e. Healthy vs COVID-19), and cell-level gene expression matrices for the identified top 4,209 intersecting HVGs.

Evaluating Performance Metrics of Original CellO Algorithm: The CellO algorithm, as described above, provides cell type classification based on single cell RNA sequencing data. The training dataset was run through the CellO pipeline, during which gene expression information is normalized, principal component analysis (PCA) is performed, and a nearest neighbor graph is created. From here, cascaded logistic regression is used to classify the cell's type, as reported in the output "most specific cell type." In this baseline analysis, the hyperparameter values were selected based on recommendations by CellO as recorded in their documentation [Bernstein, 2021]. Specifically, n_comps for PCA was set to 50, n_neighbors set to 15, and the resolution of clustering set to 2.0.

Review of Common Misclassifications Made By CellO: After a detailed analysis of classification errors made by CellO, several common themes occurred. First, the classifier appeared to have difficulty distinguishing between CD4+ and CD8+ T cells, which share many common genes but perform vastly different functions. Next, dendritic cells (DCs) were often misclassified as mononuclear cells. Although mononuclear cells is a broader umbrella classification that does include DCs, we cannot accept this broader term when the training dataset distinguishes these groups (see Appendix I). Lastly, anucleate cells (cells without nuclei, such as red blood cells) were often misclassified as platelets (which are also anucleated). Recent evidence suggests that although they are anucleate, platelets can play an essential role in the immune response, and thus should be classified as their own category [Ali, 2016]. Taking all this analysis into account, we decided to upweight the expression of T cell differentiation markers (CD4, CD8A, CD8B), DC markers (CD11B, CD11C, CD103), and platelet markers (PECAM-1) in our algorithm to improve accuracy of cell classifications.

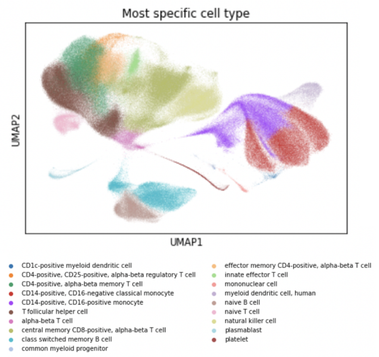
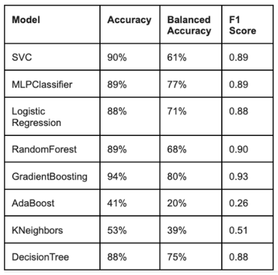


Figure 3: Relabeled cell types for test dataset using modified CellO algorithm

Optimization of CellO Algorithm: To optimize performance of this algorithm, weighting of these identified genes was accomplished by multiplying their expression by a specified weight. Next, the training dataset was split by patient into a training set (which contained cells from 61 COVID patients and 18 healthy controls) and validation set (which contained cells from 29 COVID patients and 5 healthy controls). The data set was broken into a 70%-30% training-validation data set split. The CellO model was then trained on various combinations of hyperparameters for the gene weights, n_comps for PCA, and n_neighbors for cell type prediction, and tuned based on performance on the validation set.

Development of a Cell Label Classification Algorithm Using Gradient Boosting: To determine whether we could further improve on CellO's accuracy with a different model, we decided to build a different classifier. Multiple common classification algorithms were run on a subset of the training dataset using LazyPredict and were tested using another smaller subset of the training dataset as validation data. Out of those algorithms, a Gradient Boosting algorithm gave the best results. Taking those results as a reference, a Gradient boosting algorithm using the XGBoost library was developed. To preprocess the data: normalization, log transformation and Principal Component Analysis (taking the number of components as 200) were used.



Model	Accuracy	Balanced Accuracy	F1 Score
SVC	90%	61%	0.89
MLPClassifier	89%	77%	0.89
Logistic Regression	88%	71%	0.88
RandomForest	89%	68%	0.90
GradientBoosting	94%	80%	0.93
AdaBoost	41%	20%	0.26
KNeighbors	53%	39%	0.51
DecisionTree	88%	75%	0.88

Figure 4: Performance metrics of various classifiers for cell type annotation

Development of Patient Status Prediction Algorithm: To utilize improvements to automated cell label prediction implemented in this pipeline, the amount of cells given each label by patient was calculated and analyzed as a percent of total cells from the patient. Because the prediction of patient status from scRNAseq information is a novel analysis technique, we began by considering different choices of algorithm. The LazyPredict package was used to compare various models, and an Adaboost model was selected as the most suitable for this analysis aim. The model was trained on our training dataset and hyperparameters based on performance on the validation set - specifically, we maximized the F1 score for this prediction, as we are most concerned with identifying positive cases and avoiding false negatives, as opposed to preventing false positives, given than we are predicting disease diagnosis. The results of this classifier on the validation set are detailed below.

Table 1: *Common Errors in CellO Algorithm Predictions*

Of cells misclassified as CD4+ T cells, how many are CD8+ T cells?	77.33%
Of cells misclassified as CD8+ T cells, how many are CD4+ T cells?	16.40%
Of cells misclassified as dendritic cells, how many are mononuclear cells?	87.89%
Of cells misclassified as mononuclear cells, how many are dendritic cells?	87.21%
Of cells misclassified as anucleate cells, how many are platelets?	98.38%
Of cells misclassified as platelets, how many are anucleate cells?	84.62%

6.2 Results

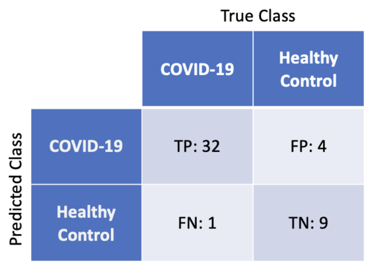
CellO Algorithm Commonly Misclassifies T Cell and Mononuclear Subsets: The original CellO algorithm, after secondary data relabeling (see Appendix I for more information), correctly predicted 71.9% of cell labels within the training dataset and 64.8% in the test dataset. However, common errors in classification were uncovered (see Table 1). As such, the decision was made to weight the expression of marker genes for these cell types, as described in the methods section above, to improve delineation of these cell type clusters.

Optimization of Hyperparameters and Weighted Gene Expression Values Improves CellO Algorithm Accuracy: After evaluating the impact of various hyperparameters on validation accuracy, the following were selected: gene weights was set to 10, n_comps set to 55, and n_neighbors set to 20. These choices of hyperparameter gave the model a validation accuracy of 88.3%. Once the algorithm was modified, the test dataset was analyzed, and accuracy of cell type prediction was found to be 71.3%. Given that the test dataset contained 12 possible cell types, a random classifier would expect accuracy of roughly 8% - as such, we feel confident reporting these results as a successful cell classification algorithm.

Table 2: *Hyperparameter Tuning and Associated Validation Accuracy*

Gene Weights	n_comps	n_neighbors	Validation Accuracy
1	50	15	78.36%
5	65	20	71.43%
3	70	15	83.85%
10	55	20	88.31%

XGBoost Algorithm Fell Victim to Overfitting and Failed to Transfer High Validation Accuracy to Test Dataset: Although initially promising with a validation accuracy of 93%, our XGBoost model had test accuracy of 21%. We found that most wrong classifications belonged to the class ‘helper T cell’, which was a major part of the training dataset. Since the training dataset had an imbalance of this cell type, this led to an overfitted model which performed well on the training dataset, but not on the test set (as it did not have the same imbalance).



Predicted Class \ True Class	COVID-19	Healthy Control
COVID-19	TP: 32	FP: 4
Healthy Control	FN: 1	TN: 9

Figure 5: Confusion matrix for Adaboost algorithm performance on test dataset

Adaboost Algorithm Accurately Predicts Patient Status Based on Predicted Cell Type Label from Modified CellO Algorithm: Initially, a support vector machine (SVM) model was presumed to be the best choice at separating out COVID-19 patient samples versus healthy controls, as they can be given a “soft” decision boundary to be robust to noise and outliers (which can be expected in patient data). However, due to a higher presence of COVID samples than healthy, this classifier chose only COVID samples as support vectors, and classified all samples as COVID positive. Although this prediction gave an overall accuracy of 85% for our validation set, we decided to select a different classifier that could more accurately distinguish between the groups. After running LazyPredict, an Adaboost classifier was selected as the optimal choice, as it gave the highest accuracy on the validation set without predicting all samples as COVID-positive. Hyperparameters for this Adaboost classifier were tuned to maximize the model's F1 score, as we are more concerned with minimizing the number of COVID patients misclassified as healthy than the opposite. From a healthcare perspective, the occasional false positive is an appropriate price to pay to ensure that all true positive cases are accurately detected. The final tuned Adaboost model yielded a weighted F1 score of 89% on our test dataset - accurately predicting all but 5 patient samples.

7) Compute/Other Resources Used

A server from the Penn Biostatistics department with additional memory storage was used to import and preprocess the original large training data set. To run our gradient boosting algorithm, Amazon Web Services (AWS) credits were used. Our datasets were in AnnData format, from which the gene expression can be extracted as a matrix of matrices. When using the dense version of the matrices, computation became exponentially expensive, while using the sparse form of the matrix proved much better. Unfortunately, a lot of traditional libraries do not accept sparse matrices (for one dimension of one data point) as input, and hence we either needed to use the dense matrix and increase computational complexity, or find another way to transform the data during preprocessing to represent it in an acceptable form. XGBoost proved to be a good library to use, since it accepted the sparse matrix data and hence saved the computational cost.

8) Conclusions

In this project, we aimed to improve analysis of single cell transcriptomics data as part of larger patient sample analysis to support disease research. Through modifications of an existing algorithm, we have improved accuracy of automated cell labeling on a single cell level based on gene expression data. These labeled cells can be used in bulk to analyze a patient's immune health and predict their disease status, as demonstrated through our Adaboost predictor of COVID infection status. Lastly, we learned that with such data, overfitting is a huge risk for cell classification and that the ML models should always be tested with datasets from different sources to determine their validity accurately.

COVID-19 continues to have a major impact on the world. Although our algorithm shows promise at both predicting immune cell type information from single cell transcriptional data and at determining patient sample origin, these results must be interpreted with caution. As with any algorithm applied to medical research, there is great possibility for bias and prediction failure that cannot be ignored. We trained our model on samples from patients in the United Kingdom early in the pandemic. As such, the algorithm may be biased towards both earlier variants of COVID-19 and towards a subset of patients. As such, these algorithms may not be generalizable to current COVID-19 samples or to the global population.

In the future, to improve the model's generalizability, larger training datasets of scRNAseq data from COVID-19 patients and healthy controls from several countries and overall health baselines must be collected and analyzed. In addition, sequencing data from people with other diseases (such as influenza, cancer, or infection) can be analyzed to determine whether this classifier is still able to isolate COVID-19 patients from other disease states, as well as from healthy controls.

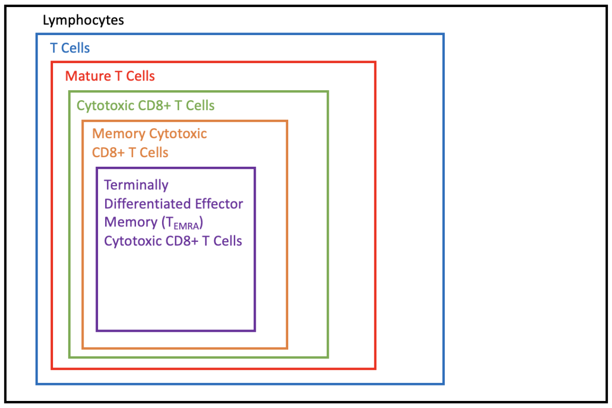
References:

1. Zhang, Allen et. al. “Probabilistic cell-type assignment of single cell RNA-seq for tumor microenvironment profiling” *Nat Methods* https://www.nature.com/articles/s41592-019-0529-1 (2019)
2. Alquicira-Hernandez, Jose et. al. “scPred: Accurate supervised method for cell-type classification from single cell RNA-seq data” *Genome Biology* https://genomebiology.biomedcentral.com/articles/10.1186/s13059-019-1862-5 (2019)
3. Pandala, Shankar Rao. “Lazy Predict” https://lazypredict.readthedocs.io/en/latest/readme.html (updated 2020).
4. Wolf, Alex et. al. “Scanpy – Single-Cell Analysis in Python” https://scanpy.readthedocs.io/en/stable/api.html#module-scanpy.pp (updated 2021)
5. Bernstein, Matthew et. al. “CellO: Cell Ontology-Based Classification” https://github.com/deweylab/CellO (updated 2021)
6. Ali, Ramadan et. al. “Platelets: essential components of the immune system” *Curr Trends Immunology* https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5096834/ (2016)

Supplemental I: The Relevance of scRNAseq Analysis, and Relabelling of Cell Types to Improve Physiological Accuracy

Single cell transcriptomics provides an incredibly detailed look into a cell's functional state. By quantifying the amount and type of mRNA transcripts present in a cell, we can determine which proteins the cell is creating, and the corresponding cell function. This technology is of particular interest in immunology, where protein expression readings taken from peripheral blood mononuclear cells (PBMCs) can determine whether a patient's immune cells are successfully fighting off infection. Quantifying the types and amounts of immune cells present in a patient can provide insight into their overall health, disease progression, and even whether or not they will respond to a particular treatment.

In the field of immunology, individual immune cells have vastly different functions, mediated in large part by their surface receptors and the genes expressed within each cell. Small differences in the expression of particular genes can define the difference between a cytotoxic T lymphocyte (capable of killing infected or damaged cells) and a regulatory T lymphocyte (capable of dampening an immune response). As a result, the "label" of a cell, or the name given to it that describes its function to researchers - is incredibly important. However, despite its importance, this labeling system is not black and white. A cell can fall under several different labels depending on the granularity of the classification - all labels are correct, but some are more detailed than others (just like how a square can also be classified as a parallelogram or a polygon). An example is demonstrated below.


Lymphocytes
T Cells
Mature T Cells
Cytotoxic CD8+ T Cells
Memory Cytotoxic CD8+ T Cells
Terminally Differentiated Effector Memory (T_{EMRA}) Cytotoxic CD8+ T Cells

To accurately train our model, we needed to match the granularity of our training cell labels to those produced by the CellO algorithm. As such, some cell types needed to be relabelled to ensure our accuracy metrics were not impacted by this issue. In total, these cells were relabelled two times. The first was to fix issues in nomenclature (for instance, the difference between "NK Cells" and "natural killer cells") and in discrepancies between the training and test dataset. Our test dataset contained some cell types that were not present in the training dataset (for instance, the test set contained granulocytes, while training did not), so these cells were relabelled as "unknown" to ensure the algorithm did not falsely place these test cells into a different category. The second relabelling was completed after the CellO algorithm was run, and corrected issues in granularity (for instance, using the example above, counting a prediction of "T_{EMRA}" as correct when the original label was "memory cytotoxic CD8+ T cell").

Predictions were accepted when they were a more granular version of the same cell type, but not if they were a less granular version (for instance, we would not accept “lymphocyte” as correct when the original label was “cytotoxic CD8+ T cell”). The details of this relabelling scheme are described below, and the code can be found in our GitHub.

The following granularity changes were made to cell labels:

Cell Type	Accepted Labels	Reasoning
B Cell	Naive B cell Class-switched memory B cell	Both naive and memory B cells are types of B cells - training dataset used broader umbrella term, so we accepted either as correct
Helper T Cell	Helper T Cell T Follicular Helper Cell	Follicular helper T cells are a specific subset of helper T cells that perform similar functions in assisting immune responses. As such, either label was accepted.
T Follicular Helper Cell	T Follicular Helper Cell Helper T Cell	Follicular helper T cells are a specific subset of helper T cells that perform similar functions in assisting immune responses. As such, either label was accepted.
Naive CD4+ T Cell	Helper T Cell T Follicular Helper Cell	All CD4+ T cells were grouped together in the training dataset, and there was no naive T label. As such, the labels of any CD4+ T cell were accepted.
CD4+ Memory T Cell	Helper T Cell T Follicular Helper Cell	All CD4+ T cells were grouped together in the training dataset, and there was no memory T label. As such, the labels of any CD4+ T cell were accepted.