scientific reports

Check for updates

OPEN

Predicting cell types with supervised contrastive learning on cells and their types

Yusri Dwi Heryanto[1,2], Yao-zhong Zhang[1,2]✉ & Seiya Imoto[1]✉

Single-cell RNA-sequencing (scRNA-seq) is a powerful technique that provides high-resolution expression profiling of individual cells. It significantly advances our understanding of cellular diversity and function. Despite its potential, the analysis of scRNA-seq data poses considerable challenges related to multicollinearity, data imbalance, and batch effect. One of the pivotal tasks in single-cell data analysis is cell type annotation, which classifies cells into discrete types based on their gene expression profiles. In this work, we propose a novel modeling formalism for cell type annotation with a supervised contrastive learning method, named SCLSC (Supervised Contrastive Learning for Single Cell). Different from the previous usage of contrastive learning in single cell data analysis, we employed the contrastive learning for instance-type pairs instead of instance-instance pairs. More specifically, in the cell type annotation task, the contrastive learning is applied to learn cell and cell type representation that render cells of the same type to be clustered in the new embedding space. Through this approach, the knowledge derived from annotated cells is transferred to the feature representation for scRNA-seq data. The whole training process becomes more efficient when conducting contrastive learning for cell and their types. Our experiment results demonstrate that the proposed SCLSC method consistently achieves superior accuracy in predicting cell types compared to five state-of-the-art methods. SCLSC also performs well in identifying cell types in different batch groups. The simplicity of our method allows for scalability, making it suitable for analyzing datasets with a large number of cells. In a real-world application of SCLSC to monitor the dynamics of immune cell subpopulations over time, SCLSC demonstrates a capability to discriminate cell subtypes of CD19+ B cells that were not present in the training dataset.

In recent years, single-cell RNA sequencing (scRNA-seq) has made remarkable advancements, enabling the analysis of gene expression profiles at the individual cell level. This technology has been instrumental in identifying rare cell populations, defining cell types, and uncovering novel cell states[1,2]. Experts in the field have collected numerous datasets using scRNA-seq, including large-scale initiatives like the Human Cell Atlas[3] and Tabula Muris Atlas[4].

The initial step in analyzing single-cell data typically involves annotating cells based on their known and novel cell types. One commonly used strategy, known as "cluster-then-annotate," involves grouping cells into clusters based on the similarity of their gene expression profiles and manually characterizing them using previously identified cell type markers[4,5]. However, this manual strategy presents several challenges. Firstly, the process of cell type annotation is labor-intensive, requiring extensive literature review of genes specific to each cluster[6]. Secondly, any changes made to the analysis, such as incorporating additional data or adjusting parameters, require the manual reevaluation of all previous annotations. Thirdly, the incompleteness of the current knowledge and researcher subjectivity may contributes to cells mislabeling[7,8]. Lastly, transferring annotations between independent datasets generated by different research groups studying related tissues is challenging, often resulting in redundant efforts.

Recognizing the limitations, a number of supervised approaches were proposed to utilize existing reference datasets as training dataset and directly annotate cells on query datasets without the clustering step. Mainstream numerical methods in this category include Seurat[9] and SingleR[10]. Seurat identifies anchor across batches using mutual nearest neighbour and then use supervised PCA on mutual neighbours to transfer reference annotations[9]. Meanwhile, SingleR utilize the Spearman correlation-based scoring to perform annotation transfer task[10]. However, both of these methods have poor scalability that led to longer runtimes and higher memory usage[11]. Kang

[1]The Institute of Medical science, The University of Tokyo, Tokyo 108-8639, Japan. [2]These authors contributed equally: Yusri Dwi Heryanto and Yao-zhong Zhang. ✉email: yaozhong@ims.u-tokyo.ac.jp; imoto@hgc.jp

et al. introduced a novel algorithm, Symphony, utilizing a linear mixture model for constructing a large-scale integrated reference dataset and and fast reference-to-query label transfer called Symphony[12]. However, Symphony is a relatively new method, and its usage and testing have not been as extensive as those of Seurat and SingleR.

Deep learning methods provide a promising solution for label transfer, with a common strategy involving neural network training for representation learning. Representation learning refers to the automated extraction of meaningful features from raw data, aiming to construct a more concise and informative representation that captures underlying patterns and structures. Then, using this new representation of data, the algorithm performs supervised learning such as K-nearest neighbors (KNN) to transfer annotations from the reference dataset to the query dataset. In the realm of representation learning models, scANVI[13] and Concerto[14] are among the state-of-the-art approaches. The scANVI model employs variational inference deep generative model to learn a compact representation of gene expression patterns in single-cell RNA sequencing (scRNA-seq) data. It integrates this learned representation with a reference dataset and subsequently transfers annotations from the reference to the query dataset through an approximate Bayesian inference procedure. On the other hand, Concerto utilizes a contrastive learning approach to learn a low-dimensional embedding. It leverages KNN in this embedding space to predict annotations for the query dataset. Both methods have demonstrated remarkable performance in label transfer tasks, surpassing other existing methods[11,13,14].

In this study, we introduce a novel framework named Supervised Contrastive Learning for Single Cell (SCLSC) for single-cell type annotation. SCLSC method consists of two steps. First, SCLSC leverages supervised contrastive learning to learn a better data representation that captures both class discrimination and underlying data structure. This representation learning leverages label information from the training data to guide the model explicitly in discerning the similarity or dissimilarity between samples during the learning process. We devised a supervised contrastive loss utilizing cell type representations, guiding the model to position a sample close to its respective representative cell in the new embedding space while maintaining distance from representative cells of different labels. Second, the annotation is transferred from the reference dataset to the test dataset based on the learned representations. Following the training phase, where the model acquires representations, the KNN classifier is employed to identify the most similar instances in the reference dataset for each instance in the test dataset.

Through a comprehensive evaluation using both real and simulated datasets, we demonstrate that the learned representations from SCLSC offer several advantages. First, they enable improved separation of cell types, effectively handling variations and noise present in the data. Second, using cell type representation simplify the contrastive learning process because the number of cell type is inherently less than the number of cells. It make SCLSC a fast and straightforward framework that scales well, even when dealing with large datasets. These characteristics make SCLSC particularly suitable for practical applications where computational efficiency is essential. Additionally, the representations capture relevant information for the label transfer task, leading to enhanced performance compared to existing state-of-the-art methods. Overall, our findings highlight the effectiveness of the supervised contrastive learning approach employed in SCLSC, showcasing its ability to achieve superior performance in single-cell label transfer tasks while maintaining simplicity, scalability, and efficiency.

Results

Overview of the SCLSC pipeline

We first introduced the complete pipeline of SCLSC. The SCLSC pipeline is divided into two main components: (1) embedding learning for cell and cell type and (2) cell annotation, shown in Fig. 1. For representing scRNA-seq data, in previous work, highly variable genes across different cell types are used as the cell profiles. However, straightforward use for cell-type annotation is less comprehensive and consistent due to data noise and batch effect. With this in mind, we design a method to map the raw gene profiles of each cell into a new embedding space. To learn embeddings of cell and cell types, we used an MLP (Multi-Layer Perceptron) encoder to translate a raw cell profile into a new embedding space that factors in its cell type annotation. Supervised contrastive learning is used to train the MLP encoder, allowing us to acquire such a new representation. To represent cell types in the same embedding space as single cells, we use the arithmetic mean of gene profile vectors from all cells annotated with the cell type as an approximation. The model parameters of the MLP encoder are shared between both cell and cell types. We optimized the supervised contrastive loss between the cell samples and the cell type representative for updating the MLP encoder. Through the supervised contrastive learning process, cells of the same type tend to become more clustered, while cells from different cell types are increasingly separate. As a result, the cell type annotation is transferred through the embedding mapping for other scRNA-seq profiles. Once the MLP encoder is learned, a KNN is then trained within the new embedding space. Subsequently, a new cell can be annotated based on the KNN within this new embedding space, as depicted in Fig. 1b.

SCLSC achieved state-of-the-art label transfer task performance

Here, we evaluate the performance of SCLSC for label transfer task. The label transfer task refers to the process of transferring known labels or annotations from one dataset to another. Our approach involves several steps: (1) calculating query embeddings using pretrained model weights, (2) locate query cells near their most similar reference cells, and (3) use a KNN classifier (with a default value of k = 10) to transfer reference annotations to the query cells.

To evaluate the effectiveness of SCLSC, we compare its performance against other methods, including Seurat based on mutual nearest neighbors, SingleR based on correlation, scANVI based on variational inference probabilistic model, Symphony based on linear mixture model, and Concerto based on contrastive learning. Two experiments were designed for evaluation: the random split experiment and the split by batch experiment. In the random split experiments, we divided the dataset into training, validation, and test datasets using stratified

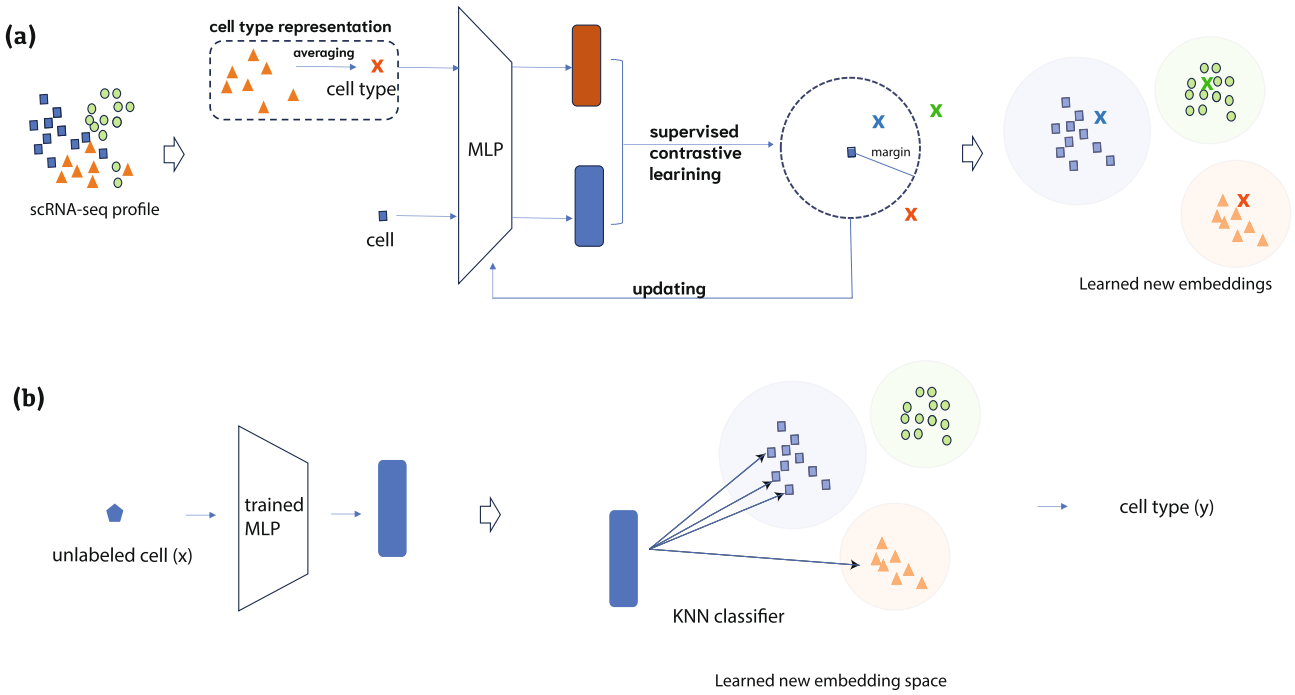

Figure 1. Overall view of SCLSC pipeline. The SCLSC pipeline can be divided into two phases: embedding learning and cell type annotation. In the first stage, as shown in (**a**), supervised contrastive learning is applied to learn new embeddings that capture the cell and cell type relationship derived from supervised data. For cell type representation, we averaged cell profile vectors in the same cell type as an approximated cell type profiling. In the second stage, as shown in (**b**), a candidate cell profile is mapped to its new embedding space based on the learned encoder in the first stage. Then, KNN is applied in the new embedding space to assign the cell type annotation for the cell.

random splits. On the other hand, in the split by batch experiment, we divided the dataset based on the batches to assess the performance under batch-specific conditions.

Based on our experiment with random splits, it was consistently observed that SCLSC consistently achieves the highest accuracy and macro-averaged F1 score across most datasets, except for the lung and pancreas dataset where it comes in a close second place (Fig. 2). Particularly in the PBMC dataset, where it achieved 11% and 27% improvements of accuracy and macro-F1 score, respectively, compared to the second-ranked method. It is

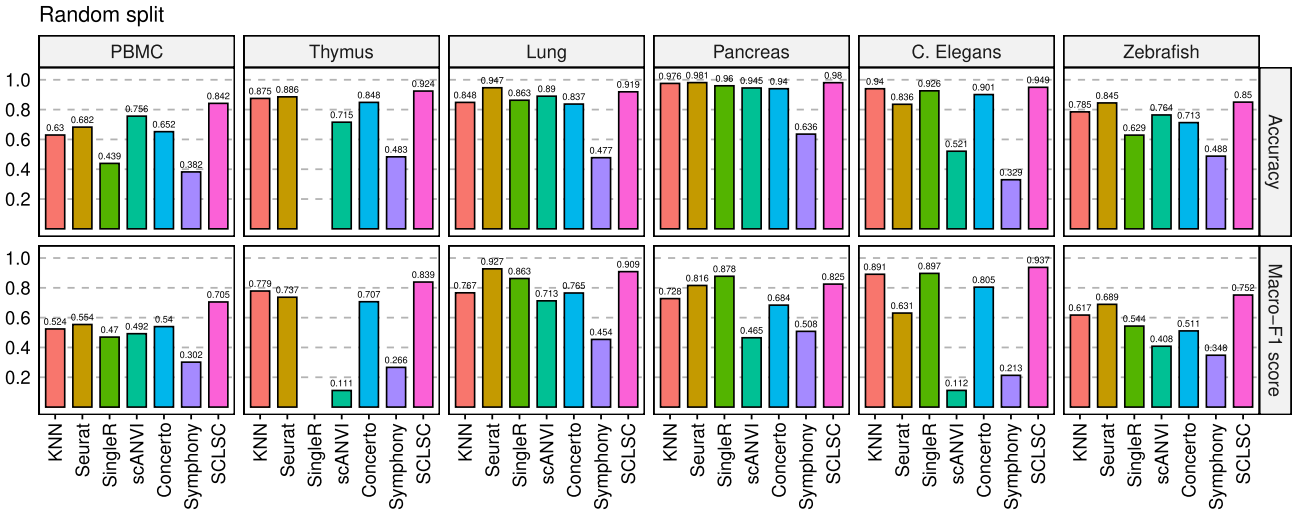

Figure 2. SCLSC achieves superior accuracy for label projection tasks. The performance of SCLSC in the label transfer task surpassed that of other state-of-the-art methods, as evidenced by its high accuracy and macro-averaged F1 score across six benchmark datasets. We excluded SingleR from the Thymus datasets assessment as it failed to generate any outputs even after running for over 6 h, leading us to terminate the process.

worth noting that each dataset has its own distinct characteristics. In the case of the PBMC dataset, challenges in cell type classification arose due to imbalanced distributions among cell types and strong correlations between them. Notably, the correlation-based approach SingleR and the linear mixture model-based method Symphony exhibited the poorest performance in this dataset, likely due to the presence of multicollinearity issues. Moving on to the CeNGEN C. Elegans dataset, it is characterized by a large number of cell types, with some cell types being rare, consisting of less than 100 cells. The neural network approach utilizing a variational inference probabilistic model, scANVI and linear mixture model framework, Symphony, demonstrated subpar performance in this dataset. In the case of the Thymus dataset, which contains a large number of samples, SingleR failed to produce any outputs even after running for more than 6 h, prompting us to terminate the process. In contrast, the consistently superior accuracy of SCLSC across all datasets demonstrates its efficacy in addressing challenges such as multicollinearity problems, imbalanced distribution of cell types, and large-scale samples. Additionally, SCLSC consistently achieves the highest macro-averaged F1 score across most of the datasets. The macro-averaged F1 score serves as a valuable performance metric, particularly in scenarios where class imbalance exists. It calculates the F1 score for each class individually and computes the average of these scores. This approach ensures that each class is given equal importance, irrespective of its prevalence in the dataset. Based on the high macro-averaged F1 score, SCLSC demonstrates superior ability in accurately separating rare cell types (Fig. 2, Supplemental Fig. S1). Furthermore, we observed that employing KNN on the representations acquired through SCLSC yielded higher accuracy compared to using KNN directly on raw data across all datasets. The potential reason for this could be that the newly acquired representations from SCLSC have lower dimensions and exhibit less noise, making it easier for the KNN classifier to categorize cell types within the learned representation as opposed to the raw data.

Cell type hierarchy is preserved in the new embedding space

In conducting supervised contrastive learning, we approximated the cell type representation by using the arithmetic mean of gene profile vectors from all cells annotated with a specific cell type. We utilized the PBMC dataset to investigate the cell type hierarchy for this method in both the raw and learned embedding spaces. Hierarchical agglomerative clustering with single-linkage was performed on 11 cell type vectors in each of these spaces. The dendogram of cell type representation from the 2000 highly variable genes (HVGs) raw data input (Fig. 3a) and the dendogram of the output cell type representation learned by SCLSC (Fig. 3b) is consistent with the immune cell lineage tree (Fig. 3c). The fact that the input HVGs dendrogram aligns with the immune cell differentiation hierarchy indicates that the selected input adequately captures the biological information within the dataset.

In the dendrogram shown in Fig. 3a, CD34+ cells are positioned at the root, from which two branches emerge. The left branch consists of myeloid cells, including CD14+ monocytes and dendritic cells, while the right branch consists of lymphoid lineage cells, namely CD56+ NK, CD19+ B, CD4+ T, and CD8+ T cells. This dendrogram structure aligns with the established immune cell lineage tree, demonstrating that the cell type representation learned by the SCLSC effectively captures and preserves the biological hierarchical structure present in the dataset.

SCLSC can learn the representation of the unseen cell in PBMC dataset while preserving the hierarchical structure

The objective of SCLSC is to transfer label annotations from a reference dataset to a query dataset. However, challenges arise when the query dataset contains cell types that were not present in the reference dataset, making it challenging for SCLSC to predict their annotations. Nevertheless, despite this limitation, SCLSC is still capable of learning valuable representations of these unseen cell types. to investigate the effect of the unseen cells, we conducted an experiment using the PBMC dataset, dividing it randomly into training, validation, and test subsets. To simulate the presence of unseen cell types, we removed CD19+ B cells from the training and validation datasets. Using the training dataset as a reference, SCLSC successfully projected the unseen CD19+ B cells separately from other cell types (refer to Fig. 4a). Moreover, the learned representation of these previously unseen cell types can maintains the hierarchical structure of immune differentiation, as illustrated in Fig. 4b. Specifically, CD19+ B cells are positioned within the same subtree as other lymphoid progenitors like T reg, cytotoxic T cells, T helper cells, and T memory cells. And, they are situated in a different subtree from myeloid progenitors such as monocytes and dendritic cells. This suggests that SCLSC holds the potential to infer representations of unseen cells from incomplete data while maintaining the structure of the complete dataset.

The effectiveness of SCLSC for label transfer task under batch specific conditions

To investigate the performance under batch-specific conditions, we partitioned some datasets based on their batches. Despite SCLSC not directly addressing batch effects, our results indicate its robust performance even in their presence. In this experiment, we focused on the methods that can align different batches so that cells from the same cell type/subpopulation will cluster together. Consequently, we omitted KNN and SingleR due to their inability to yield new alignments. SCLSC consistently achieved high accuracy and macro-averaged F1 scores, ranking first or second across all datasets splitted based on their batches (Fig. 5a). It suggests that SCLSC effectively transfer annotations from the reference dataset to query datasets from different batches. To quantitatively evaluate batch effect removal, we employed Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) metrics. SCLSC, emerging as the top performer in both NMI and ARI evaluations, demonstrates that its learned representation effectively embeds cells of the same type in the same cluster while maintaining separation between different types. Particularly in the CenGen *C. Elegans*, SCLSC exhibited a remarkable improvement, with a 27% increase in accuracy and a 52% increase in macro-F1 score compared to the second-ranked methods. The CenGen C. Elegans dataset encompasses 169 distinct cell types originating from 17 different batches. We separated the datasets into train dataset that contain 12 batches and the test dataset that contain remaining 5

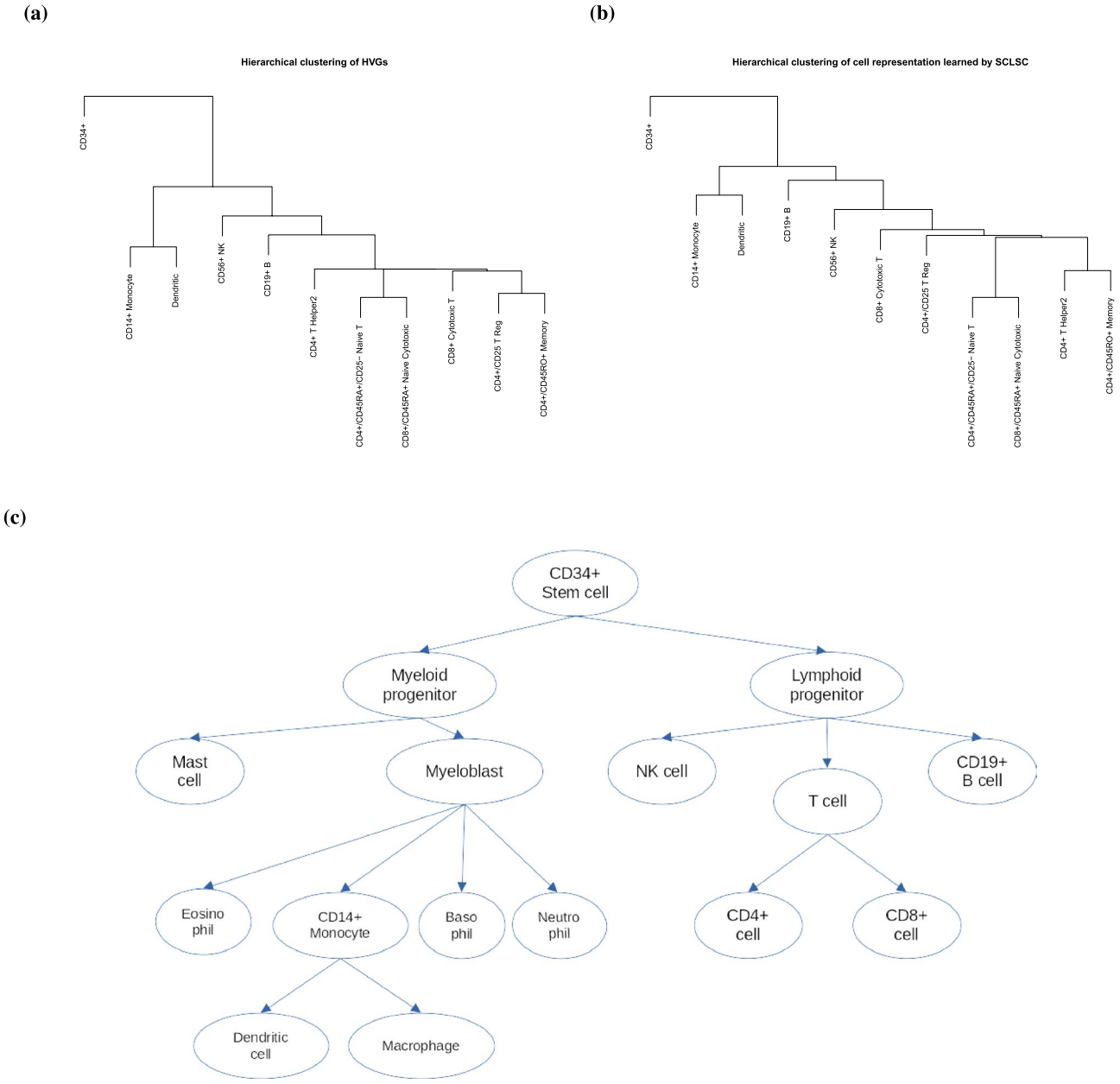

Figure 3. The comparison of dendrogram of PBMC cell type representation and the hierarchy of immune cells differentiation. (**a**) The dendrogram of the cell type representation using HVGs in original space that being used as input and (**b**) the dendrogram of the cell type representation learned by SCLSC showed that SLCSC input and output can capture and preserve (**c**) the hierarchy of immune cells differentiation.

batches. The high number of cell types and batches in this dataset poses a challenge for effective analysis by other methods. The UMAP visualization in Fig. 5b illustrates that SCLSC generates an embedding space where cells from different batches are mixed together, yet the model can still distinguish cells of different types.

There are two plausible explanations as to why our model can partially address the batch effect problem. First, contrastive learning commonly employs data augmentation techniques that introduce diverse transformations to the input data. In this context, the batch condition can be viewed as a form of data augmentation[15]. By leveraging contrastive learning, our model learns to emphasize the shared features among samples while remaining invariant to batch-specific variations. Consequently, this diminishes the impact of batch effects on the learned representations. Second, it is widely accepted that the differences between cell types are greater than differences between batch[16,17]. As SCLSC functions as a supervised model, it utilizes cell type information to guide the learning process and minimize the influence of batch effects.

SCLSC is fast and scale-well in large dataset

We used simulated datasets generated using Splatter R package[18] for scalability analysis. The dataset has 2000 features/genes and 10 cell types with equal occurrence frequencies. We performed three runtime evaluations:

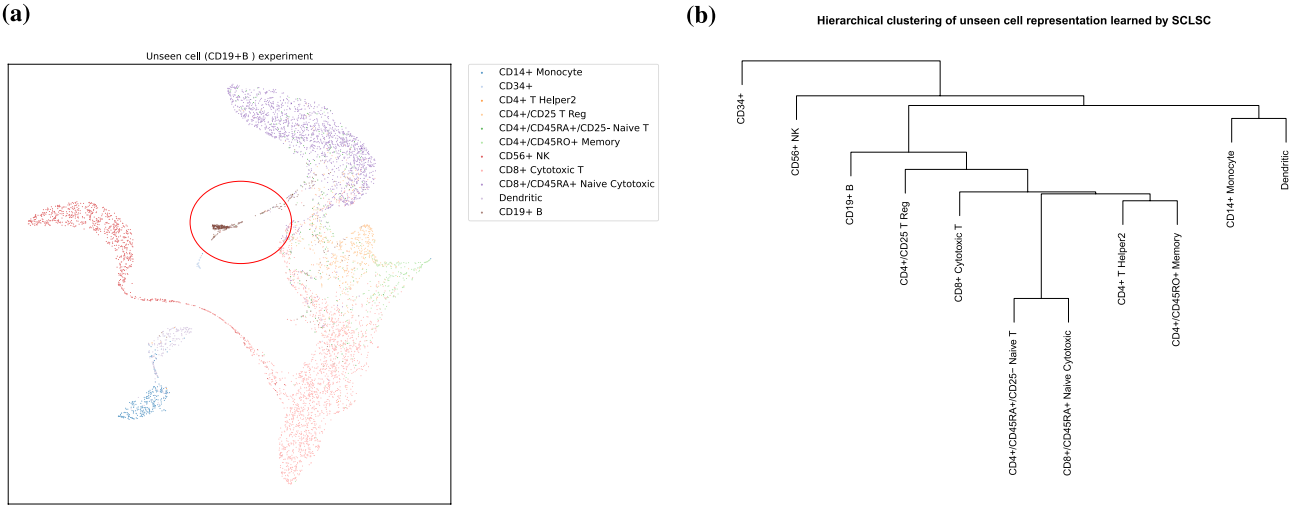

Figure 4. The projection of unseen cell types using SCLSC (**a**) SCLSC can project the unseen cell types (inside red circle) separately from other cell types. (**b**) The dendrogram of the unseen cell type representation approximately preserve the hierarchy of immune cells differentiation.

model training time, which involves calculating the duration from inputting the reference dataset until the model training is completed; label transfer time, where both the reference and query datasets are inputted to the model to transfer annotation from the reference to the query dataset; and total time, which is the sum of model training time and label transfer time. SCLSC comes second in the term of speed (total time) after Symphony compared to to other methods (Fig. 6). Symphony achieves a faster training time compared to SCLSC due to its utilization of a simpler linear mixture model in contrast to the neural network architecture used by SCLSC. Yet, upon completion of training, the label transfer runtime for SCLSC is significantly faster than that of Symphony, clocking in at 3.6 s compared to 313.9 s for a dataset of 400,000 cells. Consequently, in case of training a single reference dataset and applying it to predict the annotations of numerous query datasets, SCLSC will exhibit a faster performance compared to Symphony. SCLSC stands out as the fastest when compared to other deep neural network methods such as scANVI and Concerto in our benchmark. The SCLSC demonstrates an almost twofold increase in speed compared to another contrastive-learning-based-methods, Concerto, when applied to a dataset consisting of 400,000 samples. The results indicate that SCLSC could easily deal with large-scale datasets. SCLSC speed stems from a combination of the simplicity of the contrastive learning algorithm and the straightforwardness of the encoder architecture. The Early Stopping algorithm integrated into the SCLSC also has role to stop the training process early if the model's performance is not improving, thereby saving time and computational resources. The SingleR method was excluded from the scalability analysis alongside other methods due to its failure to generate results within the given time frame for thymus datasets, which comprised approximately 200,000 cells.

Real world application: mapping label from PBMC dataset onto immune cells from dengue datasets

As an example of real-world implementation, we use SCLSC to project cell labels from PBMC dataset from Zheng et al.[19] onto PBMC from a patient with dengue fever (DF) and a patient with dengue hemorrhagic fever (DHF)[20]. The PBMC samples of dengue dataset were gathered on specific days: at defervescence (Def) day, two days before defervescence (Day-2), one day before defervescence (Day-1)—collectively known as the febrile illness period, and two weeks after defervescence (Wk2), which represents the convalescence or follow-up phase. SCLSC can successfully transfer the label from PBMC dataset onto unlabelled cells in the dengue dataset (Fig. 7a). With the aid of these labelled cells, we can conduct downstream analysis to track the dynamics of immune cell populations during dengue virus infection, focusing on B cells in particular. Recent observations have emphasized the significant involvement of B cells during infection with dengue viruses, particularly during acute dengue infection, where a substantial increase in the number of effector B cells has been noticed[21]. To verify the accuracy of the labelling process, we conducted differential gene analysis and gene set enrichment analysis on cells identified as CD19+ B cells. The results confirmed that the labelled cells accurately represented the genetic characteristics associated with B cell cellular processes and function, including B cell activation, differentiation, proliferation, and the host's modulation of viral processes (Fig. 7b). As shown in the Fig. 7c, the proportion of the B cell is peaking in the one day before defervescence. This finding is consistent with a previous study that showed an increase in immunoglobulin-containing B cells can be observed during infection, and these cells reach their maximum levels around the time when the fever starts to subside[22,23]. Within the B cell cluster, we can also investigate the subtype of the B cells such as antibodies secreting cells (ASCs): plasmablast and plasma cell. For this purpose, we used gene markers, namely XBP1[24,25], TNFRSF17[25,26], JCHAIN[27,28], and CD27[29,30] which are ASCs markers. On the other hand, we used MS4A1[29,31,32] as a gene marker for non-ASC-B cells, excluding plasmablasts and plasma cells. Additionally, CD83 was employed to for distinguishing ASCs from non-ASCs. CD83 is an activation marker for antigen presenting cells that are expressed in dendritic cells, monocytes, T cell

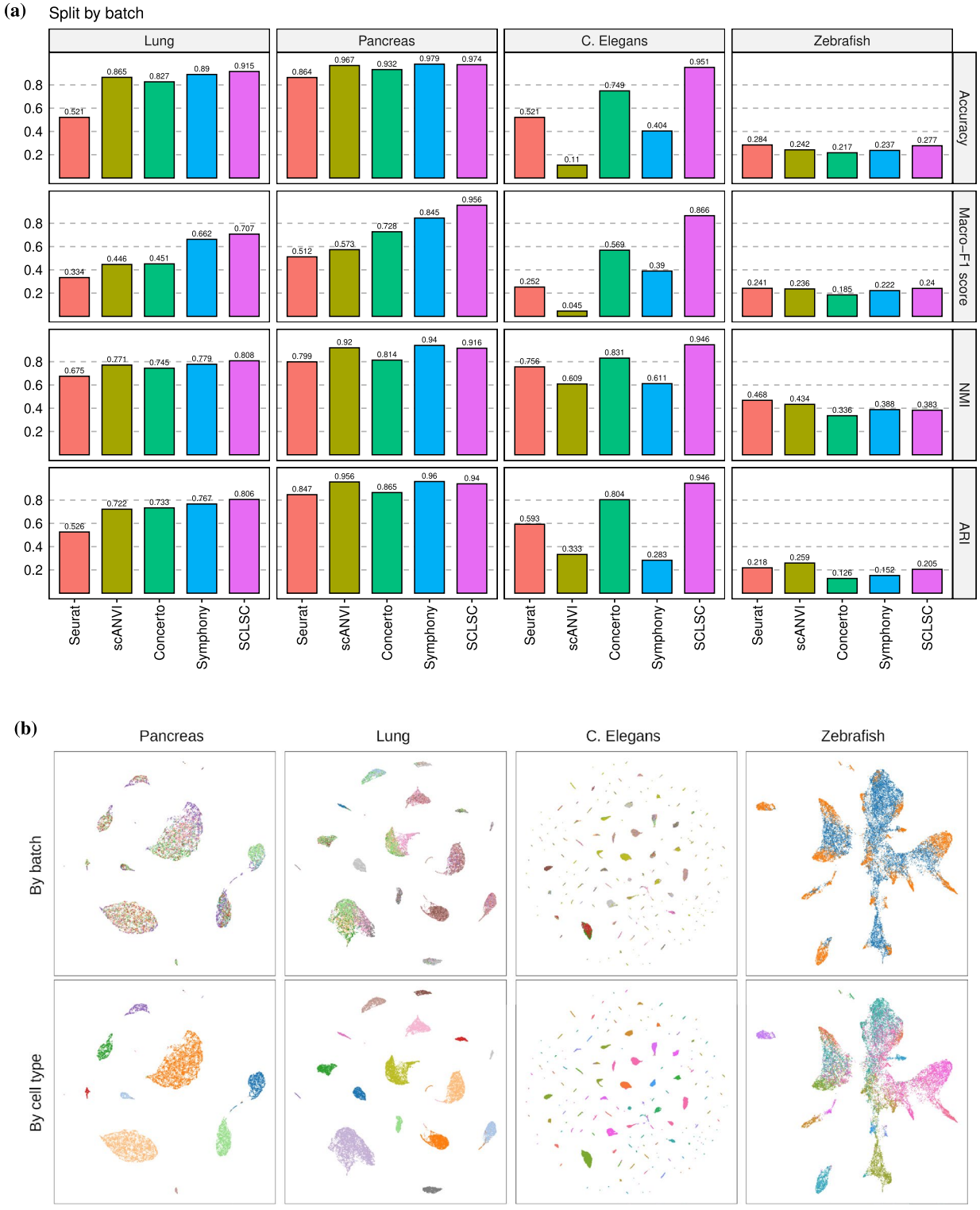

Figure 5. SCLSC can handle label transfer task under batch effects. (**a**) The incorporation of label information in SCLSC can tackles the batch effect, as demonstrated by its high accuracy, macro-F1, NMI, and ARI scores across benchmark datasets. (**b**) The benchmark datasets' learned representation by SCLSC is visualized using UMAP. The upper figure represents the color-coding of batches, while the lower figure represents the color-coding of cell types. The batch groups are well mixed within each cluster, while simultaneously maintaining clear separation of individual cell types.

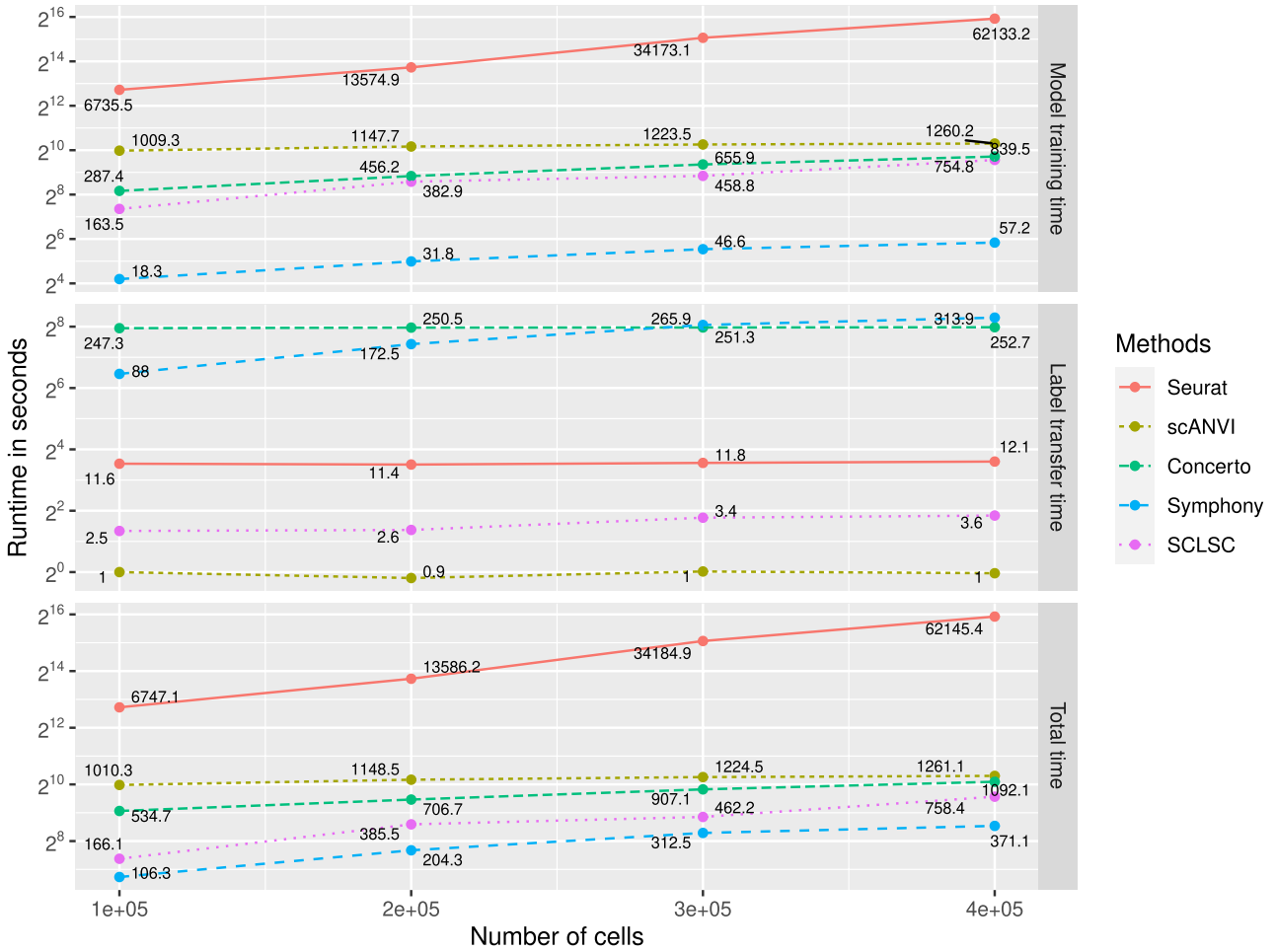

Figure 6. The runtime of the SCLSC and the competing methods against the number of cells. Due to its simplicity, the SCLSC algorithm is capable of efficiently scaling up to handle large datasets. When applied to query-to-reference label transfer tasks involving 100,000 to 400,000 cells, the SCLSC algorithm are the second fastest methods after Symphony. Nevertheless, the label transfer time of SCLSC is orders of magnitude faster than Symphony.

and B cell[33]. While CD83 is not exclusive to B cells, it has been shown that CD83 is statistically overexpressed in non-ASC-B-cell when compared to plasma cells[32]. As shown in the Fig. 7d, the SCLSC learned a representation capable of distinguishing between ASCs (plasmablasts and plasma cells) and non-ASC-B cells.

SCLSC provides stable performance across varying dimensions of input and output

SCLSC has two key parameters: the dimension of the input and the dimension of the output of the encoder. In case of input dimension, SCLSC has the capability to process input from all genes. However, a subset of genes known as highly variable genes (HVGs), which exhibit high cell-to-cell variation, have been found can help in reducing noise and emphasizing the biological signal in scRNA-seq datasets[9,34]. To investigate this further, we conducted an experiment using both all genes and HVGs as SCLSC inputs for the label transfer task on benchmark datasets. We found that using all genes as inputs did not lead to a significant difference in accuracy ($n = 5$, paired Wilcoxon test P-value = 0.11) and made the macro F1-score worse ($n = 5$, paired Wilcoxon test P-value = 0.0311) compared to using HVGs as inputs (Fig. 8a). Furthermore, utilizing all genes required more time and memory resources.

We also conducted an experiment using encoder output dimension = 8, 16, 32, and 64. We observed that the tested output dimension had minimal impact on both the accuracy and macro-F1 score. However, in most datasets, setting the output dimension to 16 resulted in a decrease in the number of epochs required for the early stopper to be triggered (Fig. 8b). This suggests that an output dimension of 16 can achieve optimal accuracy in the shortest time compared to other output dimensions. Based on these results, we used the 2000 HVGs as the input and 16 as the output dimension in our overall study.

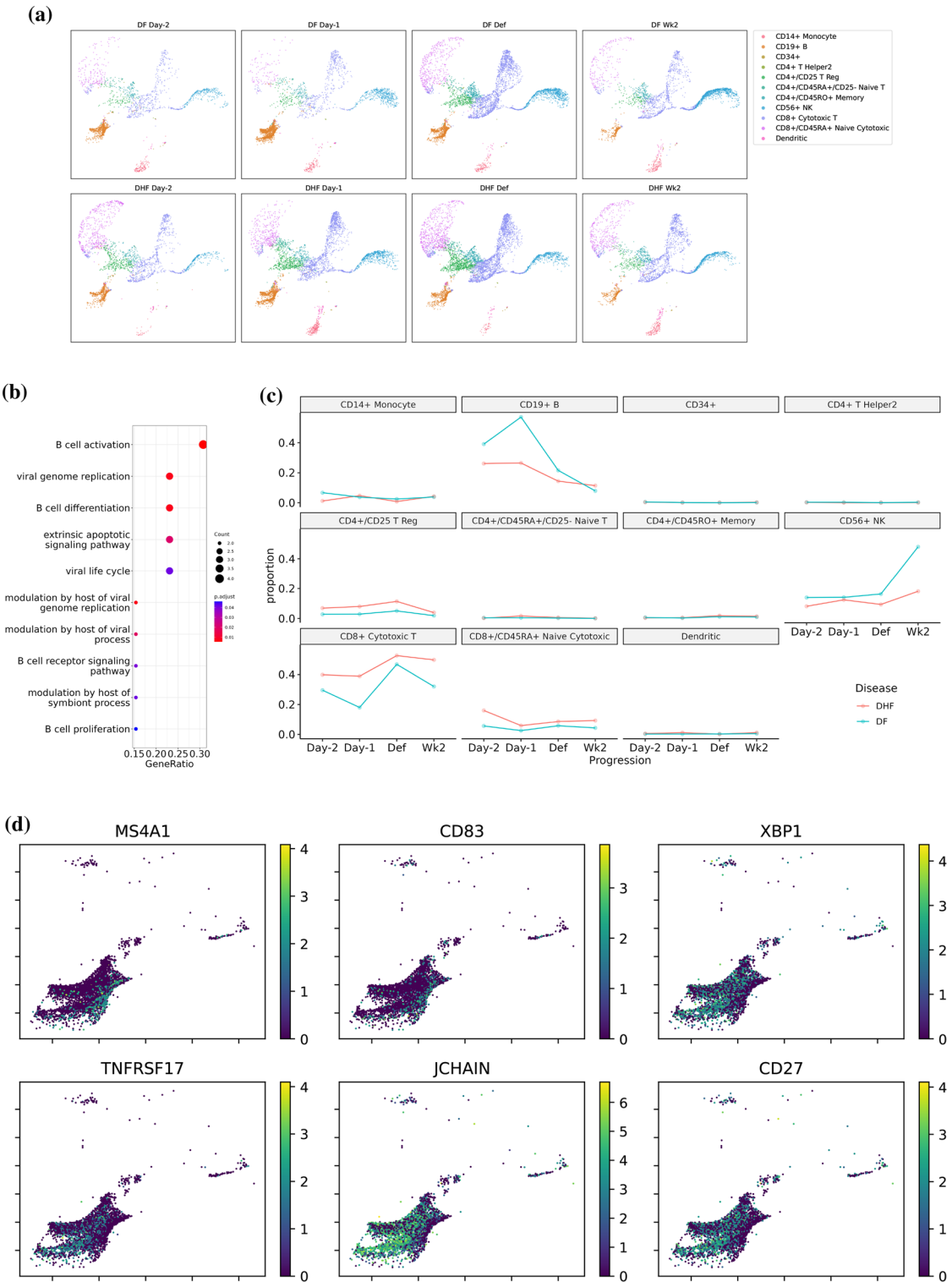

Figure 7. Downstream analysis of the representation of PBMC dengue dataset learned by SCLSC. (**a**) UMAP visualization of the representation of PBMC dataset learned by SCLSC. (**b**) The presence of enriched genes associated with B cell cellular processes in the cells labeled as CD19+ B confirms the accuracy of our labeling, confirming that these cells are indeed B cells. (**c**) The predicted cell labels can be utilized for subsequent downstream analysis, including monitoring the dynamic changes in subpopulations of PBMC cells over dengue infection progression. (**d**) The learned representation of B cells using SCLSC demonstrates the ability to distinguish subtypes that were not observed in the training dataset. The representation separates the antibodies secreting cells (ASCs): plasmablast and plasma cell, characterized by XBP1, TNFRSF17, JCHAIN, and CD27 markers, from the less differentiated non-ASC-B cells labeled by MS4A1 and CD83.

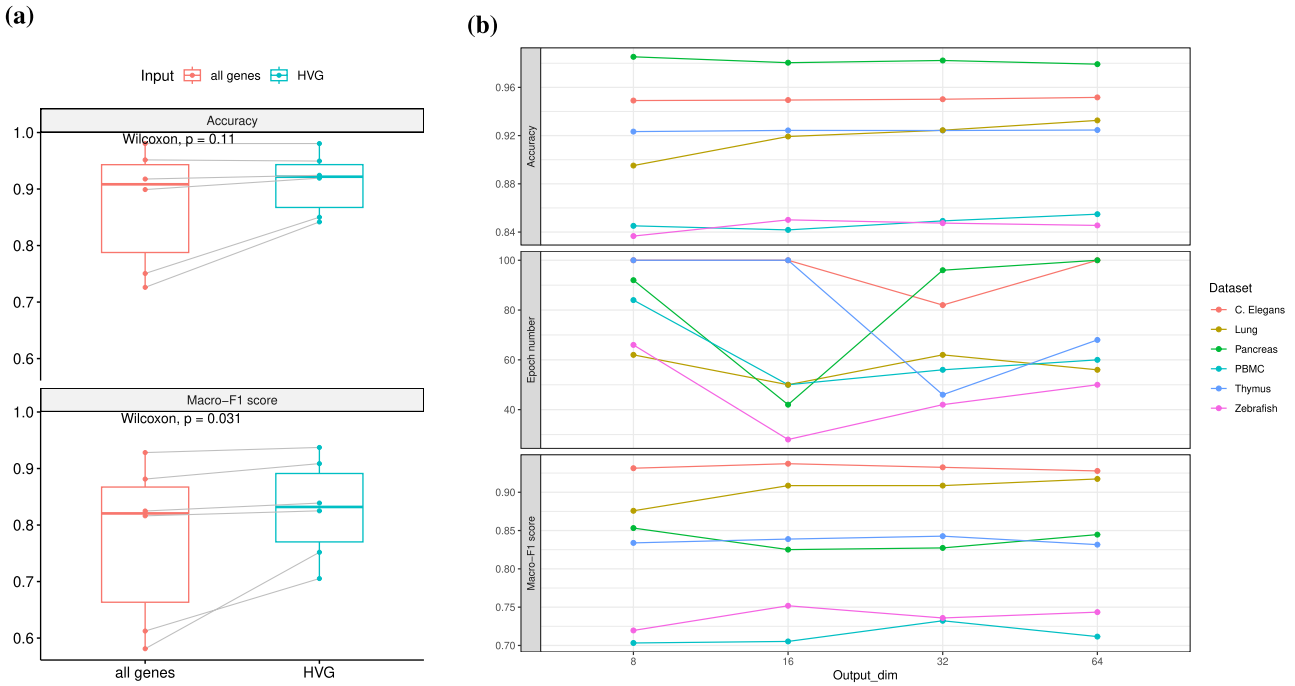

Figure 8. The impact of the encoders input and output dimension to the performance of the SCLSC (**a**) Using all genes as input does not improve accuracy ($n = 5$, paired Wilcoxon test P-value = 0.11), in fact, can be detrimental to the macro-F1 score of SCLSC ($n = 5$, paired Wilcoxon test P-value = 0.0311). (**b**) On the different choice of output dimensions, there were no drastic changes in accuracy and macro-F1 score observed . However, the epoch needed to trigger the early stopper is decreased when the output dimension is 16 in most datasets.

Discussion

In this research, we used real and simulated testing datasets to showcase the effectiveness of SCLSC in transfer cell type knowledge from annotated dataset. One key advantage of the SCLSC method is its simplicity for conducting contrastive learning for cell and cell types. Unlike the previous method doing contrastive learning for different cells, using cell types in the contrastive learning significantly reduced the number of contrastive pairs. With such a feature, SCLSC can tackle diverse datasets efficiently. SCLSC has capability to swiftly provide cell type information for hundred-thousands of cells in a matter of minutes. In our experiment, SCLSC demonstrated its scalability to large datasets and emerged as the second fastest method, following Symphony, in processing 400,000 cells. However, in comparison to Symphony, SCLSC exhibits higher accuracy and consistent performance for the label transfer task.

Previous studies have shown that contrastive learning models possess a remarkable ability to transform single cell input data into a more compact, structured, and expressive representation that captures data characteristics[14,35]. However, by incorporating the label information, supervised contrastive learning have several advantages compared to conventional contrastive learning. First, the model can learn representations that are optimized specifically for the label transfer task, resulting in more distinguishable outcomes. Second, it reduced the sensitivity to negative samples. In conventional contrastive learning, the choice of negative samples plays a crucial role in the learning process. Selecting appropriate negative samples can be challenging, and suboptimal choices may result in the model learning trivial or uninformative patterns. Supervised contrastive learning, by incorporating label information, reduces the reliance on negative samples, making it more robust and less sensitive to the choice of negatives. Third, it is enabling more effective training even with smaller labeled datasets. In traditional contrastive learning approaches, a large labeled dataset is typically required to capture similarity and dissimilarity patterns effectively. However, supervised contrastive learning can leverage labeled data, enabling more effective training even with smaller labeled datasets. Moreover, SCLSC algorithm is using cell type representation, which are the average gene expression of the cell type, to perform cell-cell type pairs instead of cell-cell pairs for contrastive learning. The number of cell types is significantly smaller than the number of cells. Thus, performing contrastive learning is easier and faster because we just need to compare the cell with small number of cell types rather than thousands of cells. Compared to our method, Concerto[14] utilizes unsupervised contrastive learning and employs an asymmetric teacher-student architecture to achieve high performance. In contrast, SCLSC employs a simpler MLP but still achieves comparable results.

There is no universal model that fits all scenarios, and our SCLSC model is no exception. The validity of our model heavily relies on the availability and accuracy of labels in the training data. Acquiring a large amount of precisely labeled single-cell data poses a challenge in the field of single-cell research. Therefore, the applicability of the SCLSC model is restricted to tasks where labeled data is readily accessible. Utilizing high-quality reference

datasets leads to superior annotations and enhances the model's performance. Additionally, the generalization capability of the model is limited when it comes to unseen classes. If the test or deployment data includes classes or samples that were not present during training, the representations produced by the SCLSC model may not generalize well to these unseen classes. To mitigate this limitation, we recommend using large-scale reference datasets that encompass a wide range of cell types during the training process. It's important to consider these limitations when applying the SCLSC model in practical applications and to take appropriate measures such as addressing batch effects, ensuring sufficient availability of accurately labeled data, and using comprehensive reference datasets to enhance generalization to unseen classes.

Methods

Training process

The MLP encoder was trained with inputs from the gene expression count matrix and cell type labels derived from the training dataset. Subsequently, representative samples were defined for each cell type by computing the mean gene expression across all samples with the same cell type. Once these representative samples were created, we employed an MLP encoder to embed both the gene expression matrix and representative samples. Within the embedding space, we calculated the supervised contrastive loss between the samples and their corresponding representative samples. The supervised contrastive loss was employed to update the MLP encoder, to bring each sample closer to its corresponding representative sample while simultaneously distancing it from other representative samples in the embedding space.

To mitigate the possibility of the overfitting, we employed an early stopping algorithm. In this algorithm, the training will stop if the validation loss is not decreasing for five validation steps. A validation step was performed for every two epoch training.

SCLSC supervised contrastive loss

We optimized the contrastive loss, defining it as follows:

$$\begin{aligned} L(c_i, ct_j, y_{ij}) &= y_{ij}\frac{1}{2}Dist\left(embed(c_i), embed(ct_j)\right)^2 \\ &+ (1 - y_{ij})\frac{1}{2}max\left\{0,\ m - Dist(embed(c_i), embed(ct_j))\right\}^2 \end{aligned}$$

The variables c_i, ct_j, y_{ij} are the cell profiles, the cell type representation profiles, and the one-hot-encoded cell label, respectively. The cell type ct_j is computed using all cells of that type from the training data. It is calculating as $ct_j = \frac{1}{K}\sum_{k=1, c_k \in ct_j}^{K} c_k$. Here, K represents the number of cell samples of the type ct_j. A margin m is employed to group cells within their respective annotated cell type, while distinguishing them from the other cell types.

Encoder structure

The encoder network accepts $X \in \mathbb{R}^d$ where d denotes the number of genes. First, the encoder feed X into a dense layer with Relu activation, dropout, and batch normalization layer to get $\mathbf{hidden_1}$. Then, the $\mathbf{hidden_1}$ is fed into a second dense layer with Relu activation, dropout, and batch normalization layer to get $\mathbf{hidden_2}$. Finally, the $\mathbf{hidden_2}$ is fed into a last dense layer to get the final output $Z \in \mathbb{R}^{d'}$ where d' denotes the number of embedding space dimensions.

$$\begin{aligned} hidden_1 &= BatchNorm(Dropout(Relu(Dense(X)))) \\ hidden_2 &= BatchNorm(Dropout(Relu(Dense(hidden_1)))) \\ Z &= Dense(hidden_2) \end{aligned}$$

Datasets

We have chosen commonly utilized datasets for benchmarking label transfer tasks (Table 1). These datasets are publicly accessible and come in a standardized format that is ready for immediate use. Each dataset possesses

Name	Description	No. cells	No. types	Download link	Ref
PBMC	Human PBMC cells	68,265	11	PBMC	19
Pancreas	Human pancreas cells	16,382	14	Pancreas	11
Thymus	Human thymus cells	223,792	44	Thymus link	36
Lung	Human lung cells	30,717	17	Lung	11
CeNGEN	*C. elegans* neuron cells	72,857	169	CeNGEN	38
Zebrafish	Zebrafish embryo cells	26,022	24	zebrafish	37
Dengue dataset	PBMC of dengue fever patients	39,591	NA	Dengue	20
Simulated dataset	Simulated dataset created by Splatter package	500,000	10	–	18

Table 1. The summary of the datasets used in our study.

distinct attributes, including imbalanced distributions among cell types, multicollinearity between cell types, a large number of cell types, a substantial number of cells, and challenges related to batch effects.

PBMC dataset
We used Zheng et al.[19] Peripheral Blood Mononuclear Cells (PBMC) dataset freely available from 10X Genomics. After preprocessing, we got total 68265 cells before splitting. This dataset contains rare cell types and the distribution of cell types is imbalanced. Moreover, the cell types in the dataset were highly correlated with each other make it difficult to differentiate them.

Pancreas dataset
The pancreas dataset is a human pancreatic islet scRNA-seq data from 6 sequencing technologies (CEL-seq, CEL-seq2, Smart-seq2, inDrop, Fluidigm C1, and SMARTER-seq)[11].

Lung dataset
Human lung scRNA-seq data were obtained from 2 different sequencing technologies: 10X and Drop-seq[11]. It can be used for analyzing batch effect in single cell analysis.

Thymus dataset
The thymus dataset were from the single-cell RNA sequencing of cells inside human thymus[36]. This large dataset contains more than 250,000 cells which make it useful for evaluating methods scalability.

Zebrafish dataset
This dataset contains the data of zebrafish embryos cells during the first day of development, with and without a knockout of chordin, an important developmental gene. This dataset contain data from 2 different laboratories. After preprocessing, this dataset had dimension 26022 cells × 2000 genes and 24 cell types[37].

CeNGEN dataset
This dataset are from The Complete Gene Expression Map of the C. elegans Nervous System (CeNGEN) project and contains FACS-isolated C. elegans neurons data sequenced on 10x Genomics[38]. This dataset is characterized by a large number of cell types and imbalanced distribution of cell types.

Dengue dataset
These data originate from PBMC cells taken from patients with acute dengue virus infection[20]. Because this dataset lacks labels, we utilized a PBMC dataset as a reference to annotate the cells in the dengue dataset for real-world application experiments.

Random split experiment

We divided the datasets into three subsets: training, validation, and test datasets. Employing stratified random splits through the *StratifiedKFold* function from the scikit-learn package with 10 splits, we designated the datasets in the first eight folds as the training dataset, the ninth fold as the validation dataset, and the last tenth fold as the test dataset. The ultimate ratio of training to validation to test datasets is 8:1:1. We used the gene expression matrix and the cell labels from the training data to train the MLP encoder. To prevent overfitting during the training of the MLP encoder, we monitored the loss changes in the validation dataset for early stopping. Following the training, we utilized the MLP encoder to embed gene expression matrices from both the training and test datasets. Finally, KNN was used to predict the labels of the test datasets with the train dataset serving as a reference.

Batch split experiment

Initially, we partitioned the datasets into two groups: train-validation datasets and test datasets, based on their batches (Table 2). Next, from the train-validation datasets, we further divided this dataset using stratified random split into train and validation datasets, with a ratio of 9:1 for training to validation. Similar to the random split experiment, we used the training and validation datasets for training the MLP encoder and implementing early stopping. The trained MLP encoder was subsequently employed to embed both the test and training datasets into an embedding space. Within this space, we used KNN to predict the labels of the test datasets, using the train dataset as a reference.

Name	Total number of batches	No. batches in training-validation dataset	No. batches in test dataset
Lung	16	15	1
Pancreas	9	8	1
CeNGEN	17	12	5
Zebrafish	2	1	1

Table 2. The batch split of the datasets.

Unseen cell experiment

We performed stratified random splits on the PBMC datasets, dividing them into training, validation, and test datasets with a ratio of 8:1:1 for training to validation to test datasets. Then, we removed the CD19+B cells from the training and validation dataset. We used training dataset as reference to predict the annotation of test dataset.

Compared benchmark methods

Seurat

Seurat is a widely used R package that has been developed for analysis and exploration of single-cell RNA sequencing data. Seurat transfer the annotation data by using a canonical correlation analysis of a set of anchor genes that are highly variable and shared between the reference and new datasets[9]. We used Seurat v4.0 and followed the transfer annotation from query datasets tutorial to perform label projection in this study.

SingleR

SingleR is an automatic annotation method for single-cell RNA sequencing implemented in R package[10]. In the SingleR pipeline, a Spearman coefficient is calculated for single-cell gene expression with each of the samples in the reference dataset, using only the variable genes in the reference dataset to increase the ability to distinguish closely related cell types. This process is performed iteratively using only the top cell types from the previous step and the variable genes among them until only one cell type remains. We used default parameter of SingleR R package in our study.

scANVI

scANVI is a semi-supervised model that employed Variational Inference to annotate a dataset of unlabelled query dataset from annotated reference dataset[13]. In our study, we used scANVI methods implemented in scvi-tools Python package with default parameter. To perform label projection, we followed the reference mapping scvi-tools tutorial.

Concerto

Concerto leverages a self-distillation contrastive learning framework to learn cell embeddings in lower dimensional space[14]. The learned cell embeddings are fed into KNN classifier to perform label projection task. In our study, we employed the Concerto source code downloaded from public repository (https://github.com/melobio/Concerto-reproducibility. We used default parameter stated in the Concerto source code.

Symphony

Symphony[12] utilizes a linear mixture model framework to iteratively assign soft-cluster memberships and compress the reference into a mappable entity with efficient summary statistics. The mapping algorithm then projects query cells into the reference's low-dimensional space, computes soft-cluster assignments, and corrects query batch effects while maintaining the stability of the reference cell embedding. This enables the transfer of annotations from reference to query cells. We used the Python implementation of Symphony, symphonypy, accessible at https://github.com/potulabe/symphonypy. The symphony parameter was configured using the default settings specified in the symphonypy source code.

K-nearest neighbors (KNN)

The KNN classifier was employed for direct label transfer from raw data. We utilized the KNN implementation in scikit-learn, specifically the *KNeighborsClassifier*. A value of $k = 10$ was set for the number of neighbors parameter, while other parameters remained at their default settings.

Data preprocessing

The datasets underwent preprocessing to eliminate cells with high mitochondrial gene expression (more than 5 percents of the cell total count), cells with minimal gene expression (number of genes per cell < 200), and genes that were only detected in a small number of cells (number of cells that expressed the gene < 3). Subsequently, We selected 2000 highly variable genes (HGV) using analytic Pearson residuals implemented in Scanpy package. Following this, we normalized the count of each cell to 10,000 counts and applied a $log(x + 1)$ transformation. The resulting dataset was then divided into training, validation, and test sets with a ratio of $8 : 1 : 1$. All of the preprocessing steps were performed using Scanpy package[39]. The summary of the dataset, reference, and download link were provided in Table 1.

Scalability analysis

For scalability analysis, simulated datasets were generated using Splatter R package[18]. The simulated datasets were generated using the following parameters: nGenes = 2000 and group.prob = rep(1/10,times = 10), while leaving the other parameters at their default values. The number of cells in the training dataset = 100,000, 200,000, 300,000, and 400,000 cells. The validation dataset and test dataset comprised 50,000 cells each, both derived from the same cell distribution as the training dataset. Then, each method was trained on each of the training datasets to make predictions regarding the annotations within the test dataset. Next, we calculated the runtime from the training until finishing prediction.

Hyperparameter analysis

We evaluated the accuracy and macro-F1 score of SCLSC across various input scenarios (utilizing all genes versus 2000 highly variable genes (HVGs)) and different encoder output dimensions (8, 16, 32, and 64) across PBMC, zebrafish, C. elegans, thymus, lung, and pancreas datasets. Additionally, we assessed the number of epochs required to trigger the early stopper in different encoder output dimension experiments. We conducted a paired Wilcoxon test to compare the accuracy and macro-F1 score between the scenarios of utilizing all genes and the 2000-HVGs experiment. A *P*-value below 0.05 is deemed statistically significant.

UMAP visualization

Cells embedding were visualized by UMAP using umap-learn Python package. The umap parameters were set at their default values

Dendogram visualization

First, we computed the averages of the 2000 High Variable Genes (HVGs) and the embedded features for each corresponding cell type, which yielded cell type representations in both the original input space and the embedding space. Then, we calculated the Euclidean distance matrix for these cell type representations in both spaces. Subsequently, we constructed dendrograms using single linkage based on the distance matrix. The dendograms were produced using hclust package.

Dengue dataset analysis

We compute a ranking for the highly differential genes in the cells labelled as CD19+ B cells using Wilcoxon method implemented in the rank_genes_groups function in Scanpy package[39]. Next, we performed gene set enrichment analysis of 15 highest rank differential genes. The enrichment analysis were performed using clusterProfiler package[40].

Evaluation metrics

Accuracy

Accuracy score is the fraction of correct predictions. It is calculated as following:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\frac{\text{Number of Correct Predictions}_i}{\text{Total Number of Predictions}_i}$$

where N is the total number of samples or instances.

Macro-F1 score

The F1 score can be seen as an average of precision and recall, achieving its highest value at 1 and its lowest at 0. Because in some datasets there are some cell types that have significantly fewer instances than others, we used we used macro-averaging F1 score (macro-F1 score). Macro F1 score gives equal weight to each class, treating them independently, which can be useful when we want to ensure that each class is adequately represented in the evaluation. The macro F1 score is calculated as the average of the individual F1 scores for each class as following:

$$F1_{\text{macro}} = \frac{1}{C}\sum_{i=1}^{C}\frac{2 \times \text{Precision}_{\text{class}_i} \times \text{Recall}_{\text{class}_i}}{\text{Precision}_{\text{class}_i} + \text{Recall}_{\text{class}_i}}$$

where C is the number of classes.

Adjusted Rand Index

The Rand Index is a measure used to assess the similarity between two data clusterings after batch removal. It evaluates the agreement between the true class labels and the predicted labels. In our study we used Adjusted Rand index (ARI) which is Rand index adjusted for chance. The formula for ARI is following:

$$ARI = \frac{\text{RI} - E[RI]}{\max(\text{RI}_{\max} - E[RI], 0)}$$

Where RI is the Rand Index,$E[RI]$ is the expected Rand Index under a random assignment, RI_{max} is the maximum possible Rand Index given the marginal totals. The Adjusted Rand Index ranges from -1 to 1, where 1 indicates perfect agreement, 0 indicates random agreement, and negative values indicate worse than random agreement.

Normalized mutual info score

The Normalized Mutual Information (NMI) is a measure of the mutual dependence between two clustering results after batch removal. It is normalized to have a value between 0 and 1, with 1 indicating perfect agreement. The formula for NMI is:

$$NMI(U, V) = \frac{MI(U, V)}{\text{mean}(H(U), H(V))}$$

where $MI(U, V)$ is the mutual information between the two clusterings, $H(U)$ and $H(V)$ are the entropies of the individual clusterings.

Data availability

All the real single-cell RNA sequencing datasets used in this study had been previously published. References to these datasets, information about their accessibility, and downloadable links can be found in Table 1.

Code availability

The source code is accessible at https://github.com/yaozhong/SCLSC.

Received: 9 September 2023; Accepted: 16 December 2023
Published online: 03 January 2024

References

1. Papalexi, E. & Satija, R. Single-cell RNA sequencing to explore immune cell heterogeneity. *Nat. Rev. Immunol.* **18**, 35–45. https://doi.org/10.1038/nri.2017.76 (2017).
2. Hwang, B., Lee, J. H. & Bang, D. Single-cell RNA sequencing technologies and bioinformatics pipelines. *Exp. Mol. Med.* **50**, 1–14. https://doi.org/10.1038/s12276-018-0071-8 (2018).
3. Regev, A. *et al.* The human cell atlas. *eLife* **6**, e27041. https://doi.org/10.7554/elife.27041 (2017).
4. Consortium, T.M. Single-cell transcriptomics of 20 mouse organs creates a tabula muris. *Nature* **562**, 367–372. https://doi.org/10.1038/s41586-018-0590-4 (2018).
5. Diaz-Mejia, J. J. *et al.* Evaluation of methods to assign cell type labels to cell clusters from single-cell RNA-sequencing data. *F1000Research* **8**, 296. https://doi.org/10.12688/f1000research.18490.3 (2019).
6. Zhang, X. *et al.* Cell marker: A manually curated resource of cell markers in human and mouse. *Nucleic Acids Res.* **47**, D721–D728. https://doi.org/10.1093/nar/gky900 (2018).
7. Grün, D. & van Oudenaarden, A. Design and analysis of single-cell sequencing experiments. *Cell* **163**, 799–810. https://doi.org/10.1016/j.cell.2015.10.039 (2015).
8. Kim, T. *et al.* Impact of similarity metrics on single-cell RNA-seq data clustering. *Brief. Bioinform.* **20**, 2316–2326. https://doi.org/10.1093/bib/bby076 (2018).
9. Stuart, T. *et al.* Comprehensive integration of single-cell data. *Cell* **177**, 1888-1902.e21. https://doi.org/10.1016/j.cell.2019.05.031 (2019).
10. Aran, D. *et al.* Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nat. Immunol.* **20**, 163–172. https://doi.org/10.1038/s41590-018-0276-y (2019).
11. Luecken, M. D. *et al.* Benchmarking atlas-level data integration in single-cell genomics. *Nat. Methods* **19**, 41–50. https://doi.org/10.1038/s41592-021-01336-8 (2021).
12. Kang, J. B. *et al.* Efficient and precise single-cell reference atlas mapping with symphony. *Nat. Commun.* **12**, 5890. https://doi.org/10.1038/s41467-021-25957-x (2021).
13. Xu, C. *et al.* Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Mol. Syst. Biol.* **17**, e9620 https://doi.org/10.15252/msb.20209620 (2021).
14. Yang, M. *et al.* Contrastive learning enables rapid mapping to multimodal single-cell atlas of multimillion scale. *Nat. Mach. Intell.* **4**, 696–709. https://doi.org/10.1038/s42256-022-00518-z (2022).
15. Sun, Y. & Qiu, P. Domain adaptation for supervised integration of scRNA-seq data. *Commun. Biol.* **6**, 274. https://doi.org/10.1038/s42003-023-04668-7 (2023).
16. Haghverdi, L., Lun, A. T. L., Morgan, M. D. & Marioni, J. C. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.* **36**, 421–427. https://doi.org/10.1038/nbt.4091 (2018).
17. Polański, K. *et al.* BBKNN: Fast batch alignment of single cell transcriptomes. *Bioinformatics* **36**, 964–965. https://doi.org/10.1093/bioinformatics/btz625 (2019).
18. Zappia, L., Phipson, B. & Oshlack, A. Splatter: Simulation of single-cell RNA sequencing data. *Genome Biol.* **18**, 174. https://doi.org/10.1186/s13059-017-1305-0 (2017).
19. Zheng, G. X. Y. *et al.* Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8**, 14049. https://doi.org/10.1038/ncomms14049 (2017).
20. Arora, J. K. *et al.* Single-cell temporal analysis of natural dengue infection reveals skin-homing lymphocyte expansion one day before defervescence. *iScience* **25**, 104034. https://doi.org/10.1016/j.isci.2022.104034 (2022).
21. Wrammert, J. *et al.* Rapid and massive virus-specific plasmablast responses during acute dengue virus infection in humans. *J. Virol.* **86**, 2911–2918. https://doi.org/10.1128/jvi.06075-11 (2012).
22. Boonpucknavig, S., Lohachitranond, C. & Nimmanitya, S. The pattern and nature of the lymphocyte population response in dengue hemorrhagic fever. *Am. J. Trop. Med. Hyg.* **28**, 885–889 (1979).
23. Jampangern, W. *et al.* Characterization of atypical lymphocytes and immunophenotypes of lymphocytes in patients with dengue virus infection. *Asian Pac. J. Allergy Immunol.* **25**, 27 (2007).
24. Hoffman, W., Lakkis, F. G. & Chalasani, G. B cells, antibodies, and more. *Clin. J. Am. Soc. Nephrol.* **11**, 137–154. https://doi.org/10.2215/cjn.09430915 (2016).
25. Tellier, J. & Nutt, S. L. Standing out from the crowd: How to identify plasma cells. *Eur. J. Immunol.* **47**, 1276–1279. https://doi.org/10.1002/eji.201747168 (2017).
26. Yang, M. *et al.* B cell maturation antigen, the receptor for a proliferation-inducing ligand and b cell-activating factor of the TNF family, induces antigen presentation in b cells. *J. Immunol.* **175**, 2814–2824. https://doi.org/10.4049/jimmunol.175.5.2814 (2005).
27. Castro, C. D. & Flajnik, M. F. Putting J chain back on the map: How might its expression define plasma cell development?. *J. Immunol.* **193**, 3248–3255. https://doi.org/10.4049/jimmunol.1400531 (2014).
28. Xu, A. Q., Barbosa, R. R. & Calado, D. P. Genetic timestamping of plasma cells in vivo reveals tissue-specific homeostatic population turnover. *eLife* **9**, e59850. https://doi.org/10.7554/elife.59850 (2020).
29. Sanz, I. *et al.* Challenges and opportunities for consistent classification of human B cell and plasma cell populations. *Front. Immunol.* **10**, 2458. https://doi.org/10.3389/fimmu.2019.02458 (2019).
30. Cancro, M. P. & Tomayko, M. M. Memory B cells and plasma cells: The differentiative continuum of humoral immunity. *Immunol. Rev.* **303**, 72–82. https://doi.org/10.1111/imr.13016 (2021).
31. Kumar, S., Kimlinger, T. & Morice, W. Immunophenotyping in multiple myeloma and related plasma cell disorders. *Best Pract. Res. Clin. Haematol.* **23**, 433–451. https://doi.org/10.1016/j.beha.2010.09.002 (2010).
32. Tarte, K., Zhan, F., De Vos, J., Klein, B. & Shaughnessy, J. Gene expression profiling of plasma cells and plasmablasts: Toward a better understanding of the late stages of B-cell differentiation. *Blood* **102**, 592–600. https://doi.org/10.1182/blood-2002-10-3161 (2003).
33. Grosche, L. *et al.* The CD83 molecule—an important immune checkpoint. *Front. Immunol.* **11**, 721. https://doi.org/10.3389/fimmu.2020.00721 (2020).

34. Brennecke, P. *et al.* Accounting for technical noise in single-cell RNA-seq experiments. *Nat. Methods* **10**, 1093–1095. https://doi.org/10.1038/nmeth.2645 (2013).
35. Ciortan, M. & Defrance, M. Contrastive self-supervised clustering of scRNA-seq data. *BMC Bioinform.* **22**, 280. https://doi.org/10.1186/s12859-021-04210-8 (2021).
36. Park, J.-E. *et al.* A cell atlas of human thymic development defines T cell repertoire formation. *Science* **367**, eaay3224. https://doi.org/10.1126/science.aay3224 (2020).
37. Wagner, D. E. *et al.* Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. *Science* **360**, 981–987. https://doi.org/10.1126/science.aar4362 (2018).
38. Hammarlund, M., Hobert, O., Miller, D. M. & Sestan, N. The CeNGEN project: The complete gene expression map of an entire nervous system. *Neuron* **99**, 430–433. https://doi.org/10.1016/j.neuron.2018.07.042 (2018).
39. Wolf, F. A., Angerer, P. & Theis, F. J. SCANPY: Large-scale single-cell gene expression data analysis. *Genome Biol.* **19**, 1–5. https://doi.org/10.1186/s13059-017-1382-0 (2018).
40. Wu, T. *et al.* clusterProfiler 4.0: A universal enrichment tool for interpreting omics data. *Innovation* **2**, 100141. https://doi.org/10.1016/j.xinn.2021.100141 (2021).

Acknowledgements
The computing resources were provided by Human Genome Center, the Institute of Medical Science, the University of Tokyo.

Author contributions
Y.D.H. was responsible for the the data curation, analyses, and visualization, and writing the original draft of the manuscript. Y.Z. was responsible for the study conceptualization, data analysis, supervision, and editing the manuscript. S.I. was responsible for the funding acquisition, project administration, supervision, and editing the manuscript. All authors reviewed the manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-023-50185-2.

Correspondence and requests for materials should be addressed to Y.Z. or S.I.