Contrastive Learning for Robust Cell Annotation and Representation from Single-Cell Transcriptomics

Leo Andrekson[1] and Rocío Mercado[1*]

[1*]Department of Computer Science and Engineering, Chalmers University of Technology, Chalmersplatsen 4, Gothenburg, 41258, Sweden.

*Corresponding author(s). E-mail(s): rocio.mercado@chalmers.se;
Contributing authors: leo.andrekson@gmail.com;

Abstract

Batch effects are a significant concern in single-cell RNA sequencing (scRNA-Seq) data analysis, where variations in the data can be attributed to factors unrelated to cell types. This can make downstream analysis a challenging task. In this study, we present a novel deep learning approach using contrastive learning and a carefully designed loss function for learning an generalizable embedding space from scRNA-Seq data. We call this model CELLULAR: CELLUlar contrastive Learning for Annotation and Representation. When benchmarked against multiple established methods for scRNA-Seq integration, CELLULAR outperforms existing methods in learning a generalizable embedding space on multiple datasets. Cell annotation was also explored as a downstream application for the learned embedding space. When compared against multiple well-established methods, CELLULAR demonstrates competitive performance with top cell classification methods in terms of accuracy, balanced accuracy, and F1 score. CELLULAR is also capable of performing novel cell type detection. These findings aim to quantify the *meaningfulness* of the embedding space learned by the model by highlighting the robust performance of our learned cell representations in various applications. The model has been structured into an open-source Python package, specifically designed to simplify and streamline its usage for bioinformaticians and other scientists interested in cell representation learning.

Keywords: scRNA-Seq, cell type annotation, novel cell type detection, machine learning, contrastive learning, bioinformatics

1 Introduction

Cells in our body exhibit remarkable diversity, and characterizing and classifying different cell types is fundamental to understanding their functions and roles in various biological processes. Learning compact vector representations of cell types enables us to organize and categorize cells based on their shared characteristics, such as their observed phenotypes, but within a lower-dimensional space than that afforded by measurements of cellular phenotypes. Due to its far-reaching implications in the biomedical sciences, including disease diagnostics and drug development, cell representation learning is an active area of research at the interface of deep learning and biology, but currently no such generalizable approach exists which can effectively do this. Such compact representations would enable us to use machine learning (ML) to learn previously unknown relationships between cells, identify disease-specific cell populations, and design targeted therapies for specific cell types and tissues.

Towards the aim of learning *meaningful* representations of cells, this study leverages single-cell RNA sequencing (scRNA-Seq) data to train a deep neural network to produce an efficient, lower-dimensional, generalizable embedding space for unique cell types. The method is called CELLULAR, or CELLUlar contrastive Learning for Annotation and Representation, and presented as an open-source Python package. This leads to the second aim of the work, which involved benchmarking CELLULAR against leading methods for scRNA-Seq data integration. Finally, the third aim of the study was to demonstrate the utility of the learned embedding space for two downstream applications: cell type annotation and novel cell type detection. This last aim allowed us to evaluate the performance of CELLULAR against previous work by validating the quality of the latent space generated for common bioinformatics tasks in single-cell transcriptomics.

In the following sections, we present an overview on single-cell omics, cell type annotation, and scRNA-Seq integration, three topics of central relevance to this study.

1.1 Single-cell omics

In the last decade, single-cell omics (sc-omics) has emerged as a rapidly growing field of biological research due to its applications in atlasing projects (e.g., the Human Cell Atlas), developmental studies, and precision medicine. [1] Popular single-cell methods include transcriptomics, epigenomics, and proteomics, where single-cell RNA sequencing (scRNA-Seq; Figure 1c), a transcriptomics approach, is currently the most used sc-omics method. [1, 2]

As various omics techniques become available for single-cell analysis, the field of single-cell multi-omics has gained prominence. [3] These methods aim to extract and integrate information from multiple omics methods within a single cell type, such that complex properties or insights about that cell family emerge. Some examples of common integrated multi-omics approaches are combinations of transcriptomics-genomics and transcriptomics-epigenomics. [4] As these technologies improve and their cost is reduced, one might eventually be able to analyze how diseases develop in patients and how they react to different therapies *at single-cell resolution*, giving unprecedented insights into disease progression at a personalized level. [1] Nevertheless, before

sc-omics data can be effectively used in this way, we must first investigate different approaches to reduce both the noise and dimensionality of these large datasets. This motivates the development of deep learning models capable of learning a generalizable latent space from scRNA-Seq data in this work, where such approaches can be collectively referred to as methods for *cell representation learning*.

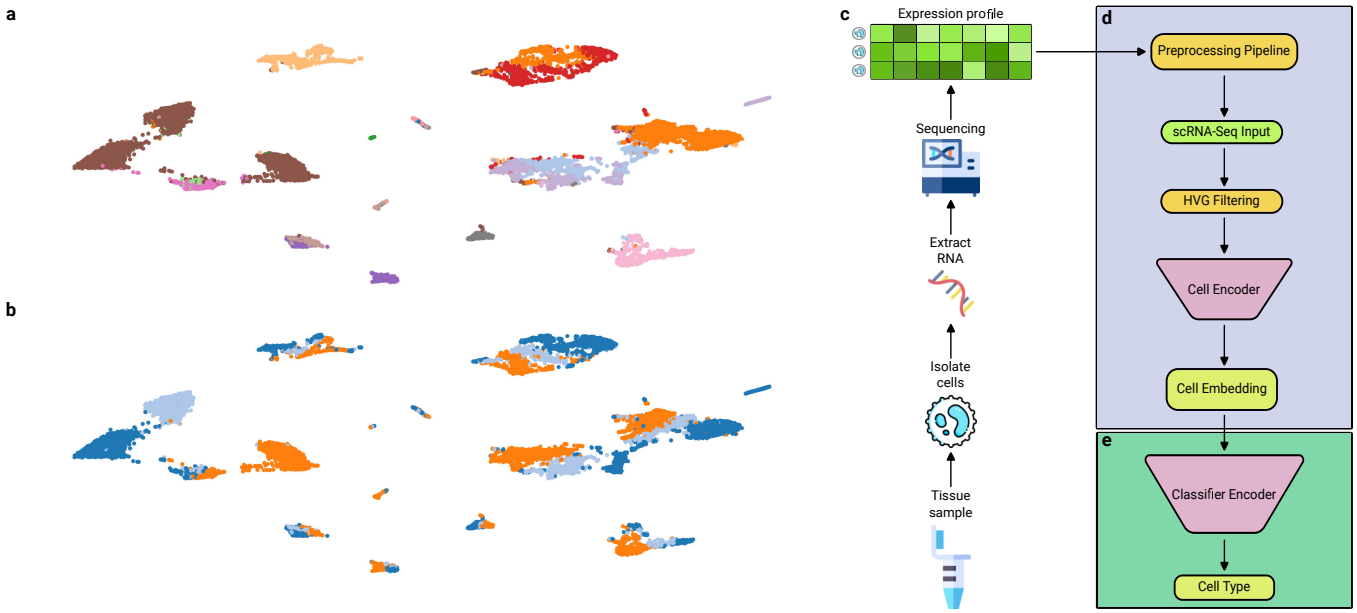

Fig. 1 (a) UMAP visualization of the kidney scRNA-Seq dataset constructed in this study, colored by cell type. (b) UMAP visualization colored by patient ID, highlighting batch effects due to patient ID. (c) Single-cell RNA sequencing generally consists of sampling, isolation of cells, RNA extraction, reverse transcription to cDNA, sequencing, and eventual data analysis. (d) Overview of CELLULAR. ScRNA-Seq data is fed through a feed-forward encoder that returns a learned embedding for each sample (cell). (e) CELLULAR embeddings can then be used as input to another feed-forward neural network to classify samples into potential cell types.

1.1.1 Cell type annotation

As the acquisition of large-scale omics data for single cells becomes increasingly accessible, the interest for automated cell annotation tools grows. This is attributed to the time-consuming and subjective nature of manual cell annotation. Recently developed strategies for cell type annotation from scRNA-Seq data typically rely on either referencing marker gene databases, correlating reference expression data, or supervised classification methods. [5] In this study, a novel method for supervised cell type annotation was developed. The model works by initially learning an embedding space from the scRNA-Seq data through contrastive learning, and subsequently using this space for cell type classification.

1.1.2 scRNA-Seq data integration

ScRNA-Seq datasets often encompass numerous samples, leading to the inevitable introduction of batch effects arising from various sources (e.g., diverse patients, sequencing methodology). [6] Here, batch effects denote sources of variation within the data attributed to non-biological factors. To emphasize the challenges imposed by batch effects, we visualize some example data in Figure 1a,b. The ideal model would integrate scRNA-Seq data into an embedding space where biological variation is maximized while minimizing variation caused by batch effects, but this is easier said than done. Many methods have been developed for this purpose, such as scANVI [7] and scGen [8], and these have been benchmarked in a recent study by Luecken et al. [9]. While these methods have shown promise in scRNA-Seq data integration, they have not been evaluated for learning a generalizable embedding space.

In this study, the scRNA-Seq integration benchmark introduced by Luecken et al. [9], aptly named *single-cell integration benchmarking* or *scIB*, was used to compare approaches utilizing transformers (TOSICA [10]), variational autoencoders (scGen [8]), and hierarchical Bayesian models (scVI [11] and scANVI [7]). By comparing all of these methods against CELLULAR, we can also see how contrastive learning fares against these methods on data integration. By including multiple modeling approaches we ensure that the benchmark covers a wide range of previous research in this area, facilitating a fair and comprehensive comparison. Many other approaches exist, such as Scanorama [12] and Harmony [13], but most of these other models are not able to make predictions on new data. Many models that have been developed are designed to integrate the data you already have, instead of being trained on one set of existing data and later used to integrate new data. The models we chose to compare in this benchmark were selected for their ability to make predictions on new data and for their diverse architectures, allowing us to compare a broad range of leading methods in the field and further evaluate their generalizability.

In Figure 1b we see how variation between patients leads to separate clusters forming for data from cells of the same type (Figure 1 a). This is not desirable, and patient ID would be considered a "batch effect" in this scenario. In the ideal model, cells of the same type would cluster together such that the data becomes completely mixed in terms of patient ID. No variation within the dataset should be attributable to batch effects (e.g., patient, laboratory) from which a dataset originated since the data in our case only contains healthy individuals. This means that the patient variable should not contain significant biological variation. Attaining complete mitigation of batch effects poses a significant challenge, prompting the development of numerous new approaches to scRNA-Seq data integration, including CELLULAR.

1.2 Previous work

1.2.1 scRNA-Seq integration

There has been significant previous work for scRNA-Seq integration. In this work, we compare CELLULAR to the following studies: TOSICA [10], scGen [8], scANVI [7], and scVI [11]. These methods were selected due to their good generalization capabilities previously demonstrated on new data.

CELLULAR is most similar to TOSICA and scGen in that they are all non-Bayesian deep learning approaches. TOSICA uses transformers for cell type annotation from scRNA-Seq data; it takes advantage of pathway and regulon information for learning. scGen uses variational autoencoders to predict cellular responses to perturbations from scRNA-Seq data. On the other hand, scVI and scANVI both use hierarchical Bayesian models where the conditional distributions are learned via neural networks; these models have the advantage of efficient training, even for very large datasets. As we demonstrate in this study, although CELLULAR uses a simpler neural network architecture than TOSICA, scVI, and scANVI, it achieves superior performance to these models on generalizable data integration, and modestly better performance to scGen. The performance of all of these models on data integration is very dataset-dependent. We do not explore the effect of perturbations on cellular phenotype in this work.

1.2.2 Cell type annotation

There has also been notable previous work for cell type annotation from scRNA-Seq data. In this work, we compare CELLULAR to the following studies: TOSICA [10], scNym [14], Seurat [15], SciBet [16], and CellID [17].

These five models cover a wide range of approaches for solving cell type annotation tasks. scNym is an adversarial neural network trained in a semi-supervised way for cell type classification from a variety of single-cell omics data; this approach can also be used for novel cell type detection. Seurat combines dictionary learning and dataset sketching for the analysis of large scale single-cell transcriptomics data. SciBet uses multinomial logistic regression for cell annotation from scRNA-Seq data. Finally, CellID is a clustering-free multivariate statistical approach to cell annotation from single-cell omics data. In this setting, CELLULAR is most similar to the TOSICA and scNym models.

In the annotation benchmark presented herein, we evaluate the aforementioned models in two ways: firstly, by training the models using all gene information (in models that can handle this); and secondly, by filtering the gene set to use only highly variable genes (HVGs) in training. In prior work, only TOSICA used HVG filtering. Nevertheless, by doing HVG filtering on all models, we can allow a more fair comparison to CELLULAR.

1.3 Motivation

Many methods have been developed to perform dimensionality reduction, and can be divided into three main groups: statistical methods, manifold methods, and neural networks. [18] This study focused on developing a neural network approach to produce generalizable cell type representations while (1) minimizing batch effects and (2) promoting biological variation. Previously developed neural network algorithms for batch correction have used methods such as autoencoders (AEs) [19, 20], variational autoencoders (VAEs) [8], generative adversarial networks (GANs) [21], AEs combined with recurrent neural networks (RNNs) [22], contrastive learning [23, 24], and transformers [10].

The approach taken in CELLULAR is novel due to the combination of contrastive learning (Equation 8) with a cell type centroid loss (Equation 9), both specifically engineered for single-cell omics data integration tasks satisfying the two aforementioned constraints. This was motivated by the hypothesis that using a contrastive loss would aid in pushing together embeddings from scRNA-Seq data of the same cell type, while the centroid loss component would help maintain the relative positioning between cell type clusters. In doing so, we demonstrate that the latent space is more robust to the inherent noise in the data, minimizes batch effects, and leads to biologically meaningful embeddings.

2 Results

2.1 Establishing patient ID as the main source of variability

During the formulation of the loss function described in Equation 9, we assume that patient ID is the main source of unwanted variation when calculating the Euclidean distances between cell type clusters. To investigate whether or not this is a reasonable assumption, we evaluated the coefficient of variation (CV) for each dataset:

$$CV = \frac{\sigma}{\mu}, \tag{1}$$

where μ is the mean Euclidean distance across patients between two cell type clusters and σ is the standard deviation of the Euclidean distance.

The CV score is a metric that can vary between 0 and infinity; a lower value is preferable. A CV score of 1, for instance, means that the standard deviation equals the mean, indicating fairly large variability in the data. Ideally, one would observe CV scores equal to 0 for all pairs of cell type clusters, indicating that the normalized Euclidean distance between cell types *within each patient* is the same. However, in reality there exist many sources of error that can affect the CV calculation, and it is therefore unlikely to ever be 0, even if the underlying assumption is true.

In the datasets explored in this work, the mean CV score tends to be around, and slightly below, 0.1, indicating a fairly low CV score for all datasets (Figure 2a). This supports the assumption that the major source of unwanted variability when calculating the Euclidean distances between cell type clusters in this work is indeed patient ID. Any remaining variation in the data is attributed to less prominent batch effects.

2.2 Learning a generalizable scRNA-Seq embedding space

An ideal data integration algorithm would be able to enforce the biological variation in the data while minimizing variability caused by additional factors, such as patient ID, tissue, sequencing technology, and so on. To investigate how well models performs on this task, the *scIB* package can be used to benchmark models and evaluate their performance [9]. It does this by computing various metrics, each of which we min-max normalized across methods for each metric and fold. We compared multiple existing methods to CELLULAR in this study so as to gain a a better understanding of how

well our approach performs on data integration and learning a generalizable embedding space. As a control, we also fed the input data directly to the benchmark and obtained the corresponding metrics; we refer to this approach as "Unint" (short for "Unintegrated") throughout the study. We benchmarked all models on the bone marrow, PBMC, kidney, pancreas, and merged datasets, i.e., datasets where patient ID is expected to be the largest batch effect.

In Figure 2b we see how *scIB* ranks CELLULAR among the best performing models for each dataset. The performance of all models, on all metrics, and for each dataset can be explored in more detail in Appendices B.1, B.2, B.3, B.4, and B.5.

To get a visual understanding of how CELLULAR embeds different cell types, uniform manifold approximation and projection (UMAP) visualizations were prepared from one of the test folds of the kidney dataset following transformation with principle component analysis (PCA; Figures 2c,e) and CELLULAR (Figures 2d,f), respectively. What we observe is a significant reduction in batch effects when using CELLULAR for embedding the data versus when using PCA; the data points are better integrated within and across clusters in Figures 2d & 2f relative to the clusters in Figures 2c & 2e. This illustrates the effectiveness of CELLULAR in data integration and learning a meaningful embedding space where biological variation is preserved while variability due to other factors is not.

2.3 Cell type annotation

To demonstrate a downstream use case for the cellular embeddings learned by CELLULAR, we used the learned embeddings for the most ubiquitous classification task in molecular biology: cell type annotation. Using four widely-used datasets (Segerstolpe, MacParland, Baron, and Zheng68k), we compared the performance of our annotator to multiple other established models (Figure 2g). The metrics considered were accuracy, balanced accuracy, and F1 score.

We found that CELLULAR was among the top performing methods for cell type annotation across all datasets and metrics explored. Nevertheless, it was not straightforward to determine which model performed best overall. To estimate the "overall performance," we computed for each model the mean metric for each dataset and min-max normalized the means across models for each dataset. This was done so as to make the metrics comparable between datasets, and the results are visualized in Figure 2h using a box plot.

The benefit of using balanced accuracy for evaluating the annotation capabilities of models from scRNA-Seq data is that it takes into account the unbalanced nature of the data by giving greater weight to the classification accuracy on more commonly occurring cell types; on the other hand, ordinary accuracy weighs all cell labels equally. While one could argue that balanced accuracy is a more informative metric in this application, when calculating the average metric score, all metrics were weighted equally; that is, we computed the arithmetic mean of the average accuracy, average balanced accuracy, and average F1 score for each model and visualized the resulting "Average Score" in a bar plot (Figure 2h). Here, the overall performance of each model becomes clearer, and we find that scNym emerges as the top-performing model when utilizing highly variable genes (HVGs), with CELLULAR following closely.

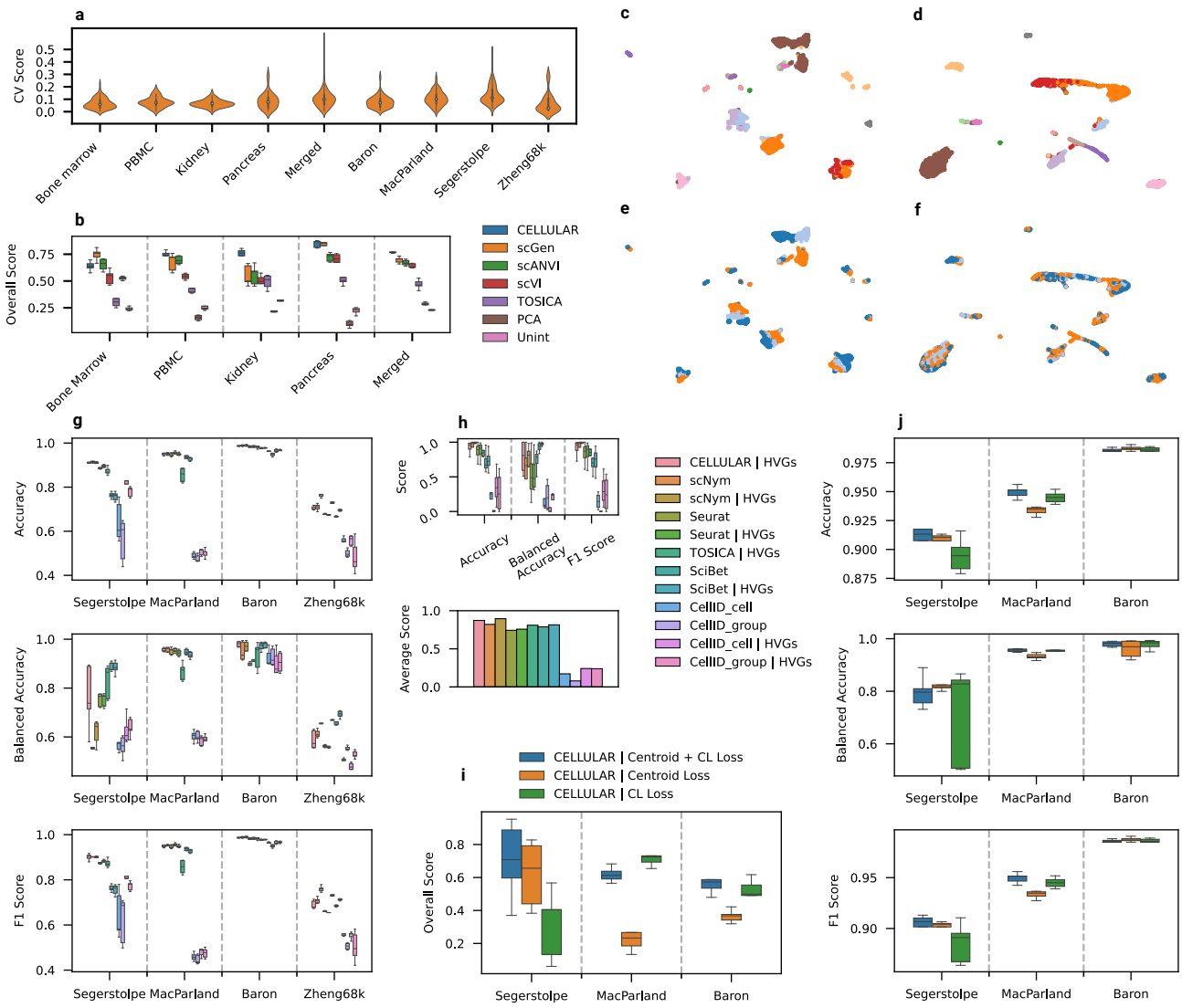

Fig. 2 (a) Density of CV scores (Equation 1) for each individual dataset explored in this work. (b) Five-fold cross-testing on the bone marrow, PBMC, kidney, pancreas, and merged dataset, where the overall scores were calculated as described in Luecken et al. [9] from min-max normalized *scIB* metrics. The following models were benchmarked: CELLULAR, scGen, scANVI, scVI, TOSICA, PCA, and Unint (Unintegrated). The input to all methods were the top 2000 HVGs. (c-f) UMAP of the first test fold for the kidney dataset when transformed by: (c) PCA, and colored by cell type, (d) CELLULAR, and colored by cell type, (e) PCA, and colored by patient ID, and (f) CELLULAR, and colored by patient ID. (g) Five-fold cross-testing on the Baron, MacParland, Segerstolpe, and Zheng68k datasets, where accuracy, balanced accuracy and F1 score are used as metrics. The following models were benchmarked: CELLULAR, scNym, Seurat, TOSICA, SciBet, and CellID. The input to the methods were either all genes or the top 2000 HVGs, specified by the "| HVGs" label. For the legend, see panel (h). (h) Box plots of min-max normalized mean metric values from panel (g) for the Baron, MacParland, Segerstolpe, and Zheng68k datasets. Below the box plots there is a bar plot of the average mean value of the metric scores for each model. (i) Five-fold cross-testing of CELLULAR on the Baron, MacParland, and Segerstolpe datasets, where the overall score was calculated as described in Luecken et al. [9] from min-max normalized *scIB* metrics. The three models benchmarked were CELLULAR using: the centroid + contrastive loss, the centroid loss alone, and the contrastive loss alone. The input to the models were the top 2000 HVGs. (j) Five-fold cross-testing of CELLULAR using different losses on the Baron, MacParland, and Segerstolpe, datasets. The input to the models were the top 2000 HVGs. Here, accuracy, balanced accuracy, and F1 score are shown. For the legend, see panel (i). In all box plots shown above, the median is displayed as the center line and the interquartile range as the box hinges, while the whiskers extend up to 1.5× the interquartile range.

While scNym and CELLULAR performed best and second-best overall, we note that in the original scNym study [14] the authors did not explore the use of HVGs; this is our contribution. In fact, without HVGs, scNym would not be among the top performers for cell type annotation. In general, we observed that the use of HVGs, rather than all genes, led to better performance for all models benchmarked, and as a result, the default approach in CELLULAR is to use HVGs. This distinction would make CELLULAR the best performing model under the "default" settings.

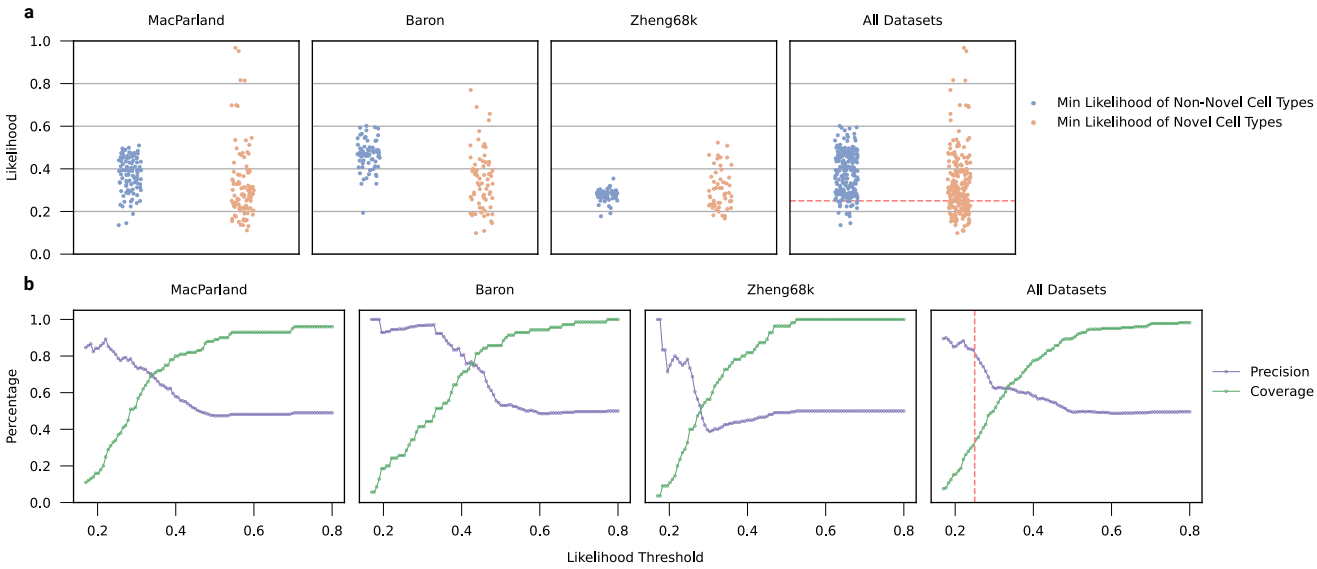

Fig. 3 (a) Jitter plot visualizing the distribution of the minimum likelihood attained by predictions made using CELLULAR for both non-novel and novel cell types. Each point represents the minimum likelihood observed for predictions of non-novel (known) and novel (unknown) cell types when one cell type was omitted from training across various test folds and datasets (MacParland, Baron, Zheng68k, and All Datasets). The red dotted line represents a likelihood threshold of 0.25; here, all data points falling below the line would contain novel cell types, according to the model. The *All Datasets* panel consists of all data points from the three other datasets combined into a single plot. Data points from MacParland, Baron, and Zheng68k are min-max normalized in relation to the maximum and minimum possible likelihood of each dataset, making datasets comparable in the *All Datasets* panel. (b) Line plot visualizing the relationship between the likelihood threshold used and the precision and coverage for detecting novel cell types using CELLULAR. The red dotted line represents a likelihood threshold of 0.25, which leads to a precision of 0.81 and a coverage of 0.32 for novel cell type detection across all datasets. The trade-off between precision and coverage is evident, allowing researchers the option to choose threshold values based on their desired model behavior for novel cell type detection.

2.4 Loss function analysis

CELLULAR uses a loss function specifically engineered for creating as generalizable an embedding space as possible from scRNA-Seq data (Equation 10). Here, we evaluate the performance of CELLULAR with each of the two main components of the

loss function individually, namely the contrastive loss (Equation 8) and centroid loss (Equation 9) terms. This comparative analysis (Figure 2j) serves to underscore the importance of using the complete loss function .

Through this analysis, we observed that using the cell type centroid loss alone (Equation 9) led to weak performance on the MacParland dataset, while using the contrastive loss alone (Equation 8) led to weak performance on the Segerstolpe dataset. The best performance on the accuracy, balanced accuracy, and F1 score (Figure 2j) was always observed when using the complete loss function; this observation holds for all three datasets explored (Segerstolpe, MacParland, and Baron). Similarly, the best performance on the overall score from the *scIB* benchmark was mainly observed when using the complete loss function (Figure 2i). To summarize, using both the contrastive and the centroid loss terms during model training led to the best and most consistent performance in CELLULAR across the three evaluated datasets.

2.5 Novel cell type detection

As seen in the previous sections, CELLULAR demonstrates promising performance for creating a generalizable embedding space for cell types, as well as good performance in cell type classification. An even more challenging application of the model would be to use it for novel cell type detection, i.e., identifying cell types that don't fit into any of the classes that the model has previously seen.

When it comes to classification tasks, it is well-known that neural networks generalize poorly to classes not seen during training due to their reliance on learned patterns from the training data, thus limiting their ability to recognize novel patterns corresponding to new classes. This limitation of neural networks can also be turned around and harnessed for novel cell type detection, where we can judge the novelty of a data point by setting a threshold on the likelihood of sampling any of the previously-seen class labels. If a prediction has a likelihood lower than this threshold for all classes, then it can be considered novel. Another way of putting it is that if the model is not particularly confident about sampling any of known cell types given a certain gene expression pattern, then it is possible that this pattern corresponds to a new type of cell. Note that by 'new' and 'novel' here, we mean that a cell type is new to the model, not necessarily new in the biological sense, although this last point would naturally be the ultimate proof of performance for novel cell type identification.

We investigated how picking different threshold values influenced the predictions made by the novel cell type detection model. A likelihood threshold value of 0.25 is illustrated in the *All Datasets* panel of Figure 3a. We found that this value led to high precision (0.81) while maintaining reasonable coverage (0.32). The impact of the threshold value on precision and coverage is further illustrated in Figure 3b. Our findings suggest that one can vary the threshold value for novel cell type detection depending on which metric one chooses to prioritize: precision, or coverage. The choice of 0.25 is used as an example, reflecting a reasonable compromise between coverage and precision, and favoring the attainment of high precision.

3 Methods

3.1 Model

CELLULAR consists of a feed-forward encoder that can be trained on scRNA-Seq data to learn a biologically-meaningful embedding space. Its general structure consists of 2 linear layers, each followed by layer normalization layer and a ReLU activation function. The encoder is designed to compress the input after each layer, ending with a final embedding space of dimension 100. Under the default settings, CELLULAR contains 2,558,600 learnable weights. The architecture is visualized in Figure 1d. CELLULAR uses the Adam optimizer for training, with a cosine warm-up scheduler. For each dataset, CELLULAR was trained on an NVIDIA A100 GPU using mini batches of size 256, where mini batches randomly sampled each epoch. Note that for both of the downstream tasks (cell annotation and novel cell type classification), the encoder weights are fixed. Only the classifier model weights are updated when training the model for cell type annotation, using the learned CELLULAR embeddings as input.

3.2 Data

Data collected from human participants were gathered from the following studies and tissues:

- Bone marrow: GSM3396161, GSM3396176, GSM3396184, GSM3396183 and GSM3396174. [25] Data available for 20 cell types and 5 patients.
- PBMC: GSE115189, GSM3665016, GSM3665017, GSM3665018, GSM3665019 and PBMC 10X data. [26, 27] Data available for 17 cell types and 6 patients.
- Pancreas: GSE81076, GSE85241, GSE86469 and E-MTAB-5061, gathered into one dataset by the Satija lab[28]. Data available for 9 cell types and 4 patients/sequencing methods. It is unclear if this data only comes from different sequencing methods, or if it also comes from different patients. For simplicity, we call this effect *patient ID*.
- Kidney: GSM4572192, GSM4572193 and GSM4572194. [29] Data available for 15 cell types and 3 patients.
- Merged: Bone marrow, PBMC, pancreas and kidney datasets all merged into one dataset consisting of 46 cell types and 18 patients.

These datasets were used to assess the models' capabilities in creating a generalizable embedding spaces from scRNA-Seq data. For exploring the models' cell type annotation capabilities, the following datasets were used:

- Baron: GSE84133. [30] Data available for 14 cell types and 4 patients.
- MacParland: GSE115469. [31] Data available for 20 cell types and 5 patients.
- Segerstolpe: E-MTAB-5061. [32] Data available for 15 cell types and 10 patients.

- Zheng68k: Data from the 'Fresh 68K PBMCs' section at 10xGenomics. [33] Data available for 11 cell types and 8 unique barcode identifiers.

After preprocessing, each of the aforementioned datasets contains the following number of cells: bone marrow – 14,779; PBMC – 22,027; kidney – 16,370; pancreas – 6,313; merged dataset – 59,489; Baron – 8,569; MacParland – 8,444; Segerstolpe – 3,514; and Zheng68k – 68,579.

3.3 Preprocessing

ScRNA-Seq data were preprocessed as follows:

1. Cells filtered by the number of unique genes a cell must reach for the cell to be considered for downstream processing. This was achieved by calculating the shifted logarithm $x = log(x+1)$ of the number of genes x and filtering out cells by setting a threshold on how many median absolute deviations (MAD) from the median a sample was allowed to be, where $MAD = median(|x - median(x)|)$. In this study the threshold was set to 5 MAD.
2. Cells filtered by the sequencing depth to ensure high quality samples. Similarly to the previous point, the shifted logarithm was calculated for the read counts and filtering was performed using a MAD value of 5.
3. Cells filtered by the proportion of mitochondrial counts. This was done to remove cells suspected of having broken membranes, which could potentially be dying cells. [34] As before, filtering was performed using MAD. The threshold was set to 5 MAD for almost all filtering, but was occasionally changed to a more appropriate value based on a visual assessment of the distribution. The MAD value was selected from a range between 5 and 10 based on visual assessment.
4. Cells filtered by the proportion of total counts corresponding to the top 20 genes with the highest number of counts per cell. As before, filtering was performed using MAD. The threshold was set to 5 MAD for almost all filtering, but was occasionally changed to a more appropriate value based on a visual assessment of the distribution. The MAD value was selected from a range between 3 and 5 based on visual assessment.
5. Genes filtered by number of unique cells expressing a gene that must be reached in order for a gene to be considered for downstream processing. In this study, genes that appeared in less than 20 cells were discarded.
6. Normalization performed by first normalizing the total number of counts so they become comparable between cells. This was achieved by first calculating the average number of counts per cell in the data $L = \frac{1}{N_c} \cdot \sum_{g,c} y_{g,c}$, where N_c is the number of cells and y is the counts matrix where g annotates the position of a gene and c the position of a cell, together specifying positions in y. Then, the sum of counts of each cell is divided by L, resulting in $S_c = \frac{1}{L} \cdot \sum_g y_{g,c}$. Next, it is good to transform the data so it resembles a normal distribution as much as possible. A shifted logarithm transform was shown to perform very

well in a recent benchmark and was therefore used in this study. [35] Normalization was performed by calculating $y_{norm_{g,c}} = log(\frac{y_{g,c}}{S_c} + 1)$, where y_{norm} is the normalized counts matrix.

7. Cell type annotation was performed using the *ScType* package. [36]
8. As a final step, one can select the top N most highly-variable genes (HVGs) in the processed data that will be used as input for the model to be trained. This can be achieved with the *highly_variable_genes* function from *Scanpy* using the *cell_ranger* flavor. [37] This method was also used in the Luecken *et al.* [9] benchmark to identify HVGs. *Scanpy* is an established tool for preprocessing single-cell omics data and is therefore used in this study. We used $N = 2000$.

A summary of the preprocessing pipeline described above is also illustrated in Figure 4a. It is worth noting that for Baron, Segerstolpe, and Zheng68k, only steps 5, 6 and 8 were performed since we assume that the quality of these samples are very high due to the vast popularity of using these datasets for benchmarking cell type annotators. For the MacParland dataset, only steps 5 and 8 were performed since it comes pre-normalized. All four of these datasets have already been annotated, such that step 7 was not required.

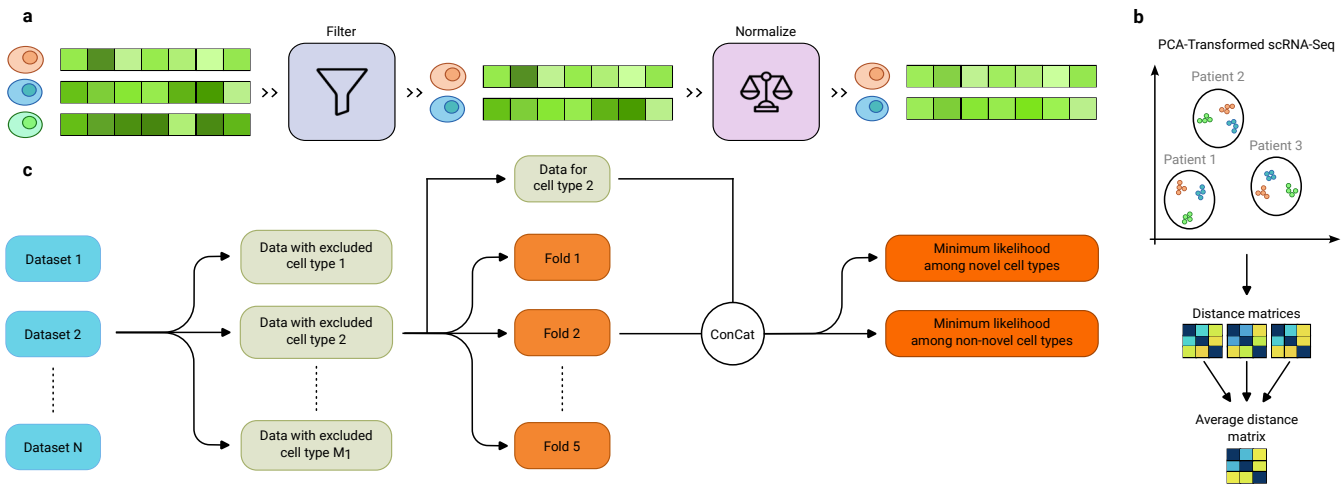

Fig. 4 (a) Summary of the preprocessing pipeline, starting with scRNA-Seq data from multiple samples. The data is then filtered to ensure sample quality, and normalized to make samples comparable. (b) Overview of steps for calculating the cell type centroid loss (Equation 1). PCA is used to reduce the dimensionality of the scRNA-Seq data. The Euclidean distance matrix is calculated between cell type clusters in different batches (i.e., patients). From this, the average distance matrix is calculated. (c) Framework for evaluating the likelihood thresholds for novel cell type detection. For each dataset, we begin by excluding one cell type at a time. Then, for each such data subset we use a five-fold split for training/testing. The data of the excluded cell type is concatenated to the test fold, acting as a "novel" cell type. Finally, the minimum likelihood produced by Model 1 is then aggregated for the non-novel and novel cell types separately.

3.4 Training

3.4.1 Contrastive learning

Models were trained using contrastive learning (CL). This is a technique where the model tries to produce outputs that are close to each other in space for augmented inputs of the same original input, called positive (+) input pairs, while being pushed away from all other inputs, called negative (-) input pairs. [38] To apply CL, one needs to define a suitable loss function. The *normalized temperature-scaled cross entropy loss* (NT-Xent; Equation 2) is a loss function that is appropriate for CL when there are multiple negative inputs. [39, 40]

$$\mathcal{L}_{NT-Xent} = -log(\frac{exp(sim(q, k_+)/\tau)}{exp(sim(q, k_+)/\tau) + \sum_{i=0}^{N} exp(sim(q, k_{-,i})/\tau)}), \tag{2}$$

where $sim()$ is the cosine similarity, τ represents the temperature parameter, N is the number of negative samples, q is the original input, k are augmented inputs where all $k_{-,i}$ are negative inputs such that $q \neq k_i$, and k_+ indicates the augmented version of the original input.

This learning strategy can be modified by instead stating that inputs belonging to the same label should strive to be as close to each other as possible while being far away from inputs of other labels, resulting in a semi-supervised approach. This is accomplished by modifying Equation 2 to be

$$\mathcal{L}_{CellType} = -\frac{1}{B}\sum_{b=0}^{B} log(\frac{\sum_{j=0}^{M} \alpha_b(k_j)}{\sum_{j=0}^{M} \alpha_b(k_j) + \sum_{i=0}^{N} \alpha_b(k_i)}), \tag{3}$$

where α_b is defined as $\alpha_b(x) = exp(sim(q_b, x)/\tau)$ and q_b is the current reference sample in the dataset, specified by the batch number b which consists of B samples. k_j are inputs of the same cell type as sample q_b while $q_b \neq k_j$ and $q_b \neq k_i$. k_i are inputs of all other cell type entries. M is the number of samples with the same label as q_b and N is the number of remaining samples. The loss calculation is performed for each sample q_b (i.e., for all data points in the batch), such that $\mathcal{L}_{CellType}$ is an expectation value. This loss function is called the *soft nearest neighbor loss*. [41]

Using the loss described in Equation 3, one can enforce the creation of cell type clusters in latent space, but this loss alone does not necessarily reduce batch effects. To do so, we define an additional loss function

$$\mathcal{L}_{BatchEffect} = -\frac{1}{B}\sum_{b=0}^{B}(log(\frac{\sum_{j=0,y_j=y_b}^{M} \alpha_b(k_j)}{\sum_{j=0,y_j=y_b}^{M} \alpha_b(k_j) + \sum_{i=0,y_i=y_b}^{N} \alpha_b(k_i)}))^{-1}, \tag{4}$$

where k_j are inputs of the same batch effect key (e.g., laboratory) as sample q_b while $q_b \neq k_j$ and $q_b \neq k_i$. k_i are inputs of all other batches. y_b is the cell type of q_b, y_j is the cell type of k_j, and y_i is the cell type of k_i, where $y_j = y_b$ and $y_i = y_b$ must

be fulfilled. Using this loss we promote the dispersion of data points of the same batch within each cell type cluster.

ScRNA-Seq data tends to be unbalanced in terms of cell types and batch effects. Hence, it can be advantageous to weight the contributions of each label type to the loss differently. Weights can be calculated as

$$\mathcal{W}_i = \frac{\frac{1}{N_i}}{\sum_j^M \frac{1}{N_j}}, \tag{5}$$

where $\mathcal{W}_i$ is the weight to be applied to the loss of cell type label i, N_i is the number of samples of label i and M is the total number of unique cell types present in the data. By calculating the weights in this way, we promote less common cell types to have a larger impact during training, which counters the label unbalance. Weights are also calculated and used for batch effects in the same manner by redefining N_i to be the number of samples of batch effect i and M as the number of unique batch effect keys.

Combining Equations 3 and 5, we arrive at the final cell type loss

$$\mathcal{L}_{CellType} = \sum_i^M \mathcal{W}_{CellType,i} \cdot \mathcal{L}_{CellType,i}, \tag{6}$$

where i is a specific cell type and M is the number of unique cell types in the dataset. Similarly, for batch effects we get

$$\mathcal{L}_{BatchEffect} = \sum_i^M \mathcal{W}_{BatchEffect,i} \cdot \mathcal{L}_{BatchEffect,i}, \tag{7}$$

where i is a specific type of batch effect and M is the number of unique batch effect keys.

Finally, the two loss terms (Equations 6 and 7) are combined to arrive at the final contrastive loss

$$\mathcal{L}_{CL} = 0.9 \cdot \mathcal{L}_{CellType} + 0.1 \cdot \mathcal{L}_{BatchEffect} \tag{8}$$

Here, the loss components are added together and weighted with 90% importance on the cell type loss and 10% on the batch effect loss. Having a small contribution from the batch effect loss is necessary so that the noise promotion does not take over, but it cannot be completely neglected. Nevertheless, these percentages can be adjusted depending on the research or project objectives.

3.4.2 Adding a cell type centroid loss component

Enforcing a meaningful relativistic positioning of cell type clusters in the latent space can help the model learn biologically meaningful embeddings. To achieve this desired clustering behavior, an additional term, referred to in this study as the *centroid loss*, was added to the final loss function.

To derive the centroid loss, the first step is to transform scRNA-Seq expression levels using PCA into a latent space of the same dimension as the intended output

of the model; in this study, this dimension is set to 100. PCA is used for its ability to extract dimensions responsible for significant variation in data, providing a lower-dimensional representation with reduced noise. From this latent space, the goal is to extract a distance matrix containing the Euclidean distances between each pair of cell type clusters' centroids (Figure 4b).

Assuming the main source of unwanted variation is due to the specified batch effect (patient ID in Figure 4b), one can expect the distances between cell type clusters within each individual batch of data to be similar. This assumption is expanded upon in Section 2.1. Calculating the average distance matrix across patients gives an approximation of the underlying relativistic positioning of cell types in the latent space. To obtain the relativistic distance matrix, one simply divides the average centroid distance matrix by the maximum value of this matrix. This is performed both on the PCA transformed space and then later on the model produced space. Ideally, the cell type clusters in the latent output space of the model should reflect this underlying relativistic positioning, which can be enforced by introducing the aforementioned centroid loss term, defined as

$$\mathcal{L}_{Centroid} = MSE(D_{Model}, D_{PCA}), \tag{9}$$

where MSE stands for mean squared error, D_{Model} is the relativistic centroid distance matrix produced by the model, and D_{PCA} is the same matrix but generated through dimensionality reduction of the scRNA-Seq via PCA. The workflow for calculating the centroid loss is further described in Appendix C. Notably, since not all cell types exist in each patient's data, some pairwise distances may not be possible to calculate. To account for this, these positions in the distance matrix are masked when evaluating Equation 9. The idea is that the existing relativistic distances from the reference data will be maintained in the latent space even after training the model.

By combining the contrastive learning loss (Equation 8) and the cell type centroid loss (Equation 9), we arrive at the final loss used in CELLULAR

$$\mathcal{L} = \mathcal{L}_{CL} + \mathcal{L}_{Centroid}. \tag{10}$$

This loss function was used to train all models developed in this work.

3.5 Evaluation

3.5.1 Data integration benchmark

To evaluate the scRNA-Seq integration performance of CELLULAR and all its variations explored in this study, it was compared to other leading methods for omics integration using the *scIB*[9] suite of established benchmarks. Performance of the models is evaluated in terms of (1) batch correction and (2) preservation of biological variation. This is evaluated using metrics such as *Normalized Mutual Information* (NMI), *Adjusted Rand Index* (ARI), *Average Silhouette Width* (ASW), *Isolated label Silhouette*, *Principal Component Regression* (PCR), *Graph Connectivity* (GC), *Isolated label F_1 score*, and *Cell-cycle Conservation* (CC), which can be effectively used

to evaluate batch correction and the conservation of biological information in the benchmarked models. All metrics are designed to fall within the range of 0 to 1, with 1 indicating better performance. Implementations of all metrics already exist in the *scIB* package developed by Luecken *et al.* [9] and were used in this study.

3.5.2 Producing cell type representation vectors

A promising application of the embedding space produced by the models is to create cell type representations. Since the model will return different outputs for different inputs (gene expression profiles) of the same cell type, a method is required to aggregate learned embeddings corresponding to the same cell type, such that there is a single embedding returned for each cell type label. Some options are: (1) calculating the centroid of each cell type cluster, (2) calculating the median of each output dimension for each cell type, or (3) calculating the medoid of each cell type cluster. All of these approaches have been implemented in CELLULAR with the intention of enabling future users of the model to effortlessly generate coherent cell type representations from scRNA-Seq data.

3.5.3 Cell annotation

Another application of the learned embedding space is to use it for classification tasks, such as cell type annotation. To explore this application, a feed-forward neural network was trained for cell type classification. This classifier accepts the learned CELLULAR embeddings as input and predicts cell types. Here, the output from the final layer of the classifier is softmaxed to return a likelihood for each cell type label. The model is trained by minimizing the cross-entropy between predicted and 'ground truth' class labels during training, using the *torch.nn.CrossEntropyLoss* implementation in *Pytorch*.[42] Because the embeddings learned by CELLULAR are generalizable, when training the classifier, only the classifier weights are updated, while the encoder weights are frozen. The model architecture for the cell type classifier is illustrated in Figure 1e.

Benchmarking of CELLULAR's cell type annotation capabilities was done by comparing to the following models: scNym [14], Seurat [15], TOSICA [10], SciBet [16], and CellID [17]. Three metrics were used to assess the performance of all models: accuracy, balanced accuracy, and F1 score. All three metrics are already implemented in the *scikit-learn* package as *sklearn.metrics.accuracy_score*, *sklearn.metrics.balanced_accuracy_score*, and *sklearn.metrics.f1_score*. [43] For *sklearn.metrics.f1_score*, the *average* parameter was set to "*weighted*".

The models developed in this study have a built-in flag for automatically optimizing the classifier hyperparameters associated using *Optuna* [44]. The model takes the data, splits it 80 : 20 for training and validation, and runs 100 *Optuna* trials to find optimum hyperparameters. This has been implemented as a default feature of the model since training the classifier is not computationally demanding. The following hyperparameters are optimized: the number of neurons in the first hidden layer, the number of neurons in the second hidden layer, the initial learning rate, and the dropout percentage.

3.5.4 Novel cell type prediction

Since a cell type annotator is trained using specific data, it can only recognize and predict the cell types present in its training dataset. If this model is later used to analyze data containing unfamiliar cell types, it will not be able to accurately identify these new types. Instead, it will mistakenly classify them as one of the known types it was trained on, leading to incorrect predictions. However, it is possible to identify whether a set of samples contains a previously unidentified cell type. One approach for detecting novel cell types is as follows:

1. Make predictions on data and retrieve the likelihood of each sample corresponding to a given cell type.
2. For each sample, min-max normalize the likelihoods across all possible cell types, where the maximum is a likelihood of 1.0, and the minimum is the inverse of the number of unique labels (cell types).
3. Extract the minimum likelihood out of the set of all predictions.
4. Compare this value to a pre-determined threshold (0.XX). A value below this threshold suggest that the sample could correspond to a novel cell type previously unseen in the data.

The intuition behind this approach is that a model which is not confident in any of the known labels (cell types) for a given sample implies that the sample is possibly a novel cell type, previously unseen.

To investigate whether CELLULAR can be used for this purpose, an experiment was performed where one cell type was left out of training at a time from each of the MacParland, Baron, and Zheng68k datasets, followed by a 5-fold split. This eventually creates $5 \cdot \sum_{i=1}^{N} \mathrm{M}_i$ total folds, where M_i is the number of cell types in dataset i and N is the number of datasets. The data of the removed cell type is then concatenated to the test fold, so that one can make predictions both on known cell types and novel (excluded from training) cell types. For each test fold, dataset, and cell type left out during training, the minimum likelihood of any non-novel prediction was calculated along with the minimum likelihood of any novel guess (Figure 4c).

4 Discussion

The model developed in this study displays impressive performance on two benchmarks. It was overall the best performing model on scRNA-Seq data integration, and the second best for cell type classification. On top of this, it also has the ability to identify the presence of novel cell types in data. Due to its robust performance on multiple tasks, we believe the developed model will be a useful tool for cell analysis to bioinformaticians and machine learning researchers alike.

4.1 Data augmentation strategies

Some augmentations that can be performed for scRNA-Seq data include randomly masking a specified percentage of gene expressions (i.e., making them zero) and adding Gaussian noise. The latter can be done by randomly replacing a specified percentage

of gene expressions with a value drawn from a Gaussian distribution that has been fit to the expression distribution of each gene in a given cell type.

The concept of artificially masking gene expressions aims to simulate the occurrence of errors during sequencing, leading to the dropout of gene expression. However, since gene expressions are typically dependent on expression of other genes, one can imagine that randomly removing expression levels, essentially removing known information, can make the model less prone to discover these connections. Hence, one could argue that this form of sc-omics data augmentation might not be desirable. Nevertheless it can also make the model better at generalizing. Depending on the needs of the user, this augmentation strategy may be worth investigating in future work.

Fitting a Gaussian distribution to the gene expression levels, then sampling noise from this Gaussian and adding it to the training data, is another potent augmentation strategy. This can be a good strategy for generating more artificial data for training, potentially making the model better at generalizing. However, for this to be a valid augmentation strategy, one must assume that the noise in gene expression within each cell type is minor, which is perhaps a strong assumption. This augmentation strategy would also be interesting to explore in future work.

4.2 Novel cell type detection

Detecting novel cell types in data is a difficult task. When looking at the precision and coverage of the Zheng68k dataset (Figure 3b) one can see how we would only achieve a high precision by sacrificing a lot of coverage. As this is not generally a worthwhile trade-off, this means the model would most often miss the presence of a novel cell type in the data. In contrast, if one looks at the Baron dataset (Figure 3b) one can achieve both a fairly high precision and coverage, making the model useful for novel cell type detection from this data. The key insight here is that the novel cell type detection capability of the model is highly dependent on the characteristics of the data. Consequently, selecting an optimal likelihood threshold that yields the best overall performance becomes challenging. An attempt at this was made by combining data points from all datasets and picking a likelihood threshold of 0.25 based on all data. This gave a high precision and a acceptable coverage. As such, we conclude that the model can serve as a valuable supplementary verification step within novel cell type detection pipelines, so as to minimize the risk in drawing conclusions based solely on this approach. In scenarios where the model detects something novel, CELLULAR can provide an extra layer of confidence in the presence of genuinely novel cell types. Conversely, if CELLULAR fails to detect any novel cell types, it is not guaranteed that there are no novel cell types present in the dataset.

4.3 The importance of HVGs

The best performing model for cell classification in this work was *scNym*, but it only outperforms CELLULAR when the input data is trimmed down to the 2000 most highly variable genes (HVGs), something which the authors did not do in their original study. [14] In fact, the use of HVGs improved the performance of all models benchmarked in this study. As such, we want to emphasize the importance of high-quality

inputs to machine learning methods for scRNA-Seq data integration, as these methods are quite sensitive to how the raw data is processed. Researchers working on the development of new methods for cell annotation and related tasks with scRNA-Seq data should consider limiting the model input data to only HVGs to promote better generalization and utility in their tools.

4.4 Future applications

4.4.1 Cell type representations

The focal point of CELLULAR is its ability to produce a generalizable embedding space. By generalizable, we mean that this embedding space can go on to be used in a variety of downstream tasks, such as cell type annotation or novel cell type detection, as demonstrated in this study. Another application which was implemented but not explored is the use of the embedding space to define cell representation vectors for each unique cell type. Three options were implemented to achieve this: calculating the centroid, median, or medoid of each cell type cluster, and using this to define a representative vector for a given cell class. However, the performance of using these representations in other machine learning applications was not investigated and is left for future research. One possible use for these representations would be to identify quantitative structure-activity relationships (QSAR) conditioned on cell type, or to provide molecular generative models with cell type information in drug discovery efforts; in this way, both QSAR and generative models would be able to make predictions conditioned on the target cell type, something which is currently lacking in the broader data-driven drug discovery literature.

4.4.2 Multi-modal deep learning approaches

4.4.3 Combining scRNA-Seq data with cell images

Due to its ability to effectively cluster data, contrastive learning can also be used to learn a joint embedding space from scRNA-Seq information and cell images, e.g., *Cell Painting* images. *Cell Painting* is a type of high-content, image-based assay that can be used to discern information about cellular states. [45] For instance, it is been shown that *Cell Painting* images can be used to discern cell health under different CRISPR perturbations, where these perturbations are designed to target genes related to known biological pathways and are thus likely to induce morphological changes in the cell [46]. *Cell Painting* images are used as an unbiased representation of quantitative information regarding the condition or status of cells[47]. As a result, they have the potential to enhance the generalizability of the latent space by exposing CELLULAR to more information related to unique cell types, and to improve the utility of the model for downstream applications. For this to work, however, one would require many *Cell Painting* images of the same cell types for which scRNA-Seq data is available, which currently presents a barrier. Due to the lack of a public *Cell Painting* image database with diverse cell types overlapping with those investigated herein, this option was not further explored. Nevertheless, we recognize it as an interesting direction for future research.

4.4.4 Integrated multi-omics

Yet another way of modifying CELLULAR would be to introduce other forms of omics data, such as single-cell assay for transposase-accessible chromatin sequencing (scATAC-Seq) data, into the model. The benefit of using contrastive learning in this case would be that one version of CELLULAR could use scRNA-Seq data as input while another version learns from scATAC-Seq. By using multiple different types of data, such as scRNA-Seq, scATAC-Seq, and *Cell Painting* images, the model may leverage complementary information from multiple data sources and thus learn a more meaningful latent space. Furthermore, the use of multiple data sources can have a regularizing effect on the model, leading to improved generalizability. As a result, the learned embedding space may more accurately represent how cell types are related to and distinct from one another. Just as it led to breakthroughs in text-conditioned image generation, the use of multi-modal deep learning in molecular biology can help transform our understanding of cellular development.

4.5 Limitations

As with most models, CELLULAR has a few limitations. One limitation is the time required for training, which presents a problem if one wants to train a model using a large number of cells. The biggest dataset used for training in this study was the Zheng68k dataset, containing around 68k cells. While it was manageable to train with this amount of data using our setup (for a dataset of approx. 50k cells: $\sim$30 mins to train the encoder, $\sim$100 mins to train the classifier), larger datasets containing on the order of millions of cells could present a problem. The reason for the relatively slow training is the complex loss function used by CELLULAR. A large amount of computations are needed per data point, creating a bottle neck. Developing methods to reduce the computational complexity of the model could thus enhance its scalability and applicability to larger datasets.

Secondly, the performances of the different models benchmarked in this work, including CELLULAR, are very dataset dependent, meaning that it is possible to have observed a different ranking for the models if different datasets were used. In an attempt to mitigate these, we purposefully used a breadth of different datasets, spanning different tissues and laboratories. Nevertheless, these datasets provide just a glimpse into the space of what can be done with scRNA-Seq and other omics technologies, and researchers should be careful when extending any conclusions to any datasets that differ significantly from the ones used herein.

5 Conclusion

In this study, we present a robust new method for leveraging scRNA-Seq data to learn a generalizable embedding space for cells using contrastive learning. The model, CELLULAR, is effective at mitigating batch effects while enforcing biologically relevant information. The approach was developed into an open-source Python package, with multiple downstream applications, including cell type annotation and novel cell type

detection, also integrated into the package. This comprehensive approach to cell representation learning can empower other researchers to explore this method for learning meaningful cellular embedding spaces from other single-cell transcriptomics datasets.

This work advances the field of bioinformatics via the following key contributions:

- We introduce a new model for cell representation learning from scRNA-Seq data, and demonstrate that it outperforms the current leading methods in data integration. The model also demonstrates robust performance on cell type annotation and novel cell type detection tasks.
- By using a carefully crafted contrastive loss, we demonstrate that our model is robust to batch effects while preserving biologically relevant information.
- We show how pruning preprocessed scRNA-Seq data to only the most highly variable genes (HVGs) improves their utility for cell annotation, regardless of the model or dataset used.

Finally, we offer suggestions for future directions in cell representation learning using this approach, highlighting the significant promise we see in multi-modal deep learning, particularly with cell images and other single-cell omics methods.

Supplementary information. The reproducibility code for this work is available at: github.com/LeoAnd00/CELLULAR_reproducibility. Most of the data for this work is available at: doi.org/10.5281/zenodo.12169799. CELLULAR has also been implemented as a stand-alone Python package, available at: github.com/LeoAnd00/CELLULAR.

Acknowledgements. The authors acknowledge computing resources from the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725. RM thanks the Wallenberg AI, Autonomous Systems, and Software Program (WASP) for funding.

Declarations

Funding

RM thanks the Wallenberg AI, Autonomous Systems, and Software Program (WASP) for funding.

Conflict of interest

The authors acknowledge no conflicting or competing interests.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Data availability

Data used in this work is available at: doi.org/10.5281/zenodo.12169799.

Materials availability

Not applicable.

Code availability

The reproducibility code for this work is available at: github.com/LeoAnd00/CELLULAR_reproducibility. CELLULAR has also been implemented as a stand-alone Python package, available at: github.com/LeoAnd00/CELLULAR.

Author contributions

RM conceived and supervised the study. RM and LA discussed and designed the machine learning models developed in this study, and designed experiments together. LA carried out all data collection, data preprocessing, model training, and computational analysis. LA wrote all code presented in this study. Both authors discussed the results. LA prepared an initial draft of the manuscript, and both authors contributed to the writing and final structure of the manuscript.

Appendix A Data preprocessing

A.1 Bone marrow dataset

UMAP of the bone marrow dataset showing all 9 cell types (Figure A1a) and 4 patient IDs / sequencing methods (Figure A1b). It is worth noting that the batch effects are not that severe for this dataset.

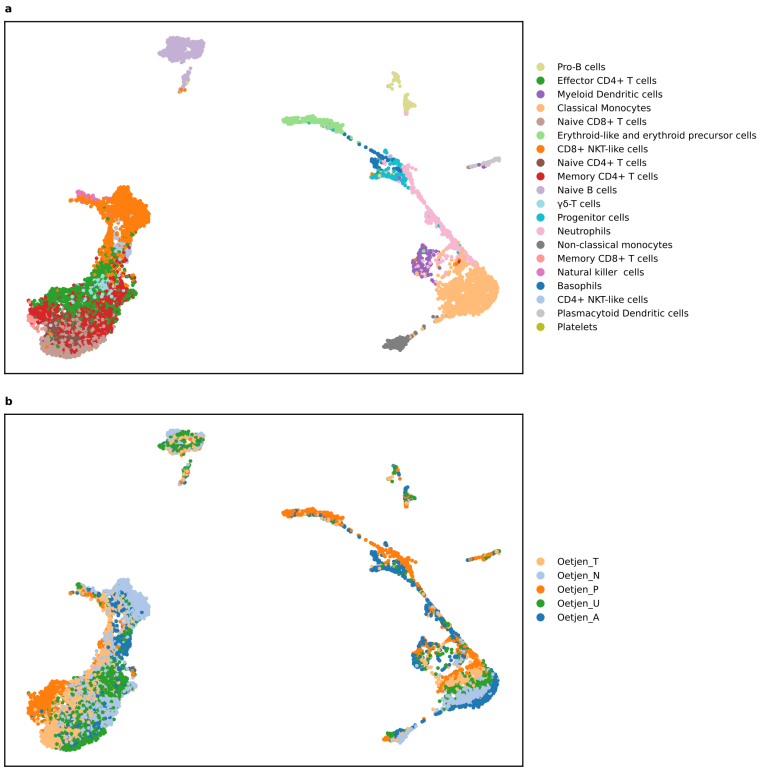

Fig. A1 **a** UMAP visualization of cell type clusters in the bone marrow dataset. **b** UMAP highlighting batch effect caused by patient ID method.

A.2 PBMC dataset

UMAP of the PBMC dataset showing all 9 cell types (Figure A2a) and 4 patient IDs / sequencing methods (Figure A2b). The batch effects are much more severe for this dataset compared to the bone marrow dataset.

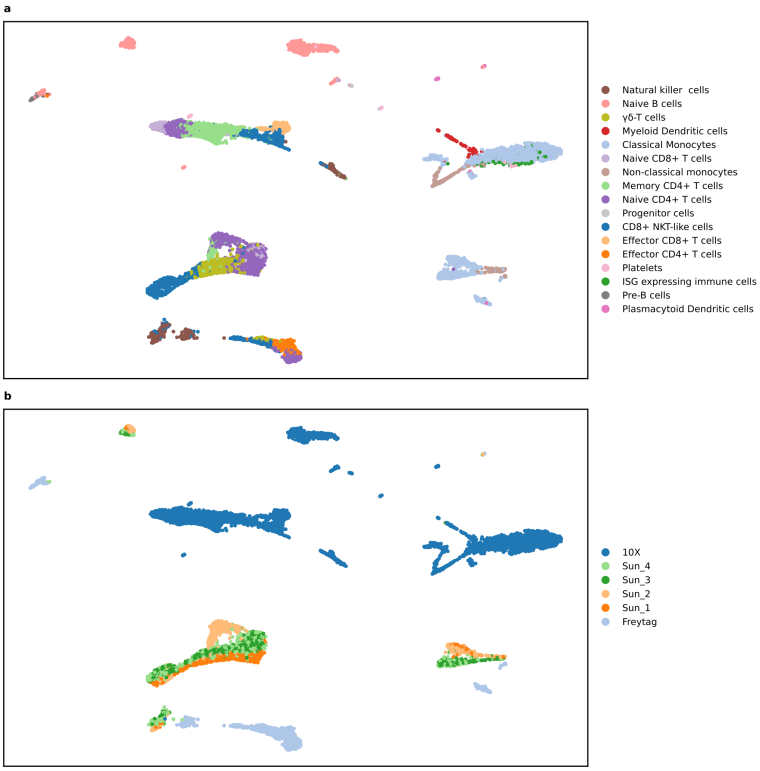

Fig. A2 **a** UMAP visualization of cell type clusters in the PBMC dataset. **b** UMAP highlighting batch effect caused by patient ID method.

A.3 Pancreas dataset

UMAP of the pancreas dataset showing all 9 cell types (Figure A3a) and 4 patient IDs / sequencing methods (Figure A3b). The batch effects are very severe for this dataset.

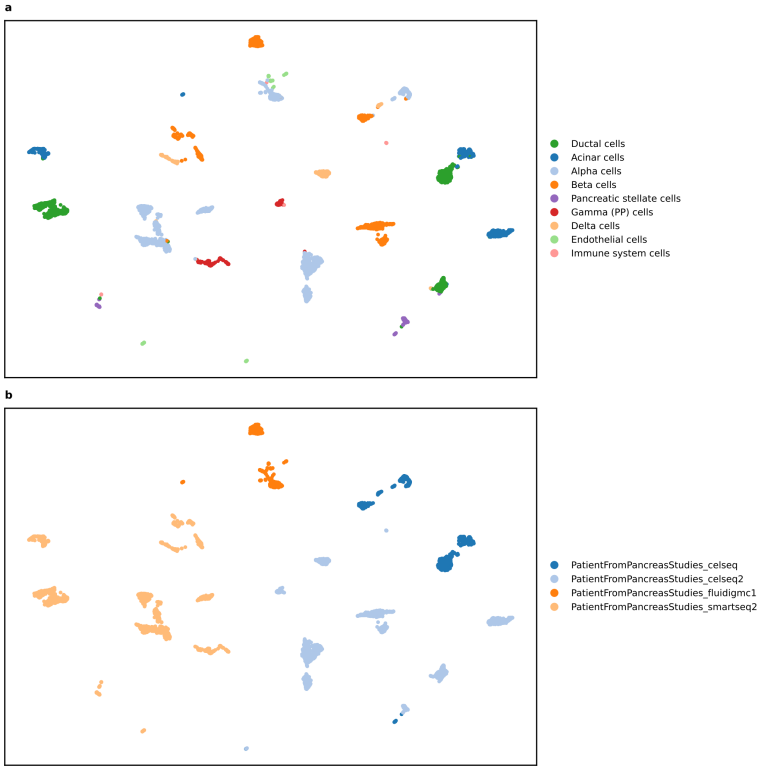

Fig. A3 **a** UMAP visualization of cell type clusters in the pancreas dataset. **b** UMAP highlighting batch effect caused by patient ID / sequencing method.

A.4 Kidney dataset

UMAP of the kidney dataset showing all 15 cell types (Figure A4a) and 3 patient IDs (Figure A4b). There are visible batch effects.

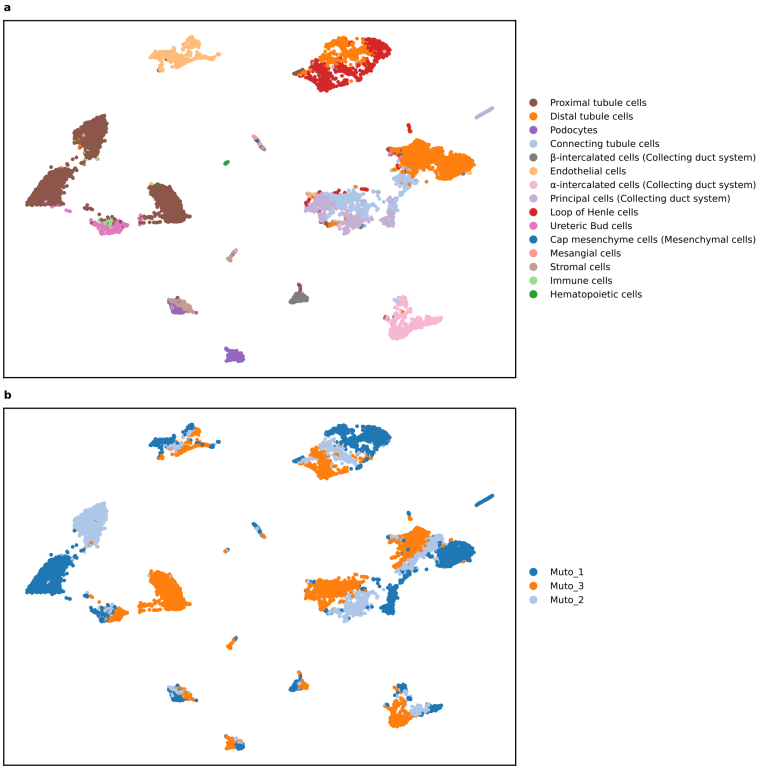

Fig. A4 **a** UMAP visualization of cell type clusters in the kidney dataset. **b** UMAP highlighting batch effect cause by patient ID.

A.5 Merged dataset

The entire merged dataset was visualized via UMAP and colored according to all 46 cell types (Figure A5a) and 18 batch effect elements (Figure A5b).

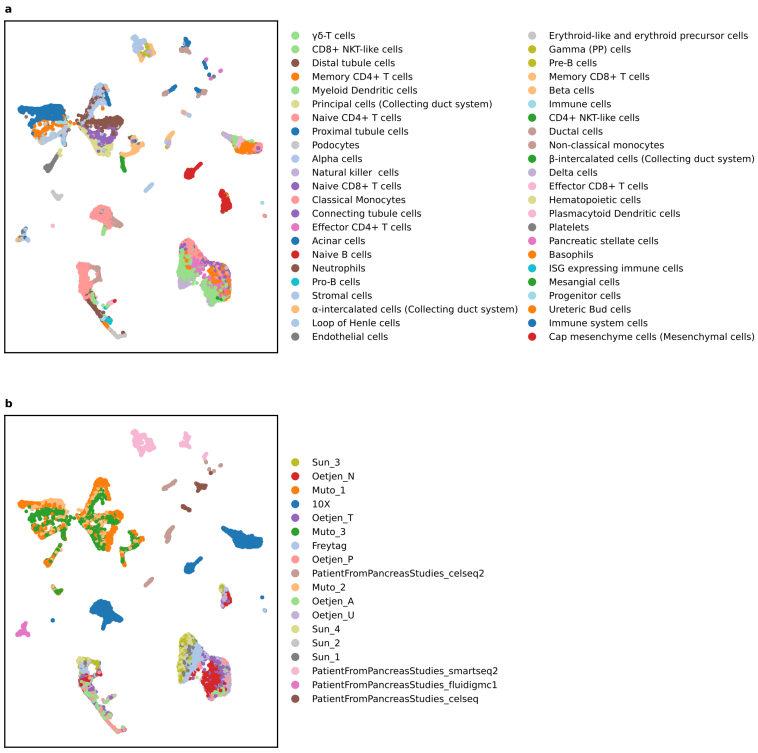

Fig. A5 **a** UMAP visualization of cell type clusters in the full merged dataset. **b** UMAP highlighting the batch effects when merging the bone marrow, PBMC, kidney, and pancreas datasets.

A.6 Baron dataset

UMAP of the Baron dataset showing all 14 cell types (Figure A6a) and 4 patient IDs (Figure A6b).

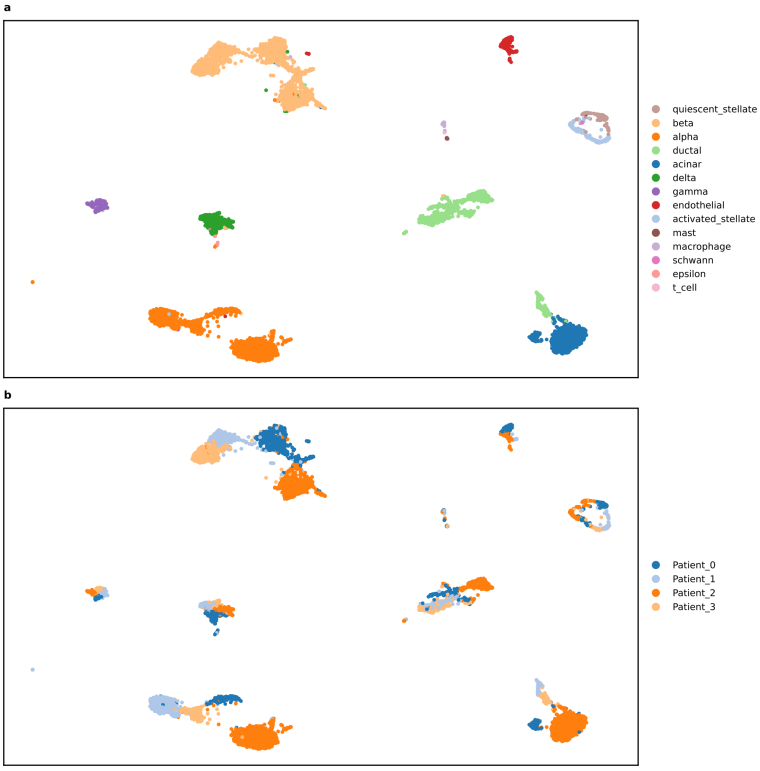

Fig. A6 **a** UMAP visualization of cell type clusters in the Baron dataset. **b** UMAP highlighting batch effect cause by patient ID.

A.7 MacParland dataset

UMAP of the MacParland dataset showing all 20 cell types (Figure A7a) and 5 patient IDs (Figure A7b).

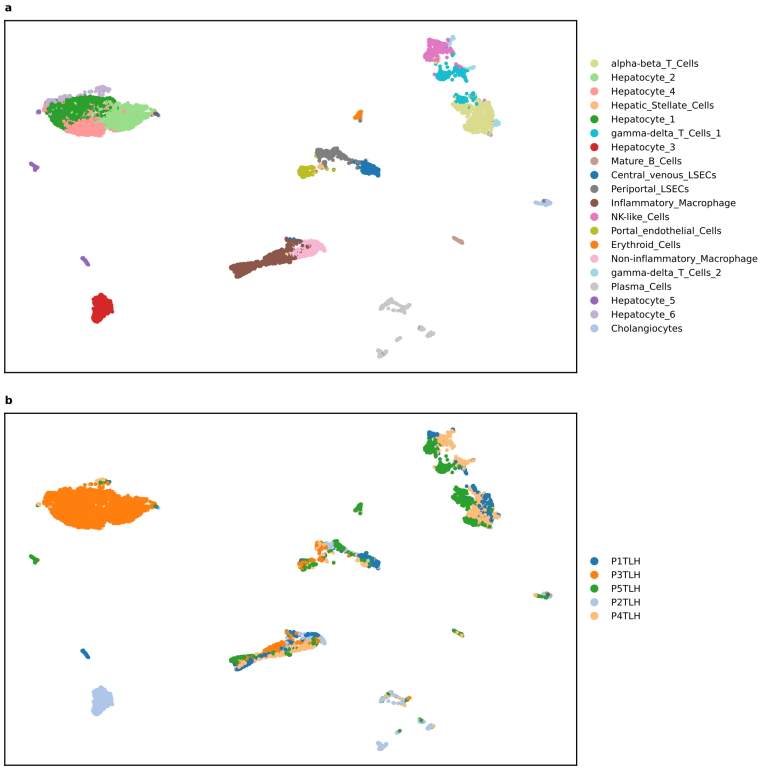

Fig. A7 **a** UMAP visualization of cell type clusters in the MacParland dataset. **b** UMAP highlighting batch effect cause by patient ID.

A.8 Segerstolpe dataset

UMAP of the Segerstolpe dataset showing all 15 cell types (Figure A8a) and 10 patient IDs (Figure A8b).

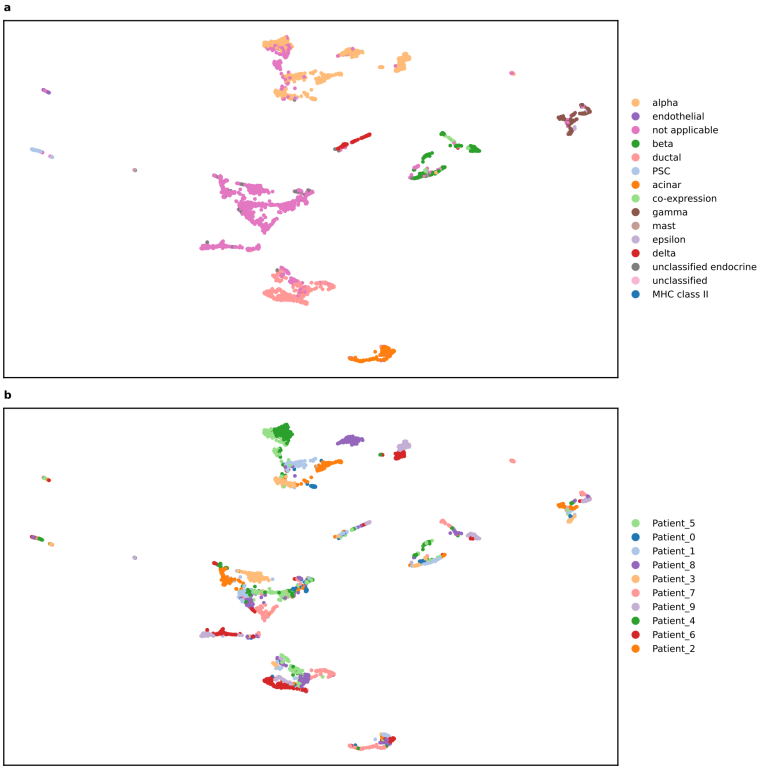

Fig. A8 **a** UMAP visualization of cell type clusters in the Segerstolpe dataset. **b** UMAP highlighting batch effect cause by patient ID.

A.9 Zheng68k dataset

UMAP of the Zheng68k dataset showing all 11 cell types (Figure A9a) and 8 unique barcode IDs (Figure A9b).

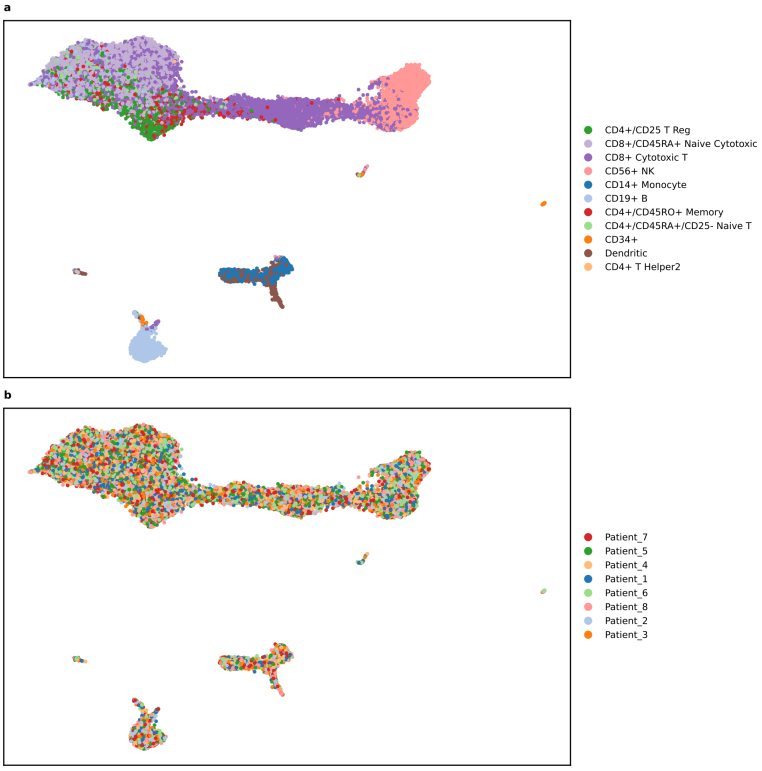

Fig. A9 **a** UMAP visualization of cell type clusters in the Zheng68k dataset. **b** UMAP highlighting batch effect cause by the barcode IDs.

Appendix B Generalizable scRNA-Seq embedding space benchmark

B.1 Bone marrow dataset

The results for all 12 metrics, including overall bio, overall batch, and overall scores for all models when trained on the bone marrow dataset are shown below.

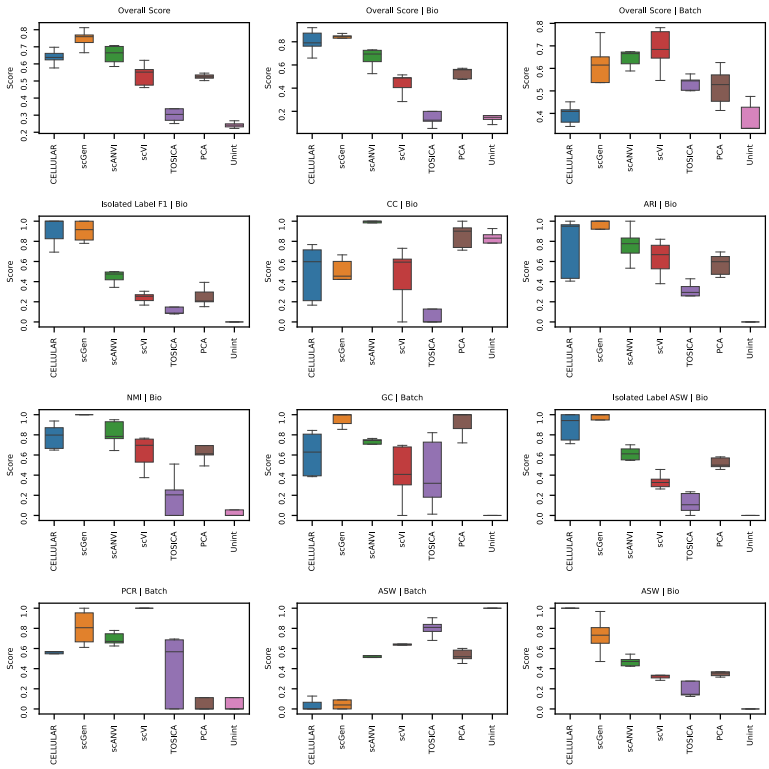

Fig. B10 Five-fold cross-testing on the bone marrow dataset, where all metrics are calculated. The following models are used in the benchmark: Unint (Unintegrated), PCA, TOSICA [10], scVI [11], scANVI [7], scGen [8], and CELLULAR. The input to the methods are the top 2000 HVGs. Box plots show the median displayed as the center line, the interquartile range as the box hinges, and the whiskers extending up to 1.5 times the interquartile range.

B.2 PBMC dataset

The results for all 12 metrics, including overall bio, overall batch, and overall scores for all models when trained on the PBMC dataset are shown below.

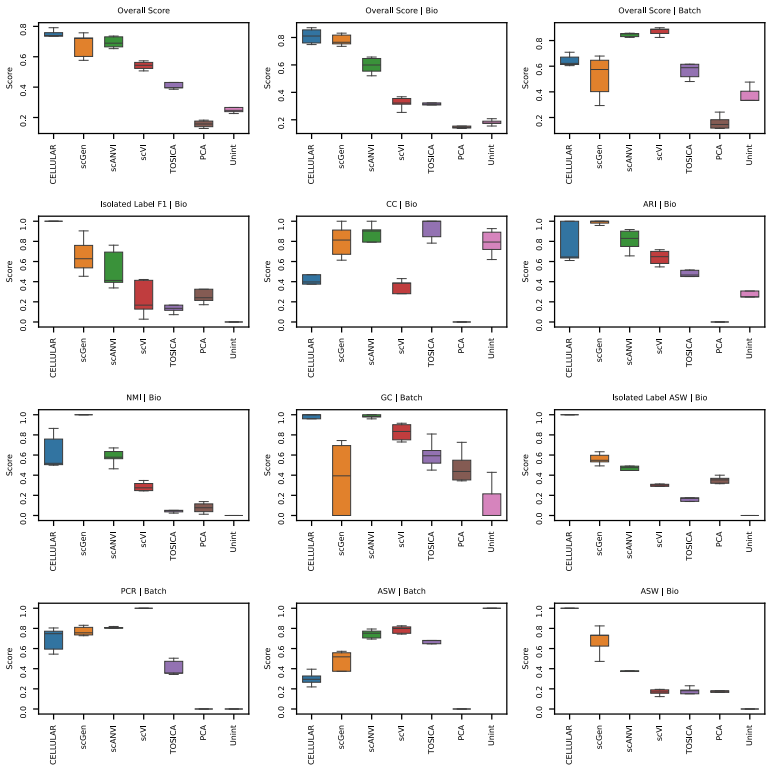

Fig. B11 Five-fold cross-testing on the PBMC dataset, where all metrics are calculated. The following models are used in the benchmark: Unint (Unintegrated), PCA, TOSICA [10], scVI [11], scANVI [7], scGen [8], and CELLULAR. The input to the methods are the top 2000 HVGs. Box plots shows the median displayed as the center line, the interquartile range as the box hinges, and the whiskers extending up to 1.5 times the interquartile range.

B.3 Kidney dataset

The results for all 12 metrics, including overall bio, overall batch, and overall scores for all models when trained on the kidney dataset are shown below.

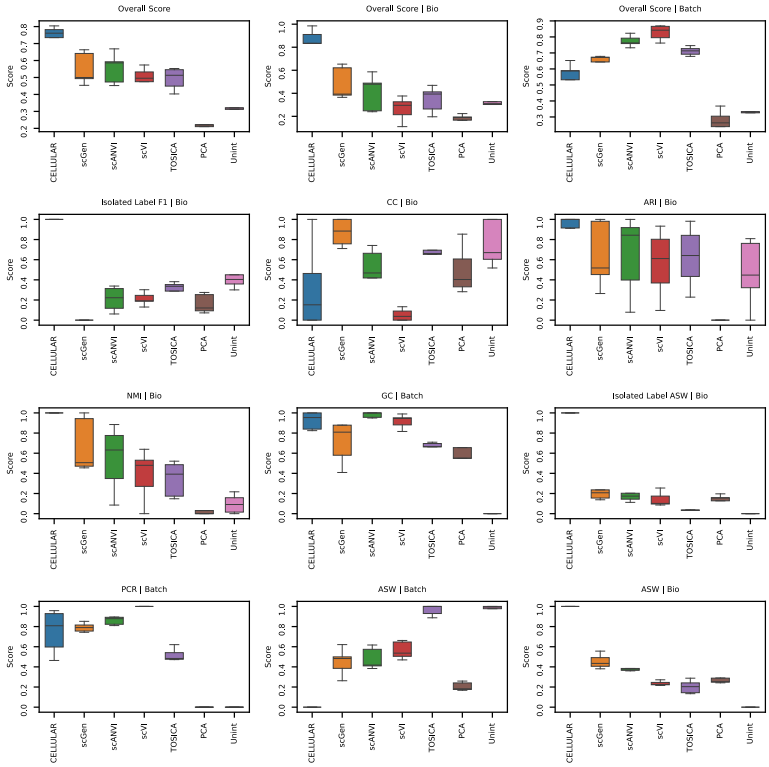

Fig. B12 Five-fold cross-testing on the kidney dataset, where all metrics are calculated. The following models are used in the benchmark: Unint (Unintegrated), PCA, TOSICA [10], scVI [11], scANVI [7], scGen [8], and CELLULAR. The input to the methods are the top 2000 HVGs. Box plots shows the median displayed as the center line, the interquartile range as the box hinges, and the whiskers extending up to 1.5 times the interquartile range.

B.4 Pancreas dataset

The results for all 12 metrics, including overall bio, overall batch, and overall scores for all models when trained on the pancreas dataset are shown below.

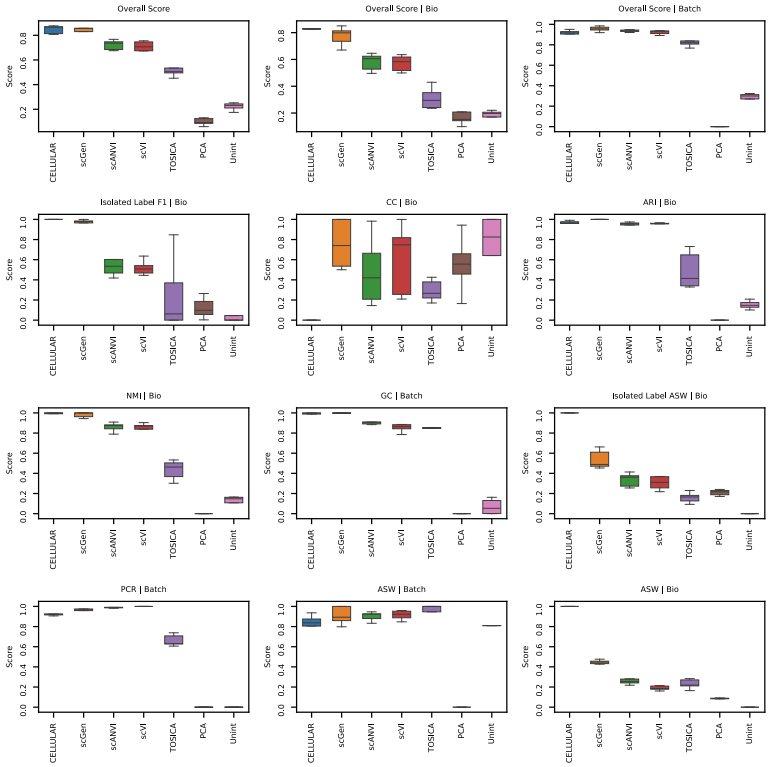

Fig. B13 Five-fold cross-testing on the pancreas dataset, where all metrics are calculated. The following models are used in the benchmark: Unint (Unintegrated), PCA, TOSICA [10], scVI [11], scANVI [7], scGen [8], and CELLULAR. The input to the methods are the top 2000 HVGs. Box plots shows the median displayed as the center line, the interquartile range as the box hinges, and the whiskers extending up to 1.5 times the interquartile range.

B.5 Merged dataset

The results for all 12 metrics, including overall bio, overall batch, and overall scores for all models when trained on the merged dataset are shown below.

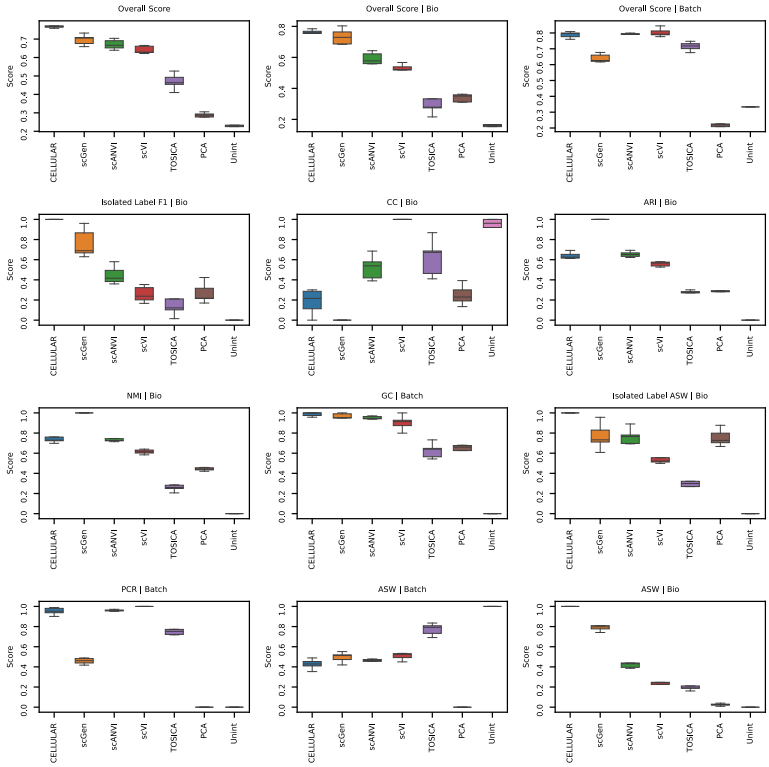

Fig. B14 Five-fold cross-testing on the merged dataset, where all metrics are calculated. The following models are used in the benchmark: Unint (Unintegrated), PCA, TOSICA [10], scVI [11], scANVI [7], scGen [8], and CELLULAR. The input to the methods are the top 2000 HVGs. Box plots shows the median displayed as the center line, the interquartile range as the box hinges, and the whiskers extending up to 1.5 times the interquartile range.

Appendix C Centroid loss algorithm

Below, Algorithm 1 outlines of the algorithm for calculating the cell type centroid loss introduced in Section 3.4.2 (note that the algorithm is split over two pages).

Algorithm 1 Cell Type Centroid Loss Algorithm

```
// Step 1: Reference Distance Matrix
Input: adata // Annotated data object
Define: X = adata.X // scRNA-Seq data
Define: X_PCA = PCA(X, dim = 100)
Define: A = "cell_type"; B = "patientID"
Define: unique_A = unique(adata.obs[A])
Define: unique_B = unique(adata.obs[B])
Define: centroids = {}
for β in unique_B do
    for α in unique_A do
        Define mask: mask = (adata.obs[B] == β) & (adata.obs[A] == α)
        Apply mask and calculate centroid:
centroids[(β, α)] = mean(X_PCA[mask], axis = 0)
    end for
end for
Initialize distance matrix:
D_ref = zeros(len(unique_A), len(unique_A))
for α_1 in unique_A do
    for α_2 in unique_A do
        δ = []
        for β in unique_B do
            δ_β = EuclideanDistance(centroids[(β, α_1)], centroids[(β, α_2)])
            δ.append(δ_β)
        end for
        Compute mean Euclidean distance across batch effect elements:
D_ref[α_1, α_2] = mean(δ)
    end for
end for
Normalize: D_ref = D_ref / max(D_ref)
```

Algorithm 2 *

Continued from previous page:

// Step 2: Calculate Loss For Mini-Batch During Training
Input mini-batch: $adata$
Define: $X = adata.X$ *// Model-generated embedding space from scRNA-Seq data*
Define: $A =$ "$cell_type$"
Define: $B =$ "$patientID$"
Define: $unique_A = unique(adata.obs[A])$
Define: $unique_B = unique(adata.obs[B])$
Define: $centroids = \{\}$
for α in $unique_A$ **do**
    Define mask: $mask = adata.obs[A] == \alpha$
    Apply mask and calculate centroid:
$centroids[\alpha] = mean(X[mask], axis = 0)$
end for
Initialize distance matrix:
$D = zeros(len(unique_A), len(unique_A))$
for α_1 in $unique_A$ **do**
    for α_2 in $unique_A$ **do**
        Euclidean distance between cell type clusters:
$\delta = EuclideanDistance(centroids[\alpha_1], centroids[\alpha_2])$
        $D[\alpha_1, \alpha_2] = \delta$
    end for
end for
Normalize: $D = \frac{D}{max(D)}$
Extract relevant part from reference distance matrix:
$D_{relevant_ref} = D_{ref}[unique_A, unique_A]$
Normalize: $D_{relevant_ref} = \frac{D_{relevant_ref}}{max(D_{relevant_ref})}$
$\mathcal{L} = MSE(D, D_{relevant_ref})$ **return** $\mathcal{L}$

Appendix D Attributions

Icons from flaticon.com were used to prepare some of the figures in this work.

References

[1] Stein, C.M., Weiskirchen, R., Damm, F., Strzelecka, P.M.: Single-cell omics: Overview, analysis, and application in biomedical science. Journal of cellular biochemistry **122**(11), 1571–1578 (2021) https://doi.org/10.1002/jcb.30134

[2] Cuomo, A.S., Nathan, A., Raychaudhuri, S., MacArthur, D.G., Powell, J.E.: Single-cell genomics meets human genetics. Nature Reviews Genetics, 1–15 (2023) https://doi.org/10.1038/s41576-023-00599-5

[3] Lee, J., Hyeon, D.Y., Hwang, D.: Single-cell multiomics: technologies and data analysis methods. Experimental & Molecular Medicine **52**(9), 1428–1442 (2020) https://doi.org/10.1038/s12276-020-0420-2

[4] Vandereyken, K., Sifrim, A., Thienpont, B., Voet, T.: Methods and applications for single-cell and spatial multi-omics. Nature Reviews Genetics, 1–22 (2023) https://doi.org/10.1038/s41576-023-00580-2

[5] Pasquini, G., Arias, J.E.R., Schäfer, P., Busskamp, V.: Automated methods for cell type annotation on scrna-seq data. Computational and Structural Biotechnology Journal **19**, 961–969 (2021) https://doi.org/10.1016/j.csbj.2021.01.015

[6] Lähnemann, D., Köster, J., Szczurek, E., McCarthy, D.J., Hicks, S.C., Robinson, M.D., Vallejos, C.A., Campbell, K.R., Beerenwinkel, N., Mahfouz, A., *et al.*: Eleven grand challenges in single-cell data science. Genome biology **21**, 1–35 (2020) https://doi.org/10.1186/s13059-020-1926-6

[7] Xu, C., Lopez, R., Mehlman, E., Regier, J., Jordan, M.I., Yosef, N.: Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. Molecular systems biology **17**(1), 9620 (2021) https://doi.org/10.15252/msb.20209620

[8] Lotfollahi, M., Wolf, F.A., Theis, F.J.: scgen predicts single-cell perturbation responses. Nature methods **16**(8), 715–721 (2019) https://doi.org/10.1038/s41592-019-0494-8

[9] Luecken, M.D., Büttner, M., Chaichoompu, K., Danese, A., Interlandi, M., Müller, M.F., Strobl, D.C., Zappia, L., Dugas, M., Colomé-Tatché, M., *et al.*: Benchmarking atlas-level data integration in single-cell genomics. Nature methods **19**(1), 41–50 (2022) https://doi.org/10.1038/s41592-021-01336-8

[10] Chen, J., Xu, H., Tao, W., Chen, Z., Zhao, Y., Han, J.-D.J.: Transformer for one stop interpretable cell type annotation. Nature Communications **14**(1), 223 (2023) https://doi.org/10.1038/s41467-023-35923-4

[11] Lopez, R., Regier, J., Cole, M.B., Jordan, M.I., Yosef, N.: Deep generative modeling for single-cell transcriptomics. Nature methods **15**(12), 1053–1058 (2018) https://doi.org/10.1038/s41592-018-0229-2

[12] Hie, B., Bryson, B., Berger, B.: Efficient integration of heterogeneous single-cell transcriptomes using scanorama. Nature biotechnology **37**(6), 685–691 (2019) https://doi.org/10.1038/s41587-019-0113-3

[13] Korsunsky, I., Millard, N., Fan, J., Slowikowski, K., Zhang, F., Wei, K., Baglaenko, Y., Brenner, M., Loh, P.-r., Raychaudhuri, S.: Fast, sensitive and accurate integration of single-cell data with harmony. Nature methods **16**(12), 1289–1296 (2019) https://doi.org/10.1038/s41592-019-0619-0

[14] Kimmel, J.C., Kelley, D.R.: scnym: Semi-supervised adversarial neural networks for single cell classification. bioRxiv, 2020–06 (2020) https://doi.org/10.1101/gr.268581.120

[15] Hao, Y., Stuart, T., Kowalski, M.H., Choudhary, S., Hoffman, P., Hartman, A., Srivastava, A., Molla, G., Madad, S., Fernandez-Granda, C., *et al.*: Dictionary learning for integrative, multimodal and scalable single-cell analysis. Nature biotechnology **42**(2), 293–304 (2024) https://doi.org/10.1038/s41587-023-01767-y

[16] Li, C., Liu, B., Kang, B., Liu, Z., Liu, Y., Chen, C., Ren, X., Zhang, Z.: Scibet as a portable and fast single cell type identifier. Nature communications **11**(1), 1818 (2020) https://doi.org/10.1038/s41467-020-15523-2

[17] Cortal, A., Martignetti, L., Six, E., Rausell, A.: Gene signature extraction and cell identity recognition at the single-cell level with cell-id. Nature biotechnology **39**(9), 1095–1102 (2021) https://doi.org/10.1038/s41587-021-00896-6

[18] Gunawan, I., Vafaee, F., Meijering, E., Lock, J.G.: An introduction to representation learning for single-cell data analysis. Cell Reports Methods **3**(8) (2023) https://doi.org/10.1016/j.crmeth.2023.100547

[19] Li, X., Wang, K., Lyu, Y., Pan, H., Zhang, J., Stambolian, D., Susztak, K., Reilly, M.P., Hu, G., Li, M.: Deep learning enables accurate clustering with batch effect removal in single-cell rna-seq analysis. Nature communications **11**(1), 2338 (2020) https://doi.org/10.1038/s41467-020-15851-3

[20] Amodio, M., Van Dijk, D., Srinivasan, K., Chen, W.S., Mohsen, H., Moon, K.R., Campbell, A., Zhao, Y., Wang, X., Venkataswamy, M., *et al.*: Exploring single-cell data with deep multitasking neural networks. Nature methods **16**(11), 1139–1145 (2019) https://doi.org/10.1038/s41592-019-0576-7

[21] Xu, Z., Luo, J., Xiong, Z.: scsemigan: a single-cell semi-supervised annotation and dimensionality reduction framework based on generative adversarial network. Bioinformatics **38**(22), 5042–5048 (2022) https://doi.org/10.1093/bioinformatics/btac652

[22] Deng, Y., Bao, F., Dai, Q., Wu, L.F., Altschuler, S.J.: Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning. Nature methods **16**(4), 311–314 (2019) https://doi.org/10.1038/s41592-019-0353-7

[23] Han, W., Cheng, Y., Chen, J., Zhong, H., Hu, Z., Chen, S., Zong, L., Hong, L., Chan, T.-F., King, I., *et al.*: Self-supervised contrastive learning for integrative single cell rna-seq data analysis. Briefings in Bioinformatics **23**(5), 377 (2022) https://doi.org/10.1093/bib/bbac377

[24] Xu, Y., Das, P., McCord, R.P.: Smile: mutual information learning for integration

of single-cell omics data. Bioinformatics **38**(2), 476–486 (2022) https://doi.org/10.1093/bioinformatics/btab706

[25] Oetjen, K.A., Lindblad, K.E., Goswami, M., Gui, G., Dagur, P.K., Lai, C., Dillon, L.W., McCoy, J.P., Hourigan, C.S.: Human bone marrow assessment by single-cell rna sequencing, mass cytometry, and flow cytometry. JCI insight **3**(23) (2018) https://doi.org/10.1172/jci.insight.124928

[26] Freytag, S., Tian, L., Lönnstedt, I., Ng, M., Bahlo, M.: Comparison of clustering tools in r for medium-sized 10x genomics single-cell rna-sequencing data. F1000Research **7**, 1297 (2018) https://doi.org/10.12688/f1000research.15809.2

[27] Sun, Z., Chen, L., Xin, H., Jiang, Y., Huang, Q., Cillo, A.R., Tabib, T., Kolls, J.K., Bruno, T.C., Lafyatis, R., *et al.*: A bayesian mixture model for clustering droplet-based single-cell transcriptomic data from population studies. Nature communications **10**(1), 1649 (2019) https://doi.org/10.1038/s41467-019-09639-3

[28] Integration and Label Transfer. https://satijalab.org/seurat/archive/v3.0/integration.html. Accessed: 2024-06-19

[29] Muto, Y., Wilson, P.C., Ledru, N., Wu, H., Dimke, H., Waikar, S.S., Humphreys, B.D.: Single cell transcriptional and chromatin accessibility profiling redefine cellular heterogeneity in the adult human kidney. Nature communications **12**(1), 2190 (2021) https://doi.org/10.1038/s41467-021-22368-w

[30] Baron, M., Veres, A., Wolock, S.L., Faust, A.L., Gaujoux, R., Vetere, A., Ryu, J.H., Wagner, B.K., Shen-Orr, S.S., Klein, A.M., *et al.*: A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure. Cell systems **3**(4), 346–360 (2016) https://doi.org/10.1016/j.cels.2016.08.011

[31] MacParland, S.A., Liu, J.C., Ma, X.-Z., Innes, B.T., Bartczak, A.M., Gage, B.K., Manuel, J., Khuu, N., Echeverri, J., Linares, I., *et al.*: Single cell rna sequencing of human liver reveals distinct intrahepatic macrophage populations. Nature communications **9**(1), 4383 (2018) https://doi.org/10.1038/s41467-018-06318-7

[32] Segerstolpe, Å., Palasantza, A., Eliasson, P., Andersson, E.-M., Andréasson, A.-C., Sun, X., Picelli, S., Sabirsh, A., Clausen, M., Bjursell, M.K., *et al.*: Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes. Cell metabolism **24**(4), 593–607 (2016) https://doi.org/10.1016/j.cmet.2016.08.020

[33] Zheng, G.X., Terry, J.M., Belgrader, P., Ryvkin, P., Bent, Z.W., Wilson, R., Ziraldo, S.B., Wheeler, T.D., McDermott, G.P., Zhu, J., *et al.*: Massively parallel digital transcriptional profiling of single cells. Nature communications **8**(1), 14049 (2017) https://doi.org/10.1038/ncomms14049

[34] Heumos, L., Schaar, A.C., Lance, C., Litinetskaya, A., Drost, F., Zappia, L., Lücken, M.D., Strobl, D.C., Henao, J., Curion, F., et al.: Best practices for single-cell analysis across modalities. Nature Reviews Genetics, 1–23 (2023) https://doi.org/10.1038/s41576-023-00586-w

[35] Ahlmann-Eltze, C., Huber, W.: Comparison of transformations for single-cell rna-seq data. Nature Methods, 1–8 (2023) https://doi.org/10.1038/s41592-023-01814-1

[36] Ianevski, A., Giri, A.K., Aittokallio, T.: Fully-automated and ultra-fast cell-type identification using specific marker combinations from single-cell transcriptomic data. Nature communications **13**(1), 1246 (2022) https://doi.org/10.1038/s41467-022-28803-w

[37] Wolf, F.A., Angerer, P., Theis, F.J.: SCANPY: large-scale single-cell gene expression data analysis. Genome biology **19**, 1–5 (2018) https://doi.org/10.1186/s13059-017-1382-0

[38] Jaiswal, A., Babu, A.R., Zadeh, M.Z., Banerjee, D., Makedon, F.: A survey on contrastive self-supervised learning. Technologies **9**(1), 2 (2020) https://doi.org/10.3390/technologies9010002

[39] Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607 (2020). https://doi.org/10.48550/arXiv.2002.05709 . PMLR

[40] Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. Advances in neural information processing systems **29** (2016)

[41] Frosst, N., Papernot, N., Hinton, G.: Analyzing and improving representations with the soft nearest neighbor loss. In: International Conference on Machine Learning, pp. 2012–2020 (2019). https://doi.org/10.48550/arXiv.1902.01889 . PMLR

[42] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc., ??? (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[43] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

[44] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2623–2631 (2019). https://doi.org/10.1145/3292500.3330701

[45] Bray, M.-A., Singh, S., Han, H., Davis, C.T., Borgeson, B., Hartland, C., Kost-Alimova, M., Gustafsdottir, S.M., Gibson, C.C., Carpenter, A.E.: Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. Nature protocols **11**(9), 1757–1774 (2016) https://doi.org/10.1038/nprot.2016.105

[46] Way, G.P., Kost-Alimova, M., Shibue, T., Harrington, W.F., Gill, S., Piccioni, F., Becker, T., Shafqat-Abbasi, H., Hahn, W.C., Carpenter, A.E., *et al.*: Predicting cell health phenotypes using image-based morphology profiling. Molecular biology of the cell **32**(9), 995–1005 (2021) https://doi.org/10.1091/mbc.E20-12-0784

[47] Caicedo, J.C., Cooper, S., Heigwer, F., Warchal, S., Qiu, P., Molnar, C., Vasilevich, A.S., Barry, J.D., Bansal, H.S., Kraus, O., *et al.*: Data-analysis strategies for image-based cell profiling. Nature methods **14**(9), 849–863 (2017) https://doi.org/10.1038/nmeth.4397