Supplementary

Methods

Data and data preprocessing

1) hECA-10M dataset for model pre-training

To facilitate the pre-training of scMulan, we meticulously curated a hECA-10M dataset drawn from seven organs with sufficient cell numbers and high-quality cell annotations available in the Human Ensemble Cell Atlas (hECA) [1]. These organs included the brain, blood, bone marrow, heart, liver, lung, and thymus. Our data curation process involved the comprehensive collection of data from the original research papers, followed by the standardization of metadata. We then harmonized the naming conventions for cell type annotations by mapping them onto a unified framework known as the unified Human Atlas Framework (uHAF) [1].

For each dataset we collected, we standardized gene symbols using a list of approved symbols from the HUGO Gene Nomenclature Committee (HGNC). Subsequently, we performed data normalization and log1p transformation. After aggregating all the datasets, we identified the top 2,000 highly variable genes across the entire dataset using the scanpy package [2]. These 2,000 highly variable genes were consistently used as features across all datasets in this study. Finally, we randomly split this comprehensive dataset into two parts, with 90% of the cells designated for scMulan's pre-training and the remaining 10% reserved for validation.

2) Datasets for cell-type annotation tasks

For the cell-type annotation experiments, we selected three datasets that were not included in the hECA-10M dataset. These datasets included a heart cell dataset from Simonson et al. [3], a liver dataset from Suo et al. [4], and a bone marrow dataset from He et al. [5], referred to as Simonson2023, Suo2022, and AHCA_BoneMarrow, respectively. For the intestine dataset used in the fine-tuning experiments, we selected the intestine cells from the Human Cell Landscape [6] and named it Intestine_HCL. We followed similar preprocessing steps as those employed in the construction of the hECA-10M dataset. These steps included metadata standardization, gene symbol uniformity, normalization, and log1p transformation. For cell-type labels, we mapped the original annotations onto the organ-specific uHAF derived from hECA and provided the mapping relationship in Supplementary file 1.

For training cellTypist and fine-tuning scGPT and Geneformer, we randomly sampled 200,000 cells from the hECA-10M dataset. This data size was chosen to ensure that the fine-tuning process could be completed within a 24-hour period.

3) Datasets for data integration Task

For the Lung integration experiment, we collected data from the benchmark paper scIB [7] and followed similar preprocessing steps as mentioned previously to uniform the gene symbols. We used the original batch information and cell type annotation to evaluate the integration performance

For the COVID-19 integration experiment, we downloaded the dataset from the link provided by [8], which was also used in the scGPT paper [9]. Since the provided cell type annotations had already been unified, we conducted no further processing on the metadata and annotations. We simply selected the same 2,000 genes that were used in scMulan as the features for this dataset.

Model details

scMulan is a Transformer decoder-only model [10]. It comprises 20 Transformer decoder blocks, each with 20 attention heads, and a hidden layer dimension of 1120. It has additional embedding layers for entity and value embedding, and prediction heads for entity and value prediction. The entity embedding layer takes gene entities as input and projects the entities into the embedding dimension of 1120. The value embedding layer is similar to

scGPT, which takes genes' expression as input, bins the expression into discrete levels, and projects the expression levels into the same embedding dimension as entities. The prediction heads predict the subsequent entities and values in the c-sentences, at each time step of the decoder.

During the training phase, scMulan employs an autoregressive approach, effectively utilizing contextual information to generate output sequences. Furthermore, the model is implemented on the PyTorch 2.0 platform, incorporating Flash Attention technology [11] to optimize computational efficiency during training and enhance training speed. We pretrain scMulan for 245,000 steps on hECA-10M. The batch size is set to be 40, with gradient accumulation increasing to 320. This setup equates to processing roughly 320,000 words (tokens) per batch. In total, this batch size and number of steps corresponds to pre-training on 78B words. The pretraining process took 192 hours on four Nvidia-A800 GPUs.

In the inference phase, scMulan introduces the KV cache technique, which effectively reduces the computational requirements for inference.

It supports extensive 2k-context input. The parameter of scMulan reaches 386 million.

Downstream tasks

1) Zero-shot cell type annotation

Zero-shot annotation requires the generalization ability of the foundation model. Here we used unseen external datasets to evaluate the zero-shot annotation performance. We benchmarked scMulan against scGPT [9] and Celltypist [12], without additional tuning on these new datasets. The external datasets including Simonson2023 [3], Suo2022 [4], and AHCA_BoneMarrow [5] are described in the supplementary materials. We evaluated cell type annotation performance based on three classification metrics, accuracy, weighted precision, and weighted-F1 score, computed by the scikit-learn package [13]. As the models are pre-trained or fine-tuned on hECA-10M, their output predicted labels would include all kinds of cell types from various organs, but the predicted labels may only contain few cells. Thus, precision and F1 scores were weighted by the number of cells per cell type. Besides, we relaxed the predictions as correct if they matched a subclass of the true label. For instance, a cell classified as a CD8 T cell by the model was counted as correct if the true label was a T cell or a Lymphoid cell.

In this experiment, we used scMulan's generated fine-grained cell types as prediction, prompted by the input cell's gene expression (W_gs) followed by a '<cell type annotation>' entity.

2) Cell type annotation through fine-tuning

To test scMulan's capability for domain transferring, we fine-tuned scMulan to extend the cell type prediction on an unseen tissue, the intestine, not covered in the hECA-10M. We used the Intestine_HCL_55k [6] dataset in this experiment, split into training, and testing sets in a 9:1 ratio. We compared the performance of scMulan with Celltypist and scGPT on the testing set, after fine-tuning on the training set. Additionally, we assessed scMulan's transfer learning effectiveness across various training sample sizes, from 0% to 100%. The evaluation metrics are accuracy, precision, and macro-F1 score, as in this experiment the predicted labels and true labels are from the same context.

We fine-tuned scMulan only using the '<cell type annotation>' task entity prompted c-sentences from the new dataset, following the same cell language modeling optimization as pre-training.

3) Zero-shot batch integration

Foundation models are supposed to provide robust cell embeddings zero-shot without applying batch correction fine-tuning on datasets. We benchmarked zero-shot batch integration of scMulan on two datasets, Covid-19

containing 18 batches and Lung containing 16 batches. Detailed descriptions of these datasets are provided in the supplementary materials. We compared the integration performance of scMulan with 9 methods, BBKNN [14], Harmony [15], Seurat v3 [16], FASTMNN [17], scVI [18], and scANVI [18], fine-tuned scGPT (scGPT_finetune), pre-trained scGPT without fine-tuning (scGPT_zeroshot) and pre-trained 6-layer Geneformer without fine-tuning (Geneformer_zeroshot). We computed biological conservation metric (AvgBIO) and batch correction metric (AvgBATCH) suggested by scGPT paper. The AvgBIO metric is the average of metrics including normalized mutual information, adjusted Rand index, and average silhouette width of cell types. The AvgBATCH metric is the average of average silhouette width of batches and graph connectivity. We calculated all the metrics with scib-metrics package [19].

For the methods including BBKNN, Harmony, FASTMNN, scVI, scANVI and Seurat v3, we followed tutorials of these methods with default parameters. For Geneformer_zeroshot, we used the pre-trained 6-layer model and followed 'extract_and_plot_cell_embeddings' tutorial without further hyperparameter selection. For scGPT_zeroshot and scGPT_finetune, we followed 'Tutorial_Integration' with the given parameters.

For the purpose of generating embeddings, we extracted cell representations from the final transformer layer of scMulan. This process involved inputting a c-sentence composed of a cell's gene expression profile followed by the task entity '<cell type annotation>'.

4) Conditional cell generation

To evaluate scMulan's capacity for conditional generation, we established various organ and cell type pairings as joint conditions under which the model should generate cell expression profiles. We chose organs including the heart, lung, brain, blood, bone marrow, and liver, and for each, we randomly generated 3,000 cells, each assigned a cell type corresponding to the particular organ. This process yielded a total of 18,000 organ-cell type pairs and their associated cells generated by scMulan.

For comparison, we sampled an equivalent number of real cells from hECA-10M, maintaining the same proportion of cell types. We visualized both the generated cells and the sampled cells using UMAP for a direct comparison. Furthermore, we compared the generated and real dataset by assessing the distributions of mean expression (ME) of genes across cells and the number of expressed genes (EN) via Q-Q plots.

The organ-cell type pairs were converted into c-sentences, appended with the task-specific prompt '<cell generation>' to direct scMulan for conditional cell generation. For example, the c-sentence '(Heart,0), (T cell, 0), (<cell generation>,0)' would prompt scMulan to generate a T cell specific to the heart.

5) Model explanation for marker gene retrieval

We employed gradient-based explanation methods to determine the importance of each input feature, particularly gene entities, in the model's prediction of cell types. This was accomplished by analyzing the partial derivatives of the predicted cell type with respect to the entity embedding of each input gene. Specifically, we utilized the Gradient x Input method described in [19] to calculate attribution scores for each gene within a c-sentence under the cell type prediction task. We then identified and recorded the top 5 genes based on their attribution scores for each cell. Subsequently, we compiled the 10 most frequently occurring genes within these top attribution scores for each cell type, referred to as saliency genes. The results are illustrated in a bubble plot, highlighting genes that potentially serve as markers for the respective cell types.

Reference

1. Chen S, Luo Y, Gao H, Li F, Chen Y, Li J, et al. hECA: The cell-centric assembly of a cell atlas. iScience. 2022;25:104318.

2. SCANPY: large-scale single-cell gene expression data analysis | Genome Biology [Internet]. [cited 2023 Nov 8]. Available from: https://link.springer.com/article/10.1186/s13059-017-1382-0

3. Simonson B, Chaffin M, Hill MC, Atwa O, Guedira Y, Bhasin H, et al. Single-nucleus RNA sequencing in ischemic cardiomyopathy reveals common transcriptional profile underlying end-stage heart failure. Cell Reports. 2023;42:112086.

4. Suo C, Dann E, Goh I, Jardine L, Kleshchevnikov V, Park J-E, et al. Mapping the developing human immune system across organs. Science. 2022;eabo0510.

5. He S, Wang L-H, Liu Y, Li Y-Q, Chen H-T, Xu J-H, et al. Single-cell transcriptome profiling of an adult human cell atlas of 15 major organs. Genome Biology. 2020;21:294.

6. Construction of a human cell landscape at single-cell level | Nature [Internet]. [cited 2023 Nov 8]. Available from: https://www.nature.com/articles/s41586-020-2157-4

7. Luecken MD, Büttner M, Chaichoompu K, Danese A, Interlandi M, Mueller MF, et al. Benchmarking atlas-level data integration in single-cell genomics. Nat Methods. 2022;19:41–50.

8. Lotfollahi M, Naghipourfar M, Luecken MD, Khajavi M, Büttner M, Wagenstetter M, et al. Mapping single-cell data to reference atlases by transfer learning. Nat Biotechnol. 2022;40:121–30.

9. Cui H, Wang C, Maan H, Pang K, Luo F, Wang B. scGPT: Towards Building a Foundation Model for Single-Cell Multi-omics Using Generative AI [Internet]. bioRxiv; 2023 [cited 2023 Sep 22]. p. 2023.04.30.538439. Available from: https://www.biorxiv.org/content/10.1101/2023.04.30.538439v2

10. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention Is All You Need. Advances in neural information processing systems. 2017;30.

11. Dao T, Fu D, Ermon S, Rudra A, Ré C. Flashattention: Fast and Memory-Efficient Exact Attention with Io-Awareness. Advances in Neural Information Processing Systems. 2022;35:16344–59.

12. Domínguez Conde C, Xu C, Jarvis LB, Rainbow DB, Wells SB, Gomes T, et al. Cross-tissue immune cell analysis reveals tissue-specific features in humans. Science. 2022;376:eabl5197.

13. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research. 2011;12:2825–30.

14. Polański K, Young MD, Miao Z, Meyer KB, Teichmann SA, Park J-E. BBKNN: fast batch alignment of single cell transcriptomes. Bioinformatics. 2020;36:964–5.

15. Korsunsky I, Millard N, Fan J, Slowikowski K, Zhang F, Wei K, et al. Fast, sensitive and accurate integration of single-cell data with Harmony. Nature methods. 2019;16:1289–96.

16. Stuart T, Butler A, Hoffman P, Hafemeister C, Papalexi E, Mauck WM, et al. Comprehensive integration of single-cell data. Cell. 2019;177:1888–902.

17. Zhang F, Wu Y, Tian W. A novel approach to remove the batch effect of single-cell data. Cell discovery. 2019;5:46.

18. Lopez R, Regier J, Cole MB, Jordan MI, Yosef N. Deep generative modeling for single-cell transcriptomics. Nature methods. 2018;15:1053–8.

19. Alammar J. Ecco: An Open Source Library for the Explainability of Transformer Language Models. In: Ji H, Park JC, Xia R, editors. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations [Internet]. Online:

Association for Computational Linguistics; 2021 [cited 2023 Nov 8]. p. 249–57. Available from: https://aclanthology.org/2021.acl-demo.30

Supplementary Tables

Table S1. The datasets used for scMulan, scGPT, and Celltypist in pre-training, fine-tuning, and testing in the zero-shot cell type annotation task. ScMulan is pre-trained on hECA-10M, and applied on the testing set without fine-tuning. scGPT is pre-trained on Whole-human 33M, fine-tuned on hECA-10M-sampled, and applied on the testing set. Celltypist requires no pre-training process, and it's trained on hECA-10M-sampled, and applied on the testing set. Geneformer is pre-trained on Genecorpus-30M, fine-tuned on hECA-10M-sampled, and applied on the testing set. The new test datasets include AHCA_BoneMarrow, Simonson2023, and Suo2022.

Model	Pre-train	Fine-tune	Test
scMulan	hECA-10M	-	New test dataset
scGPT	Whole-human 33M	hECA-10M-sampled	New test dataset
Celltypist	-	hECA-10M-sampled	New test dataset
Geneformer	Genecorpus-30M	hECA-10M-sampled	New test dataset

Table S2. The datasets used for scMulan, scGPT, and Celltypist in pre-training, fine-tuning, and testing in the cell type annotation task with fine-tuning. ScMulan is pre-trained on hECA-10M, and fine-tuned with varying fractions of cells (ranging from 20% to 100%) from the training set. scGPT is pre-trained on Whole-human 33M, and fine-tuned on 100% samples of the training set. Celltypist requires no pre-training process, and it's trained on 100% samples of the training set. Geneformer is pre-trained on Genecorpus-30M, and fine-tuned on 100% samples of the training set. Training and testing samples are randomly split with a ratio of 9:1.

Model	Pre-train	Fine-tune	Test
scMulan	hECA-10M	Intestine_HCL_55k_train (Varying from 20% to 100%)	Intestinal_55k_test
scGPT	Whole-human 33M	Intestine_HCL_55k_train 100%	Intestinal_55k_test
Celltypist	-	Intestine_HCL_55k_train 100%	Intestinal_55k_test
Geneformer	Genecorpus-30M	Intestine_HCL_55k_train 100%	Intestinal_55k_test

Table S3. The datasets used for scMulan, scGPT, and Celltypist in pre-training, fine-tuning, and testing in the cell type annotation task with fine-tuning. ScMulan is pre-trained on hECA-10M and applied zero-shot. scGPT_finetune and scGPT_zeroshot were pre-trained on the whole-human-33M dataset, with scGPT_finetune undergoing additional fine-tuning on the testing datasets. Geneformer were pre-trained on the Genecorpus-30M dataset. The other methods are directly applied to the testing datasets. The other methods include BBKNN, Harmony, Seurat v3, FASTMNN, scVI, and scANVI. Testing datasets include COVID-19 and Heart_adults.

Model	Pre-train	Fine-tune	Test
scMulan	hECA-10M	-	Testing dataset
scGPT_zeroshot	Whole-human 33M	-	Testing dataset
scGPT_finetune	Whole-human 33M	Testing dataset	Testing dataset
Geneformer	Genecorpus-30M	-	Testing dataset
other methods	-	Testing dataset	Testing dataset

Supplementary figures

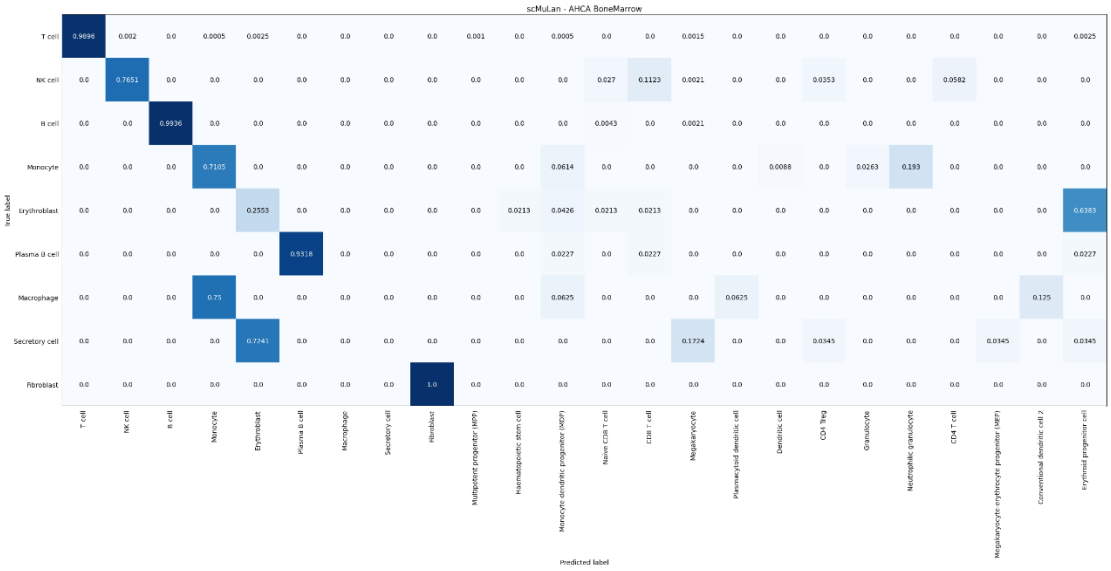

Figure S1. The confusion matrix of scMulan annotation on AHCA_BoneMarrow dataset without fine-tuning

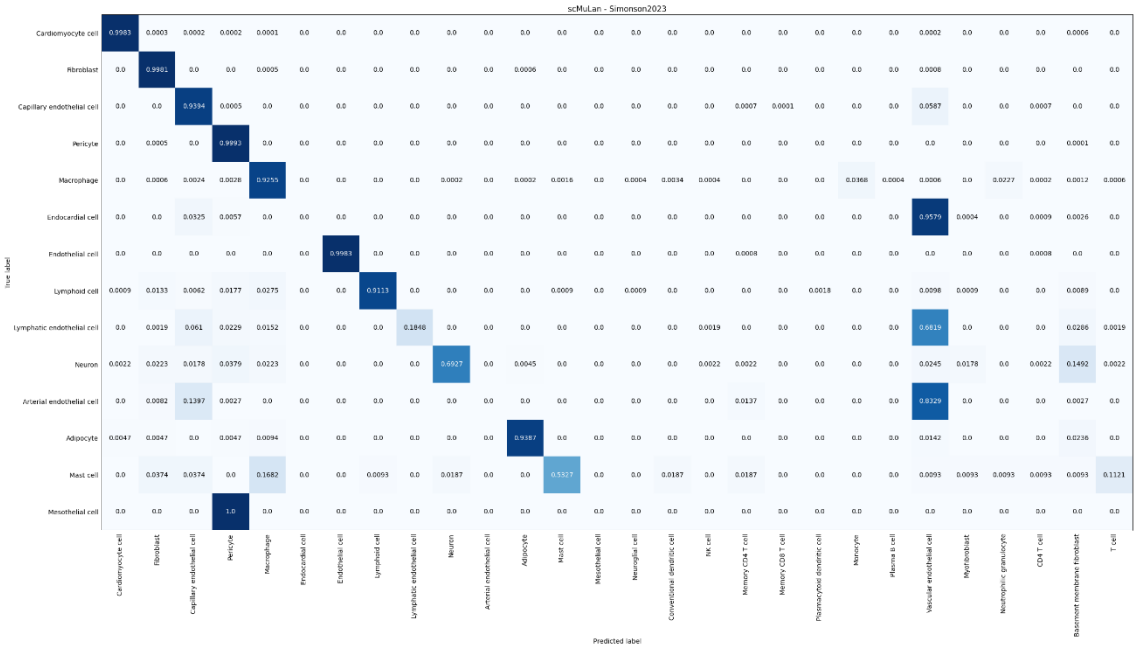

Figure S2. The confusion matrix of scMulan annotation on Simonson2023 dataset without fine-tuning

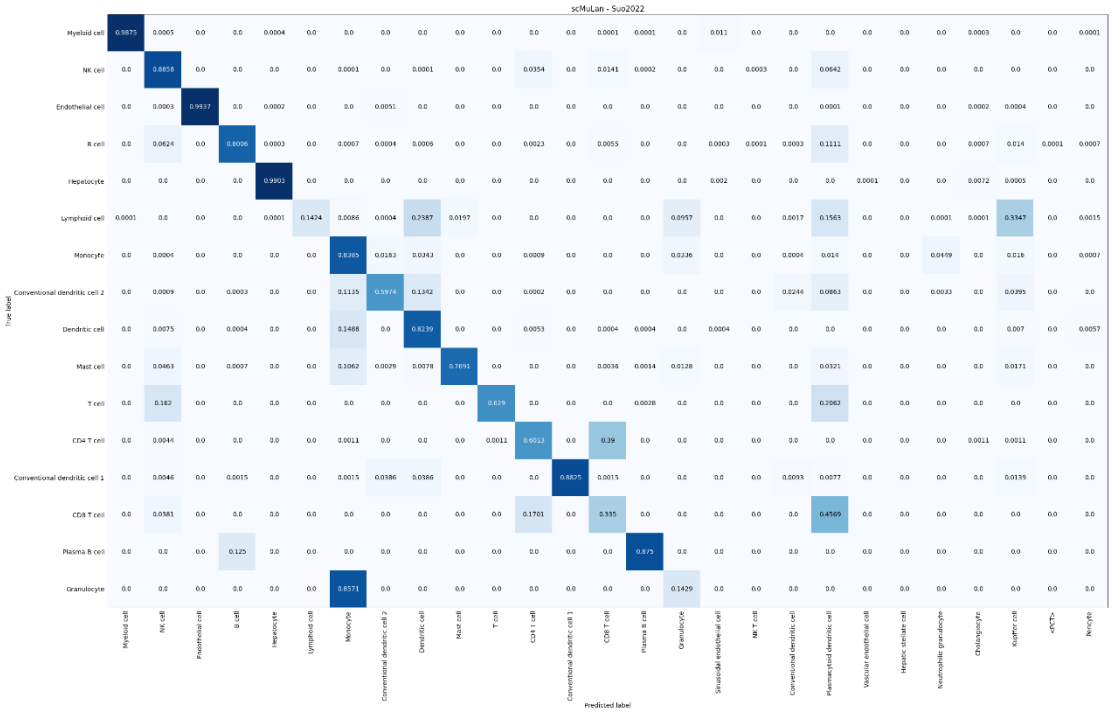

Figure S3. The confusion matrix of scMulan annotation on Suo2022 dataset without fine-tuning

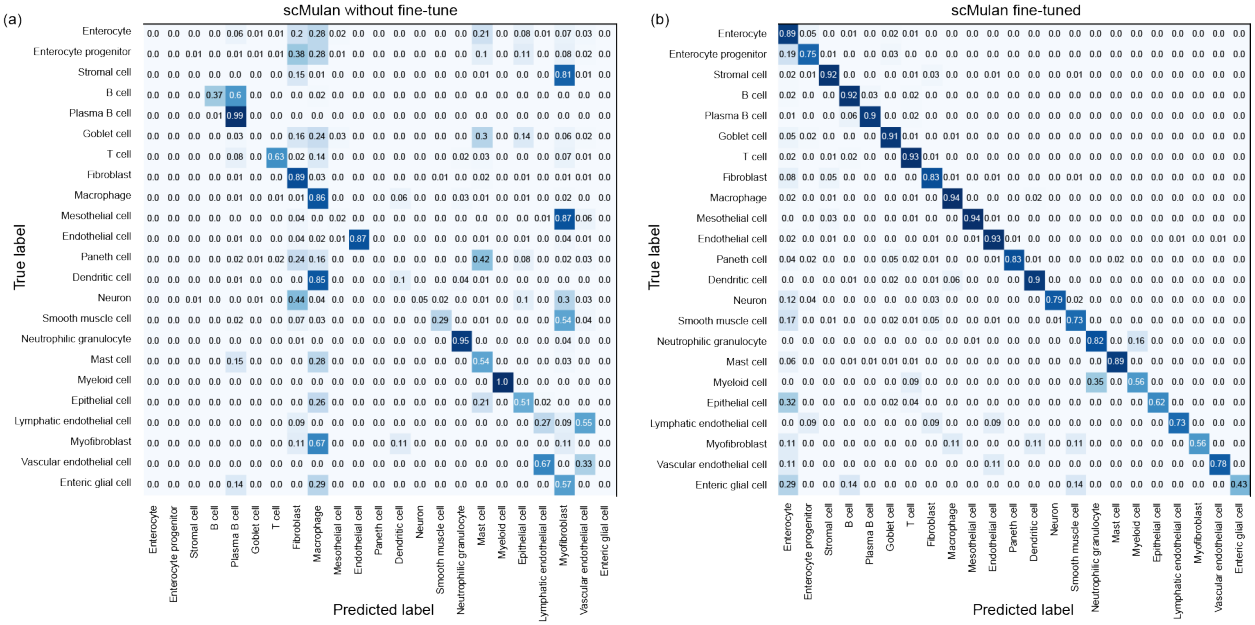

Figure S4. The confusion matrix of scMulan prediction on the Intestine_HCL dataset. (a) Results before fine-tuning. (b) Results after fine-tuning.

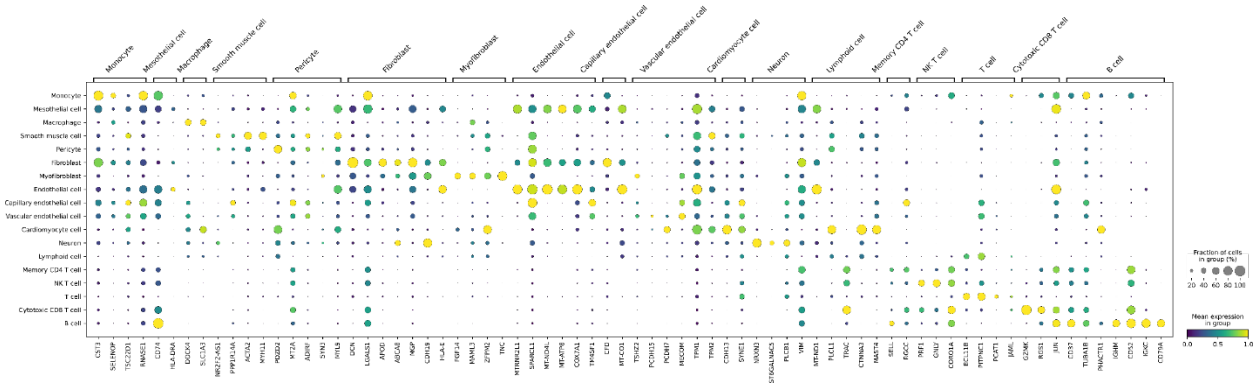

Figure S5: Bubble plot visualization of saliency genes extracted by scMulan in the sampled heart cells of hECA-10M Dataset. Each row corresponds to a specific cell type. Each column represents a gene, grouped by its associated cell type. The size of the bubbles reflects the proportion of cells expressing a given gene, with larger bubbles indicating a higher fraction of expression. The color of the bubbles represents the mean expression level of the gene within the respective cell type.