scGPT: Towards Building a Foundation Model for Single-Cell Multi-omics Using Generative AI

Haotian Cui[1,2,3]*, Chloe Wang[1,2,3]*, Hassaan Maan[1,3,4], Bo Wang[1,2,3,4,5]†

[1]*Peter Munk Cardiac Centre, University Health Network, Toronto, ON, Canada*
[2]*Department of Computer Science, University of Toronto, Toronto, ON, Canada*
[3]*Vector Institute, Toronto, ON, Canada*
[4]*Department of Medical Biophysics, University of Toronto, Toronto, ON, Canada*
[5]*Department of Laboratory Medicine and Pathobiology, University of Toronto, Toronto, ON, Canada*

Abstract

Generative pre-trained models have achieved remarkable success in various domains such as natural language processing and computer vision. Specifically, the combination of large-scale diverse datasets and pre-trained transformers has emerged as a promising approach for developing foundation models. While texts are made up of words, cells can be characterized by genes. This analogy inspires us to explore the potential of foundation models for cell and gene biology. By leveraging the exponentially growing single-cell sequencing data, we present the first attempt to construct a single-cell foundation model through generative pre-training on over 10 million cells. We demonstrate that the generative pre-trained transformer, scGPT, effectively captures meaningful biological insights into genes and cells. Furthermore, the model can be readily finetuned to achieve state-of-the-art performance across a variety of downstream tasks, including multi-batch integration, multi-omic integration, cell-type annotation, genetic perturbation prediction, and gene network inference. The scGPT codebase is publicly available at https://github.com/bowang-lab/scGPT.

1 Main

Generative pre-trained models have recently achieved unprecedented success in many domains. The most well-known applications include computer vision and natural language generation (NLG) [44, 43, 45]. These foundation models such as DALL-E2 and GPT-4 follow a similar paradigm of pre-training transformers on large-scale diverse datasets [43, 45]. These foundation models can be readily tailored to a variety of downstream tasks and scenarios. More interestingly, they demonstrate improved performance on multiple tasks compared to task-specific models trained from scratch [22, 58, 47]. This showcases strong evidence of a task-agnostic and "deep" understanding

*These authors contributed equally.
†Corresponding author. Email: bowang@vectorinstitute.ai

of knowledge in these domains. Despite the success of foundation models in other domains, currently machine-learning based discovery in single-cell research is rather distributed, with specific models dedicated to specific analysis tasks [32, 42, 27]. Often the breadth and scale of the datasets used in each study are also limited, due to sequencing capacity as well as the scope of the research question [2]. This calls for a foundation model pre-trained on large-scale data to achieve a general understanding of single-cell biology. We expect such a model to serve as a strong foundation and contribute to the discovery of new biological insights with the help of the learned knowledge from millions of sequenced cells.

While the feasibility of generative pre-training in single-cell biology remains largely unexplored, we can draw inspirations about modelling and the data-centric perspectives from other domains. From a modelling perspective, the self-attention transformer has been verified as an efficient and effective architecture to model input tokens of words. While texts are made up of words, cells can be characterized by genes. We can learn and extract gene and cell representations simultaneously in a similar fashion as word and sentence embeddings in NLG. The flexible vocabulary structure also allows easy addition of new features and meta information. The attention mechanism of the transformers could be further explored to inform on gene-to-gene and cross-modality associations [60]. From a data perspective, the vast-scale atlases of single-cell RNA sequencing (scRNA-seq), such as the Human Cell Atlas, already encompass tens of millions of cells, and the scale of accessible omic data continues to grow exponentially [26, 51, 23, 2]. This opens ample opportunities to employ self-supervised learning techniques to learn from diverse cell types and tissues, and integrate across different organs and species.

In this work, we present the first attempt to build a single-cell foundation model, scGPT, by generative pre-training on over 10 million cells. We introduce several new techniques to address the methodology and engineering challenges of pre-training on large-scale single-cell omic data. To handle the large-scale data, we use an in-memory data structure to store hundreds of datasets that allow fast access. We establish a unified generative pre-training workflow specifically for the non-sequential omic data, and adapt the transformer architecture to simultaneously learn cell and gene representations. We also provide reusable finetuning pipelines and objectives designed for a variety of downstream tasks to help users apply the pre-trained model with ease.

We demonstrate that the pre-trained model captures meaningful biological insights on both gene and cell levels. The learned gene embedding maps decode known pathways by grouping together genes that are functionally relevant. With zero-shot learning, the pre-trained model is able to reveal meaningful cell clusters on unseen datasets. With finetuning in a few-shot learning setting, the model achieves state-of-the-art performance on a wide range of downstream tasks, including batch correction, multi-omic integration, cell type annotation, genetic perturbation prediction, pseudo-cell generation, and gene network inference. The release of the scGPT model and workflow aims to facilitate future research in all related areas. We envision that the adoption of pre-trained foundation models will further our understanding of cellular biology, and pave the foundation for future discoveries.

2 Results

2.1 Single-cell transformer foundation model overview

Single-cell sequencing captures the genetic profiles at the individual cell level. For instance, scRNA-seq measures transcriptomic activities from RNA abundance, which informs on cell identity, stage, and functionality. Recent cellular reference maps such as the human cell atlas comprise of millions

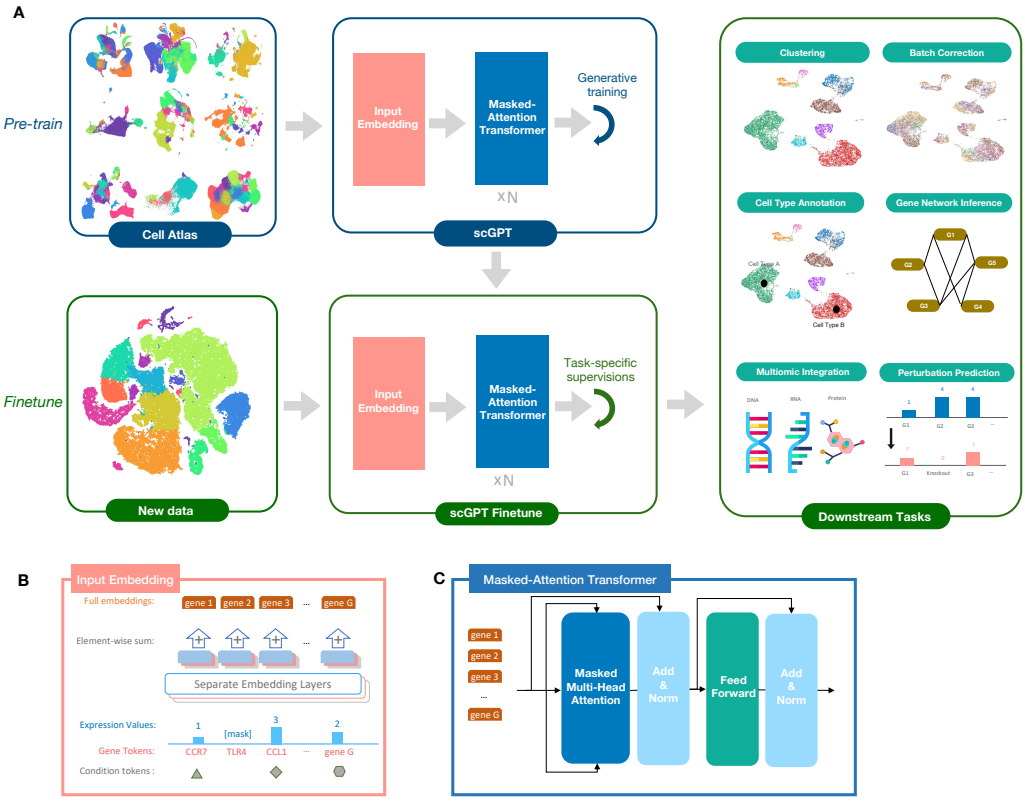

Figure 1: Model Schematic. *(a)* The workflow of scGPT. The model is firstly generatively trained on large-scale scRNA-seq data from cell atlas. For a downstream application, the pre-trained model weights can be finetuned on the new data. The core component of scGPT contains stacked transformer blocks with specialized attention masks for generative training. We applied scGPT on a variety of tasks including clustering, batch correction, cell type annotation, multi-omics, genetic perturbation prediction, and gene network inference. *(b)* The zoom-in view of the input embeddings. The input contains three layers of information, the gene token, expression value, and condition tokens (modality, batch, perturbation condition, et al.). *(c)* The zoom-in view of the scGPT transformer layer. We introduced a specially designed attention mask to conduct generative pre-training on single-cell sequencing data.

of single cells from diverse organs and tissues, offering an unparalleled representation of cellular heterogeneity [26, 52]. We introduce scGPT as the generative pre-trained foundation model in the single-cell domain. The core model has stacked transformer layers with multi-head attention that learns cell and gene embeddings simultaneously (See Online Methods 4.2). Figure 1A illustrates a two-stage workflow involving pre-training and fine-tuning of scGPT. In the pre-training stage, we collected over 10.3 million scRNA-seq data of blood and bone marrow cells from the CellXGene portal [11] for training. We introduce a specially designed attention mask and generative training pipeline to train scGPT in a self-supervised manner to jointly optimize cell and gene representations (See Online Methods 4.3). During training, the model gradually learns to generate gene expression of cells based on simple cell or gene expression cues. In the fine-tuning stage, researchers can apply the pre-trained model to new datasets and specific tasks (See online Methods 4.5). We offer flexible finetuning pipelines suitable for a variety of downstream tasks essential in single-cell research, including scRNA-seq integration with batch correction, cell type annotation, multi-omic integration, perturbation prediction, and gene regulatory network inference.

scGPT learns cell and gene representations from diverse single-cell data through gene expression modelling. To facilitate gene representation learning, we employed Gene Expression Prediction (GEP) as the generative self-supervised objective to iteratively predict gene expression values of unknown tokens from known tokens in an auto-regressive manner (See Online Methods 4.4). To enhance cell representation learning, we designed Gene Expression Prediction for Cell Modelling (GEPC) objective, where the model predicts gene expression values from cell representations (See Online Methods 4.4). This creates a direct link between the gene expression profile and cellular heterogeneity, allowing for joint optimization within the scGPT framework. Furthermore, scGPT's embedding architecture can easily extend to multiple sequencing modalities, batches, and perturbation states. This is achieved by adding new condition tokens for sequencing modalities, batches, and perturbation states. This unique model flexibility enables the pre-trained model to seamlessly combine with any additional information required for specific downstream tasks. See model architecture illustration in Figure 1B and more details in Online Methods 4.

scGPT serves as a powerful single-cell feature extractor on previously unseen datasets. In benchmark experiments, scGPT outperformed recent methods and achieved state-of-the-art results across all downstream tasks. This demonstrates the benefits of pre-training and the transferability of learned knowledge across diverse use cases. By providing a robust and unified framework, scGPT enables single-cell researchers to easily leverage the pre-trained foundation model in related studies.

2.2 Integration of multiple scRNA-seq data with batch correction

Clustering and visualization of single-cell sequencing data encounter a significant challenge in the presence of batch effects arising from the utilization of multiple datasets or sequencing batches as input. By employing a finetuning workflow, the scGPT framework effectively tackles this challenge by incorporating customized finetuning objectives (refer to Online Methods 4.4). This approach successfully corrects for batch effects while preserving the true biological signals inherent in the data.

scGPT achieves the state-of-the-art performance in preserving the biological variance of the integrated datasets upon batch correction. We benchmarked scGPT with three popular integration methods, scVI [34], Seurat [55], and Harmony [29] on two integration datasets Immune Human (10 batches) [36] and PBMC 10K (2 batches) [21]. As shown in Figure 2A in the Immune Human dataset, scGPT successfully integrated all batches of CD4+ T cells, CD8+ T cells, and CD14+ Monocytes into their individual clusters, whereas Seurat produced a few sub-clusters corresponding to sequencing batches within each of these cell types (See batch visualizations in Supplementary

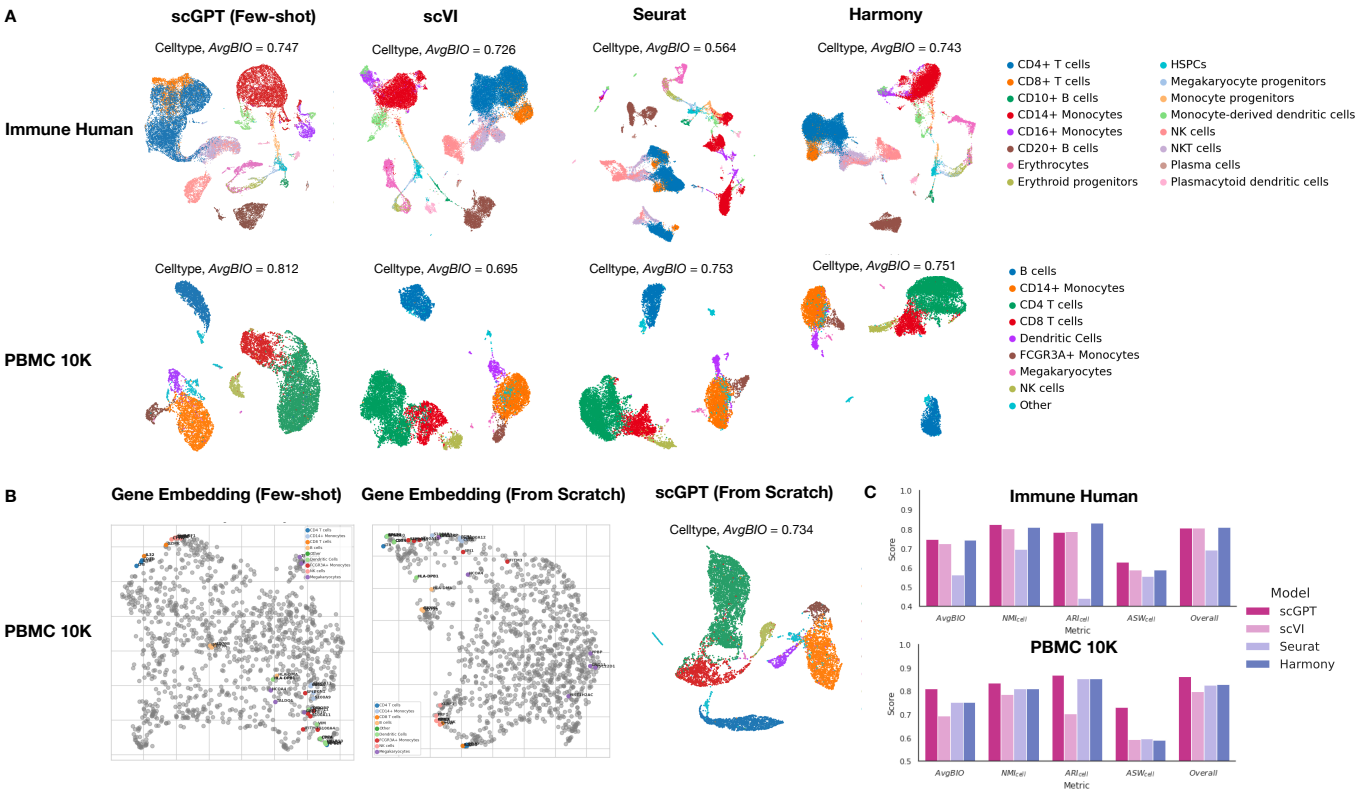

Figure 2: *(a)* Benchmark of the few-shot scGPT model with scVI [34], Seurat Seurat [55], and Harmony Harmony [29] on the Immune Human (10 batches) [36] and PBMC 10K (2 batches) [21] datasets for cell type clustering performance upon batch integration. The UMAP plot of learned cell embeddings was colored by *cell types*. *(b)* Comparison of gene embedding maps of the few-shot and trained-from-scratch scGPT models. The highly variable genes from each *cell type* were highlighted. The UMAP plot of learned cell embeddings from trained-from-scratch scGPT model was visualized. *(c)* Comparison of the scGPT model with other benchmarked methods on $AvgBIO$, the detailed biological conservation metrics (NMI_{cell}, ARI_{cell}, ASW_{cell}), and the $Overall$ score.

Figure S3). scGPT also managed to separate the Monocyte-derived dendritic cells from the CD16+ Monocytes, but scVI and Harmony both saw a significant overlap of the two clusters. Moreover, in the PBMC 10K dataset, scGPT is the only method that clearly separated out the cell type Other from the annotated clusters. In contrast, scVI, Seurat and Harmony all confused this Other cell type with CD14+ Monocytes and CD8 T cells. This inaccuracy is visualized as these blue dots from cell type Other scattering all over the orange and red clusters. scGPT's superior clustering performance is also reflected in the biological conservation score, where scGPT achieves an $AvgBIO$ score of 0.812, which is 5% higher than Seurat and Harmony, and 10% higher than another deep learning method, scVI. In Figure 2C, scGPT presents competitive scores across all cell type clustering metrics attributing to biological conservation. scGPT also ranked top in the Overall metric considering both biological conservation and batch correction performance (See detailed metrics in Supplementary Table S.1, and batch visualizations in Supplementary Figure S3).

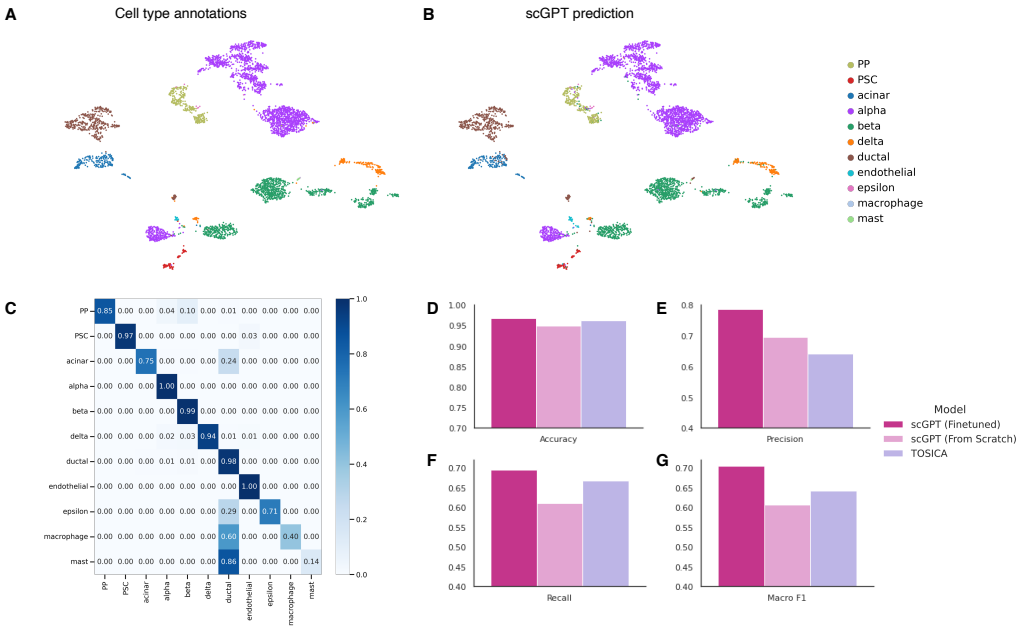

Figure 3: *(a)* UMAP visualization of cells colored by ground-truth *cell types* in the hPancreas query set. *(b)* UMAP visualization of cells colored by scGPT predicted *cell types*. *(c)* Normalized confusion matrix by *cell types*. $(b) - (g)$ Comparison of the finetuned scGPT model with trained-from-scratch model and existing method on *Accuracy*, *Precision*, *Recall* and $MacroF1$ scores.

We further highlight the benefits of pre-training by showing a significant performance boost in the finetuned model in comparison to the trained-from-scratch model in the PBMC 10K dataset (See Figure 2B). The finetuned gene embeddings produced more compact networks for highly variable genes of the CD4 T cells and Megakaryocytes, compared to the trained-from-scratch model. We observe similar results in the cell embeddings, as the cell type clusters become more defined in the finetuned model with a 8% improvement in the $AvgBIO$ score. As a sanity check, we validated that the pre-trained model in zero-shot is also able to produce meaningful cell type clusters with an $AvgBIO$ score of 0.728, on par with the trained-from-scratch model (See Supplementary Figure S2). This presents the zero-shot model as a generalizable feature extractor that can be readily applied to unseen datasets. Furthermore, the 8% performance boost from pre-training demonstrates the benefits of leveraging these pre-trained feature extractors and the knowledge learned. The foundation model is proven to be not only easily transferable to new datasets but also more powerful than learning from limited data from scratch.

2.3 Cell type annotation

Cell type annotation is a crucial step in single-cell analysis after clustering, as it resolves heterogeneity in sequenced tissues and lays the foundation for further investigation of cell and gene functions to gain biological and pathological insights. While several methods have been proposed for cell annotation, such as cellAssign [64], singleR [3], and Chetah [17], they typically require dimension reduction prior to model input, which can lead to information loss. In contrast, scGPT's transformer model can directly take in gene expressions in an unbiased manner, with full resolution on the entire highly variable gene set as input. This approach provides greater reliability and improved accuracy in cell type classification, as demonstrated in our subsequent analyses.

For the cell type annotation task, we finetuned the pre-trained scGPT model using cross-entropy loss against ground-truth labels from a new reference dataset (See Online Methods 4). Using the hPancreas dataset of human pancreas cells as an example, we trained the scGPT model on the reference set and validated the classification performance on a different query set. Figure 3 panels A and B present the cell embeddings colored by ground-truth versus predicted cell types, where the scGPT model demonstrates faithful prediction with high accuracy score of 96.7%. The model also achieved high precision in predicting majority of the cell types, except for the rare cell types with extremely low cell numbers in the reference set (See Figure 3C). For example, fewer than 50 cells belong to the mast and macrophage cell types out of the 10.6 thousand cells in the reference set. To benchmark the performance of scGPT, we compared it with TOSICA [12], a recent transformer-based annotation method. scGPT outperforms TOSICA on all classification metrics, including *Accuracy*, *Precision*, *Recall*, and $MacroF1$, as shown in Figure 3D-G. We also trained a separate scGPT model from scratch without the pre-trained model parameters. It achieves reasonable accuracy on the query set. The performance improvement of the finetuned model from the trained-from-scratch model demonstrates the benefits of transfer learning with pre-trained scGPT.

2.4 Perturbation Prediction

Sequencing and gene editing techniques have recently facilitated high-throughput experiments, enabling the exploration of cellular responses to multiple genetic perturbations. The approach holds immense promise for uncovering novel gene interactions and advancing regenerative medicine. However, the vast combinatorial space of potential gene perturbations quickly surpasses the practical limits of experimental feasibility. To overcome this limitation, scGPT can be employed to leverage the knowledge gained from cellular responses in known experiments and extrapolate to predict responses in unknown scenarios. The utilization of self-attention mechanisms over the gene dimension enables the encoding of intricate interactions between perturbed genes and the responses of other genes. By leveraging this capability, scGPT can effectively learn from existing experimental data and accurately predict gene expressions following perturbation.

For the perturbation prediction task, we evaluated our model using two perturbation datasets pre-processed by Roohani, Huang, and Leskovec [53]: (1) the Pertub-seq dataset of K562 leukemia cell line [1], which comprises 87 one-gene perturbations, with approximately 100 cells per perturbation and a minimum of 7,000 unperturbed cells, and (2) the other Norman Perturb-Seq dataset [41], consisting of 131 two-gene perturbations and 105 one-gene perturbations.

We assessed the perturbation prediction by calculating the Pearson correlation (*corr*) between the predicted and the corresponding ground-truth expression values after perturbation. In addition, we introduced a variant of the Pearson metric, denoted as *corr(Δ)*, which measures the correlation based on the magnitude of expression change post-perturbation compared to the control. We have presented Pearson metrics for various gene sets, namely all genes (*ALL*) and the top 20 differentially expressed genes (*DE*). For detailed information on how these metrics were calculated, refer to Supplementary Online Methods S.2.

Table 1: Results of perturbation prediction

	Norman et al. [41]				Adamson et al. [1]			
	DE		ALL		DE		ALL	
Model	corr	corr(Δ)	corr	corr(Δ)	corr	corr(Δ)	corr	corr(Δ)
MLP	0.909	0.428	0.987	0.408	0.948	0.729	0.991	**0.656**
GEARS	0.917	0.508	0.986	0.387	0.961	0.726	0.991	0.652
scGPT	**0.923**	**0.546**	**0.988**	**0.459**	**0.971**	**0.775**	**0.992**	0.647

We conducted a performance comparison between scGPT, the recent GEARS method [53], and the multi-layer perception (MLP) baseline. Our results demonstrate that scGPT achieves the highest correlation across seven out of eight evaluation metrics. It is worth noting that approximately 50% of the gene expression counts are zero in the raw scRNA-seq data. Therefore, we contend that evaluating differentially expressed genes, specifically the DE columns in Table 1, provides more compelling evidence. Notably, scGPT exhibits significant improvements in the correlation of the (Δ) change of the top 20 differentially expressed genes, which is arguably the pivotal metric.

2.5 scGPT facilitates multi-omic integration and multi-modal representation learning

Single-cell multi-omic (scMultiomic) data presents multiple views of genetic regulation all at once, including epigenetic, transcriptomic, and translation activities [57, 37]. It provides an ample opportunity to enhance feature and cell representation learning beyond gene expressions. However, the challenge lies in how to reliably aggregate cell representations from multiple views while preserving biological signals.

The scGPT framework can be readily extended to integrate multiple sequencing data modalities. Each omic type in scMulti-omic data (e.g., gene expression, chromatin accessibility, and protein abundance) is similar to a different language in NLG. Analogously, scGPT supports joint optimization of multi-omic tokens from diverse sequencing modalities. This framework also allows seamless addition of new sequencing modalities to existing pre-trained network by extending the "vocabulary". In the benchmark experiments, scGPT demonstrates outstanding performance in cell representation learning and multiomic batch integration tasks compared to existing state-of-the-art methods (See Figure 4).

scGPT effectively extracts integrated cell embeddings from paired scMultiomic data. In this paired data integration setting, each sequenced cell contains all the data modalities. We used the 10X Multiome PBMC [14] dataset with joint gene expression and chromatin accessibility measurements as an example. We benchmarked scGPT with two state-of-the-art methods scGLUE[9] and Seurat v4[24] on cell type clustering performance. As shown in Figure 4, scGPT is the only method that produced a clear separate cluster for CD8 Naive cells, whereas the other two methods failed. scGPT also differentiated Memory B cells from Naive B and Intermediate B cell clusters, yet scGLUE produced a merged cluster of all three types of B cells. scGPT separated the CD4 and CD8 cell groups into two distinct groups of clusters, surpassing the results of Seurat v4. scGPT demonstrates superior cell type clustering performance overall ($AvgBIO$=0.767) and robustness across the diverse biological conservation metrics benchmarked (See Figure 4 and Supplementary Table S.1).

scGPT simultaneously integrates multi-modal batches from mosaic scMultiomic data via joint representation learning. In the mosaic data integration setting, sequenced samples share a few but not all data modalities. We used the ASAP human PBMC dataset [38] with four sequencing batches and three data modalities as an example. The first two sequencing batches contain gene expression and protein abundance data from CITE-seq, and the second two batches have chromatin accessibility and protein abundance measurements from ASAP-seq. In the benchmark experiment with scMoMat[65], scGPT demonstrates superior batch correction performance as shown in Figure 4, especially in the rarer cell group NK cell. In comparison, scMoMat produced two distinct clusters for each cell type corresponding to the first two and second two batches, indicating failure to mitigate modality differences. scGPT achieves a overall batch correction score $AvgBATCH$ of 0.948, with a close-to-perfect $GraphConn$ score of 0.992 and a significantly higher ASW_{batch} score of 0.904 compared to scMoMat ($ASW_{batch} = 0.849$). scGPT's biological conservation metrics

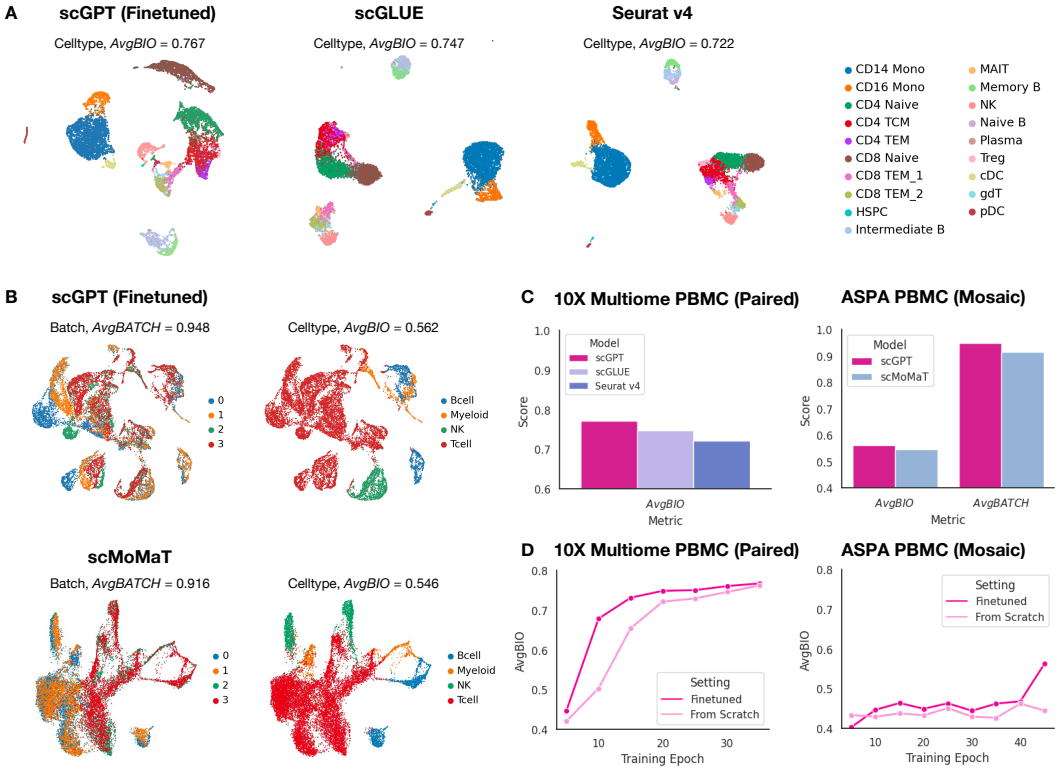

Figure 4: *(a)* Benchmark of the few-shot scGPT model with scGLUE[9] and Seurat v4[24] on the 10x Multiome PBMC [14] dataset (paired data setting) for cell type clustering task. The UMAP plot of learned cell embeddings was colored by *cell types*. *(b)* Benchmark of scGPT with scMoMat[65] on the ASAP PBMC [38] dataset (mosaic data setting) for batch correction and cell type clustering tasks. The UMAP plots of learned gene embeddings were colored by *sequencing batches* (left) and *cell types* (right). *(c)* Comparison of the scGPT model with other benchmarked methods on $AvgBIO$ and $AvgBATCH$ scores. *(d)* Comparison of training progress curves of the few-shot and trained-from-scratch models with $AvgBIO$ scores evaluated every five epochs.

also compare favorably to scMoMat's, which further indicates the robustness of multi-modal batch correction without interfering with the biological signals (See Figure 4 and Supplementary Table S.1).

scGPT readily supports the addition of new data modalities to exisiting pre-trained models. We compared scGPT's training progress curves in two settings, finetuned on the pre-trained model and trained from-scratch, to demonstrate the benefits of pre-training in the multi-omic integration task. As shown in Figure 4, for the 10X Multiome PBMC dataset, the finetuned model reached the best AvgBIO scores in the 70% range 5 epochs earlier than the trained-from-scratch model. This demonstrates the benefits of the pre-trained model in leading training progress and faster convergence. In the more challenging mosaic integration setting with the ASAP PBMC dataset, the finetuned model's AvgBIO scores increased steadily as training proceeded, whereas the trained-from-scratch model did not see much progress with training. At epoch 45, the pre-trained model finished at an AvgBIO score of 0.562, which is more than 10% higher than the trained-from-scratch model with a score of 0.444. This suggests that the model leverages the learned gene embeddings from large atlases to guide the learning of peak and protein embeddings in the pre-trained setting.

2.6 Gene embeddings for Gene Regulatory Network inference

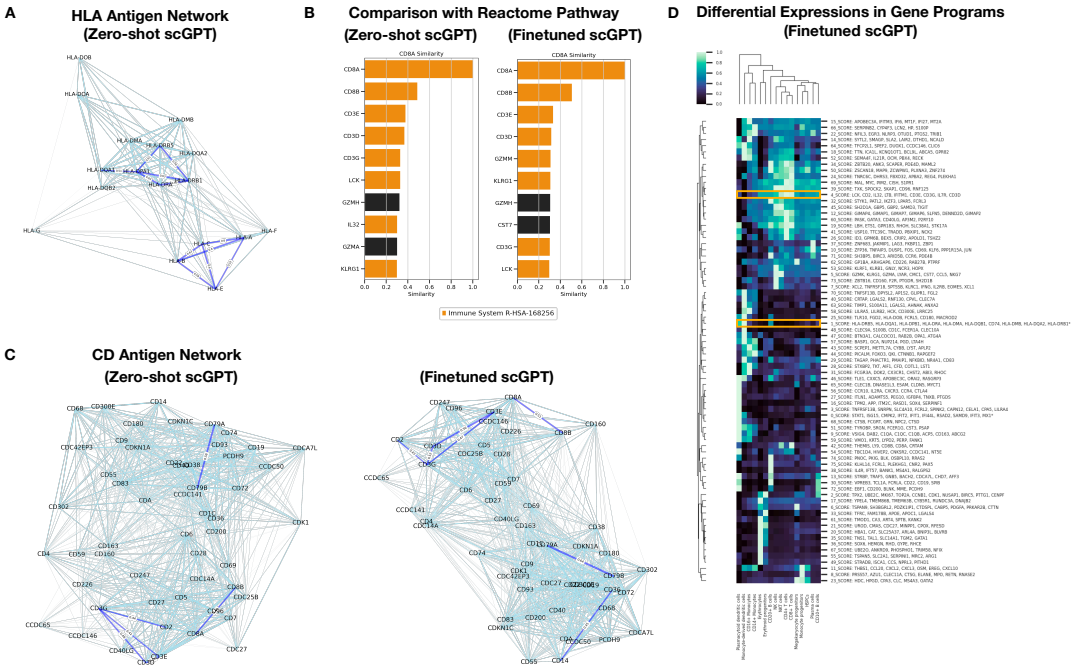

Figure 5: *(a)* HLA antigen network from zero-shot scGPT. *(b)* CD8A gene neighbors from zero-shot and finetuned scGPT models, ranked by embedding similarity colored by ground-truth signalling pathway from Reactome. *(c)* CD antigen network from zero-shot and finetuned scGPT on the Immune Human dataset. *(d)* Differential expressions among scGPT-extracted gene programs by cell types in the Immune Human dataset.

The interactivity between transcription factors, cofactors and target genes underlying a Gene Regulatory Network (GRN) mediates important biological processes. Existing GRN inference methods often rely on correlation in static gene expressions or pseudo-time estimates as a proxy for causal graphs [46]. scGPT, optimized by the generative training of gene tokens, implicitly encodes such relationships in its gene embeddings. The gene embeddings can therefore be applied to construct similarity networks that entail gene-gene interactions. We hereby validate the scGPT's gene embedding network against known biology, and then explore its applicability to gene program discovery.

scGPT demonstrates its ability to group the functionally related genes and differentiate functionally different genes from its gene embedding network. In Figure 5A, we visualized the similarity network of the human leukocyte antigens (HLA) antigens from the pre-trained gene embeddings. In the zero-shot setting, the scGPT model highlights two clusters corresponding to the two well-characterized HLA classes that trigger different immune responses, namely HLA class I and HLA class II. The HLA class I antigens HLA-A, -C, and -E are recognized by CD8+ T cells to mediate cell killing, whereas HLA class II antigens HLA-DR, -DP, and –DQ are recognized by CD4+T cells to trigger broader helper functions [13]. For the finetuned scGPT model on the Immune Human dataset, we explored the CD antigen network specific to the immune cell types present in this dataset (See Figure 5C). The pre-trained scGPT is able to identify CD3E, D, and G genes as a group that encodes the T3 complex for T-cell activation, CD79A and B for B-cell signalling, and CD8A and B as co-receptors for HLA class 1 molecules [40]. The finetuned scGPT further

highlights the connection between CD36 and CD14 as markers for monocytes and macrophages. This demonstrates scGPT's ability to generalize from the knowledge learned in pre-training and extract specific information related to the dataset at hand.

scGPT reconstructs meaningful gene programs in a purely unsupervised workflow. In Figure 5D, we visualized the gene programs extracted by the finetuned scGPT model on the Immune Human dataset, and their differential expressions by cell types. These gene programs are selected in an unsupervised manner by first clustering the gene embeddings and then thresholding on clusters that consist of 5 or more genes, following Ceglia et al. [10]'s proposed pipelines (See Online Methods 4). We observed that the same HLA antigen cluster was identified as group 1. Similarly, the CD3 genes involved in T3 complex were identified as group 4, with highest expressions in T cells. This confirms that scGPT's inferred gene programs correspond to function groups that are biologically meaningful. We further validated the gene similarity relationships encoded by the scGPT model against the known Reactome database [50]. Using the CD8A gene as an example, we demonstrate that its closest neighbors are more likely to be part of the Immune System R-HSA-168256 pathway than genes that are further away, from both zero-shot and finetuned scGPT models (See Figure 5B). The top three genes closest to CD8A remain consistent before and after finetuning. The finetuned model highlighted one additional gene GZMM that is commonly enriched in NK, NKT and T cell subclusters [4] which are major cell types found in the Immune Human dataset. We further validated this correlation on the entire gene set across the Reactome database. Among pairs of genes, there exists a positive correlation between the cosine similarity score of the gene embeddings and the number of common pathways shared by these genes, with a Pearson correlation score of 0.316.

While a more comprehensive evaluation pipeline is to be established, these findings showcase that scGPT has learned meaningful biological patterns from generative pre-training in the zero-shot setting. More specifically, we demonstrate its ability to perform unsupervised gene program discovery on new datasets along with other cell-level analysis tasks by leveraging the pre-trained model. We envision this attempt as one of the first steps towards knowledge discovery in the single-cell domain assisted by foundation models.

3 Discussion

We hereby present scGPT, the first foundation model that leverages pre-trained transformers learned from over 10 million single-cell data. The self-supervised pre-training paradigm with increasingly large amount of training data has created powerful language models such as chatGPT and GPT4 [44, 43]. These successes inspired us to apply the same pre-training paradigm to the single-cell domain, with the aim of decoding complex biological interactions with the pre-trained transformers. The transformer models naturally support the joint learning of gene and cell embeddings, analogous to word and sentence embeddings in NLG. These technical advantages create a solid foundation for modelling these different aspects of cellular processes together at once.

We demonstrate the benefits of pre-training with comprehensive experiments in both zero-shot and finetuning settings. The pre-trained model itself is a universal feature extractor. It showcases strong capabilities of extrapolating to unseen datasets, presenting meaningful cell clustering in zero-shot experiments. The learned gene networks also reflect known gene programs and their functional roles. These abilities give us confidence in the pre-trained model that it has not only memorized but also synthesized the patterns from the large-scale single-cell data. We also observed a consistent contribution of the pre-trained model in multiple downstream tasks via transfer learning. For example, in both multi-batch and multi-omic integration tasks, the finetuned model

has demonstrated superior performance in cell-type clustering, with an 8 to 12% increase in the biological conservation score compared to the trained-from-scratch models.

To implement generative pre-training for the non-sequential single-cell data, we introduced specialized attention masking to support generation and joint gene and cell representation learning. In the finetuning pipeline, we offer setups for a diverse range of downstream tasks such as batch correction, cell type assignment, multi-omic integration, perturbation prediction, and gene network inference. We hereby release the scGPT codebase and the pre-trained model. We hope that this provides a unified framework to help researchers easily adapt the pre-trained models to their own tasks at hand.

For future directions, we plan to pre-train on a larger-scale dataset with more diversity, including multi-omic data, spatial omics, and diseased conditions. It is also interesting to incorporate perturbation and temporal data in the pre-training stage for causal discovery. More importantly, we would like to validate the pre-trained model on a wider range of biologically meaningful tasks to understand and interpret what the pre-trained model has learned. We also aim to explore in-context instruction learning for single-cell data. The goal is to have a pre-trained model that understands different tasks and contexts in the zero-shot setting without having to finetune. scGPT thus serves as the first step to use large-scale pre-trained foundation model to understand the context and nuances in cell biology. We envision that the pre-training paradigm be readily integrated into single-cell research, and serve as a foundation to leverage the existing knowledge from the exponentially growing cell atlases for new discoveries.

4 Methods

4.1 Input embeddings

The single-cell sequencing data is processed into a cell-gene matrix, $\boldsymbol{X} \in \mathbb{R}^{N \times G}$, where each element $X_{i,j} \in \mathbb{R}^{+}$ represents the read count of a RNA for scRNA-seq or a peak region if scATAC-seq. For example in scRNA-seq, the element denotes the RNA abundance for gene $j \in 0, 1, \ldots, G$ in cell $i \in 0, 1, \ldots, N$. In subsequent sections, we will refer to this matrix as the raw matrix. The input to scGPT consists of three main components: (1) gene (or peak) tokens, (2) expression values, and (3) condition tokens. For each modeling task, the gene tokens and expression values are pre-processed from the raw count matrix $\boldsymbol{X}$ accordingly:

Gene Tokens In the scGPT framework, each gene is considered the smallest unit of information, equivalent to a word in natural language generation (NLG). We therefore use gene names as tokens, and assign each gene g_j a unique integer identifier $id(g_j)$ within the complete vocabulary of tokens. This approach offers great flexibility to harmonize multiple studies with different gene sets (i.e., generated by distinct sequencing technologies or pre-processing pipelines). Specifically, different sets of gene tokens can be integrated into a common vocabulary by taking the union set of all genes across studies. Additionally, we incorporate special tokens in the vocabulary, such as $< cls >$ for aggregating all genes into a cell representation, and $< pad >$ for padding the input to a fixed length. Conceptually, we draw parallels between gene tokens and word tokens in NLG. The input gene tokens of each cell i are hence represented by a vector $\boldsymbol{t}_g^{(i)} \in \mathbb{N}^M$:

$$\boldsymbol{t}_g^{(i)} = \left[id(g_1^{(i)}), id(g_2^{(i)}), \ldots, id(g_M^{(i)}) \right], \tag{1}$$

where M is a pre-defined input length, and usually equals to the number of selected highly variable genes.

Expression Values The gene expression matrix X requires additional processing before being used as input for modeling. A fundamental challenge in gene expression modeling is the variability in absolute magnitudes across different sequencing protocols [54]. Due to variations in sequencing depths and the presence of sparsely expressed genes, the data scales differ significantly among different batches of sequencing samples. These differences are not easily mitigated with common preprocessing techniques such as transcripts per million (TPM) normalization and $log1p$ transformation [25]. To make it more clear, the same absolute value can convey different "semantic" meanings across sequencing batches. To address this scale difference, we propose the ***value binning*** technique to convert all expression counts into relative values. For each non-zero expression count in each cell, we calculate the raw absolute values and divide them into B consecutive intervals $[b_k, b_{k+1}]$, where $k \in \{1, 2, \ldots, B\}$. Each interval represents an equal portion of all expressed genes $(1/B)$. It is important to note that a new set of bin edges is computed for each cell, so the interval edges b_k may vary among cells. The binned expression value $x_j^{(i)}$ for cell i is defined as:

$$x_j^{(i)} = \begin{cases} k, & \text{if } X_{i,j} > 0 \text{ and } X_{i,j} \in [b_k, b_{k+1}], \\ 0, & \text{if } X_{i,j} = 0. \end{cases} \tag{2}$$

Through this binning technique, the semantic meaning of $x_j^{(i)}$ is consistent across sequencing batches. For instance, a value of $x_j^{(i)} = B$ consistently indicates the highest expression among genes. Before applying the value binning step, we performed $log1p$ transformation, and highly variable gene selection [35]. To simplify the notation, we use $X_{i,j}$ to represent both the raw and preprocessed data matrices prior to binning. Therefore, the final input vector of binned expression values for cell i is denoted as

$$\boldsymbol{x}^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, \ldots, x_M^{(i)}\right]. \tag{3}$$

Condition Tokens The condition tokens encompass diverse meta information associated with individual genes, such as functional pathways (represented by pathway tokens) or perturbation experiment alterations (indicated by perturbation tokens). To represent position-wise condition tokens, we utilize an input vector that shares the same dimension as the input genes. This vector is denoted as:

$$\boldsymbol{t}_c^{(i)} = \left[t_{c,1}^{(i)}, t_{c,2}^{(i)}, \ldots, t_{c,M}^{(i)}\right], \tag{4}$$

where $t_{c,j}^{(i)}$ represents an integer index corresponding to a condition.

Embedding layers We utilize the conventional embedding layers[1] $\texttt{emb}_\texttt{g}$ and $\texttt{emb}_\texttt{c}$ for the gene tokens and condition tokens, respectively, to facilitate the mapping of each token to a fixed-length embedding vector of dimension D. We employ fully connected layers, denoted as $\texttt{emb}_\texttt{x}$, for the binned expression values to enhance expressivity. This choice enables the modeling of the continuum of gene expression values. Consequently, the final embedding $h^{(i)} \in \mathbb{R}^{M \times D}$ for cell i is defined as,

[1] pytorch embedding layer

$$\boldsymbol{h}^{(i)} = \texttt{emb}_\texttt{g}(\boldsymbol{t}_g^{(i)}) + \texttt{emb}_\texttt{x}(\boldsymbol{x}^{(i)}) + \texttt{emb}_\texttt{c}(\boldsymbol{t}_c^{(i)}). \tag{5}$$

4.2 Cell and gene expression modeling by transformers

4.2.1 scGPT Transformer

We employ the self-attention transformer [60, 18] to encode the complete input embedding $\boldsymbol{h}^{(i)}$ in equation 5. The self-attention mechanism operates on the sequence of M embedding vectors, making it particularly suitable for capturing interactions between genes. The output of the stacked transformer blocks can be defined as follows:

$$\begin{aligned} \boldsymbol{h}_0^{(i)} &= \boldsymbol{h}^{(i)} \\ \boldsymbol{h}_l^{(i)} &= \texttt{transformer\_block}(\boldsymbol{h}_{l-1}^{(i)}) \quad \forall l \in [1, n] \end{aligned} \tag{6}$$

We utilize the resulting representation $\boldsymbol{h}_n^{(i)} \in \mathbb{R}^{M,D}$ for both gene-level and cell-level tasks. Gene-level finetuning objectives (See Online Methods 4.4) are directly applied. Examples include the gene expression prediction (GEP) objective and the perturbed expression prediction task (perturb-GEP). For cell-level tasks, we first integrate $\boldsymbol{h}_n^{(i)}$ into a cell embedding vector. (See Online Methods 4.2.2). An example would be the cell type assignment task, where the cell embeddings are used to predict cell type labels by an added classifier in the CLS training objective.

The input dimension M can reach tens of thousands of genes, significantly exceeding the input length of conventional transformers commonly used in NLG. To address this challenge and ensure efficient self-attention mechanisms, we leverage more advanced approaches such as Flash-Attention [16]. This implementation effectively enhances the model capacity and enables effective processing of large-scale input dimensions. Other efficient transformers can also be utilized, such as Transformers with linear complexity (Linformer) [61] and Kernelized Self-Attention (KSA) [28].

4.2.2 Cell representation

Each cell is considered a "sentence" composed of genes, and its representation $\boldsymbol{h}_c^{(i)} \in \mathbb{R}^D$ is obtained by aggregating the learned gene-level representations $\boldsymbol{h}_n^{(i)}$. Various pooling operations, such as element-wise mean-pooling or weighted-pooling, can be readily employed in this context. In this study, we opt to employ a special token $< cls >$ for the cell representation, enabling the model to learn the pooling operation within transformer blocks. The $< cls >$ token is appended to the beginning of the input tokens, and the final embedding at this position is extracted as the cell representation. Consequently, we have the cell embedding $\boldsymbol{h}_c^{(i)}$ equals to a row in the stacked final-layer embeddings $\boldsymbol{h}_n^{(i)}[< cls >]$, where the $[< cls >]$ operation retrieves the row at the index corresponding to the $< cls >$ token in the input.

4.2.3 Condition tokens for batch and modality

We use additional sets of tokens to represent different batches and sequencing modalities, specifically for the scRNA-seq and scMultiomic integration tasks. This is similar to conditon tokens

introduced in Online Methods 4.1, and implemented similarly using the standard embedding layers. The modality tokens $\boldsymbol{t}_m^{(i)}$ are associated with individual input features g_j (e.g., to indicate whether it is a gene, region or protein). The batch tokens are on the cell level originally but can be propagated to all features of a single cell as well. In other words, the same batch token $t_b^{(i)}$ can be repeated up to the length M of input features of single cell i:

$$\boldsymbol{t}_b^{(i)} = \left[t_{b,1}^{(i)}, t_{b,2}^{(i)}, \ldots, t_{b,M}^{(i)}\right] = \left[t_b^{(i)}, t_b^{(i)}, \ldots, t_b^{(i)}\right]. \tag{7}$$

The difference between the tokens described in Online Methods 4.1 and the batch and modality tokens is that these embeddings are not used as input to the transformer blocks. Instead, they are concatenated with the transformer output on either feature or cell level prior to entering specific fine-tuning objectives. This is to prevent the transformer from amplifying the attention within features of same modalities while underestimating those of different modalities. Furthermore, knowing the modality and/or batch identities facilitates gene expression modelling in the downstream fine-tuning objectives. As the model learns to predict expression values conditioned on modality and/or batch identities, such biases are implicitly removed from the gene and cell representations themselves. This serves as a technique to facilitate batch correction.

As an example, in the scMultiomic integration task, we concatenate the transformer output with the sum of batch and modality embeddings. This serves as input to the downstream fine-tuning objectives for expression modelling:

$$\boldsymbol{h}_n'^{(i)} = concat(\boldsymbol{h}_n^{(i)}, \texttt{emb}_\texttt{b}(\boldsymbol{t}_b^{(i)}) + \texttt{emb}_\texttt{m}(\boldsymbol{t}_m^{(i)})), \tag{8}$$

where $\texttt{emb}_\texttt{b}$ and $\texttt{emb}_\texttt{m}$ denote the batch and modality embedding layers respectively.

Alternatively, in the scRNA-seq integration task, concatenation of batch embedding with the cell representation yields the following representation:

$$\boldsymbol{h}_c'^{(i)} = concat(\boldsymbol{h}_c^{(i)}, \texttt{emb}_\texttt{b}(t_b^{(i)})), \tag{9}$$

where $t_b^{(i)}$ denotes the batch identity of cell i.

4.3 Generative pre-training

4.3.1 Foundation model pre-training

The foundation model is designed to be a generalizable feature extractor that can benefit a diverse range of downstream tasks. It contains the entire set of genes in the human genome. The expression values were normalized, log-transformed, and binned prior to model training (See Online Methods 4.1). To speed up the training, we restrict the input to only genes with non-zero expressions for each input cell. This strategy provides useful pre-trained results contributing to the subsequent finetuning stage, where we include all genes with zero expressions as well by default. To efficiently train the model to capture gene-gene relation and gene-cell relation, we introduce a new generative training strategy with specialized attention masks as detailed in the following section.

4.3.2 Attention mask for generative pre-training

Self-attention has been widely used to capture the co-occurrence patterns among tokens. In natural language processing, this has been achieved mainly in two ways: (1) masked token prediction used in transformer encoder models such as BERT [18] and Roberta [33], where randomly masked tokens in the input sequence are predicted in the model's output; (2) auto-regressive generation with sequential prediction in causal transformer decoder models such as the OpenAI GPT series [48, 49, 6, 43]. The generative pre-training used in OpenAI GPT3 [6] and GPT4 [43] employs a unified framework in which the model predicts the most likely next token from a "prompt" consisting of known input tokens. This framework offers great flexibility to be utilized in various natural language generation (NLG) applications and demonstrates new capabilities such as contextual awareness in zero-shot and few-shot settings [7]. We believe that the generative training can be beneficial to single-cell models in a similar manner. Specifically, we are interested in two tasks: (1) generating unknown gene expression values based on known gene expressions, i.e., generation by "gene prompts", and (2) generating whole genome expressions given an input cell type condition, i.e., generation by "cell prompts".

Despite similar usage of tokens and prompts, modelling genetic reads is inherently different from natural language due to the non-sequential nature of the data. Unlike words in a sentence, the order of genes within a cell is interchangeable, and there is no equivalent concept of "next gene" to predict. This makes it challenging to apply the causal masking formulation from GPT models directly in single-cell domain. To address this challenge, we developed a specialized attention masking mechanism for scGPT that defines the order of prediction based on attention scores.

The scGPT's attention mask supports both gene-prompt and cell-prompt generations in a unified way. The binary attention mask is applied on the self-attention map in the transformer blocks. For an input $\boldsymbol{h}_l^{(i)} \in \mathbb{R}^{M \times D}$ of M tokens (See Online Methods 4.2.1), the transformer block will generate M query and key vectors to compute the attention map, $\boldsymbol{A} \in \mathbb{R}^{M \times M}$. The attention mask is of the same size $M \times M$. We visualize the attention mask in Supplementary Figure S1A, where queries are organized in rows and keys in columns. The token identity associated with each column of the mask is annotated at the bottom of the figure, namely $< cls >$, known genes, and unknown genes. Each token in the input embedding $\boldsymbol{h}_l^{(i)}$ can be one of these three groups: (1) the reserved $< cls >$ token for cell embedding (introduced in Online Methods 4.2.2), (2) known genes with token embeddings and expression value embeddings, and (3) unknown genes whose expression values are to be predicted. The rule of thumb for scGPT's attention-masking is to only allow attention computation between embeddings of the "known genes" and the query gene itself. In each generation iteration, scGPT predicts the gene expression values of a new set of genes, and these genes in turn become the "known genes" in the next iteration for attention computation. This approach reflects the casual masking design with next token prediction in the conventional transformer decoders by making sequential predictions in the non-sequential single-cell data.

As illustrated in Supplementary Figure S1A, during training, we randomly pick a ratio of the genes as unknown so their expression values are omitted in the input. The queries on the positions of these unknown genes are only allowed with attention computation on the known genes and the query gene itself. For example, the last gene to predict at position M has attention scores with the cell embedding, known genes and itself only, but not the other unknown genes, as illustrated in the last row of the attention mask. The scGPT model predicts the expressions for these unknown genes via the stacked transformer blocks with the masked attention map described above. The inference steps are illustrated in Supplementary Figure S1B. During the inference for cell-prompt generation, scGPT generates all genome-wide gene expressions conditioned on the specific cell types. A trained cell embedding is inputted at the first position representing the cell type condition. The whole generation process of thousands of gene expressions is conducted in K iterative steps (i.e., $K = 3$

steps in Supplementary Figure S1B). For example, in one iteration $i \in \{1, 2, \dots K\}$, the attention masking mechanism allows attention with all predicted genes from previous 0 to $i-1$ iterations. In each iteration, scGPT selects the top $1/K$ genes from the unknown set with the highest prediction confidence to be included as known genes in the next iteration $i+1$. Intuitively, this workflow streamlines the generation of large groups of gene expressions in an auto-regressive manner, where gene expressions with highest prediction confidence are first generated and used to help subsequent rounds of generation. The gene-prompt generation works similarly in an iterative manner. The difference is that it starts with a set of known genes with observed expression values, instead of a cell embedding.

The scGPT attention masking unifies the encoding process of known genes and the generation on unknown genes. It also stands as one of the first transformer schemes to conduct auto-regressive generation for non-sequential data.

4.4 Fintuning objectives

scGPT leverages various fine-tuning objectives to facilitate the learning of biologically valid representations of cells and genes, as well as for regularization purposes such as batch correction.

Gene Expression Prediction (GEP) To encourage the learning of gene-gene interactions, scGPT incorporates gene expression prediction. Within each cell, a subset of genes and their corresponding expression values $\boldsymbol{x}^{(i)}$ are randomly masked. scGPT is optimized to accurately predict the expression values at the masked positions. This fine-tuning objective benefits the model in effectively encoding co-expressions among the genes in the dataset. Specificcaly, we employ a fully connected MLP to estimate the expression value for M genes, on the transformer output. The optimization of this objective involves utilizing the cross entropy loss at the masked positions, denoted as $\mathcal{M}_{mask}$. The GEP works as follows,

$$\begin{aligned} \tilde{\boldsymbol{x}}^{(i)} &= \texttt{MLP}(\boldsymbol{h}_n^{(i)}), \\ \mathcal{L}_{GEP} &= \frac{1}{|\mathcal{M}_{mask}|} \sum_{j \in \mathcal{M}_{mask}} \texttt{ce}(\tilde{x}_j^{(i)}, x_j^{(i)}). \end{aligned} \tag{10}$$

Here, $\tilde{\boldsymbol{x}}^{(i)} \in \mathbb{N}^M$ represents the row of expression estimates for cell i, and $\texttt{ce}$ denotes the cross entropy function. It is worth noting that in integration tasks, we use $\boldsymbol{h}_n'^{(i)}$ in Equation 8 instead of $\boldsymbol{h}_n^{(i)}$.

GEP presents a general self-supervised finetuning objective, which aims to forecast gene expression values. In certain downstream tasks, such as the perturbation prediction, the model is required to predict perturbed gene expression values instead of the original values. We refer to this variation as **perturb-GEP**. We maintain the MLP estimator in equation 10, but utilize the gene expressions post-perturbation as the target $x_j^{(i)}$. In perturb-GEP, the predicted expression values are simply altered to apply to all valid target positions instead of solely the masked positions in GEP.

Gene Expression Prediction for Cell Modelling (GEPC) This finetuning objective operates similarly to GEP, but predicts gene expression values based on the cell representation $\boldsymbol{h}^{(i)}c$ to

explicitly foster cell representation learning. For each gene j in an input cell i, we create a query vector $\boldsymbol{q}_j$ and utilize the parameterized inner product of $\boldsymbol{q}_j$ and the cell representation $\boldsymbol{h}^{(i)}c$ as the predicted expression value.

$$\begin{aligned} \boldsymbol{q}_j &= \texttt{MLP}(\texttt{emb}_\texttt{g}(\boldsymbol{t}_g^{(i)})), \\ \tilde{x}_j^{(i)} &= \boldsymbol{q}_j \cdot \boldsymbol{W}\boldsymbol{h}_c^{(i)}, \\ \mathcal{L}_{GEPC} &= \frac{1}{|\mathcal{M}_{mask}|} \sum_{j \in \mathcal{M}_{mask}} \texttt{ce}(\tilde{x}_j^{(i)}, x_j^{(i)}) \end{aligned} \tag{11}$$

GEPC inherits the gene token embedding, $\texttt{emb}_\texttt{g}(\boldsymbol{t}^{(i)}g)$, from Equation equation 5. In integration tasks, we utilize $\boldsymbol{h}c'^{(i)}$ from Equation 9 instead of $\boldsymbol{h}_c^{(i)}$, and we also concatenate the modality and/or batch embeddings to $\texttt{emb}_\texttt{g}$. In our experiments, we observed that combining GEP and GEPC leads to significantly improved performance compared to using either method individually.

Elastic Cell Similarity (ECS) This finetuning objective enhances cell representations through the utilization of a similarity learning loss [31]:

$$\mathcal{L}_{ECS} = -(\texttt{sim}(\boldsymbol{h}_c^{(i)}, \boldsymbol{h}_c^{(i')}) - \beta)^2, \tag{12}$$

where $\texttt{sim}$ represents the cosine similarity function, while i and i' refer to two cells within the mini-batch. Additionally, β denotes a predefined threshold. The underlying idea behind this approach is to enhance the similarity between pairs exhibiting cosine similarity values above β, thereby making them even more similar. Conversely, dissimilar pairs are encouraged to be further apart.

Domain Adaptation via Reverse Back-propagation (DAR) Cell representation learning is hindered by the presence of batch effects, which result from non-biological batch differences introduced by sequencing technologies [19, 59]. To mitigate this problem, we employ a distinct multi-layer perceptron (MLP) classifier to predict the sequencing batch associated with each input cell, and modify the back-propagation process by reversing the gradients within the model. This approach leverages insights from the robust domain adaptation method proposed by Ganin and Lempitsky [20].

Cell Type Classification (CLS) This finetuning objective is designed to leverage the learned cell representations to annotate single cells. We use a separate MLP classifier to predict the cell types from their cell representations $\boldsymbol{h}_c^{(i)}$. This finetuning objective is optimized with cross entropy loss *ce* between the predicted cell type probabilities and ground-truth labels.

4.5 Finetuning on downstream tasks

Batch correction on integrating multiple scRNA-seq datasets Batch effects can be a major confounder in cell type clustering when the input contains multiple datasets from different sequencing batches or technologies. Therefore, we aim to correct batch effects while preserving biological variances when integrating multiple scRNA-seq datasets. For finetuning on this integration task, the common set of gene tokens between the pre-trained foundation model and the current

dataset were retained. We further selected a subset of highly variable genes from the common set as input. The gene expression values were normalized, log-transformed and binned prior to model training. All pre-trained model weights were used to initialize the finetuned model. All gene tokens with both zero and non-zero expression values were used in training. In addition to GEP and GEPC, the ECS, DAR and DSBN finetuning objectives were optimized simultaneously for enhanced cell contrastive learning and explicit batch correction through reverse back-propagation and domain-specific normalization.

Cell type annotation For the cell type annotation task, we finetuned the model on a reference set with ground-truth labels, and validated the annotation performance on an external query set. The common set of gene tokens between the pre-trained foundation model and the reference set was retained. We pre-processed the expression values prior to model training similar to the integration task. All pre-trained model weights were used to initialize the finetuned model. All gene tokens with both zero and non-zero expression values were used in training. The CLS finetuning objective was used to minimize the classification loss.

Perturbation prediction Gene editing techniques applied to scRNA-seq experiments have revealed cellular responses to various genetic perturbations. However, the vast combinatorial space of potential gene perturbations quickly surpasses the limits of feasible experimentation. Consequently, machine learning approaches have been employed to leverage known perturbations and predict unknown ones. To fine-tune the perturbation prediction task, we initially selected highly variable genes and pre-processed the expression values prior to model training. All pre-trained model weights were utilized for initializing the fine-tuned model. During training, all gene tokens with both zero and non-zero expression values were included. To achieve this, we adopted the perturb-GEP finetuning objective with two modifications to the training setup. Firstly, instead of utilizing the masked and unmasked versions of the same cell as input and learning target, we employed a control cell as the input and the perturbed cell as the target. This was achieved by randomly pairing a non-perturbed control cell with each perturbed cell to construct the input-target pairs. Secondly, rather than randomly masking gene positions as in the original GEP setting, we employed the target perturbed genes as positions for prediction. The input values consisted of the non-perturbed gene expression values rather than mask values. Consequently, the model learned to predict the post-perturbation responses based on the control gene expressions.

Integrative representation learning for scMultiomic data scMultiomic data may contain different sequencing modalities in each batch, which presents a more challenging scenario for integrative analysis. We examined two data integration settings, paired and mosaic, for scMultiomic data. In the paired setting, all samples (cells) share all the data modalities sequenced. In the mosaic setting, some batches share a few common data modalities but not all. Due to the presence of additional ATAC and/or protein tokens, we inherited the trained gene embeddings for RNA data only, and trained the additional token embeddings and rest of the model from scratch. All tokens with both zero and non-zero expression values were used in training. We used an additional set of modality tokens to indicate the data type of each token (i.e., gene, region, or protein) and to facilitate the masked gene and value prediction in GEP and GEPC finetuning objectives (See Online Methods 4.2.3). In the paired setting, the model was optimized with GEP and GEPC finetuning objectives. In the mosaic setting, DAR was included to facilitate multi-modal batch correction.

Gene Regulatory Network Inference In the zero-shot setting, we extracted the gene similarity network from scGPT models's gene embeddings based on cosine similarities. In the finetuned setting, we constructed the gene networks in a similar manner from the scGPT model finetuned on the Immune Human dataset. Following Ceglia et al. [10]'s pipelines, we further extracted gene programs from the gene embedding clusters that consist of five or more genes. See Online Methods 4.7 for more details on gene network analysis and validation.

4.6 Datasets

CELLxGENE scRNA-seq human PBMC Collection We retrieved over 10.3 million human PBMC scRNA-seq samples from the CELLxGENE portal [11] for foundation model pre-training. A total of 65 datasets were collected from CELLxGENE by filtering on Organism (i.e., Homo sapiens), Tissue (i.e., blood, bone marrow), and Disease (i.e., normal, COVID-19, influenza).

PBMC 10K The PBMC 10K dataset comprises two scRNA-seq batches of human peripheral blood mononuclear cells (PBMCs) obtained from a healthy donor. The dataset was re-processed by Gayoso et al. [21], resulting in the identification of 3,346 differentially expressed genes. The first batch encompasses 7,982 cells, while the second batch encompasses 4,008 cells. The cell groups annotated using Seurat [55] consist of 9 categories, namely B cells, CD4 T cells, CD8 T cells, CD14+ Monocytes, Dendritic Cells, NK cells, FCGR3A+ Monocytes, Megakaryocytes, and Other.

Immune Human The Immune Human dataset encompasses five scRNA-seq datasets: one derived from human bone marrow and four from human peripheral blood. Various sequencing technologies were employed, including 10X Genomics, 10X Genomics v2, 10X Genomics v3, and Smart-seq2. The dataset comprises a total of 33,506 cells and includes 12,303 genes. The ten distinct batches were defined based on the origin of the donors. The harmonized data encompass 16 cell groups. We used the data re-processed and the annotations provided by Luecken et al. [36].

hPancreas The hPancreas dataset contains five scRNA-seq datasets of human pancreas cells, re-processed by Chen et al. [12] for the cell type assignment task. The five datasets were split into reference and query sets by data sources. The reference set consists of Braon[5] and Muraro[39] datasets, and the inference set consists of Xin[63], Segerstolpe[56], and Lawlor[30] datasets. The reference and query sets both have 3,000 genes, and ground-truth annotations retained from their original publications. The reference set contains 10,600 cells of 13 cell groups (alpha, beta, ductal, acinar, delta, PSC, PP, endothelial, macrophage, mast, epsilon, schwann, and t_cell). The query set contains 4,218 cells of 11 cell groups (alpha, beta, ductal, PP, acinar, delta, PSC, endothelial, epsilon, mast, MHC class II). Note that MHC class II is a new cell type in the query set not previously seen in the reference set.

Adamson The Adamson perturbation dataset contains gene expression data from the K562 leukemia cell line perturbed by Pertub-seq [1]. This dataset includes 87 unique one-gene perturbations, each replicated in around 100 cells.

Norman The Norman perturbation dataset contains gene expression data from the K562 leukemia cell line perturbed by Pertub-seq [41]. This dataset has 131 two-gene perturbations and 105 one-

gene perturbations. Each perturbation is replicated in around 300-700 cells.

10X Multiome PBMC The 10X Multiome PBMC dataset [14] contains paired single-cell RNA and ATAC data on human PBMC cells sequenced by the 10X Single Cell Multiome protocol. In this dataset, all samples came from the same healthy donor. Each cell contains both gene expression and chromatin accessibility measurements. The processed data by Cao and Gao [9] contains 9,631 cells with counts from 29,095 genes and 107,194 regions. The annotations include 19 cell groups (CD14 Mono, CD16 Mono, CD4 Naive, CD4 TCM, CD4 TEM, CD8 Naive, CD8 TEM_1, CD8 TEM_2, HSPC, Intermediate B, MAIT, Memory B, NK, Naive B, Plasma, Treg, cDC, gdT, and pDC).

ASAP PBMC The ASAP PBMC dataset contains four sequencing batches with three data modalities (gene expression, chromatin accessibility, and protein abundance) [38]. The four batches each contain 5,023, 3,666, 3,517, and 4,849 cells respectively. In batches 1 and 2, all samples have 4,768 genes and 216 protein measurements from CITE-seq. In batches 3 and 4, all samples have 17,742 regions and the same 216 protein measurements from ASAP-seq. The annotations by [38] contain 4 cell groups (Bcell, Myeloid, NK, and Tcell).

4.7 Experiment Setup

scRNA-seq batch integration In this work, we compared the performance of scGPT with three other methods, namely Seurat [55], Harmony [29], and scVI [34]. The evaluation covers batch correction and cell type clustering on two integration datasets: PBMC 10K [21] and Immune Human [36]. Harmony and scVI are highlighted as the top-performing methods in the recent integration benchmark conducted by Luecken et al. [36]. To ensure a fair comparison, all methods were provided with the same number of 1,200 highly variable genes as input. Gene expression values were normalized per cell by considering the total counts across all genes and subsequently log-transformed. The integrated cell embeddings were obtained after the completion of training and were used for evaluation.

The evaluation of the integrated cell embeddings was performed using biological conservation metrics proposed by Luecken et al. [36]. These metrics include the normalized mutual information (NMI_{cell}), adjusted Rand index (ARI_{cell}), and average silhouette width (ASW_{cell}). These scores measure the consistency between the derived cell type clusters and the ground truth labels. For easier comparison, we also computed the average of these metrics, referred to as $AvgBIO$. Additionally, we reported the batch correction metrics proposed by Luecken et al. [36] to assess batch mixing. The batch correction performance was quantified using the inverse of the average silhouette width for batch clustering, denoted as *ASWbatch*, and the graph connectivity measure, denoted as *GraphConn*. We computed *AvgBATCH* as the average of *ASWbatch* and *GraphConn* to summarize the batch mixing performance. Furthermore, we introduced an *Overall* score, which is a weighted sum of *AvgBIO* and *AvgBATCH*, consistent with the approach taken by [36]. See Supplementary Online Methods S.2 for details of metric calculations.

scRNA-seq cell type annotation We benchmarked scGPT against the recent transformer-based cell type annotation method TOSICA [12] on the hPancreas dataset. We used the same pre-processed reference and query sets by Chen et al. [12] for model training and validation. The predicted cell type labels on the query set were retrieved for evaluation.

We evaluated cell type assignment performance based on four standard classification metrics, *Accuracy*, *Precision*, *Recall*, and $MacroF1$. *Accuracy*, *Precision*, and *Recall* are calculated globally for overall performance, whereas $MacroF1$ is averaged per class to increase the weighing of rare cell types. We also reported a normalized confusion matrix with *Precision* by cell type for additional details. See Supplementary Online Methods S.2 for details on metric calculations.

scRNA-seq perturbation We compare scGPT against the recent perturbation prediction method GEARS [53]. To ensure consistency, we followed the pre-processing steps outlined by Roohani, Huang, and Leskovec [53] in their benchmark. Initially, gene expression values were normalized per cell using the total counts across all genes, and a logarithmic transformation was applied. Subsequently, we selected 5,000 highly variable genes and incorporated any perturbed genes that were not initially considered into the gene set. In the experiments, for one-gene perturbations in both datasets Adamson et al. [1] and Norman et al. [41], the perturbations are split to ensure that test perturbations are not seen in training, i.e., no cells in training set has undergone any of the test perturbations. For two-gene perturbations in the Norman et al. [41] dataset, the train-test split consists of three scenarios with increasing difficulty: (1) 0/2 unseen genes, (2) 1/2 unseen genes, and (3) 2/2 unseen genes in the training set.

To evaluate the accuracy of perturbation prediction, we employed the Pearson correlation coefficient (*corr*) between the predicted gene expressions and the ground-truth expression values. Additionally, we calculated a variant of the Pearson metric based on the amount of change in expression post-perturbation compared to the control, denoted as *corr(*Δ*)*. Furthermore, we reported these Pearson metrics for different gene sets, including all genes (ALL), and the top 20 differentially expressed genes (DE). Thus, we presented four evaluation metrics in total, namely *corr* and *corr(*Δ*)* for the ALL and DE conditions, respectively. See Supplementary Online Methods S.2 for details of metric calculation.

scMultiomic integration We benchmarked scGPT in two integration settings, paired and mosaic, against the recent scMultiomic integration methods Seurat v4 [24], scGLUE [9] and scMoMat [65] respectively. In the paired data integration experiment, we benchmarked scGPT with scGLUE [9] and Seurat v4 [24] on the 10X Multiome PBMC [14] dataset. The same 1,200 highly variable genes and 4,000 highly variable peaks were used as input to all methods. In the mosaic data integration experiment, we benchmarked scGPT with scMoMat [65] on the ASAP PBMC [38] dataset. The same 1,200 highly variable genes, 4,000 highly variable peaks, and all 216 protein features were used as input to both methods. While keeping the input feature set consistent, we used each method's custom pre-processing pipeline to normalize the expression values. The integrated cell embeddings were retrieved for evaluation after training.

In both paired and mosaic data integration settings, we evaluated cell embedding quality on the four biological conservation metrics NMI_cell, ARI_cell, AWS_cell, and $AvgBIO$. In the mosaic data integration setting, we further evaluated mixing of different omic batches with the three batch correction metrics AWS_batch, $GraphConn$, and Avg_BATCH. An overall score was also reported on the mosaic integration experiment. See Supplementary Online Methods S.2 for details on metric calculation.

Gene Regulatory Network Inference We validated scGPT's gene similarity network against the known HLA and CD antigen networks. For each network, we first defined the related gene set by filtering on gene names with set prefixes (i.e., HLA- and CD-). We then filtered on genes involved in the Immune System R-HSA-168256 pathway from the Reactome 2022 database [50]. For the CD

antigens, we used the common genes with the selected HVG set from the Immune Human dataset for the ease of comparison between pre-trained and finetuned models. We then extracted gene embeddings of these selected genes from the scGPT model and constructed a similarity network based on cosine similarity. We highlighted sub-networks of strong connections by selecting edges with cosine similarities greater than a certain threshold (i.e., 0.5 for HLA and 0.4 for CD antigen network). We then compared the sub-networks against known functional groups in immunology. Furthermore, we evaluated the gene similarity relationships encoded by the scGPT model with Reactome. Following Ceglia et al. [10]'s pipelines, we first evaluated whether the neighbors of a gene involved in a known pathway belong to that pathway. Using CD8A gene as an example, we ranked its 10 nearest neighbors (including itself) by cosine similarity in the selected HVG set from the Immune Human dataset, and examined their membership in the Immune System R-HSA-168256 pathway. Subsequently, on a system level, we examined the relationship between cosine similarity of pairs of genes and the number of common pathways that a gene pair is involved in. We reported the Pearson correlation score between cosine similarity score and pathway coverage on the entire gene set across all pathways in Reactome.

4.8 Implementation Details

The pretrianed foundation model has an embedding size of 512. It consists of 12 stacked transformer blocks with 8 attention heads each. The fully connected layer has hidden size of 512. In pre-training, we randomly split the data and used 97% (10 million) of the data for training and 3% (0.3 million) for validation. We set the ratio of genes to generate to be uniformly sampled from three options of 0.25, 0.50 and 0.75. The model was optimized by the Adam optimizer, using a mini-batch size of 32, at a starting learning rate of 0.0001 and a 0.9 weight decay after each epoch. The model was trained for a total of 6 epochs.

For the tasks of scRNA-seq batch integration, cell type annotation, and perturbation prediction, we utilized the same model configuration inherited from the pre-trained model. During the fine-tuning process, we initiated with a learning rate of 0.0001, which decayed to 90% after each epoch. The mask ratio for GEP and GEPC was set to 0.4, while the parameter β in ECS was set to 0.6. When combined with other losses, ECS was assigned a weighting of 10. To divide the datasets into training and evaluation sets, we employed a ratio of 9:1. The model was trained for a fixed duration of 30 epochs, and after each epoch, the GEP loss value was evaluated on the validation set. The reported results correspond to the model with the best validation score. Notably, for the perturbation task (refer to Section section 2.4), we observed that the model typically converged within 3 epochs, and we report the best-validated model accordingly.

For the multi-omic integration task, we loaded the gene embeddings from the pre-trained model and used the same embedding size of 512 for all tokens (i.e., including gene, ATAC-peak, and/or protein). The main model is set to have 4 stacked transformer blocks with 8 attention heads each, and a hidden layer size of 512. Each dataset is split into train and evaluation sets at 9:1 ratio. We used DAR weighing of 1.0 for batch integration. We used a starting learning rate of 0.001 and a weight decay of 0.95 after each epoch. We trained the model for fixed 60 epochs and similarly reported the best-validated model.

We used the SCANPY python library [62] for gene expression pre-processing, including normalization, log-transformation and highly variable gene selection. We used the EpiScanpy python library [15] on chromatin accessibility data for highly variable peak selection. In the scRNA-seq batch integration and scMultiomic integration tasks, the evaluation metrics are calculated using the implementation in `scib.metrics` by Luecken et al. [36]. In the cell-annotation task, the evaluation metrics are implemented using the `scikit-learn` package.

5 Acknowledgement

We would like to express our sincere gratitude to, Dr. Nan Duan, for his invaluable guidance and support throughout the project. We appreciate the valuable feedback from Dr. Lin Zhang during the writing of the manuscript.

References

[1] Britt Adamson et al. "A multiplexed single-cell CRISPR screening platform enables systematic dissection of the unfolded protein response". In: *Cell* 167.7 (2016), pp. 1867–1882.

[2] Philipp Angerer et al. "Single cells make big data: New challenges and opportunities in transcriptomics". In: *Current opinion in systems biology* 4 (2017), pp. 85–91.

[3] Dvir Aran et al. "Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage". In: *Nature immunology* 20.2 (2019), pp. 163–172.

[4] Britta Bade et al. "Differential expression of the granzymes A, K and M and perforin in human peripheral blood lymphocytes". In: *International immunology* 17.11 (2005), pp. 1419–1428.

[5] Maayan Baron et al. "A single-cell transcriptomic map of the human and mouse pancreas reveals inter-and intra-cell population structure". In: *Cell systems* 3.4 (2016), pp. 346–360.

[6] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[7] Sébastien Bubeck et al. *Sparks of Artificial General Intelligence: Early experiments with GPT-4*. 2023. arXiv: 2303.12712 [cs.CL].

[8] Lars Buitinck et al. "API design for machine learning software: experiences from the scikit-learn project". In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 2013, pp. 108–122.

[9] Zhi-Jie Cao and Ge Gao. "Multi-omics single-cell data integration and regulatory inference with graph-linked embedding". In: *Nature Biotechnology* 40.10 (2022), pp. 1458–1466.

[10] Nicholas Ceglia et al. "GeneVector: Identification of transcriptional programs using dense vector representations defined by mutual information". In: *bioRxiv* (2022), pp. 2022–04.

[11] Chanzuckerberg Initiative. *CZ CELLxGENE Discover*. https://cellxgene.cziscience.com/. Online; accessed 26 December 2022. 2022.

[12] Jiawei Chen et al. "Transformer for one stop interpretable cell type annotation". In: *Nature Communications* 14.1 (2023), p. 223.

[13] Paola Cruz-Tapias, John Castiblanco, and Juan-Manuel Anaya. "Major histocompatibility complex: antigen processing and presentation". In: *Autoimmunity: From Bench to Bedside [Internet]*. El Rosario University Press, 2013.

[14] Darren A Cusanovich et al. "Multiplex single-cell profiling of chromatin accessibility by combinatorial cellular indexing". In: *Science* 348.6237 (2015), pp. 910–914.

[15] Anna Danese et al. "EpiScanpy: integrated single-cell epigenomic analysis". In: *Nature Communications* 12.1 (2021), p. 5228.

[16] Tri Dao et al. "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness". In: *arXiv preprint arXiv:2205.14135* (2022).

[17] Jurrian K De Kanter et al. "CHETAH: a selective, hierarchical cell type identification method for single-cell RNA sequencing". In: *Nucleic acids research* 47.16 (2019), e95–e95.

[18] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[19] Michael Eisenstein. "Single-cell RNA-seq analysis software providers scramble to offer solutions." In: *Nature Biotechnology* 38.3 (2020), pp. 254–257.

[20] Yaroslav Ganin and Victor Lempitsky. "Unsupervised domain adaptation by backpropagation". In: *International conference on machine learning*. PMLR. 2015, pp. 1180–1189.

[21] Adam Gayoso et al. "A Python library for probabilistic analysis of single-cell omics data". In: *Nature Biotechnology* 40.2 (2022), pp. 163–166.

[22] Suchin Gururangan et al. "Don't stop pretraining: Adapt language models to domains and tasks". In: *arXiv preprint arXiv:2004.10964* (2020).

[23] Xiaoping Han et al. "Mapping the mouse cell atlas by microwell-seq". In: *Cell* 172.5 (2018), pp. 1091–1107.

[24] Yuhan Hao et al. "Integrated analysis of multimodal single-cell data". In: *Cell* 184.13 (2021), pp. 3573–3587.

[25] Ashraful Haque et al. "A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications". In: *Genome medicine* 9.1 (2017), pp. 1–12.

[26] HCA. *HCA DCP*. https://data.humancellatlas.org/. Online; accessed 12 April 2023. 2023.

[27] Yuge Ji et al. "Machine learning for perturbational single-cell omics". In: *Cell Systems* 12.6 (2021), pp. 522–537.

[28] Angelos Katharopoulos et al. "Transformers are rnns: Fast autoregressive transformers with linear attention". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5156–5165.

[29] Ilya Korsunsky et al. "Fast, sensitive and accurate integration of single-cell data with Harmony". In: *Nature methods* 16.12 (2019), pp. 1289–1296.

[30] Nathan Lawlor et al. "Single-cell transcriptomes identify human islet cell signatures and reveal cell-type–specific expression changes in type 2 diabetes". In: *Genome research* 27.2 (2017), pp. 208–222.

[31] Chundi Liu et al. "Guided similarity separation for image retrieval". In: *Advances in Neural Information Processing Systems* 32 (2019).

[32] Jiajia Liu et al. "Machine intelligence in single-cell data analysis: advances and new challenges". In: *Frontiers in Genetics* 12 (2021), p. 655536.

[33] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[34] Romain Lopez et al. "Deep generative modeling for single-cell transcriptomics". In: *Nature methods* 15.12 (2018), pp. 1053–1058.

[35] Malte D Luecken and Fabian J Theis. "Current best practices in single-cell RNA-seq analysis: a tutorial". In: *Molecular systems biology* 15.6 (2019), e8746.

[36] Malte D Luecken et al. "Benchmarking atlas-level data integration in single-cell genomics". In: *Nature methods* 19.1 (2022), pp. 41–50.

[37] Zhen Miao et al. "Multi-omics integration in the age of million single-cell data". In: *Nature Reviews Nephrology* 17.11 (2021), pp. 710–724.

[38] Eleni P Mimitou et al. "Scalable, multimodal profiling of chromatin accessibility, gene expression and protein levels in single cells". In: *Nature biotechnology* 39.10 (2021), pp. 1246–1258.

[39] Mauro J Muraro et al. "A single-cell transcriptome atlas of the human pancreas". In: *Cell systems* 3.4 (2016), pp. 385–394.

[40] Philip S Norman. "Immunobiology: The immune system in health and disease". In: *Journal of Allergy and Clinical Immunology* 96.2 (1995), p. 274.

[41] Thomas M Norman et al. "Exploring genetic interaction manifolds constructed from rich single-cell phenotypes". In: *Science* 365.6455 (2019), pp. 786–793.

[42] Sergio Oller-Moreno et al. "Algorithmic advances in machine learning for single-cell expression analysis". In: *Current Opinion in Systems Biology* 25 (2021), pp. 27–33.

[43] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: 2303.08774 [cs.CL].

[44] OpenAI. *CZ CELLxGENE Discover*. https://openai.com/blog/chatgpt. Online; accessed 10 April 2023. 2023.

[45] OpenAI. *CZ CELLxGENE Discover*. https://openai.com/product/dall-e-2. Online; accessed 10 April 2023. 2023.

[46] Aditya Pratapa et al. "Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data". In: *Nature methods* 17.2 (2020), pp. 147–154.

[47] Xipeng Qiu et al. "Pre-trained models for natural language processing: A survey". In: *Science China Technological Sciences* 63.10 (2020), pp. 1872–1897.

[48] Alec Radford et al. "Improving language understanding by generative pre-training". In: (2018).

[49] Alec Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[50] Reactome. *Reactome Pathway Database: Home*. https://reactome.org/. 2022.

[51] Aviv Regev et al. "Science forum: the human cell atlas". In: *elife* 6 (2017), e27041.

[52] Dennis Thompson Healthday Reporter. *'human cell atlas' maps 1 million cell types in 33 organs*. 2022. URL: https://medicalxpress.com/news/2022-05-human-cell-atlas-million.html.

[53] Yusuf Roohani, Kexin Huang, and Jure Leskovec. "GEARS: Predicting transcriptional outcomes of novel multi-gene perturbations". In: *bioRxiv* (2022).

[54] Abhishek Sarkar and Matthew Stephens. "Separating measurement and expression models clarifies confusion in single-cell RNA sequencing analysis". In: *Nature genetics* 53.6 (2021), pp. 770–777.

[55] Rahul Satija et al. "Spatial reconstruction of single-cell gene expression data". In: *Nature biotechnology* 33.5 (2015), pp. 495–502.

[56] Åsa Segerstolpe et al. "Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes". In: *Cell metabolism* 24.4 (2016), pp. 593–607.

[57] Indhupriya Subramanian et al. "Multi-omics data integration, interpretation, and its application". In: *Bioinformatics and biology insights* 14 (2020), p. 1177932219899051.

[58] Chi Sun et al. "How to fine-tune bert for text classification?" In: *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*. Springer. 2019, pp. 194–206.

[59] Hoa Thi Nhu Tran et al. "A benchmark of batch-effect correction methods for single-cell RNA sequencing data". In: *Genome biology* 21.1 (2020), pp. 1–32.

[60] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[61] Sinong Wang et al. "Linformer: Self-attention with linear complexity". In: *arXiv preprint arXiv:2006.04768* (2020).

[62] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. "SCANPY: large-scale single-cell gene expression data analysis". In: *Genome biology* 19.1 (2018), pp. 1–5.

[63] Yurong Xin et al. "RNA sequencing of single human islet cells reveals type 2 diabetes genes". In: *Cell metabolism* 24.4 (2016), pp. 608–615.

[64] Allen W Zhang et al. "Probabilistic cell-type assignment of single-cell RNA-seq for tumor microenvironment profiling". In: *Nature methods* 16.10 (2019), pp. 1007–1015.

[65] Ziqi Zhang et al. "scMoMaT jointly performs single cell mosaic integration and multi-modal bio-marker detection". In: *Nature Communications* 14.1 (2023), p. 384.

S Supplementary

S.1 Benchmarking results on downstream tasks

Dataset	Model	Biological Conservation				Batch Correction			
		AvgBIO	NMI_{cell}	ARI_{cell}	ASW_{cell}	**AvgBATCH**	ASW_{batch}	$GraphConn$	**Overall**
Immune Human [36]	scGPT	**0.746**	0.824	0.784	0.630	0.894	0.817	0.970	**0.804**
	scVI [34]	0.726	0.803	0.787	0.587	**0.923**	0.871	0.975	0.805
	Seurat [55]	0.564	0.697	0.441	0.555	0.883	0.858	0.908	0.692
	Harmony [29]	0.743	0.810	0.832	0.588	0.914	0.859	0.968	0.811
PBMC 10K [21]	scGPT	**0.812**	0.834	0.869	0.732	0.940	0.949	0.931	**0.863**
	scVI	0.695	0.786	0.704	0.593	**0.950**	0.971	0.930	0.797
	Seurat	0.753	0.810	0.854	0.595	0.934	0.937	0.931	0.826
	Harmony	0.751	0.810	0.855	0.589	0.945	0.967	0.923	0.829

Table 2: scRNA-seq Integration Benchmark Results. scGPT was benchmarked with scVI [34], Seurat [55], and Harmony [29] on the Immune Human (10 batches) [36] and PBMC 10K (2 batches) [21] datasets for cell type clustering and batch correction performance. We present three aggregate scores $AvgBIO$, $AvgBATCH$, and $Overall$. These aggregate scores were calculated from three detailed biological conservation metrics (NMI_{cell}, ARI_{cell}, ASW_{cell}) and two batch correction metrics (ASW_{batch}, $GraphConn$). See metric details in Supplementary Online Methods S.2

Dataset	Model	Classification Metrics			
		Accuracy	*Precision*	*Recall*	*MacroF*1
hPancreas [12]	scGPT	**0.967**	0.785	0.694	0.705
	TOSICA [12]	0.961	0.641	0.667	0.641

Table 3: Cell Type Annotation Benchmark Results. scGPT was benchmarked with TOSICA [12] on the hPancreas [12] dataset for cell type annotation performance. We present four classification evaluation metrics *Accuracy*, *Precision*, *Recall*, and *MacroF*1. See metric details in Supplementary Online Methods S.2.

Dataset	Model	Biological Conservation				Batch Correction			Overall
		AvgBIO	NMI_{cell}	ARI_{cell}	ASW_{cell}	**AvgBATCH**	ASW_{batch}	*GraphConn*	
10X Multiome PBMC [14] (Paired)	scGPT	**0.767**	0.818	0.822	0.661	-	-	-	-
	scGLUE [9]	0.747	0.815	0.806	0.619	-	-	-	-
	Seurat v4 [24]	0.722	0.784	0.691	0.691	-	-	-	-
ASAP PBMC [38] (Mosaic)	scGPT	**0.562**	0.601	0.472	0.614	**0.948**	0.904	0.992	**0.716**
	scMoMat [65]	0.546	0.448	0.557	0.633	0.916	0.849	0.983	0.667

Table 4: scMultiomic Integration Benchmark Results. For the paired 10X Multiome PBMC [14] dataset, scGPT was benchmarked with scGLUE [9] and Seurat v4 [24] for cell type clustering performance evaluated on four biological conservation metrics. Batch correction metrics are not applicable to this setting. For the mosaic ASAP PBMC [38] dataset, scGPT was benchmarked with scMoMat [65] for cell type clustering and multiomic integration performance, evaluated on eight biological conservation and batch correction metrics.

S.2 Evaluation Metric Calculations

S.2.1 Single-cell integration

We adopted the evaluation metric calculations outlined by Luecken et al. [36] in their benchmark study. Each metric is described below.

Normalized Mutual Information

To quantify the concurrence between the cell type labels based on ground truth and the Louvain cluster labels obtained from integrated cell embeddings, we computed the normalized mutual information (NMI) score. The Louvain clustering was conducted across resolutions ranging from 0.1 to 2, with increments of 0.1. The best score will be selected. The NMI score for cell types, referred to as $\boldsymbol{NMI_{cell}}$, ranges between 0 and 1, where a higher score indicates a better match of cell types.

Adjusted Rand Index

The adjusted rand index (ARI) was employed to assess both the agreement between the annotated labels and the MNI-optimized Louvain clusters. Furthermore, the rand index was adjusted to account for randomly correct labels. The ARI score for cell types, denoted as $\boldsymbol{ARI_{cell}}$, ranges from 0 to 1, where 0 corresponds to random labeling and 1 represents a perfect match.

Average Silhouette Width

The silhouette width assesses the relationship between a cell's within-cluster distances and its distances to the closest cluster boundaries. By averaging the silhouette widths of all cells, we calculate the average silhouette width (ASW) score. This score ranges from -1 to 1, where a score of 1 indicates well-separated clusters, while scores from -1 to 0 suggest overlapping clusters and misclassification.

For evaluating cell type clustering, we compute the ASW score based on cell type labels, represented as $\boldsymbol{ASW_{cell}}$. To obtain this score, we utilize the following formula:

$$ASW_{cell} = (ASW_C + 1)/2$$

Here, C represents the cell types.

Regarding batch mixing evaluation, we calculate the ASW score considering batch labels and adjust it by subtracting 1. This score is denoted as $\boldsymbol{ASW_{batch}}$. The calculation is as follows:

$$ASW_{batch} = 1 - |ASW_B|$$

Both $\boldsymbol{ASW_{cell}}$ and $\boldsymbol{ASW_{batch}}$ have values between 0 and 1. Higher scores indicate better cell-type clustering or batch-mixing performance.

Graph Connectivity

The graph connectivity metric quantifies the average proportion of cells within each cell type that are connected through a kNN (k-nearest neighbors) graph. For every cell identity c in the set C, we compute the size of the largest connected component using kNN among cells exclusively

belonging to identity c. This value is divided by the total number of cells with identity c to obtain a normalized measure. The ***GraphConn*** score is then reported as the average across all cell types:

$$GraphConn = \frac{1}{|C|} \sum_{c \in C} \frac{|LCC(G_c^{kNN})|}{N_c}$$

Here, LCC represents the largest connected component, and N denotes the number of cells of each celltype.

Aggregated Metrics

The aggregated metric ***AvgBIO*** calculates the average of biological conservation metrics:

$$AvgBIO = (ARI_{cell} + NMI_{cell} + ASW_{cell})/3$$

Similarly, the aggregated metric ***AvgBATCH*** computes the average of batch mixing metrics:

$$AvgBATCH = (ASW_{batch} + GraphConn)/2$$

In accordance with the convention established in [36], an ***Overall*** metric is derived as the weighted average of ***AvgBIO*** and ***AvgBATCH***:

$$AvgBATCH = 0.6 * AvgBIO + 0.4 * AvgBATCH$$

S.2.2 Cell Type Assignment

We used the standard classification metrics $Accuracy$, $Precision$, $Recall$, and $MacroF1$ to evaluate cell type assignment performance. The $Accuracy$, $Precision$, $Recall$, and $MacroF1$ scores are calculated from true positives (tp), false positives (fp), true negatives (tn), and false negatives (fn) globally or averaged per class.

The $Accuracy$, $Precision$ and $Recall$ scores are calculated globally:

$$Accuracy = \frac{tp}{tp + fp + tn + fn}, \quad Precision = \frac{tp}{tp + fp}, \quad Recall = \frac{tp}{tp + fn} \ .$$

The $MacroF1$ score is calculated per cell type c first and averaged across cell types:

$$MacroF1 = \sum_{c \in C} \frac{F1_c}{N_c}, \quad \text{where} \quad F1_c = \frac{2 \times Precision_c \times Recall_c}{Precision_c + Recall_c}.$$

The above metrics are calculated using `scikit-learn`'s implementations [8].

Supplementary Figures

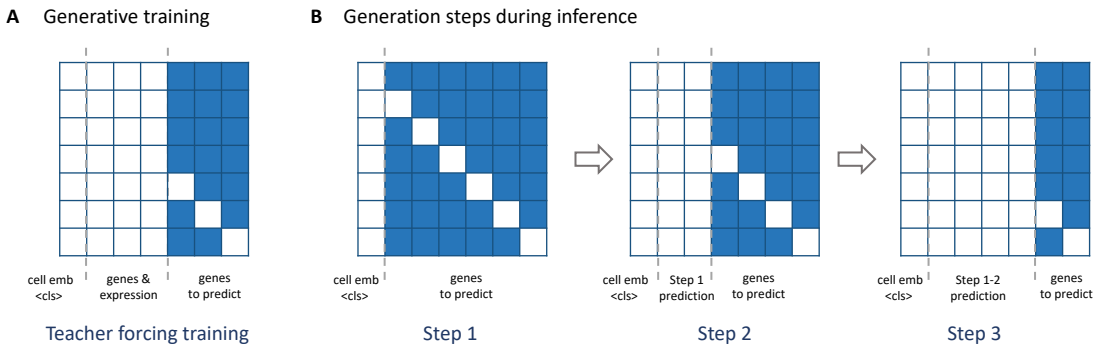

Figure S1: The scGPT Attention Mask. The masked positions are colored in blue, and the allowed positions in white. These masked and unmaksed positions correspond to the $M \times M$ attention map for M input tokens. The row indices correspond to queries and columns correspond to keys. In the self-attention computation of transformers, the attention scores on the marked positions will be removed. The token identity associated with each column is annotated below, namely cell emb $< cls >$ for cell embedding, genes & expressions for known genes, and genes to predict for unknown genes. *(a)* scGPT attention mask in training where only query gene and the known genes participate in attention computation. *(b)* At inference time, the attention mask at each step during the iterative process of scGPT cell-prompt generation.

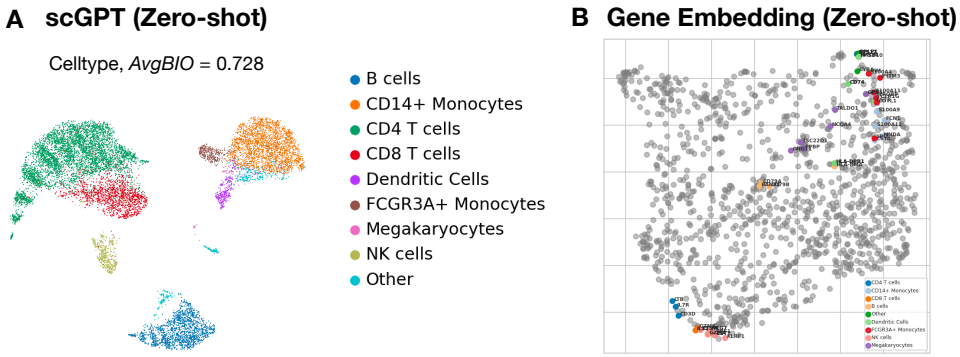

Figure S2: Visualization of Zero-shot scGPT Learned Cell and Gene Embeddings on the PBMC 10K Dataset. *(a)* UMAP visualization of cell embeddings colored by *cell types*. *(a)* UMAP visualization of gene embeddings. The highly variable genes corresponding to each *celltype* were colored accordingly.

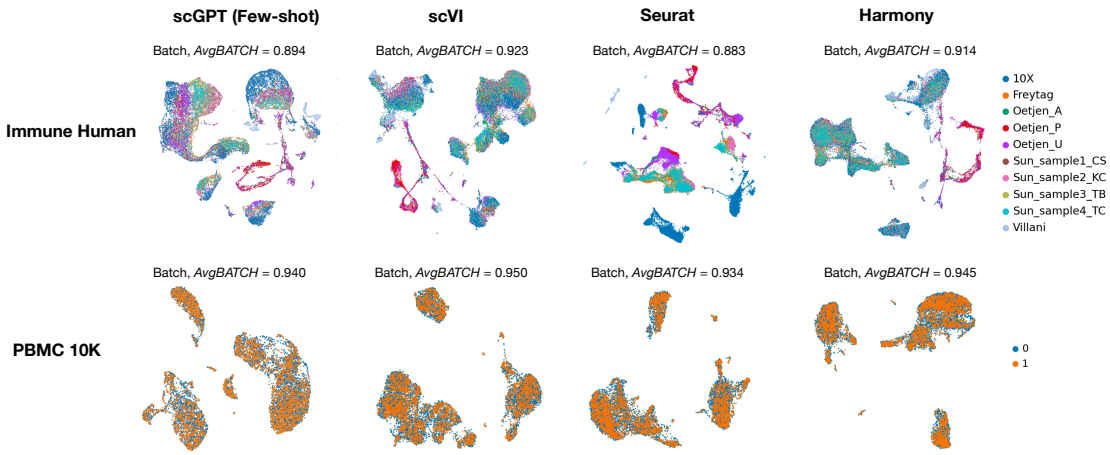

Figure S3: Benchmark of scGPT with scVI [34], Seurat [55], and Harmony [29] on the Immune Human (10 batches) [36] and PBMC 10K (2 batches) [21] Datasets for Batch Correction. UMAP visualization of cell embeddings colored by *sequencing batches*.