xTrimoGene: An Efficient and Scalable Representation Learner for Single-Cell RNA-Seq Data

Jing Gong [*1] **Minsheng Hao** [*2] **Xin Zeng** [1] **Chiming Liu** [1] **Taifeng Wang** [1] **Jianzhu Ma** [2] **Xuegong Zhang** [2] **Xingyi Cheng** [1] **Le Song** [1 3]

Abstract

The advances in high-throughput sequencing technology have led to significant progress in measuring gene expressions at the single-cell level. The amount of publicly available single-cell RNA-seq (scRNA-seq) data is already surpassing 50M records for human with each record measuring 20,000 genes. This highlights the need for unsupervised representation learning to fully ingest these data, yet classical transformer architectures are prohibitive to train on such data in terms of both computation and memory. To address this challenge, we propose a novel asymmetric encoder-decoder transformer for scRNA-seq data, called xTrimoGene, which leverages the sparse characteristic of the data to scale up the pre-training. This scalable design of xTrimoGene reduces FLOPs by one to two orders of magnitude compared to classical transformers while maintaining high accuracy, enabling us to train the largest transformer models over the largest scRNA-seq dataset today. Our experiments also show that the performance of xTrimoGene improves as we increase the model sizes, and it also leads to SOTA performance over various downstream tasks, such as cell classification, perturb-seq effect prediction, and drug combination prediction.

1. Introduction

Single-cell RNA sequencing (scRNA-seq) technology has transformed the field of cell biology and enabled us to understand cell-cell, cell-gene and gene-gene relations at the cellular level (Jovic et al., 2022; Chen et al., 2019). This technique captures the expression levels of thousands of genes in parallel, facilitating the study of cellular heterogeneity (Chen et al., 2022; Li et al., 2022). Integrating and modeling such large-scale scRNA-seq data can reveal rich cellular information and benefit various biological task learning.

Representation learning from scRNA-seq data (Flores et al., 2022) has been an active area of research in past decades. The first published pre-trained model for single-cell data is scBERT, which uses a low-rank transformer (Yang et al., 2022) to analyze the scRNA data. It learns the cellular representation by randomly masking a percent of non-zero gene expression values and tries to recover them. scBERT has achieved state-of-the-art results for cell-type annotation tasks. The study shows the potential of a pre-training strategy for single-cell biology research. However, scBERT has certain limitations in fully utilizing scRNA-seq data properties. These limitations include:

(1) Scalability. The large number of genes (almost 20,000) and the sparsity of scRNA-seq data, with nearly 90% of values being zero, lead to many redundant computations (e.g., self-attentions between zero tokens). (2) Limited resolution for expression values. scBERT rounds the gene expression values into integer values, which limits the model's ability to distinguish closeness and similarity between gene expression values. The strategy leads to a loss of resolution and introduces bias during model training, resulting in sub-optimal performance.

scRNA-seq data are characterized by high dimensional (approximately 20,000 genes), high sparsity (90% zero in a typical dataset (Ruochen Jiang, 2022; Jiarui Ding, 2020)) and the gene expression values are continuous scalars. To address the challenges associated with scRNA-seq data modeling and consider the unique nature of this data, we present a novel and efficient framework, xTrimoGene, for pre-training large-scale scRNA-seq data. Our framework makes the following key contributions:

(1) We design an asymmetrical encoder-decoder architecture to guide the pre-training process, which enables us to learn a high-capacity model for single-cell RNA-seq data. Our

[*]Equal contribution [1]BioMap Research, California, USA [2]Tsinghua University, Beijing, China [3]Mohamed bin Zayed University of Artificial Intelligence, Masdar City, Abu Dhabi. Correspondence to: Le Song <songle@biomap.com>.

The 2023 ICML Workshop on Computational Biology. Baltimore, Maryland, USA, 2023.

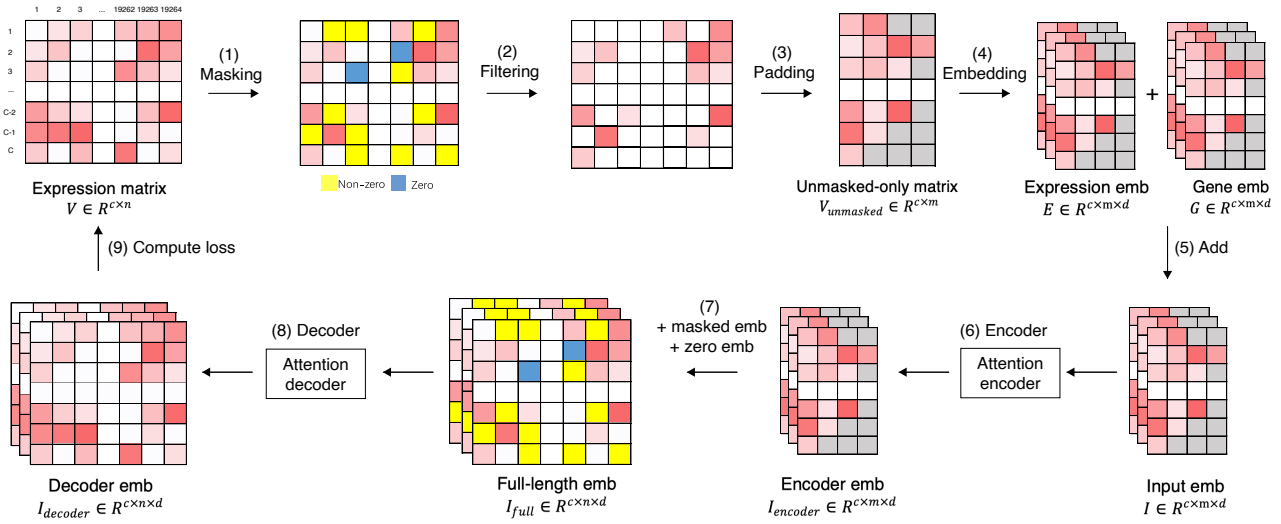

Figure 1. The xTrimoGene Framework: (1) Random positions (including both zero and non-zero values) are masked for prediction. (2) Masked and zero-valued positions are filtered out. (3) Remaining unmasked positions are aligned with padding tokens (grey) to ensure maximum length consistency within a batch. (4) Gene expression values and gene embeddings are separately projected into embeddings. (5) These two embeddings are element-wise added. (6) The resulting input is fed into the encoder. (7) The intermediate encoder embedding is combined with embeddings for masked positions and zero embeddings. (8) This combined representation is then fed into the decoder. (9) Decoder embedding is projected to model output with a MLP layer. The MSE loss is calculated between the model output and ground truth values for the masked positions. Detailed description can be referred to appendix A.1.

model achieves an improvement in the speed of pre-training of over 3 times compared to previous encoder-only models.

(2) We illustrate that the efficiency and scalability of our model allow us to train the largest single-cell pre-trained model to date, with approximately 100 million parameters for the xTrimoGene-100M model, using a curated scRNA-seq dataset of approximately 50 billion effective gene tokens.

(3) The pre-trained model xTrimoGene achieved remarkable results in multiple downstream tasks, including cell type annotation, perturbation prediction and synergistic drug combination prediction.

2. xTrimoGene Architecture

xTrimoGene is a highly efficient framework for pre-training large-scale single-cell RNA-seq data (illustrated in Figure 1 and appendix A.1). The training process is based on a regression masked task, aimed at accurately recovering masked values in the expression matrix (appendix A.2.1). The setting is more fitted than previous classification task (appendix Figure 3). We also ablated the masking strategy (appendix Table 3) and observed both agreements and discrepancies with NLP tasks (appendix Figure 4,5,6). Notably, a specific optimized asymmetrical encoder-decoder framework is employed to accelerate the learning of sparse matrices. This is achieved by feeding only the unmasked non-zero positions (less than 10% of the full length) into the encoder, while the largely masked and zero positions are input into a lightweight decoder with a reduced number of layers and attention heads. A similar rational design has been proven powerful in masked autoencoders (MAE) (He et al., 2021), which is tailored for CV data pre-training. In addition, a novel auto-discretization strategy is introduced to project continuous expression values into a latent embedding space. Instead of rounding to the nearest integer, values are directly mapped to the latent space allowing for the representation of closely-related values. The strategy has been valid effective (appendix Figure 7) and advantageous (appendix Figure 8) over previous binning methods.

3. xTrimoGene achieves high computational efficiency and scalability

We quantitatively compared the training cost of xTrimoGene with other two encoder-only models, including full-length attention Transformer and kernel-based approximation Performer (scBERT). We observed that total FLOPs for Performer decreased to 10% of native Transformer (see Table 1). Notably, xTrimoGenes runs a 3-times faster than Performer. The results validate the efficiency of xTrimoGene, which is

readily adapted for large-scale data pre-training.

Table 1. Computational efficiency comparison between different algorithms. The resource column is normalized by the Transformer row.

Model name	Parameter (M)	Total train (FLOPs)	Resource
Transformer	11.3	2.46E+20	100%
Performer	8.9	2.65E+19	10.8%
xTrimoGene	9.8	8.38E+18	3.4%

The Deep Learning community has shown significant interest in the scalability of proposed models (Kaplan et al., 2020; Brown et al., 2020). To test the scale-up ability of xTrimoGene, we pre-trained three models across multiple compute regions and scales (e.g., from 3M to 100M parameters, appendix Table 4). The training curve clearly shows all models are steadily down to a lower loss when training steps increase (appendix Figure 9). More importantly, xTrimoGene-100M model obtains a significant improvement over xTrimoGene-10M model, which is also superior over xTrimoGene-3M model. The tendency is consistent across different data size. The results suggest xTrimoGene framework is robust to scale-up, making it possible and convenient to pre-train larger models with more data.

4. xTrimoGene demonstrates superior performance on both single cell and bulk level downstream tasks

Currently, multiple tasks have been established to evaluate different models, including cell type annotation and recently developed perturbation response prediction tasks. We first assessed the performance of xTrimoGene on these single-cell tasks. Additionally, we explored the potential application on bulk RNA-sequencing data, with a focus on synergistic drug combination prediction.

4.0.1. Cell type annotation

First, we evaluated xTrimoGene's performance on cell type annotation task with Zheng68K (Zheng et al., 2017) and Segerstolpe (Segerstolpe et al., 2016) dataset, which has been widely benchmarked. We compared the xTrimoGene against other several methods, including scBERT (Yang et al., 2022), ACTINN (Ma & Pellegrini, 2020), Scanpy (Wolf et al., 2018), CellTypist (Domínguez Conde et al., 2022), scVI (Lopez et al., 2018) and singleCellNet (Yuqi Tan, 2019). For xTrimoGene model, we added a max-pooling layer and a linear layer to predict cell type labels with fine-tuning mode (appendix A.5.1). For other methods, we followed their instruction with the default parameter setting. We observed that xTrimoGene achieves a high Precision and F1 score, surpassing all the other methods (Table 2). The results indicated xTrimoGene learns a well-represented cellular embedding by simply aggregating contextual gene embedding.

4.0.2. Perturbation response prediction

The perturbation response prediction task evaluates what is the expression value of genes after perturbation. GEARS (Roohani et al., 2022) is a newly developed method for this purpose. We compared the native GEARS (Roohani et al., 2022) model with and without incorporating embeddings from xTrimoGene (appendix A.5.2). The evaluated dataset (Norman et al., 2019) contains both single and double gene perturbation and we thus assess the performance across different perturbation levels. As shown in Figure 2A, GEARS with xTrimoGene embedding scores a lower MSE (decreased 14.8%) for top20 differential expressed genes. Notably, the tendency is consistent across different perturbation levels, regardless the perturbed target is seen or not. The results demonstrated that the pre-training strategy empowers xTrimoGene to capture constraints under various circumstances, including post-perturbations. The application further proved the efficacy and potential of xTrimoGene to boost scRNA-seq based tasks.

4.0.3. Synergistic drug combinations prediction

The drug synergistic task evaluates how patients or cells respond to a drug combination intervention (Mokhtari et al., 2017). Since the generated wet-lab experimental data only covers a tiny search space of possible drug combinations, multiple models have been proposed to accelerate predicting the synergistic landscape of drugs (Preuer et al., 2017; Wang et al., 2022). Similar to the perturbation prediction test, we adapted xTrimoGene to DeepDDS (Wang et al., 2022) with the intermediate context embedding (appendix A.5.3). We also included DeepSynergy and Random Forest for comparison. As illustrated in Figure 2B, utilizing embedding from xTrimoGene model outperforms all the other models. The result proved xTrimoGene can accurately capture cell level representation, even for bulk sequencing data. This also opens the avenue for xTrimoGene to be applied across other biological modeling tasks, especially where bulk level transcriptome data is available.

5. Explainable case study of biological representation

The pre-training task endows the model to capture the relationship between genes, which can be demonstrated by a biologically meaningful binarized vector. Specifically, we collected the cell type-specific marker genes of B cells and Fibroblast from PanglaoDB (Franzén et al., 2019), and

Table 2. The cell annotation performance on the Zheng68K and Segerstolpe dataset. xTrimoGene is evaluated with 10M parameter model.

Method Name	Zheng68K		Segerstolpe	
	Precision	F1 score	Precision	F1 score
xTrimoGene	0.7335 ± 0.0226	$\mathbf{0.7354} \pm 0.0189$	$\mathbf{0.8112} \pm 0.0009$	$\mathbf{0.8140} \pm 0.0008$
scBERT	0.7029 ± 0.0115	0.6695 ± 0.0077	0.6818 ± 0.0736	0.6703 ± 0.0653
ACTINN	0.6720 ± 0.0021	0.6486 ± 0.0041	0.7545 ± 0.0018	0.7219 ± 0.0073
Scanpy	0.6111 ± 0.0017	0.5474 ± 0.0085	0.6274 ± 0.0000	0.5398 ± 0.0000
CellTypist	$\mathbf{0.7454} \pm 0.0009$	0.7151 ± 0.0038	0.7923 ± 0.0003	0.8117 ± 0.0001
scVI	0.4883 ± 0.0005	0.4843 ± 0.0008	0.5101 ± 0.0022	0.5208 ± 0.0016
singleCellNet	0.6452 ± 0.0013	0.5982 ± 0.0027	0.7551 ± 0.0096	0.8055 ± 0.0076

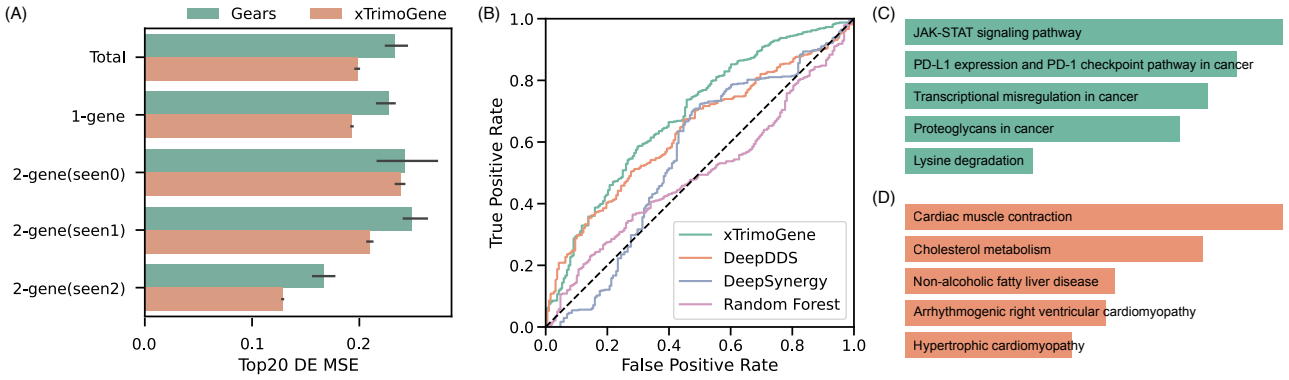

Figure 2. Evaluation and explainable study of xTrimoGene. (A) The MSE of the top 20 deferentially expressed (DE) genes for different models on perturbation response prediction. "Total" denotes evaluating all test perturbation set. "1-gene" denotes sub-test set on the single gene perturbation, where the perturbed target is not seen in the training set. "2-gene" represents the sub-test set for perturbing two genes simultaneously. "seen0","seen1" and "seen2" denotes zero, one or two perturbed targets are not seen in the training set, respectively. Black line denotes 95% confidence interval. (B) ROC curve of different models on drug combination synergy prediction task. (C)(D) Bar plot of KEGG enrichment analysis of top expressed genes activated by particular marker genes, for B cell (C) and Fibroblast cell (D), respectively.

selected the top 50 genes with high ubiquitousness index, respectively. We set two zero vectors and made the corresponding positions of the 50 marker genes into 1, for building the artificial binary vectors. Then we input these two vectors into our model. We took out the name of the top 500 highly expressed genes in the model outputs and found 73 genes are specific to each cell type. Then KEGG pathway enrichment analysis was performed on these 73 genes via the enrichR online tools (Chen et al., 2013; Kuleshov et al., 2016). As shown in Figure 2C, the highly expressed genes for B cell are enriched in JAK-STAT, PD-1 and other human immunity and cancer-related pathways, while for Fibroblastthose are enriched in cardiac fibrosis, liver fibrosis and other fibrosis diseases related pathways (Figure 2D). This illustrates the ability of our model to infer the hidden relation between genes in different cell types from very little gene expression information.

6. Conclusion

xTrimoGene is a new, efficient framework for learning scRNA-seq data. The proposed asymmetric encoder-decoder framework takes advantage of the sparse gene expression matrix, and establishes the projection strategy of continuous values with a higher resolution. The results show that xTrimoGene is scalable and performs well on tasks like cell type annotation, perturbation response prediction, and synergistic drug combination prediction. The experiments demonstrate the efficacy of pre-training in single-cell biology. xTrimoGene has been integrated into BioMap's single-cell analysis platform, functioning as a fundamental and essential model (appendix Figure 10). The codebase, pre-trained model, and accompanying training and evaluation pipelines will be publicly available for access on GitHub soon. In the future, with the increase of data, larger pre-trained models are expected to drive more advancements in various downstream task learning.

References

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020.

Chen, E. Y., Tan, C. M., Kou, Y., Duan, Q., Wang, Z., Meirelles, G. V., Clark, N. R., and Ma'ayan, A. Enrichr: interactive and collaborative html5 gene list enrichment analysis tool. *BMC bioinformatics*, 14(1):1–14, 2013.

Chen, G., Ning, B., and Shi, T. Single-cell rna-seq technologies and related computational data analysis. *Frontiers in genetics*, pp. 317, 2019.

Chen, S., Luo, Y., Gao, H., Li, F., Chen, Y., Li, J., You, R., Hao, M., Bian, H., Xi, X., et al. heca: The cell-centric assembly of a cell atlas. *Iscience*, 25(5):104318, 2022.

Chen, T., Xu, B., Zhang, C., and Guestrin, C. Training deep nets with sublinear memory cost. *CoRR*, abs/1604.06174, 2016.

Domínguez Conde, C., Xu, C., Jarvis, L., Rainbow, D., Wells, S., Gomes, T., Howlett, S., Suchanek, O., Polanski, K., King, H., et al. Cross-tissue immune cell analysis reveals tissue-specific features in humans. *Science*, 376 (6594):eabl5197, 2022.

Flores, M., Liu, Z., Zhang, T., Hasib, M. M., Chiu, Y.-C., Ye, Z., Paniagua, K., Jo, S., Zhang, J., Gao, S.-J., et al. Deep learning tackles single-cell analysis—a survey of deep learning for scrna-seq analysis. *Briefings in bioinformatics*, 23(1):bbab531, 2022.

Franzén, O., Gan, L.-M., and Björkegren, J. L. M. PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data. *Database*, 2019, 04 2019. ISSN 1758-0463. baz046.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners, 2021.

Jiarui Ding, Xian Adiconis, S. K. S. M. S. K. C. C. H. N. D. M. T. K. H. M. H. W. T. B. L. T. N. J. Y. H. K. B. B. W. G. A. J. K. S. C. S. L. N. H. O. R.-R. A. K. S. A.-C. V. A. R. . J. Z. L. Systematic comparison of single-cell and single-nucleus rna-sequencing methods. *Nature Biotechnology*, 38(6):737–746, Jun 2020. ISSN 1546-1696. doi: 10.1038/s41587-020-0465-8. URL https://doi.org/10.1038/s41587-020-0465-8.

Jovic, D., Liang, X., Zeng, H., Lin, L., Xu, F., and Luo, Y. Single-cell rna sequencing technologies and applications: A brief overview. *Clinical and Translational Medicine*, 12(3):e694, 2022.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models, 2020.

Kuleshov, M. V., Jones, M. R., Rouillard, A. D., Fernandez, N. F., Duan, Q., Wang, Z., Koplev, S., Jenkins, S. L., Jagodnik, K. M., Lachmann, A., et al. Enrichr: a comprehensive gene set enrichment analysis web server 2016 update. *Nucleic acids research*, 44(W1):W90–W97, 2016.

Li, M., Zhang, X., Ang, K. S., Ling, J., Sethi, R., Lee, N. Y. S., Ginhoux, F., and Chen, J. Disco: a database of deeply integrated human single-cell omics data. *Nucleic acids research*, 50(D1):D596–D602, 2022.

Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., and Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, 2018.

Ma, F. and Pellegrini, M. Actinn: automated identification of cell types in single cell rna sequencing. *Bioinformatics*, 36(2):533–538, 2020.

Mokhtari, R. B., Homayouni, T. S., Baluch, N., Morgatskaya, E., Kumar, S., Das, B., and Yeger, H. Combination therapy in combating cancer. *Oncotarget*, 8(23): 38022, 2017.

Norman, T. M., Horlbeck, M. A., Replogle, J. M., Ge, A. Y., Xu, A., Jost, M., Gilbert, L. A., and Weissman, J. S. Exploring genetic interaction manifolds constructed from rich single-cell phenotypes. *Science*, 365(6455):786–793, 2019. doi: 10.1126/science.aax4438. URL https://www.science.org/doi/abs/10.1126/science.aax4438.

Preuer, K., Lewis, R. P. I., Hochreiter, S., Bender, A., Bulusu, K. C., and Klambauer, G. DeepSynergy: predicting anti-cancer drug synergy with Deep Learning. *Bioinformatics*, 34(9):1538–1546, 12 2017. ISSN 1367-4803.

Rajbhandari, S., Rasley, J., Ruwase, O., and He, Y. Zero: Memory optimizations toward training trillion parameter models. ArXiv, May 2020.

Roohani, Y., Huang, K., and Leskovec, J. Gears: Predicting transcriptional outcomes of novel multi-gene perturbations. *bioRxiv*, 2022.

Ruochen Jiang, Tianyi Sun, D. S. . J. J. L. Statistics or biology: the zero-inflation controversy about scrna-seq

data. *Genome Biology*, 23(1):31, Jan 2022. ISSN 1474-760X. doi: 10.1186/s13059-022-02601-5. URL https://doi.org/10.1186/s13059-022-02601-5.

Segerstolpe, Å., Palasantza, A., Eliasson, P., Andersson, E.-M., Andréasson, A.-C., Sun, X., Picelli, S., Sabirsh, A., Clausen, M., Bjursell, M. K., Smith, D. M., Kasper, M., Ämmälä, C., and Sandberg, R. Single-cell transcriptome profiling of human pancreatic islets in health and type 2 diabetes. *Cell Metab.*, 24(4):593–607, October 2016.

Wang, J., Liu, X., Shen, S., Deng, L., and Liu, H. Deepdds: deep graph neural network with attention mechanism to predict synergistic drug combinations. *Briefings in Bioinformatics*, 23(1):bbab390, 2022.

Wolf, F. A., Angerer, P., and Theis, F. J. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19:1–5, 2018.

Yang, F., Wang, W., Wang, F., Fang, Y., Tang, D., Huang, J., Lu, H., and Yao, J. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature Machine Intelligence*, 4(10):852–866, 2022.

Yuqi Tan, P. C. Singlecellnet: A computational tool to classify single cell rna-seq data across platforms and across species. *Cell Systems*, 19(2):207–213, 2019.

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

A. Methods

A.1. xTrimoGene model framework

The xTrimoGene framework consists of the following components:

Masking: A portion of the normalized gene expression matrix V is masked for prediction, including both zero and non-zero positions. c denotes cell sample size, and n denotes gene number (19,264 in our setting, appendix A.3 for data collection and processing).

Filtering: The masked and zero-valued embeddings are filtered out, yielding a variable-length sequence of valuable information that is prepared for encoding.

Padding: The remaining unmasked positions are aligned with padding tokens, resulting in a much smaller unmasked-only matrix $V_{unmasked}$. m denotes the maximum length of the unmasked sample.

Embedding: Expression value and gene embeddings are separately projected. d denotes the dimension of the embedding. The expression embedding is calculated through an auto-discretization mapping. The gene embedding is retrieved from a randomly initialized lookup table.

Combining Expression and Gene Embeddings: The expression and gene embeddings (E and G) are element-wise added to form the input embedding, which is then fed into the encoder of the model.

Encoding: The sum of the embeddings is input into the encoder, which implements self-attention mechanisms using a Transformer-like architecture.

Extending masked and zero embeddings: The intermediate encoder embedding $I_{encoder}$ is combined with embeddings for masked and zero-value positions.

Decoding: The combined embeddings are processed by the decoder, utilizing self-attention mechanisms instead of the typical casual attention used in NLP decoders.

Loss Computation: Decoder embedding is projected to model output with a MLP layer. The mean squared error (MSE) loss is computed between the predicted masked values from the model and their corresponding ground truth values.

A.1.1. Encoder

The scRNA-seq data is characterized by its high sparsity, with cell information largely concentrated in the non-zero expression values. Thus, the encoder is designed to focus only on the non-zero part of the unmasked matrix, $V_{unmasked}$. The encoder is based on a traditional multi-head attention transformer and takes the combination of value embedding, E, and gene embedding, G, as its in-

put, $I \in \mathbb{R}^{c\times m\times d}$. The value and gene embeddings are similar to the word and positional embeddings in natural language modeling, respectively. The value embedding, E, is generated using the auto-discretization strategy discussed previously, while the gene embedding, G, is retrieved from the embedded vocabulary.

$$\begin{aligned} E &= \text{Autobin}(V_{unmasked} \odot M_{nonzero}) \\ G &= \text{Lookup}(genes) \\ I &= E + G \end{aligned} \tag{1}$$

Then the encoder processes the input embeddings I and generates the high-level gene representations $I_{encoder} \in \mathbb{R}^{b\times m\times d}$ via the multi-head attention mechanism.

$$I_{encoder} = \text{Trm}(f_Q(I), f_K(I), f_V(I)) \tag{2}$$

where f_Q, f_K, f_V are the project functions. Trm denotes the Transformer block.

A.1.2. DECODER

Unlike the encoder which focuses on the main information (non-zero expression values) in the cells, the decoder in the system performs full-length feature abstraction and extraction. The input to the decoder, I_{full}, comprises three token types: the output from the encoder, $I_{encoder}$, the genes with zero expression embs I_{zero}, and the mask token embs I_{masked}.

$$I_{full} = W_p(I_{encoder} \oplus I_{zero} \oplus I_{masked}) + b_p \tag{3}$$

where $\oplus$ represents the concatenation operation, and W_p and b_p are learnable parameters that project the decoder's embedding size.

The decoder transforms the input I_{full} into final gene-level embeddings, $I_{decoder} \in \mathbb{R}^{b\times n\times d}$, and predicts the masked values through a shared linear layer, $W \in \mathbb{R}^{d\times 1}$, applied across all genes. The operations are expressed as follows:

$$\begin{aligned} I_{decoder} &= \text{Trm}((f_Q(I_{full}), f_K(I_{full}), f_V(I_{full})) \\ \tilde{V} &= I_{decoder} \cdot W \end{aligned} \tag{4}$$

A.1.3. AUTO-DISCRETIZATION STRATEGY

The expression value ($V \in \mathbb{R}^{c\times n}$) is transformed into a hidden embedding (E) for addtion with the gene embedding (G) via an auto-discretization block. This block uses a random look-up table ($EXP\_lookup \in \mathbb{R}^{d\times b}$), where d is the embedding dimension and b is the number of tokens (default 100). The expression value is first transformed with a linear layer ($v_1 = V \cdot w_1$), where w_1 is the weight vector, then passed through a leaky ReLU activation ($v_2 = leaky\_relu(v_1)$). A cross-layer projection ($v_3 = w_2 \cdot v_2 + \alpha \cdot v_2$), with weight vector w_2 and scaling mixture factor α, follows. The bin weights are normalized using the Softmax function ($v_4 = softmax(v_3)$). The final output is a weighted combination of individual embeddings from the look-up table ($output = EXP\_lookup \cdot v_4$), where the weights are learnable parameters.

To validate the effectiveness of the expression value projection, we conducted an analysis of viewing the weight distribution pattern for continuous values. Our results showed that the normalized weight distribution of the close values exhibited smooth transitions and that of the distant values being clearly distinguishable (appendix Figure 7). This supports the conclusion that the auto-discretization strategy effectively represents continuous values with high resolution while preserving relatively rich meaning.

We also compared the performance of the proposed auto-discretization strategy with three other discretization methods: (1) Round bin with zero, in which values are rounded to the nearest integer, and zeros are kept as it is, (2) Up bin without zero. Values greater than zero are converted to the nearest ceiling integer, while zero is represented as individual 0. (3) Equal bin. All the values fall into a fixed percentage interval, which is calculated by value distribution and frequency. We evaluated the different strategies on a standard cell clustering task (appendix A.4) and found that the proposed auto-discretization strategy outperformed the others (as shown in appendix Figure 8), demonstrating the importance of high-resolution projections in handling expression values.

A.2. Training strategy

A.2.1. REGRESSION MASKED TASK

The traditional masked language task is a multi-class classification problem, where the predicting target is a single token with limited, naturally distinct categories. In contrast, the normalized gene expression value is a continuous scalar. To fit the data property, we modify the pre-trained learning objective to a regression task, aimed at recovering the absolute value of the masked positions. The loss function employed is the Mean Square Error (MSE) between the ground truth and the predicted values:

$$\text{Loss} = \frac{1}{(n-m)*c}\sum(V_{i,j} - \tilde{V}_{i,j})^2 \tag{5}$$

where n represents the number of all genes, m represents the maximum length of the unmasked positions in a sample, and c represents the number of cells. To evaluate the efficacy of this modification, we compared the regression setting with the classification setting on the cell clustering task. The results indicate that the regression model outperforms the classification model (appendix Figure 3),

providing evidence of the benefits of learning a more fitted representation.

A.2.2. MASKING STRATEGY

We mask both non-zeros and zeros positions though the scRNA-seq expression matrix is highly sparse (where zero percentage is usually over 90%). As the zero positions percentage is much higher than non-zero positions, the masked ratio can't be the same for the two types. Otherwise, the model tends to predict all zeros and still obtains a low error level. We propose to mask an almost equal number of positions for zero and non-zeros positions (appendix Table 3). The setting enforces the model to learn embeddings for all values and not to be dominated by zero representation. We found zero values supervision is necessary to boost the performance (appendix Figure 6), which demonstrates that some zeros represent the true extremely low expression level. This type of zeros is informative to illustrate how the gene abundant behaves inside the cell.

The recovery of masked tokens in NLP is challenging due to the fact that word comprehension relies heavily on long-range interactions rather than local context. Accurate inference of the missing tokens can be achieved at low masking ratios (15%) where the information in the entire sentence is still relatively redundant and encoded by the unmasked tokens. We investigated the density of information needed for the scRNA-seq regression task by training models with different masking ratios (for non-zero values, the ratio was set 10 times higher than for zero values) ranging from 15% to 90% with a 15% interval. The models were then evaluated on the cell clustering task, with the results showing that performance improved first and then degraded as the masking ratio increased. When the masking ratio was close to 30%, the majority of metrics reached a peak (appendix Figure 5). We also found percentage of [MASK] token agrees well with NLP tasks (appendix Figure 4). These results suggest that the scRNA-seq expression vector contains more redundant information than a sentence and highlight the role of hidden regulations between genes in constraining the inference of expression values.

A.2.3. ACCELERATION STRATEGY

The attention mechanism in masked language modeling is computationally expensive for long sequences, as time and space complexity grows quadratically along with sequence length. Though multiple attention architectures have been proposed to reduced the complexity to near linear, it's still slow to train large models with billions of data. We have adopted multiple techniques to boost the training speed as following.

Since FP16 or BFLOAT16 Tensor Core have twice the computational throughput compared to TF32 on NVIDIA Ampere GPU, and, additionally, FP16 training also reduces residual memory consumption, xTrimoGene was conducted mainly with mixed-precision training strategy to optimize computational efficiency.

Distributed Data Parallelism is another training strategy used in our work, which handles large corpus on HPC clusters. In our setting, one single Ampere GPU provides sufficient amounts of memory for one model replica of billions of parameters performing forward and backward passes, and gradient accumulation is used to raise effective batch size to enhance large model training.

To scale up the model size, ZeRO-DP stage two (Rajbhandari et al., 2020) and checkpointing (Chen et al., 2016) techniques are experimentally tested in our setting. The results verified that both strategies reduce the model and residual state memory without expanding training time too much.

A.3. scRNA-seq data collection and processing

Recently, scRNA-seq data is rapidly accumulated and mostly has been uploaded to Gene Expression Omnibus (GEO) repository (https://www.ncbi.nlm.nih.gov/geo/). We major collected data from GEO and processed data with a unified pipeline.

Downloading data and preparing raw count matrix. We first search scRNA-seq related data sets in GEO with multiple keywords, including "scRNA-seq", "single cell RNA-seq", "single cell RNA-seq sequencing". The search processes return a list of GSE ID from different studies. After removing duplicated GSE ID, we downloaded the particular expression or count matrix. Most of the samples provide a raw count matrix. For samples with normalized expression matrices, we converted the matrix back to a count matrix. The conversion strategy is as follows: the minimal non-zero value in the whole normalized matrix is thought to have raw count 1, then all the other normalized values can be converted by scaling to this minimum value.

Matrix mapping to the reference gene list. After preparing all the count matrices, we mapped the matrix to our reference gene list. We downloaded the human protein-coding gene list (about 19,000) from the HUGO Gene Nomenclature Committee (HGNC, https://www.genenames.org/download/archive/), plus common mitochondrial genes, jointly constitute our full reference list (n = 19,264). For each count matrix, values of those genes not mapped in the reference list are filled with zero.

Quality control. To filter low-quality samples, we only keep samples with over 200 genes expressed (i.e., expression vector with non-zero value count greater than 200) for subsequent training and analysis.

Normalization. We followed the standard process in Scanpy (https://scanpy-tutorials.readthedocs.io/en/latest/pbmc3k.html) (Wolf et al., 2018) to obtain the normalized expression value. There are two steps: (1) for each sample normalize the library size to 10,000. (2) scale the values into a log space.

In summary, all the scRNA-seq data are collected from Gene Expression Omnibus (GEO) repository with a keyword searching and data retrieval process. Then the downloaded count matrices are processed with a unified pipeline, including reference gene list mapping, quality control and normalization. In total, we curated about 5 million scRNA-seq data for training. The full data set is randomly split into train, validation and test sets with ratio of 96:2:2.

A.4. Clustering task evaluation and data sets

Cell clustering is an essential task for single-cell research, reflecting the ability of cell embeddings to remove noise and preserve biological signals. In the ablation experiments, we benchmarked the models' clustering performance on two cell-type annotated datasets.

PBMC This dataset is processed by Scanpy python library (Wolf et al., 2018) and contains 2,638 cells. The cell types are annotated by the human with known markers and cover the major immune cells including B cells, CD4 &CD8 T cells, Monocytes, Dendritic cells, Megakaryocytes and NK cells.

Experiments and Evaluation Metric. For every single cell, the expression values are fed into the model and the max pooling layer is applied across all genes' output embedding to get a cell embedding. We then perform the usual single-cell clustering analysis step on these embeddings: 1) build the neighboring graph based on these embeddings 2) use the Leiden algorithm to cluster the cell into groups. Since the Leiden algorithm requires resolution rather than the number of clusters, we used a dichotomy method to find an optimal resolution reaching the number of cell types given in the dataset.

Several evaluation metrics are applied to access the performance of the clustering results in different aspects, including: Adjusted Rand index (ARI), Normalized Mutual Information (NMI), Homogeneity(HOMO), Completeness (CP) and Silhouette Coefficient (SIL). All these metrics are the higher the better. ARI and NMI measures the similarity of the clustering results from the statistics and information entropy theory view, respectively. HOMO and CP are intuitive metrics using conditional entropy analysis. HOMO measures how much the sample in a cluster are similar, and CP measures how much similar samples are put together. SIL measures the similarity of the embeddings to its cluster member compared to other clusters.

A.5. Evaluation on downstream tasks

A.5.1. Cell type annotation task

We downloaded the Zheng68K expression matrix dataset from (Zheng et al., 2017) and mapped the matrix to our reference gene list. Then, the dataset is split into training, validation and test sets with a ratio of 8:1:1. All the methods are trained on the training set and the best model is selected according to the performance on validation set. Evaluation metrics (macro F1-score and marco precision) are calculated for individual testing set.

In training process, the expression matrix is fed into the encoder of the xTrimoGene model and the gene embedding is obtained. Then we used a max-pooling layer to aggregate all gene embeddings into one cell embedding, and use a single linear layer to predict cell types from the embeddings.

A.5.2. Perturbation effect prediction

The Norman dataset is downloaded from a previous study (Roohani et al., 2022). The expression matrix data is mapped to our reference gene list. We reproduced results of GEARS with original codes and settings (https://github.com/snap-stanford/GEARS). All the data processing are the same as GEARS, including data split, pre-post sample pairing strategy and evaluation metrics calculation.

While training GEARS with xTrimoGene, the expression matrix is fed into xTrimoGene and intermediate context embedding is obtained. The context embedding is then input to the co-expression graph network branch, all the other parts remain unchanged.

A.5.3. Drug combination prediction

To test how xTrimoGene adapted to DeepDDS (Wang et al., 2022) for synergistic drug combination prediction, we first reproduced DeepDDS algorithm. Both data and original codes are downloaded from Github repository (https://github.com/Sinwang404/DeepDDs/tree/master). We use data in "new_labels_0.csv" file for training and "independent_set" for testing. The genomic expression data are all mapped to our reference gene list. Models are trained 5 times and evaluated on the testing set. For all metrics, the averaged value and the standard deviation are reported. We keep the overall framework of DeepDDS while testing xTrimoGene. The genomic expression matrix is fed to xTrimoGene and the intermediate context embedding is obtained. The embedding replaces raw expression profile for MLP branch input.

A.6. Website deployment of xTrimoGene model

xTrimoGene has been proven advantageous in gene representation and cell context embedding extraction. To facilitate its wide application for single-cell RNA-seq data analysis, we deployed the xTrimoGene model within Biomap corporation. On the website, the xTrimoGene is implemented as a standard operator and serves multiple downstream tasks, including cell clustering, dimension reduction and batch removal appendix (Figure 10). The interactive page is user-friendly and feasible to evaluate performance with rich visualizations.

B. Appendix tables

Table 3. Masking strategy for gene expression matrix. The gene expression matrix is masked by selecting a predetermined number of positions for prediction. $\sim$ 1,100 positions, including $\sim$600 non-zero and $\sim$540 zero expressions, are masked in a matrix with $\sim$20,000 genes. The performance of the model is evaluated using Mean Squared Error (MSE) loss on these masked positions.

Value	Masked	Unmasked	Total
$\neq 0$	600	1,400	2,000
$= 0$	540	17,460	18,000
sum	1,140 (5.7%)	18,860 (94.3%)	20,000 (100%)

C. Appendix figures

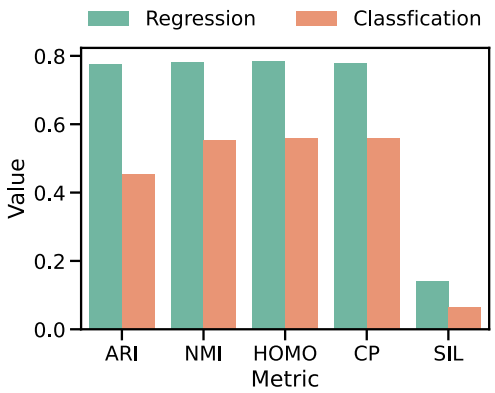

Figure 3. Performance of pre-trained models with different task mode, including regression and classification setting. The cell clustering task is evaluated. Detailed descriptions can be referred to the main text. ARI for Adjusted Rand index, NMI for Normalized Mutual Information, HOMO for Homogeneity, CP for Completeness and SIL for Silhouette Coefficient. The details of the definition and calculation for all metrics are referred to in appendix A.4.

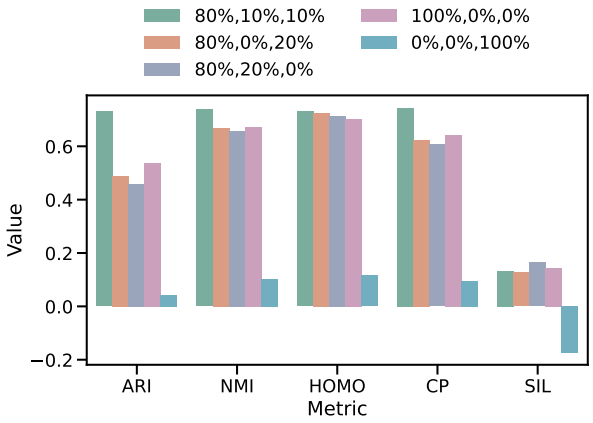

Figure 4. Comparison of performance for xTrimoGene model trained with different masking strategy. percentage1, percentage2, percentage3 denote corresponding replacing probability for three types of tokens: percentage1 for [MASK] token, percentage2 for random expression token and percentage for original token.

Table 4. Size and hyper-parameters of the pre-trained models. All models are set to train on a 5 million datasets and for 5 epochs.

Model name	Parameter (M)	Encoder depth	Encoder heads	Encoder dim	Decoder depth	Decoder heads	Decoder dim
xTrimoGene-3M	3	4	2	128	2	2	128
xTrimoGene-10M	10	4	8	256	2	4	256
xTrimoGene-100M	100	12	12	768	6	8	512

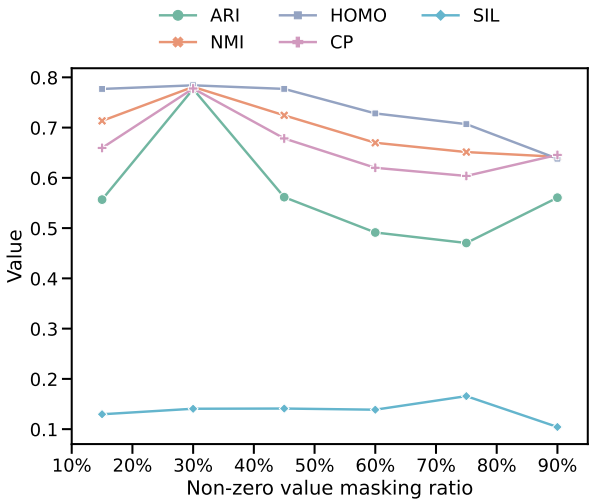

Figure 5. Model performance under different mask ratios of non-zero values. The cell clustering task is evaluated.

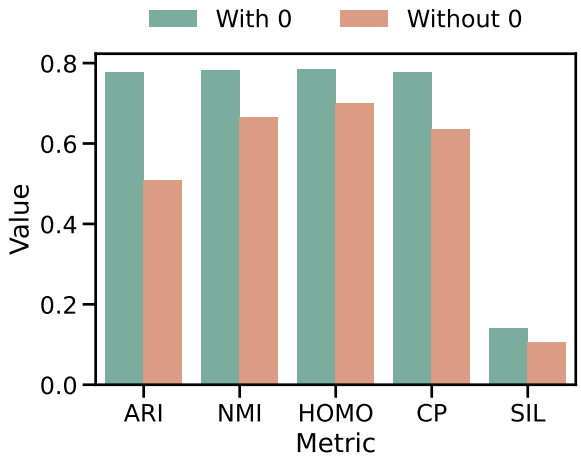

Figure 6. Cell clustering performance for xTrimoGene model considering masking zero (With 0) values or not (Without 0).

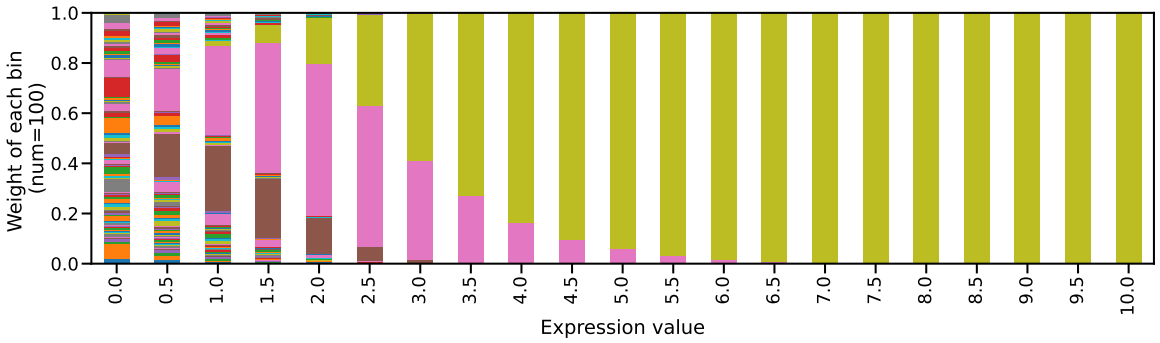

Figure 7. Weight distribution across bins for various expression values. The auto-discretization strategy was applied to each expression value in the range from 0 to 10, producing a corresponding weight vector with a length equal to the number of bins (100 in this case). The weight vectors were normalized to sum to 1 and visualized as stacked plots.

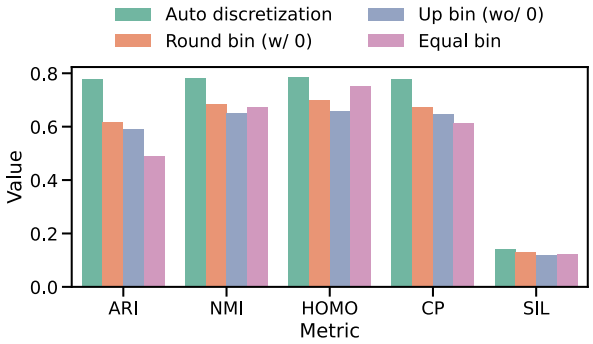

Figure 8. Perofrmance comparison between auto discretization strategy and other binning methods for expression value projection. The cell clustering task is evaluated and five metrics are displayed.

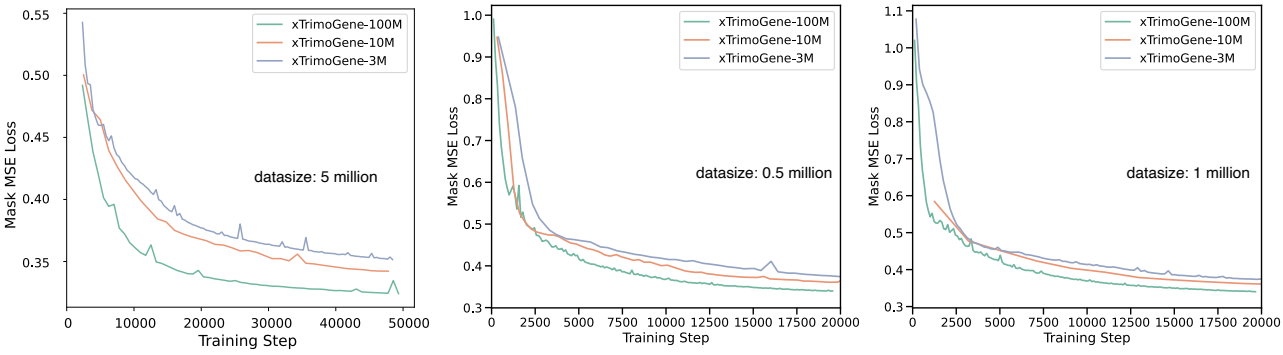

Figure 9. The learning curve of pre-trained xTrimoGene models with different parameter scale. The loss curve measures MSE for masked positions during the pre-training stage, and only the validation set is displayed.

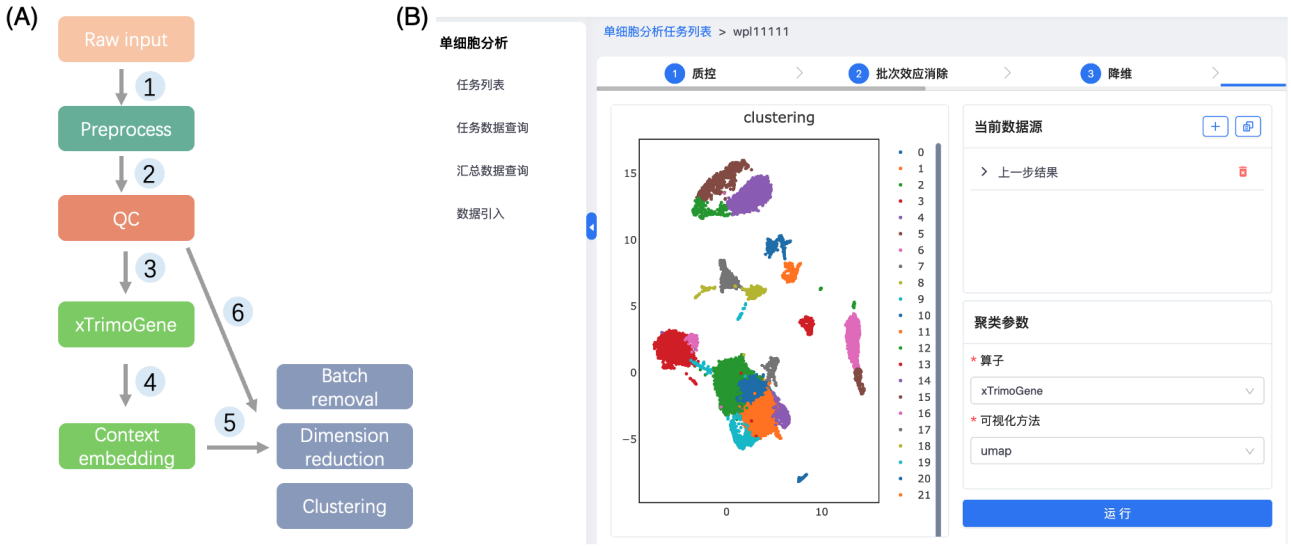

Figure 10. The deployment of xTrimoGene model on a website is depicted in this figure. Figure A shows the overall pipeline, which includes the following steps: (1) User-uploaded raw input undergoes preprocessing and filtration through (2) quality control, (3) feeding the processed data into xTrimoGene for (4) context embedding extraction. The model supports multiple downstream applications such as (5) cell clustering, dimension reduction, and batch removal. The extracted expression profile can also be directly utilized by other algorithms. Figure B provides a snapshot of a clustering task in action using xTrimoGene's context embeddings.