GREmLN: A Cellular Regulatory Network-Aware Transcriptomics Foundation Model

Mingxuan Zhang*
Department of Systems Biology
Columbia University
mz2934@cumc.columbia.edu

Vinay Swamy*
Department of Biomedical Informatics
Columbia University
vss2134@cumc.columbia.edu

Rowan Cassius
Department of Systems Biology
Columbia University
rc36864@cumc.columbia.edu

Léo Dupire
Department of Systems Biology
Columbia University
ld3154@cumc.columbia.edu

Charilaos Kanatsoulis
Department of Computer Science
Stanford University
charilaos@cs.stanford.edu

Evan Paull#
AI Science Program
Chan Zuckerberg Biohub New York
evan.paull@czbiohub.org

Mohammed AlQuraishi#
Department of Systems Biology
Columbia University
ma4129@cumc.columbia.edu

Theofanis Karaletsos#
AI Science
Chan Zuckerberg Initiative
tkaraletsos@chanzuckerberg.com

Andrea Califano#
Department of Systems Biology
Columbia University
ac2248@cumc.columbia.edu

*: These authors contributed equally
#: Corresponding authors

Abstract

The ever-increasing availability of large-scale single-cell profiles presents an opportunity to develop foundation models to capture cell properties and behavior. However, standard language models such as transformers are best suited for sequentially structured data with well defined absolute or relative positional relationships, while single cell RNA data have orderless gene features. Molecular-interaction graphs, such as gene regulatory networks (GRN) or protein-protein interaction (PPI) networks, offer graph structure-based models that effectively encode both non-local gene-gene dependencies, as well as potential causal relationships. We introduce GREmLN (**G**ene **R**egulatory **Em**bedding-based **L**arge **N**eural model), a foundation model that leverages graph signal processing to embed gene-regulatory network structure directly within its attention mechanism, producing biologically informed single cell specific gene embeddings. Our model faithfully captures transcriptomics landscapes and achieves superior performance relative to state-of-the-art baselines on both cell type annotation and graph structure understanding tasks. It offers a unified and interpretable framework for learning high-capacity

Preprint. Under review.

foundational representations that capture complex, long-range regulatory dependencies from high-dimensional single-cell transcriptomic data. Moreover, the incorporation of graph-structured inductive biases enables more parameter-efficient architectures and accelerates training convergence.

1 Introduction

Gene regulatory and signaling networks have been broadly used to represent the molecular logic that mediates the effect of genetic and epigenetic events in determining cell states and whose dysregulation is responsible for a variety of human diseases, including cancer[5, 6]. We propose that advances in multi-omics single cell profiling, such as single cell RNA-seq (scRNA-seq), coupled with advanced cellular network models, provide a unique opportunity to create foundation models that improve our ability to elucidate the mechanisms that underlie tumorigenesis, tumor plasticity, and drug responses. Recent attempts to train large-scale transformer-based foundation models with scRNA data demonstrated potential in terms of both cell-level tasks (e.g.: cell type prediction, batch correction) and gene-level tasks (e.g.: gene regulatory network analysis, perturbation response prediction)[7, 12, 34]. However, these models represent genes as discrete tokens and model cells as sequences, applying standard multi-head self-attention mechanisms, either in their original form [34], with attention biases informed by gene-gene relationships [7], or using randomized nonlinear kernel functions to encode structural priors [12]. These formulations overlook the fact that genes products in the cell, as represented in scRNA-seq profiles, lack inherent sequential order or positional semantics, which is essential for inducing stable long-range attention decay patterns that facilitate the Transformer's capacity to model dependencies and generalize to sequences beyond those encountered during training[18, 33, 19].

In this work, we introduce GREmLN(**G**ene **R**egulatory **Em**bedding-based **L**arge **N**eural model), a scRNA foundation model that leverages gene regulatory networks to encode biologically meaningful relative position information and long-range dependency into single cell level gene embeddings. We first detail the formulation of our attention mechanism based on graph signal processing and numerical analysis. By applying a diffusion kernel to the graph Laplacian, we construct a kernel Gram matrix that can be used to transform the query embeddings, thereby enabling the self-attention mechanism to be constrained and thus structured by the underlying graph. The diffusion process itself further helps capture long-range dependencies between gene nodes in the network. We also implement a Chebyshev polynomial based approximation of the kernel Gram matrix to scale to large graphs and long gene sequences. We then evaluate model performance with both cell level and gene level tasks such as cell type annotation with cell embeddings and graph structure understanding of gene embeddings to demonstrate superior performances over scRNA foundation model baselines.

2 Related Work

Gene Regulatory Networks Gene Regulatory Networks (GRNs) describe a system of interconnected gene regulators and target genes that control cellular phenotype and function, and can be applied as useful predictive or descriptive models in various biological applications. There are many established algorithms for accurate inference of GRNs, including data-driven models that can exploit large scRNA datasets [24, 13, 23].

Graph Transformers A growing body of work integrates graph structural priors into Transformer architectures to enhance relational reasoning and introduce domain-specific inductive biases [9]. These methods generally fall into two categories: (1) incorporating graph structure into positional embeddings, and (2) modifying attention computations using graph information.

Popular positional encoding schemes include classical Laplacian eigenvectors [8] and their permutation-equivariant variants [21, 14]. Other approaches include SVD-derived embeddings [15], centrality and degree features [38], and hop-based relative embeddings via auxiliary GNNs [40]. However, such embeddings often scale poorly to large or high-order graphs and risk introducing noisy or biased priors, limiting generalization. The work in [17] recently proposed learnable, GNN-generated encoding that act as drop-in, computationally efficient alternatives for Laplacian-based methods.

For attention modification, methods include designing equivariant attention mechanisms [35], using ego-centric subgraphs [41], or appending GNN-based features with permutation-invariant Transformers [37]. Others use attention masking [25, 10] or bias terms derived from shortest path distances or proximity metrics [38, 41]. Yet, hard masking blocks all information flow and impairs gradient stability, while additive biases only shift attention scores without enabling structural message passing.

In contrast, we embed graph structure into a spectral kernel that filters noise and injects multi-hop, spectrum-aware inductive bias at low cost. This enhances convergence, preserves gradient propagation across regulatory edges, and enables each query to integrate contextual information from biologically related neighbors.

Biological Foundation Models Similar to their use in natural language understanding, foundation models learn robust representations from biological data. Early work such as ESM[30] or ProGen[11] focused on proteins sequences but have since expanded to many other modalities such as RNA sequences[32], small molecules [31], and most relevantly scRNA-seq[7],[34]. Foundation models for scRNA-seq frame the problem analogously to natural language, where each cell is a document and each gene is a word. All methods aim to develop a meaningful latent representation of RNA expression in a given cell that can subsequently be used for downstream tasks such as batch effect removal [7], cell type classification [34], drug sensitivity prediction[12], or the effects of genetic perturbations[34].

One key difference of scRNA-seq to other modalities is that scRNA-seq represents the expression value of each gene as a continous rather than discrete variable. Most models solve this by simply discretzing expression into bins and modeling expression using standard auto-regressive [7] or masked-token modeling[34] objectives. Newer models such as scFoundation model the read depth of each cell and employ a masked regression objective to better model the continuous nature of RNA expression [12]. Another challenge with modeling scRNA-seq is the inherent lack of sequential structure. scRNA-seq expression vectors are unordered sets, and so standard sequential based positional encodings cannot be applied. Existing methods simply fix an arbitrary gene order; thus, this challenge remains unsolved.

3 Model Architecture

We propose a new model architecture that integrates gene regulatory network structure to facilitate the learning of single-cell-specific gene embeddings. By employing this approach, the model gains the ability to capture meaningful long-range dependencies among gene tokens, while also enriching the feature space with relevant positional and structural relationships between these tokens. Such a framework could be easily extended to arbitrary interaction graphs, including protein-protein interaction networks and protein subcellular localization networks, particularly in scenarios where incorporating graph-based dependency models into learned token representations is desirable.

3.1 Pre-training Data Preparation

The pre-training dataset consists of 11M scRNA-seq profiles spanning 19K genes from healthy human cells, sourced from the expansive CELLxGENE dataset (`https://cellxgene.cziscience.com/collections`). Our single cell dataset covers 162 cell-types from various tissues (heart, blood, brain, lung, kidney, intestine, pancreas, and others). The specificity of the cell-types is as indicated in the $obs.cell_type$ attribute of the CELLxGENE corpus for corresponding cells.

Our pre-processing pipeline consists of the following steps: (1) quality control by UMI count and log1p normalization of expression values, (2) quantization of expression values into 100 discrete bins for computational efficiency, (3) cell-type specific Gene Regulatory Network generation using the ARACNe algorithm[23], detailed in 3.2 below, and (4) curating single-cell-specific networks from the cell-type-specific GRNs as the subgraph of a cell's expressed genes, followed by splitting the dataset into unseen-graph validation, seen-graph validation, and training sets. These steps are further detailed in Appendix A.

3.2 Tokenization and Initial Embeddings

We can extract two types of information from a scRNA count matrix: (1) Gene identity, and (2) Gene expression value, which is a measurement of how expressed a gene is in a cell. An unprocessed scRNA count matrix (raw count) contains positive integers representing the number of RNA transcript copies of a gene. The inputs to the model are a gene identity embedding matrix (denoted as E^g), and a gene rank embedding matrix (denoted as E^r). We construct the gene identity embedding for cell i as:

$$E^g = emb_g(id(g_1^i),\ id(g_2^i),\ \dots,\ id(g_G^i))$$

where emb_g is a learnable embedding layer. Gene IDs are never masked. To construct the gene rank embedding, we first compute the rank of gene j in cell i as:

$$r(g_j^i) = k, \hat{X}_{ij} \in [b_k, b_{k+1}]$$

where $\hat{X}$ is the normalized expression matrix, and b_k, b_{k+1} are lower and upper bounds of bin k. The normalized expression matrix is obtained by normalizing the total counts of all cells to sum up to a fixed number (in our case 10^6) and then log-plus-one transforming it. The <MASK> rank token is randomly assigned to 15% of the rank values. The gene rank embedding of cell i is then given by:

$$E^r = emb_r(r(g_1^i),\ r(g_2^i),\ \dots,\ r(g_G^i))$$

where emb_r is a learnable embedding layer. The input embedding to the transformer is then given by the concatenation of two representations:

$$E = [E^g || E^r]$$

We also prepend a <CLS> token to the beginning of each cell sentence that attends to all genes. The <CLS> token is not part of the graph and is not involved in any graph operations.

3.3 Gene Regulatory Network Construction

To construct gene regulatory networks, we use the extensively validated ARACNe algorithm[23] which infers the transcriptional targets of regulatory proteins (represented as edges in a network) by estimating the mutual information between their gene expression via bootstrapping and adaptive partitioning, followed by elimination of indirect targets via the data processing inequality theorem. ARACNe is a key component of several well-validated algorithms that have been effective in elucidating master regulator proteins representing mechanistic determinants of tumor cell states as well as in determining their sensitivity to small molecules, including use in clinical trials[2, 3, 26, 28, 29, 39], thus demonstrating the biologically meaningful nature of the inferred networks, which makes ARACNe networks a natural and potentially useful choice for implementing graph-enriched embeddings in foundation models.

Gene regulatory networks vary widely across cell types, so prior to model pre-training, we infer a unique GRN for each cell type in our training corpus. To reconstruct each network, we first run ARACNe within every Louvain-algorithm-inferred cluster. The resulting subnetworks are then consolidated into a single network by evaluating the likelihood of edge recurrence under a binomial null model, allowing the false discovery rate to be controlled at 5%. Forming a consensus network from cluster-specific subnetworks offers several advantages: (1) consensus networks are robust to potentially significant batch effects; (2) almost all cells of each cell type contribute to the final network, and (3) multiple possible cell states are faithfully represented in the single network, enabling embeddings to capture the full granularity of single-cell biological states.

3.4 Graph Diffusion Kernel Attention

Leveraging the fact that we observe existing token graphs in the form of gene regulatory networks, we incorporate graph-based structural information into the attention mechanism by transforming each query embedding with a diffusion kernel Gram matrix derived from the token graph. In doing so, the query vectors become conditioned on graph topology—effectively biasing the attention process to reflect the gene-gene relationships encoded in the graph. Key and value embeddings, meanwhile, are not transformed to preserve the original information in token embeddings. We aim to develop general

embeddings that inherently capture token graph structures. More specifically, assume the token graph adjacency matrix is A, we compute the normalized Laplacian matrix as:

$$L = I - D^{-1/2} A D^{-1/2},$$

where D is the diagonal degree matrix. Using the normalized Laplacian matrix ensures that the eigenvalues remain bounded when mapping the graph to the spectral domain. To that end, let $L \in \mathbb{R}^{G \times G}$ be a symmetric matrix with spectral decomposition

$$L \;=\; U \Lambda U^{\top}, \qquad U = \big[u_1, \ldots, u_G\big],\; \Lambda = \mathrm{diag}\big(\lambda_1, \ldots, \lambda_G\big).$$

For a non–negative spectral filter $\kappa : \mathbb{R} \to \mathbb{R}_{\geq 0}$ we define the *kernel Gram matrix*

$$\Phi_L \;=\; \sum_{i=1}^{G} \kappa(\lambda_i)\, u_i u_i^{\top} \;=\; U\, \kappa(\Lambda)\, U^{\top}$$

with $\kappa(\Lambda) = \mathrm{diag}\big(\kappa(\lambda_1), \ldots, \kappa(\lambda_G)\big)$. Because $\kappa(\lambda_i) \geq 0$ for every i, the matrix Φ_L is symmetric positive semidefinite and therefore constitutes the matrix representation of an orthogonal projection operator onto the Reproducing Kernel Hilbert Space $\mathcal{H}_\kappa$ whose reproducing kernel satisfies

$$k_\kappa(i,j) \;=\; e_i^{\top} \Phi_L e_j, \qquad i,j \in \{1, \ldots, G\}.$$

where $\kappa(.,.)$ is any kernel function. Let $\kappa(\lambda_i) = \exp(-\beta\lambda_i)$ the Gram matrix is given by:

$$\Phi_L(Q) = U \exp(-\beta\Lambda) U^T Q$$

We then compute dot-product attention with the diffused query embedding as:

$$Attn(Q, K, V, L) = \mathrm{softmax}(\frac{\Phi_L(Q) K^T}{\sqrt{d}}) V$$

where un-softmaxed attention score for each pair is given by the following similarity measure:

$$S_{ij} = q_i^{\top} \Phi_L^{1/2} k_j$$

This bilinear score incurs asymmetric smoothing which steers the attention mechanism toward interactions that respect the graph-implied low-frequency (long-range) decay, while still allowing keys to contribute high-frequency detail.

3.5 Training Objectives

We formulate the pre-training objective as a masked modeling problem conditioned on graphs. Assume we have G genes, let $X^r = (X_1^r, X_2^r, .., X_G^r)$ be the bin values of genes in a cell, L^c be the normalized Laplacian matrix of the gene regulatory network specific to cell type c which the cell belongs to, $m = (m_1, m_2, ..., m_G)$ be a random binary mask vector where $m = 1$ represent genes being masked. The likelihood of all masked genes is given by

$$p_\theta(X_m^r | L^c, X_{\bar{m}}^r, E^g) = \prod_{i, \forall m_i = 1} p_\theta(x_i^r | L^c, X_{\bar{m}}^r, E^g)$$

where E^g is the gene identity embedding, X_m^r are the bin values of masked genes, θ are all learnable parameters, and $X_{\bar{m}}^r$ are the bin values of unmasked genes. Denote the transformer encoder with graph diffusion kernel attention as enc and the linear decoder network as dec, a masked gene i's conditional distribution can be parameterized as:

$$\begin{aligned}
&E_{\bar{m}}^r = emb_r(X_{\bar{m}}^r) \\
&H = enc(L^c, E_{\bar{m}}^r, E^g) \\
&p_\theta(x_i^r | L^c, X_{\bar{m}}^r, E^g) = dec(H_i) = \sigma(W H_i + b)
\end{aligned}$$

where H is the learned embedding by the transformer encoder. The pre-training objective is then given by maximizing the expected conditional log likelihood:

$$\mathcal{L} = -\mathbb{E}_{X^r \sim \mathcal{D}} \mathbb{E}_{m \sim p(m)} [\sum_{i, \forall m_i = 1} \log p_\theta(x_i^r | L^c, X_{\bar{m}}^r, E^g)]$$

where $\mathcal{D}$ is some ground truth data distribution and $p(m)$ is the sampling distribution for the mask.

3.6 Approximation of Large Kernel Gram Matrices

Due to the size of gene regulatory networks and length of cell sentences, performing operations such as matrix exponential and spectral decomposition every batch to transform query embeddings becomes computationally expensive and difficult to scale. To alleviate this issue, we take advantage of Chebyshev polynomials to approximate the kernel Gram matrix. More specifically, if a token graph has G nodes, we approximate the transformed query embedding with Chebyshev polynomial truncated at term $K \ll G$ as:

$$\begin{aligned}\Phi_L(Q) &= U \exp(-\beta\Lambda) U^T Q \\ &\approx \sum_{k=1}^{K} c_k T_k(L) Q\end{aligned}$$

to efficiently compute the kernel gram matrix of large graphs. This approximation can be solved with Gauss-Chebyshev quadrature, which we detail in Appendix B.

4 Validation Experiments

In this section we compare our model to state-of-the-art transformer-based single cell transcriptomics foundation models, namely scGPT[7], Geneformer[34], and scFoundation[12]. We restrict our evaluation to tasks involving frozen embeddings, without additional fine-tuning, to ensure fair comparison across models and to reflect realistic deployment scenarios. Additionally, we employ evaluation models with minimal representational capacity—such as linear models and shallow MLPs—to ensure that performance predominantly reflects the quality of the pre-trained embeddings rather than the expressiveness of the downstream model. First, we assess the model's performance on gene expression reconstruction and cell type classification. Subsequently, we evaluate the extent to which the learned embeddings capture and encode underlying graph structural information.

4.1 Evaluation datasets

We select evaluation datasets that are non-overlapping with those used during pre-training to assess the generalizability of all models. In particular, we focus on a dataset consisting of healthy human immune cells, as we are interested in downstream applications of our model to human immune cell state engineering aimed at effectuating onco-reversion. Aside from healthy immune cells, we also test the model on diverse datasets derived from cancer-infiltrating immune cells and non-immune cells.

Human Immune Cells This dataset includes cells from both human bone marrow and peripheral blood, covering 16 distinct immune cell types. We use the version provided by scGPT (`https://figshare.com/ndownloader/files/25717328`), with original cell type annotations from Luecken et al[22]. This dataset serves as our primary benchmark for evaluating the model's ability to capture gene regulatory graph structure and to learn meaningful cell embeddings. We apply the same preprocessing steps as those used for the CELLxGENE corpus for model pre-training.

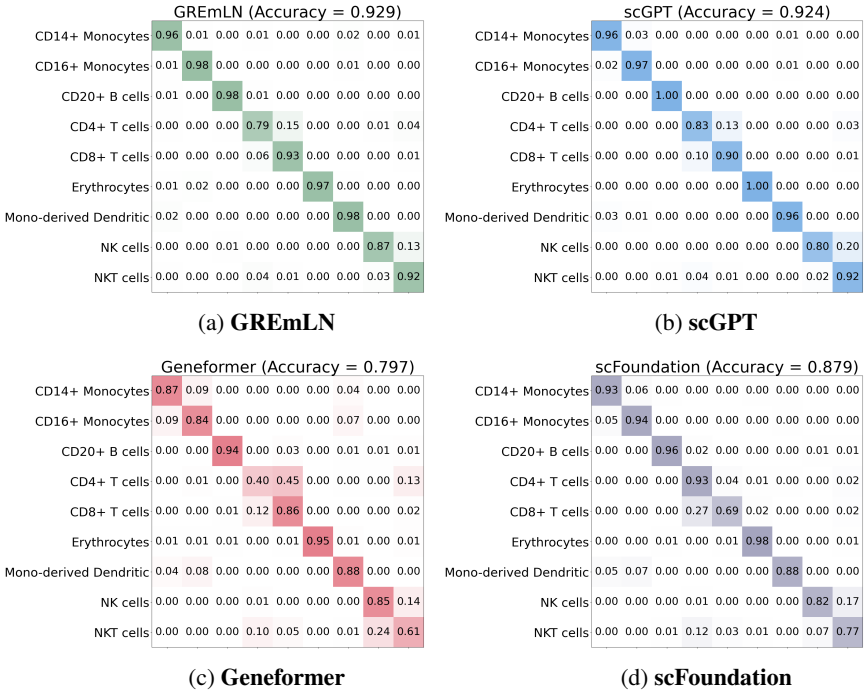
Cancer Infiltrating Myeloid This dataset comprises tumor-infiltrating myeloid cells (TMIs) from nine different human cancer types. TMIs are key regulators of tumor progression, with properties and functions that vary across cancer types. Similar to scGPT, we designate TMIs from six cancer types as a reference set for training and use the remaining types for out-of-distribution inference (query set). Pre-processing steps are identical to those applied to the CELLxGENE corpus. The dataset is available at: `https://drive.google.com/drive/folders/1VbpApQufZq8efFGakW3y8QDDpY9MBoDS`

Held Out Non-Immune Cells This dataset consists of 10 non-immune cell types that our model does not see during pre-training and validation, sourced from the CELLxGENE corpus (available at: `https://cellxgene.cziscience.com/collections`) with the same pre-processing steps as the training dataset. The selected cell types are retinal rod, epithelial, perivascular, amacrine, respiratory basal, smooth muscle, neural progenitor, secretory, myofibroblast, and megakaryocyte, testing the model's ability to perform zero-shot annotation across distinct non-immune cell types.

4.2 Transcriptomic Landscape Learning and Cell Type Annotation

We first validate of our model's effective learning of mRNA transcripts by training a simple linear regression model to reconstruct gene expression. Our model is able to accurately reconstruct gene

expression for both healthy and diseased human cells(Figure 4). We then assess the quality of our learned embeddings' cellular representations by measuring their ability to recapitulate cell type.



(a) **GREmLN** (b) **scGPT**

(c) **Geneformer** (d) **scFoundation**

Figure 1: **Cell-type classification performance on the Human Immune dataset.** Shown are confusion matrices produced by (a) GREmLN, (b) scGPT, (c) Geneformer, and (d) scFoundation. Each heat-map cell uses the same colour scale so the three models can be compared directly.

Since our model requires graphs as input, we only allow the model to see integrated networks that combines all cell-type specific GRNs to avoid inadvertently leaking cell-type labels.

Bayesian Graph Integration Let X_{train} be the training gene–expression matrix and let $X_{\text{test}}^{(i)}$ denote the profile of a single test cell i. For every cell type $c \in \mathcal{C}$ we compute the posterior edge-exclusion probability $\Pr\big(e \notin \mathcal{G}_c \mid X_{\text{train}}, C = c\big)$ and declare the edge present in the type-specific estimate $\widehat{\mathcal{G}}_c$ whenever this posterior does not exceed a user-defined false-edge level α,

$$e \in \widehat{\mathcal{G}}_c \iff \Pr\big(e \notin \mathcal{G}_c \mid X_{\text{train}}, C = c\big) \le \alpha. \tag{1}$$

A k-nearest-neighbor classifier fitted exclusively on X_{train} yields an empirical posterior $w_c^{(i)} = \Pr\big(C = c \mid X_{\text{test}}^{(i)}\big)$, with $\sum_{c \in \mathcal{C}} w_c^{(i)} = 1$. Using the law of total probability,

$$p_i(e) = \sum_{c \in \mathcal{C}} w_c^{(i)} \ \Pr\big(e \notin \mathcal{G}_c \mid X_{\text{train}}, C = c\big). \tag{2}$$

The regulatory graph supplied to the model for cell i is finally obtained via the same α-level threshold,

$$\widehat{\mathcal{G}}_i = \big\{\, e : \ p_i(e) \le \alpha \big\}. \tag{3}$$

Equations (1)–(3) collectively recast the procedure as two successive Bayesian updates: first, learning type-specific edge posteriors on X_{train}; second, averaging these posteriors for each unseen cell under the k-NN–derived type posterior, thereby integrating multiple cell-type-specific graphs without ever exposing the true test-cell labels.

Training cell type classifier For each cell i, we begin with its embedding matrix $H_i \in \mathbb{R}^{L \times d}$, where L is the gene dimension and d is the embedding dimension. Next, we obtain a fixed-size cellular representation $h_i \in \mathbb{R}^d$ via mean pooling over the gene sequence dimension. Next, we fit a linear classifier to predict cell type given h_i and the integrated graph $\widehat{\mathcal{G}}_i$ by minimizing cross-entropy

Dataset	Human Immune Cells			Non-Immune Cells(zero-shot)		
Model	Precision	Recall	Macro F1	Precision	Recall	Macro F1
GREmLN	**0.931 ± 0.002**	**0.929 ± 0.003**	**0.929 ± 0.001**	0.904 ± 0.002	0.903 ± 0.001	**0.903 ± 0.001**
scGPT	0.928 ± 0.003	0.924 ± 0.001	0.924 ± 0.002	-	-	-
scFoundation	0.887 ± 0.013	0.879 ± 0.017	0.879 ± 0.015	**0.913 ± 0.002**	**0.904 ± 0.003**	0.903 ± 0.003
Geneformer	0.800 ± 0.012	0.797 ± 0.010	0.792 ± 0.010	0.774 ± 0.015	0.768 ± 0.010	0.768 ± 0.012

Table 1: **Test cell type annotation performance across multiple tissue types.** CI is computed across 3 random initializations of test splits. scGPT is excluded from evaluation on the Held-out Non-Immune Cell dataset, as it does not constitute a zero-shot setting.

loss between the estimated class distribution and the true cell type label. More specifically, assuming we have cell type label y_i, we aim to learn the distribution of cell type labels conditioning on the cell representation and the input graph. For cell type c the conditional density is given by:

$$p(y_i = c|h_i, \widehat{\mathcal{G}}_i, f_\theta)$$

where f_θ are the classifiers being trained. We learn f_θ by optimizing cross entropy loss as:

$$\mathcal{L}(\theta) = -\sum_{c \in C} \sum_{i=1}^{N} \log p(y_i = c|h_i, \widehat{\mathcal{G}}_i, f_\theta)$$

Our model achieved highest performances on all classification metrics on the human immune cells dataset(Figure 1, Table 1). Additionally, for held-out non-immune cells we test the model's ability to perform zero-shot cell type annotation. Since scGPT are trained with the full CELLxGENE dataset, we exclude the model from this evaluation. We achieve competitive zero-shot performances on the held-out non immune cell dataset(Table 1) for all metrics. We also point out that GREmLN contains only 10.3 million learnable parameters (less than one third of all baselines' parameter sets and one tenth of scFoundation, see details in Appendix C), which highlights the predictive power of gene regulatory network guidance. Our key contribution lies in tailoring the attention mechanism to encode biological inductive bias, rather than increasing model size.

4.3 Graph Structure Understanding

We assess the learned embeddings' capacity to capture and generalize graph structure by evaluating edge prediction on unseen gene regulatory networks from held-out cells. Specifically, we randomly mask a subset (15%) of edges and measure how well the model recovers them when conditioned on the learned embeddings and the incomplete graph. Let H be learned gene embeddings from a model

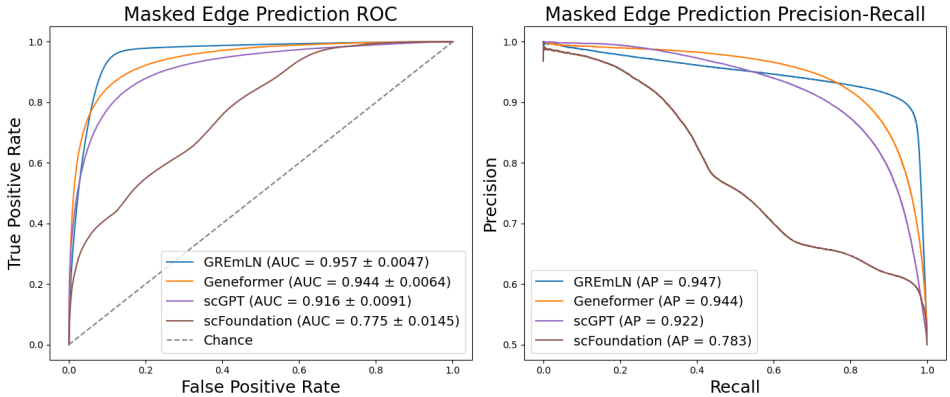


Figure 2: **Benchmarking performance for graph understanding on the Human Immune Dataset.** Left: Receiver operating characteristic curve with area under curve (AUROC) metrics. Confidence intervals computed with cross validation. Right: Precision-recall curve with average precision (AP) metrics.

and A be the adjacency matrix of the graph, the objective is to learn the conditional probability of a masked edge existing, i.e.:

$$p_\theta(A_{i,j} = 1|\tilde{A} = A \odot \mathcal{M}, H)$$

We use the learned embeddings H from pre-trained models and parameterize the conditional likelihood with a model f_ψ where ψ are learned model parameters. More specifically, for an edge (i, j):

$$p_\theta(A_{i,j} = 1 | \tilde{A} = A \odot \mathcal{M}, H) = f_\psi(i, j, \tilde{A}, H_i, H_j)$$

where H_i, H_j are learned embeddings of nodes i, j respectively. To learn the model f_ψ we treat an edge-label pair $\{(i, j), y_{ij}\}$ as i.i.d samples from some data generating distributions. The label $y_{ij} = 1$ if the edge (i, j) is a real masked edge and $y_{ij} = 0$ if the edge is a fake edge that does not exist in A. The training objective is then given by:

$$\mathcal{L}(\psi) = - \sum_{\{(i,j),y_{ij}\} \sim \mathcal{D}} [y_{ij} \log f_\psi(i, j, \tilde{A}, H_i, H_j) + (1 - y_{ij}) \log(1 - f_\psi(i, j, \tilde{A}, H_i, H_j))]$$

In practice, we freeze our encoder weights and incorporate the partially observed graph $\tilde{A}$ through the attention mechanism to condition our embeddings. For baseline models, we similarly condition on $\tilde{A}$ by passing the learned gene embeddings through a graph attention network using the same masked graph. We first benchmark the GRN structure understanding task on healthy human datasets to evaluate model performances with seen cell types but unseen data/graph. GREmLN achieves the best performance by a significant margin in terms of both ROC and PRC, while Geneformer is also competitive due to their tokenization and pre-trianing objective focusing on learning gene embeddings that are GRN-aware (Figure 2).

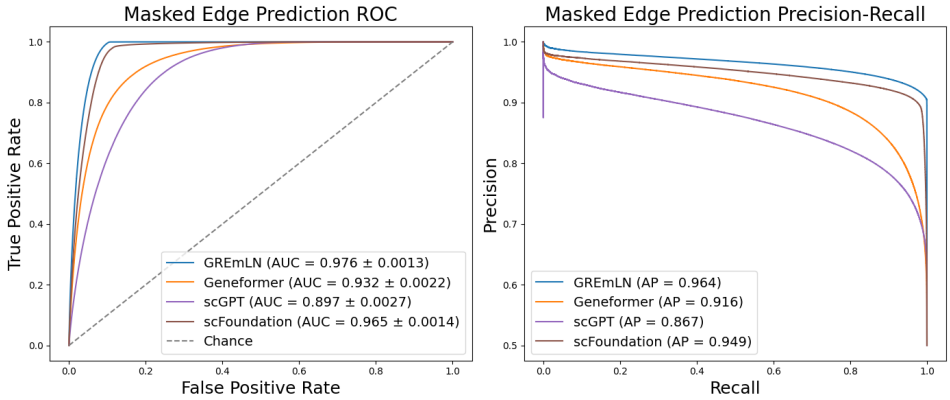


Figure 3: **Benchmarking performance for graph understanding on the Cancer Infiltrating Myeloid dataset.** Left: Receiver operating characteristic curve with area under curve (AUROC) metrics. Confidence intervals computed with cross validation. Right: Precision-recall curve with average precision (AP) metrics.

To further evaluate the model's capacity for structural generalization, we assessed performance on graph structure understanding tasks involving previously unseen cell types. Given that pathological conditions such as cancer often induce substantial rewiring of gene regulatory networks, we specifically tested the model on cancer-infiltrating myeloid cells from distinct tissue origins. Evaluation was conducted on held-out cancer types to simulate out-of-distribution generalization. GREmLN consistently outperformed all baseline models across all performance metrics (Figure 3), demonstrating superior generalization to previously unseen regulatory graphs within complex and heterogeneous biological systems.

5 Limitations

Static Graph: We use the graph structure as inductive bias to reduce the learning space, so the graph structure needs to remain static during training and evaluation. However, future work can extend this framework to model the stochasticity of the graph and construct prior distributions that contain information related to the generative process of cells. Moreover, gene regulatory networks are only the most straightforward solution to modeling molecular interactions as a graph. Future inclusion of protein-protein, microRNA, and receptor/ligand interaction should further improve the quality of our embeddings.

Sensitivity to Graph Quality: The model's embedding quality is sensitive to network quality. For example, single cell derived networks that are too sparse or noisy will naturally lead to worse embedding qualities due to noisy propagation. A solution for this is to construct cleaner networks, for example integrating single cell level networks with bulk RNA-seq based networks to increase edge confidence.

Spectral Constraints: For the current framework, the graph adjacency matrix needs to be symmetric [27]; this constraint forces us to remove direction from edges but allows us to maintain the full structure, including cycles, that are usually present in real world GRNs. Derivations of spectral representations of DAGs can be potentially incorporated in a future version of the model.

6 Conclusion and Discussion

High-throughput single-cell datasets are enabling unprecedented insights into therapeutically relevant genetic mechanisms. As these datasets grow, scRNA foundation models offer a scalable framework to generalize across biological contexts, perturbation regimes, and experimental conditions. However, the absence of inherent sequential order or positional semantics in gene tokens poses a challenge to Transformer-based architectures, which rely on such structure to model long-range dependencies and generalize beyond the training distribution.

To address this, we introduce a graph-aware model that integrates gene regulatory networks, enabling the learning of biologically informed gene embeddings through graph-induced ordering. The learned embeddings can be used to reconstruct gene expression with a linear regression model, demonstrating that transcriptomics information is well captured. The model additionally achieves state-of-the-art performance in predicting held-out cell types and unseen regulatory structures, illustrating its ability to capture generalizable rules of gene regulation and representation of cell states.

While the functional search space for gene interactions is intractable even for large models, incorporating validated regulatory interactions significantly improves performance across downstream tasks [16, 4, 2]. Our architecture supports arbitrary molecular interaction graphs and leverages well-established methods for network inference [23] alongside diffusion-based graph kernels [28, 20]. Future directions include fine-tuning cell and gene embeddings for specific tasks—such as modeling combinatorial perturbations and identifying optimal interventions—while using attention-based interpretability to recover core regulatory modules.

References

[1] Paszke Adam, Gross Sam, Massa Francisco, and et. al. Pytorch: An imperative style, high-performance deep learning library. *arxiv*, 2019.

[2] Mariano J Alvarez, Yao Shen, Federico M Giorgi, Alexander Lachmann, B Belinda Ding, B Hilda Ye, and Andrea Califano. Network-based inference of protein activity helps functionalize the genetic landscape of cancer. *Nature Genetics*, 48(8):838, 2016.

[3] Mariano J Alvarez, Yao Shen, Federico M Giorgi, Alexander Lachmann, B Belinda Ding, B Hilda Ye, and Andrea Califano. Functional characterization of somatic mutations in cancer using network-based inference of protein activity. *Nature Genetics*, 2019.

[4] Ziv Bar-Joseph, Georg K Gerber, Tong Ihn Lee, Nicola J Rinaldi, Jane Y Yoo, François Robert, D Benjamin Gordon, Ernest Fraenkel, Tommi S Jaakkola, Richard A Young, et al. Computational discovery of gene modules and regulatory networks. *Nature biotechnology*, 21(11):1337–1342, 2003.

[5] Andrea Califano and Mariano J. Alvarez. The recurrent architecture of tumour initiation, progression and drug sensitivity. *Nature Reviews Cancer*, 2017.

[6] James C. Chen, Mariano J. Alvarez, and Flaminia Talos et al. Identification of causal genetic drivers of human disease through systems-level analysis of regulatory networks. *Cell*, 2014.

[7] Haotian Cui and Chloe Wang et al. scgpt: Toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, 2024.

[8] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arxiv*, 2020.

[9] Min Erxue, Runfa Chen, and Yatao Bian. Transformer for graphs: An overview from architecture perspective. *arxiv*, 2022.

[10] Yu Rong Erxue Min, Tingyang Xu, and Yatao Bian. Masked transformer for neighhourhood-aware click-through rate prediction. *arxiv*, 2022.

[11] Madani et al. Progen: Language modeling for protein generation. *arxiv*, 2020.

[12] Minsheng Hao and Jing Gong et al. Large-scale foundation model on single-cell transcriptomics. *Nature Methods*, 2024.

[13] Michael Hecker, Sandro Lambeck, Susanne Toepfer, Eugene Van Someren, and Reinhard Guthke. Gene regulatory network inference: data integration in dynamic models—a review. *Biosystems*, 96(1):86–103, 2009.

[14] Yinan Huang, William Lu, Joshua Robinson, Yu Yang, Muhan Zhang, Stefanie Jegelka, and Pan Li. On the stability of expressive positional encodings for graphs. In *The Twelfth International Conference on Learning Representations*.

[15] Md Shamim Hussain, Mohammed J Zaki, and Dharmashankar Subramanian. Edge-augmented graph transformers: Global self-attention is enough for graphs. *arxiv*, 2021.

[16] Trey Ideker and Nevan J Krogan. Differential network biology. *Molecular systems biology*, 8(1):565, 2012.

[17] Charilaos I Kanatsoulis, Evelyn Choi, Stephanie Jegelka, Jure Leskovec, and Alejandro Ribeiro. Learning efficient positional encodings with graph neural networks. In *The Thirteenth International Conference on Learning Representations*, 2025.

[18] Amirhossein Kazemnejad and Inkit Padhi et. al. The impact of positional encoding on length generalization in transformer models. *NeurIPS*, 2023.

[19] Guolin Ke, Di He, and Tie-Yan Liu. Rethinking positional encoding in language pre-training. *ICLR*, 2020.

[20] Mark DM Leiserson, Fabio Vandin, Hsin-Ta Wu, Jason R Dobson, Jonathan V Eldridge, Jacob L Thomas, Alexandra Papoutsaki, Younhun Kim, Beifang Niu, Michael McLellan, et al. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nature genetics*, 47(2):106–114, 2015.

[21] Derek Lim, Joshua David Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. In *The Eleventh International Conference on Learning Representations*.

[22] Malte D. Luecken, M. Büttner, and et. al. Benchmarking atlas-level data integration in single-cell genomics. *Nature Methods*, 2021.

[23] Adam A Margolin and Ilya Nemenman. Aracne: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 2006.

[24] Daniele Mercatelli, Laura Scalambra, Luca Triboli, Forest Ray, and Federico M Giorgi. Gene regulatory network inference resources: A practical overview. *Biochimica et Biophysica Acta (BBA)-Gene Regulatory Mechanisms*, 1863(6):194430, 2020.

[25] Gregoire Mialon, Dexiong Chen, Margot Selosse, and Julien Mairal. Graphit: Encod- ing graph structure in transformers. *arxiv*, 2021.

[26] Grunn A et al. Mundi PS, Dela Cruz FS. A transcriptome-based precision oncology platform for patient-therapy alignment in a diverse set of treatment-resistant malignancies. *Cancer Discovery*, 2023.

[27] Bogdan Nica. *A brief introduction to spectral graph theory*, volume 3. European Mathematical Society Zürich, 2018.

[28] Evan O Paull, Daniel E Carlin, Mario Niepel, Peter K Sorger, David Haussler, and Joshua M Stuart. Discovering causal pathways linking genomic events to transcriptional states using tied diffusion through interacting events (tiedie). *Bioinformatics*, 29(21):2757–2764, 2013.

[29] Jones SJ et al. Paull EO, Aytes A. A modular master regulator landscape controls cancer transcriptional identity. *Cell*, 2021.

[30] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

[31] Gabriel Grand Seyone Chithrananda and Bharath Ramsundar. Chemberta: Large-scale self-supervised pretraining for molecular property prediction. *arxiv*, 2020.

[32] Sazan Mahbub Caleb N. Ellington Robin Algayres Dian Li Yonghao Zhuang Hongyi Wang Le Song Eric P. Xing Shuxian Zou, Tianhua Tao. A large-scale foundation model for rna function and structure prediction. *bioarxiv*, 2024.

[33] Jianlin Su and Yu Lu et. al. Roformer: Enhanced transformer with rotary position embedding. *arxiv preprint*, 2021.

[34] Christina V. Theodoris and Ling Xiao et al. Transfer learning enables predictions in network biology. *Nature*, 2023.

[35] Philipp Tholke and Gianni De Fabritiis. Torchmd-net: Equivariant transformers for neural network based molecular potentials. *ICLR*, 2022.

[36] Dao Tri, Fu Daniel, Ermon Stefano, and et. al. Flashattention: Fast and memory-efficient exact attention with io-awareness. *arxiv*, 2022.

[37] Zhanghao Wu and Paras Jain et. al. Representing long-range context for graph neural networks with global attention. *NeurIPS*, 2021.

[38] Chengxuan Ying, Tianle Cai, and et. al. Do transformers really perform bad for graph representation? *arxiv*, 2021.

[39] Tizita Z Zeleke, Qingfei Pan, and et. al. Network-based assessment of hdac6 activity predicts preclinical and clinical responses to the hdac6 inhibitor ricolinostat in breast cancer. *Nature Cancer*, 2023.

[40] Jiawei Zhang, Haopeng Zhang, and et. al. Graph-bert: Only attention is needed for learning graph representations. *arxiv*, 2020.

[41] Jianan Zhao and et. al. Chaozhou Li. Gophormer: Ego-graph transformer for node classification. *arxiv*, 2021.

A Preprocessing Pipeline and Implementation Details

Preprocessing Pipeline Single cell RNA sequencing data is inherently noisy, subject to batch effects, sequencing errors, and high dropout rates. In fact, the sparsity of the expression profiles in our dataset is approximately 80%. Extracting a meaningful signal from such profiles is challenging; thus, it is critical to pre-process this data appropriately before feeding it to the model. Our pre-processing pipeline consists of the following steps: quality control and normalization, quantization, regulatory network generation, and caching. (1) The cells are controlled for quality by filtering out any cells not meeting our UMI sequencing depth requirements. For a given cell, the UMI count of a gene is a measure of that gene's expression level. UMI count thresholds are set to a minimum and maximum of 1K and 100K reads, respectively, and - if needed - are dynamically updated to keep only the middle 90% of cells by total UMI count. Cells with insufficient total UMI counts are removed as they likely represent faulty or empty reads from the sequencing process, and cells with exceedingly large total UMI counts are removed as they likely represent doublets and multiplets (the read contains information from more than one cell). The remaining expressions are log1p normalized to reduce the range of expression values and stabilize the expression variance. (2) After quality controlling, the cells are quantized into 100 expression bins. The zero-th bin is assigned to all zero-expression genes and the remaining expressed genes are assigned to their corresponding quantile for that cell; the bin number increasing with expression. This provides a discretized relative expression of the genes in each cell, preserving cell insights while reducing the computational burden of processing this data during training. (3) Gene Regulatory Networks (GRNs) are then generated for each cell-type using the ARACNe algorithm[23], as described in 3.2. The use of GRNs is at the heart of the novelty of this model, framing expression signatures in a regulatory context. (4) In the final caching step, a GRN sub-network including only the non-zero expressed genes is created for each cell. It is also required that each single cell meets the minimum number of 500 expressed genes.

To evaluate the model's zero-shot capabilities, 15% of cell-types, along with their GRNs, are held out as the unseen-graph validation set. The cells of the remaining 85% are randomly split into the seen-graph validation set and the training set. The seen-graph validation set accounts for approximately 15% of total cells, leaving the last 70% for the training set.

Implementation Details The model is implemented using PyTorch[1] and Flashattention[36]. Optimization are performed with the AdamW optimizer in PyTorch for both pre-training and validation models. Validation-based early stopping are implemented for all downstream regressors/classifiers that takes in frozen embeddings. The architecture of downstream models for all validation experiments are identical across the different foundation models within the same task.

B Chebyshev Approximation of Diffused Query Embeddings

We first compute the Chebyshev coefficients c_k. Since our Laplacian matrix is normalized, the range of eigen values is $[0, 2]$. Since Chebyshev polynomials are orthogonal on $[-1, 1]$, we first map to this domain by setting

$$x = \frac{2\lambda}{\lambda^*} - 1$$

where λ^* is the max eigenvalue, which means $\lambda^* = 2$. Solving for λ yields:

$$\lambda = \frac{\lambda^*}{2}(x+1) = (x+1)$$

We can then compute the Chebyshev coefficients numerically with Gauss-Chebyshev quadrature. The Chebyshev coefficients are given by:

$$c_k = \frac{2}{\pi}\int_{-1}^{1} \frac{f(\lambda)T_k(x)}{\sqrt{1-x^2}}dx \tag{4}$$

$$= \frac{2}{\pi}\int_{-1}^{1} \frac{\exp(-\beta\lambda)T_k(x)}{\sqrt{1-x^2}}dx \tag{5}$$

$$= \frac{2}{\pi}\int_{-1}^{1} \frac{\exp(-\beta(x+1))T_k(x)}{\sqrt{1-x^2}}dx \tag{6}$$

To compute this integral numerically, the non-angular quadrature points x_i can be converted to a function of angles θ as:

$$x_i = \cos\theta_i = \cos(\frac{\pi(i-\frac{1}{2})}{N})$$

where N are number of total quadrature points. Notice that $T_k(x) = T_k(\cos(\theta)) = \cos(k\theta)$. We differentiate both sides to obtain:

$$\frac{dx}{d\theta} = -\sin\theta \rightarrow dx = -\sin\theta d\theta$$

Denote $g(x) = \exp(-\beta(x+1))$, substitute this change-of-variables form into the original integral yields:

$$c_k = \frac{2}{\pi}\int_{-1}^{1} T_k(x)\frac{g(x)}{\sqrt{1-x^2}}dx \tag{7}$$

$$= \frac{2}{\pi}\int_{0}^{\pi} \cos(k\theta)\frac{g(\cos\theta)}{\sin\theta} * \sin\theta d\theta \tag{8}$$

$$= \frac{2}{\pi}\int_{0}^{\pi} \cos(k\theta)g(\cos\theta)d\theta \tag{9}$$

Notice that in (5) the integral range after variable change becomes $[\pi, 0]$ and we flip the range because of the negative sign from the change-of-variables form. The resulting integral after change-of-variables can be approximated as:

$$c_k \approx \frac{2}{N}\sum_{i=1}^{N} g(\cos\theta_i)\cos(k\theta_i)$$

which can be efficiently computed. The Chebyshev polynomial terms and the subsequent transform of the query embedding Q is given by:

$$T_0(L)Q = Q$$
$$T_1(L)Q = LQ$$
$$T_k(L)Q = 2LT_{k-1}(L)Q - T_{k-2}(L)Q$$

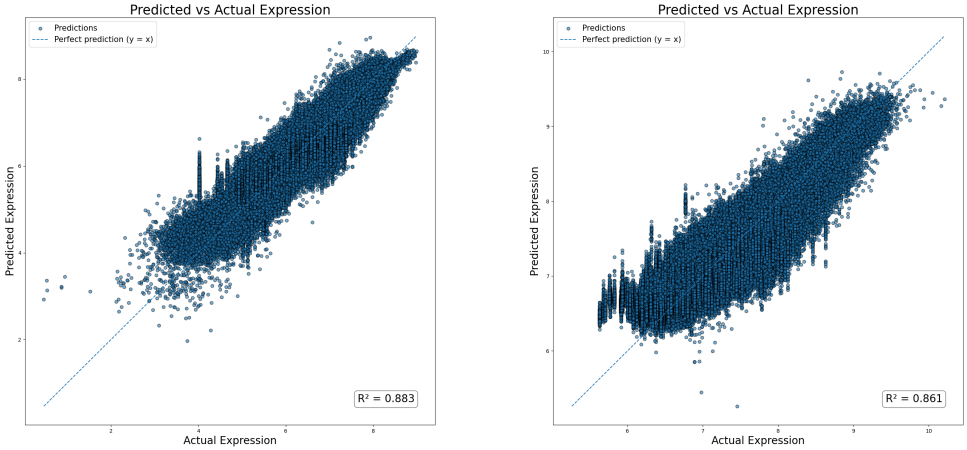
C Hyperparameters and Training Details

All hyperparameters and their values used to train the model are listed in the following table. The model tested contains 10.3 million parameters.

Hyperparameters	Value
Transformer model dimension	512
Number of Transformer blocks	3
Number of attention heads per block	8
Activation	GeLU
Peak learning rate	3e-4
Batch size	16
Chebyshev truncation index(K)	64
Diffusion rate(β)	0.1
Number of quadrature points(N)	100
Graph integration false edge threshold(α)	0.05

We apply dynamic learning rate scheme to ensure scalable learning behavior. At the start of pre-training, we first apply a linear ramping of learning rate with start factor 1e-5 that lasts for 10% of the total steps. After peak learning rate is achieved, we apply cosine annealing for the rest of the training steps to promote generalization and avoid forgetting. The model was pre-trained for 1 full epoch on 8 Nvidia H100 80G GPUs in parallel.

D Linear Probing of Gene Expression



(a) **Human Immune Cells,** $R^2 = 0.883$ (b) **Cancer Infiltrating Myeloid,** $R^2 = 0.861$

Figure 4: **Reconstructing gene expression** Shown are predicted and ground truth gene expression from the learned GREmLN embedding. A linear regressor was trained on GREmLN embedding to predict gene expression.

E Baseline Model Checkpoints

Here we list the exact pre-trained checkpoint for the models used in the experiments:

scGPT We used the 33M parameter model pre-trained on normal human cells, which is available at `https://github.com/bowang-lab/scGPT`

Geneformer We used the 30M parameter model with version number GF-6L-30M-i2048, which is available at `https://huggingface.co/ctheodoris/Geneformer`

scFoundation We use the 100M parameter model available at `https://github.com/biomap-research/scFoundation`