



This MICCAI paper is the Open Access version, provided by the MICCAI Society. It is identical to the accepted version, except for the format and this watermark; the final published version is available on SpringerLink.

# Spatial Transcriptomics Analysis of Zero-shot Gene Expression Prediction

Yan Yang<sup>1,2</sup> ✉, Md Zakir Hossain<sup>1,2,3</sup>, Xuesong Li<sup>1,2</sup>, Shafin Rahman<sup>4</sup>, and Eric Stone<sup>1</sup>

<sup>1</sup> Biological Data Science Institute,  
The Australian National University, Canberra, Australia

[Yan.Yang@anu.edu.au](mailto:Yan.Yang@anu.edu.au)

<sup>2</sup> Agriculture and Food, CSIRO, Canberra, Australia

<sup>3</sup> School of Elec Eng, Comp and Math Sci, Curtin University, Perth, Australia

<sup>4</sup> Department of ECE, North South University, Dhaka, Bangladesh

**Abstract.** Spatial transcriptomics (ST) captures gene expression fine-grained distinct regions (i.e., windows) of a tissue slide. Traditional supervised learning frameworks applied to model ST are constrained to predicting expression of gene types seen during training from slide image windows, failing to generalize to unseen gene types. To overcome this limitation, we propose a semantic guided network, a pioneering zero-shot gene expression prediction framework. Considering a gene type can be described by functionality and phenotype, we dynamically embed a gene type to a vector per its functionality and phenotype, and employ this vector to project slide image windows to gene expression in feature space, unleashing zero-shot expression prediction for unseen gene types. The gene type functionality and phenotype are queried with a carefully designed prompt from a pre-trained large language model. On standard benchmark datasets, we demonstrate competitive zero-shot performance compared to past state-of-the-art supervised learning approaches. Our code is available at <https://github.com/Yan98/SGN>.

**Keywords:** Spatial transcriptomics · Computational pathology · Gene expression prediction · Tissue slide image · Zero-shot learning.

## 1 Introduction

Spatial transcriptomics (ST) facilitates the exploration and diagnosis of diseases, providing gene expression for fine-grained regions, referred to as windows, in tissue slides. However, acquiring gene expression data for tissue slide windows involves resource-intensive experiments utilizing specialized equipment, typically operated by human experts [15]. This inevitably presents a challenge for collecting datasets for training end-to-end neural networks to predict gene expression from windows of easily obtainable tissue slide images. Furthermore, after deploying the trained network, a new demand may arise to predict the expression of gene types that are not used/seen during network training, i.e., unseen

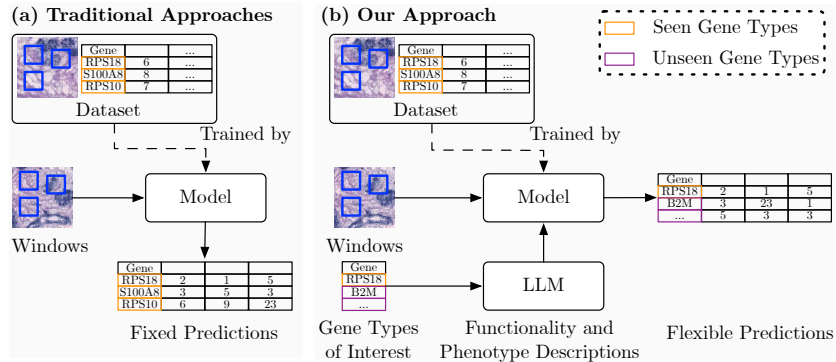


Fig. 1: Overview of fields. Training by a dataset of **seen** gene types, (a) traditional approaches predict expression of fixed (i.e., **seen**) gene types for **windows**; (b) by using a large language model (LLM) to describe gene type functionality and phenotype, we flexibly predict expression of **seen** and **unseen** gene types, i.e., zero-shot learning.

gene types, necessitating a revisit of the data collection process for network re-training [19,21,18]. Therefore, to address these challenges, this paper pioneeringly studies zero-shot gene expression prediction from tissue slide image windows. Our method not only enhances the efficiency and effectiveness of gene expression prediction but also accommodates the prediction of unseen gene types.

Recently, the computer vision community has studied gene expression prediction of tissue slide image windows from two perspectives: individual and joint gene expression prediction. In individual prediction approaches, such as those proposed by [9,5,22,2,1], networks are trained to predict gene expression of each window independently. While these approaches have promising results, they neglect spatially nearby slide image windows often share similar gene expression that could mutually benefit each other and overcome spatial transcriptomics data noises in the prediction task.

Considering this insight, the core of joint gene expression prediction, as outlined in [23,16], is to embed each window of a tissue slide image into features, connect windows as a graph, apply graph convolutions networks [12,8] to establish dependency among windows for refining window features, and predict gene expression of each window from the refined window features. The graph is constructed by treating each window as a node and connecting edges among spatially nearby window nodes. However, the similar windows within a slide image could also mutually benefit each other for each gene expression prediction, which is underexplored in past works.

Nevertheless, existing individual and joint gene expression prediction approaches focus on traditional supervised learning, thereby restricting expression prediction to gene types seen during training. Refer to Fig. 1 for examples. This paper takes a pioneering step forward by studying zero-shot gene expression prediction of windows in a tissue slide image, extending the prediction ability to unseen gene types by presenting our semantic guided network (SGN). Our key

idea is to describe a gene type by its functionality and phenotype and use the description to project tissue slide windows to gene expression in feature space.

Formally, our SGN implements zero-shot gene expression prediction in three stages. Firstly, we extract a feature vector for each window using a pre-trained network. Inspired by [23], in a slide image, we treat each window as a node, and construct a heterogeneous graph by connecting nearby windows and windows with similar extracted features. A graph convolution network [8] is then applied to refine the features of each window, benefiting from spatially nearby and feature-similar windows. Concurrently, to obtain the functionality and phenotype descriptions of the gene type of interest, we design a prompt to leverage a pre-trained large language model (LLM) for querying the description. As a general-purpose LLM potentially lacks domain-specific gene type knowledge, when internet access is available, we automatically scrap references related to the gene type to supplement the knowledge of the LLM for describing gene type functionality and phenotype. We then embed the gene type description into a projection vector by using a neural network. Finally, we perform a dot product between the refined window features and gene type projection vector to derive the gene expression. Experimentally, our method achieves competitive performance compared to the state-of-the-art traditionally supervised approach on standard benchmark datasets, evaluating our method on unseen gene types.

## 2 Method

**Overview.** We distinguish gene types into seen gene types  $\mathcal{C}^s$  and unseen gene types  $\mathcal{C}^u$ , ensuring  $\mathcal{C}^s \cap \mathcal{C}^u = \emptyset$ . Given a slide image containing  $N$  windows  $\{\mathbf{X}_i\}_{i=1}^N$  and the gene type  $c \in \mathcal{C}^s \cup \mathcal{C}^u$  of interest, our goal is to predict the expression  $\{\hat{y}_{i,c}\}_{i=1}^N$  of gene type  $c$  for all windows  $\{\mathbf{X}_i\}_{i=1}^N$  in the slide image, where  $\mathbf{X}_i \in \mathbb{R}^{H \times W \times 3}$ ,  $\hat{y}_{i,c} \in \mathbb{R}$ , and  $H$  and  $W$  are height and width of the window  $\mathbf{X}_i$ , respectively. This zero-shot learning framework is trained by using ground truth expression  $\{y_{i,c}\}_{i=1}^N$  of gene type  $c \in \mathcal{C}^s$  to supervise the predicted expression  $\{\hat{y}_{i,c}\}_{i=1}^N$ , and has three steps: i) window embedding, composing of feature extraction and refinement to obtain  $D$ -dimensional feature vectors  $\{\mathbf{z}_i\}_{i=1}^N$ , i.e.,  $\mathbf{z}_i \in \mathbb{R}^{1 \times D}$ ; ii) gene type embedding, obtaining functionality and phenotype descriptions  $\mathbf{T}_c$  of gene type  $c$  by using a pre-trained LLM, and deriving a projection vector  $\mathbf{v}_c \in \mathbb{R}^{1 \times D}$  for gene type  $c$  per the description  $\mathbf{T}_c$ ; iii) gene expression prediction, computing gene expression by applying a dot product  $\{\hat{y}_{i,c}\}_{i=1}^N = \{\mathbf{z}_i \cdot \mathbf{v}_c^\top\}_{i=1}^N$ . In testing, we evaluate our zero-shot ability by repeating the above three steps to predict the expression of the unseen gene type  $c \in \mathcal{C}^u$  on the slide image. We show the overall framework in Fig. 2.

### 2.1 Window Embedding

We use graph-based settings to embed the windows  $\{\mathbf{X}_i\}_{i=1}^N$  into  $\{\mathbf{z}_i\}_{i=1}^N$  through feature extractions and refinements, providing discriminative features for our zero-shot gene expression prediction task.

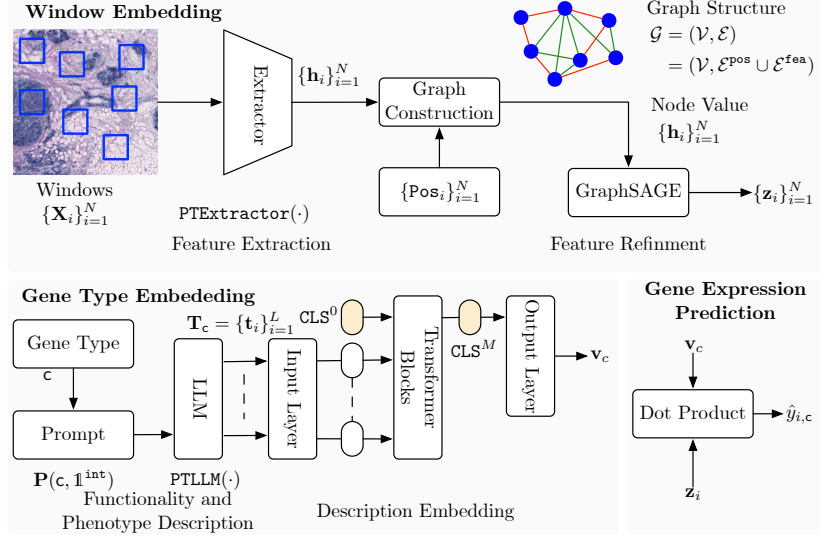


Fig. 2: Our framework. We have three stages: i) window embedding, extracting and refining features from each window by using an extractor and a GraphSAGE network that explores relations of window spatial positions and feature similarities; ii) gene type embedding, querying gene type functionality and phenotype from an LLM, and embedding the description; iii) gene type prediction, performing dot product between window embedding and gene type embedding to compute gene expression.

**Feature Extraction.** We use a pre-trained network [10],  $\text{PTExtractor}(\cdot)$ , to extract window features  $\{\mathbf{h}_i\}_{i=1}^N = \{\text{PTExtractor}(\mathbf{X}_i)\}_{i=1}^N$ , where  $\mathbf{h}_i \in \mathbb{R}^{1 \times D^{\mathbf{H}}}$  and  $D^{\mathbf{H}}$  is the extracted feature dimension. However, the window features  $\{\mathbf{h}_i\}_{i=1}^N$  are independently and locally extracted. They are short of context/global information of the slide image which has been proven to be beneficial for the prediction task [23], motivating us to refine the features in the next section.

**Feature Refinement.** We define a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  with a node set  $\mathcal{V}$  and an edge set  $\mathcal{E}$  for windows in the slide image, and use graph convolution network to refine window features  $\{\mathbf{h}_i\}_{i=1}^N$  based on the graph structure  $\mathcal{G}$ .

To construct  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , we set each window  $\mathbf{X}_i$  as a node to form the node set  $\mathcal{V} = \{v_i\}_{i=1}^N$ , and consider two edge types  $\mathcal{E} = \mathcal{E}^{\text{pos}} \cup \mathcal{E}^{\text{fea}}$  to explore relations of spatial position and feature similarity among windows, i.e., the two edge type gives context/global relations among the graph. With  $\text{Pos}_i \in \mathbb{R}^2$  describing spatial position of each window  $\mathbf{X}_i$  in the slide image, we have

$$\mathcal{E}^{\text{pos}} = \{e_{ij} \mid v_i, v_j \in \mathcal{V} \times \mathcal{V} \wedge \phi(\text{Pos}_i, \text{Pos}_j, \{\text{Pos}_k\}_{k=1}^N)\}, \quad (1)$$

$$\mathcal{E}^{\text{fea}} = \{e_{ij} \mid v_i, v_j \in \mathcal{V} \times \mathcal{V} \wedge \phi(\mathbf{h}_i, \mathbf{h}_j, \{\mathbf{h}_k\}_{k=1}^N)\}. \quad (2)$$

$\phi(\cdot, \cdot, \cdot)$  is a k-nearest neighbors (k-NN) function, determining nearest neighbors, e.g.,  $\phi(\text{Pos}_i, \text{Pos}_j, \{\text{Pos}_k\}_{k=1}^N)$  computes if  $\text{Pos}_i$  is one of the nearest neighbors of  $\text{Pos}_j$  among  $\{\text{Pos}_k\}_{k=1}^N$ .

With the graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , we refine the window features  $\{\mathbf{h}_i\}_{i=1}^N$  into  $\{\mathbf{z}_i\}_{i=1}^N$  with a GraphSAGE network [8] along the two edges  $\mathcal{E}^{\text{pos}} \cup \mathcal{E}^{\text{fea}}$ . Mathematically, the refined features  $\{\mathbf{z}_i\}_{i=1}^N$  is defined by

$$\mathbf{z}_i = \left[ \mathbf{h}_i \parallel \frac{1}{|\mathcal{N}_i^{\text{pos}}|} \sum_{j \in \mathcal{N}_i^{\text{pos}}} \mathbf{h}_j \parallel \frac{1}{|\mathcal{N}_i^{\text{fea}}|} \sum_{k \in \mathcal{N}_i^{\text{fea}}} \mathbf{h}_k \right] \mathbf{W}^{\mathbf{z}}, \quad (3)$$

where  $\mathbf{W}^{\mathbf{z}} \in \mathbb{R}^{(3 \times D^{\text{H}}) \times D}$  is a linear weight matrix,  $[\cdot \parallel \cdot \parallel \cdot]$  is a concatenation operator, and  $\mathcal{N}_i^{\text{pos}} = \{j \mid e_{jk} \in \mathcal{E}^{\text{pos}} \wedge k = i\}$  and  $\mathcal{N}_i^{\text{fea}} = \{k \mid e_{kj} \in \mathcal{E}^{\text{fea}} \wedge j = i\}$  give indexes of nodes connected to  $v_i$  along  $\mathcal{E}^{\text{pos}}$  and  $\mathcal{E}^{\text{fea}}$ , respectively. The refined features  $\{\mathbf{z}_i\}_{i=1}^N$  are finally used to perform our zero-shot gene expression prediction in Sec. 2.3.

## 2.2 Gene Type Embedding

We generate the functionality and phenotype description for a gene type  $\mathbf{c}$ , and dynamically embed the description into a vector  $\mathbf{v}_{\mathbf{c}}$  that can project refined window features  $\{\mathbf{z}_i\}_{i=1}^N$  into the gene type expression.

**Functionality and Phenotype Description.** To generate descriptions of functionality and phenotype for a gene type  $\mathbf{c}$ , we leverage a pre-trained LLM, and design a prompt  $\mathbf{P}(\mathbf{c}, \mathbf{1}^{\text{int}})$  that is adjusted by internet access availability  $\mathbf{1}^{\text{int}}$ . If the internet access is absent, i.e.,  $\mathbf{1}^{\text{int}} = \text{false}$ , the prompt  $\mathbf{P}(\mathbf{c}, \mathbf{1}^{\text{int}})$  directly query the functionality and phenotype description from the LLM. Conversely, with the internet access, i.e.,  $\mathbf{1}^{\text{int}} = \text{true}$ , we supplement the knowledge base of the LLM by providing a domain-specific gene type reference, conditionally prompting the LLM to generate the functionality and phenotype description. The generated description  $\mathbf{T}_{\mathbf{c}}$  are obtained by

$$\mathbf{T}_{\mathbf{c}} = \text{PTLLM}(\mathbf{P}(\mathbf{c}, \mathbf{1}^{\text{int}})), \quad \mathbf{T} \in \mathbb{R}^{L \times D^{\text{T}}}. \quad (4)$$

Here,  $\text{PTLLM}(\cdot)$  is a pre-trained LLM model [11],  $L$  is the length of the description, and  $D^{\text{T}}$  is the feature dimension of  $\text{LLM}(\cdot)$ . We preserve the feature representation capability of the LLM by discarding its classification layer, using the final embedding layer output as our description  $\mathbf{T}_{\mathbf{c}}$ .

**Description Embedding.** We embed  $\mathbf{T}_{\mathbf{c}}$  into a vector  $\mathbf{v}_{\mathbf{c}}$  by using a transformer [6], aligning them to the feature space of the refined window features  $\{\mathbf{z}_i\}$ , and summarising information beneficial to the expression prediction of gene type  $\mathbf{c}$ . The transformer has an input layer, a list of transformer blocks, and an output layer in order. The computation is described as follows. We have an input layer that projects  $\mathbf{T}_{\mathbf{c}}$  to a  $D^{\text{E}}$ -dimensional matrix by using a weight matrix  $\mathbf{W}^{\mathbf{T}_{\mathbf{c}}} \in \mathbb{R}^{D^{\text{T}} \times D^{\text{E}}}$ , and append a  $[\text{CLS}^0]$  token to the projected matrix, obtaining  $\mathbf{E}_{\mathbf{c}}^0$ ,

$$\mathbf{E}_{\mathbf{c}}^0 = \{\text{CLS}^0, \mathbf{t}_1 \mathbf{W}^{\mathbf{T}_{\mathbf{c}}}, \mathbf{t}_2 \mathbf{W}^{\mathbf{T}_{\mathbf{c}}}, \dots, \mathbf{t}_L \mathbf{W}^{\mathbf{T}_{\mathbf{c}}}\}, \quad (5)$$

where  $\mathbf{t}_i$  is the  $i$ -th token embedding of  $\mathbf{T}_{\mathbf{c}}$ , i.e.,  $\mathbf{T}_{\mathbf{c}} = \{\mathbf{t}_i\}_{i=1}^L$ . Assuming there are  $M$  transformer blocks, for  $1 \leq m \leq M$ , we compute embedding  $\mathbf{E}_{\mathbf{c}}^m$  of the

Table 1: Quantitative gene expression prediction comparisons with SOTA methods on the STNet dataset and 10xProteomic dataset. We present the performance of our method trained by traditional supervised learning as our performance upper bound. For the zero-shot setting, we set gene types selected by past works as unseen gene types, and test these gene types.

Method	Zero-shot	MSE $_{\times 10^2}$	MAE $_{\times 10^1}$	PCC@F $_{\times 10^1}$	PCC@S $_{\times 10^1}$	PCC@M $_{\times 10^1}$
Experiments on the STNet dataset.						
STNet [9]	✗	4.52	1.70	0.05	0.92	0.93
NSL [5]	✗	-	-	-0.71	0.25	0.11
EGN [22]	✗	4.10	1.61	1.51	2.25	2.02
HSANet [2]	✗	4.00	1.59	1.60	2.28	2.38
CFNet [1]	✗	6.30	1.66	2.12	3.06	3.00
EGGN [23]	✗	3.94	1.61	2.12	3.05	2.92
Ours	✗	4.38	1.72	2.00	3.03	2.83
Ours	✓	11.86	2.88	1.79	2.89	2.69
Experiments on the 10xProteomic dataset.						
STNet [9]	✗	12.40	2.64	1.25	2.26	2.15
NSL [5]	✗	-	-	-3.73	1.84	0.25
EGN [22]	✗	5.49	1.55	6.78	7.21	7.07
HSANet [2]	✗	4.00	1.54	6.93	7.43	7.20
CFNet [1]	✗	4.00	1.49	8.00	8.16	8.02
EGGN [23]	✗	3.52	1.31	7.06	7.60	7.44
Ours	✗	4.27	1.67	8.22	8.38	8.15
Ours	✓	13.05	2.70	6.33	6.51	6.48

$m$ -th block as

$$\mathbf{E}_c^m = \text{FFN}^m \left( \text{Attention}^m \left( \mathbf{E}_c^{m-1}, \mathbf{E}_c^{m-1}, \mathbf{E}_c^{m-1} \right) \right). \quad (6)$$

$\text{FFN}^m(\cdot)$  and  $\text{Attention}^m(\cdot, \cdot, \cdot)$  are respectively a feedforward layer and an attention layer in the  $m$ -th block [6]. In an output layer, we then pop out the  $[\text{CLS}^M]$  token from  $\mathbf{E}_c^M$ , and project it to  $\mathbf{v}_c$  by using a weight matrix  $\mathbf{W}^v \in \mathbb{R}^{D^E \times D}$ .

### 2.3 Gene Expression Prediction

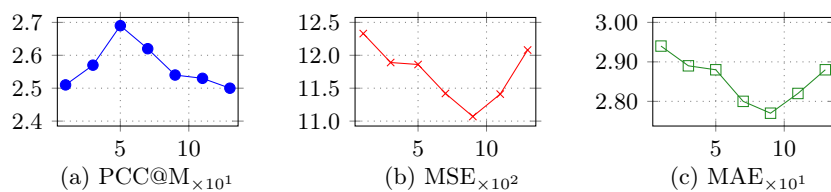
With refined window features  $\{\mathbf{z}_i\}_{i=1}^N$  and gene type embedding  $\mathbf{v}_c$  in a shared feature space,  $\mathbf{v}_c$  is used to project each  $\mathbf{z}_i$  for performing zero-shot expression prediction of gene type  $c$  as

$$\{\hat{y}_{i,c}\}_{i=1}^N = \{\mathbf{z}_i \cdot \mathbf{v}_c^\top\}_{i=1}^N. \quad (7)$$

With the prediction and ground truth, we optimize our network with a mean square error  $\mathcal{L}_{\text{mse}}$  and batch-wise Pearson correlation coefficient (PCC) loss  $\mathcal{L}_{\text{pcc}}$ .

Table 2: Ablation of our model components. We use ‘FI’ and ‘PT’ for shorts of functionality and phenotype.

GraphSAGE		LLM		MSE $_{\times 10^2}$	MAE $_{\times 10^1}$	PCC@M $_{\times 10^1}$
$\mathcal{E}^{\text{pos}}$	$\mathcal{E}^{\text{fea}}$	FI	PT			
$\times$	$\times$	✓	✓	10.70	2.71	2.58
✓	$\times$	✓	✓	11.59	2.85	2.65
$\times$	✓	✓	✓	10.66	2.73	2.48
✓	✓	$\times$	✓	12.53	2.97	2.61
✓	✓	✓	$\times$	11.52	2.83	2.58
✓	✓	✓	✓	11.86	2.88	2.69

Fig. 3: Ablation study on the number of neighbors used in constructing  $\mathcal{E}^{\text{fea}}$ .

### 3 Experiment

**Datasets.** We experiment with the STNet dataset [9] and 10xProteomic datasets<sup>5</sup>. The STNET dataset and 10xProteomic dataset have 30,612 windows from 68 slide images and 24,263 windows from 6 slide images, respectively. We follow the dataset pre-processing and cross-fold validation settings of [22,23]. Past works select 250 gene types with the largest mean across the dataset as prediction targets. To compare with the past gene expression prediction works, we use their unselected gene types in training as seen gene types and their selected gene types in testing as unseen gene types.

**Evaluation Metrics.** Our method is evaluated with mean squared error (MSE), mean absolute error (MAE), first quartile of PCC (PCC@F), median of PCC (PCC@S), and mean of PCC (PCC@M).

#### 3.1 Experimental Result

We compare with state-of-the-art methods on the STNet dataset and 10xProteomic dataset in Tab. 1. Among all the methods, we are the only method that predicts gene expression in a zero-shot manner. Though our method performs poorly on absolute gene expression prediction evaluation metrics, MSE and MAE, our method successfully captures the relative variations of gene expression across different windows, i.e., PCC@F, PCC@S, and PCC@M. For example, on

<sup>5</sup> <https://www.10xgenomics.com/resources/datasets>

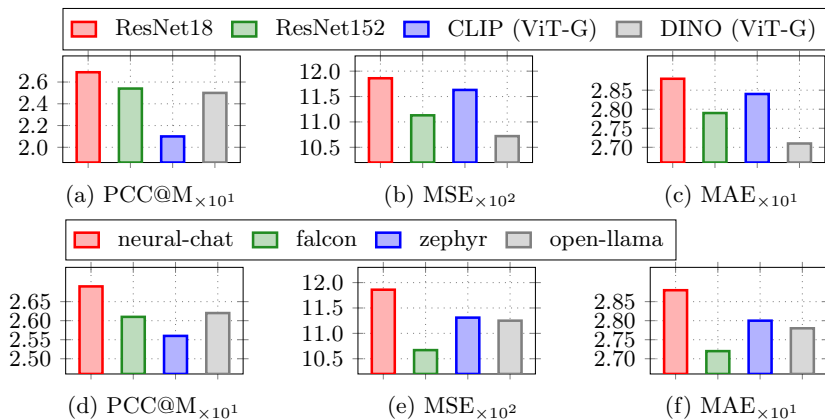


Fig. 4: Ablation study of the pre-trained feature extractor (a-c) [10,3,17] and LLM (d-f) [11,7,20,14].

the STNet dataset, our method finds 0.269 PCC@M that is 0.031 PCC@M lower than the state-of-the-art method, CFNet (0.300 PCC@M). As demonstrated by [22,23], capturing relative variations of gene expression are most important in our task, and our method shows competitive zero-shot gene expression prediction performance against the state-of-the-art traditional supervised learning approaches in these metrics. This validates the performance of our zero-shot gene expression prediction framework.

### 3.2 Ablation

**Model Component.** We ablate the our model components in Tab. 2. When disabling the GraphSAGE, a single linear layer is used to unify the feature dimensions. Consistently, our components improve the prediction performance. **Number of Neighbors.** We study the number of k-NN edges used for constructing  $\mathcal{E}^{\text{fea}}$  in Fig. 3. Again, we are biased on PCC-based evaluation metrics. Having 5 kNN edges finds the most balanced performance.

**Extractor and LLM.** We ablate the pre-trained feature extractor and LLM in Fig. 4. In contrast to recent trends that prefer using a ViT-G feature extractor [13,24,4] for multi-modality feature interaction, with using neural-chat as our pre-trained LLM, our result suggests that using ResNet18 as a feature extractor finds the best performance.

**Reference for LLM.** By default, our model is evaluated by automatically scraping gene type references from the internet. When internet access is disabled, we find a lower performance, i.e., 0.256 PCC@M, 0.124 MAE, and 0.295 MSE. This shows the effectiveness of scraping gene type references for supplementing the knowledge base of the LLM.

**Scalability.** When using 25%, 50%, and 100% of data, the PCC@M is 0.207, 0.240, and 0.269, respectively. By halving and doubling the model parameters,



we have PCC@M of 0.245 and 0.273, respectively, excluding pre-trained feature extractor and LLM parameters. This investigates our data and model scalability.

## 4 Conclusion

This paper studies an SGN framework for zero-shot gene expression prediction of windows in a tissue slide image. Given a gene type of interest, we design a prompt to query the functionality and phenotype of the gene from a pre-trained LLM. The obtained gene type description is then used to project each window to the expression of the gene type in feature space. Finally, we compare our zero-shot gene expression prediction framework with the past state-of-the-art supervised learning approaches, and experimentally demonstrate competitive gene expression prediction performance.

**Acknowledgments.** The authors would like to thank Machine Learning & Artificial Intelligence Future Science Platforms, CSIRO for computation resource funding. S.R. is grateful for support from the Conference Travel and Research Grants (CTRG) for 2023-2024 from North South University, under Grant ID: CTRG-23-SEPS-20.

**Disclosure of Interests.** The authors declare no conflicts of interest.

## References

1. Chen, C., Zhang, Z., Mounir, A., Liu, X., Huang, B.: Spatial gene expression prediction using coarse and fine attention network. In: Liu, F., Sadanandan, A.A., Pham, D.N., Mursanto, P., Lukose, D. (eds.) PRICAI 2023: Trends in Artificial Intelligence - 20th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2023, Jakarta, Indonesia, November 15-19, 2023, Proceedings, Part III. Lecture Notes in Computer Science, vol. 14327, pp. 396–408. Springer (2023). [https://doi.org/10.1007/978-981-99-7025-4\\_34](https://doi.org/10.1007/978-981-99-7025-4_34), [https://doi.org/10.1007/978-981-99-7025-4\\_34](https://doi.org/10.1007/978-981-99-7025-4_34)
2. Chen, C., Zhang, Z., Tang, P.: Spatial gene expression prediction using hierarchical sparse attention. In: Luo, B., Cheng, L., Wu, Z., Li, H., Li, C. (eds.) Neural Information Processing - 30th International Conference, ICONIP 2023, Changsha, China, November 20-23, 2023, Proceedings, Part X. Communications in Computer and Information Science, vol. 1964, pp. 594–606. Springer (2023). [https://doi.org/10.1007/978-981-99-8141-0\\_44](https://doi.org/10.1007/978-981-99-8141-0_44), [https://doi.org/10.1007/978-981-99-8141-0\\_44](https://doi.org/10.1007/978-981-99-8141-0_44)
3. Cherti, M., Beaumont, R., Wightman, R., Wortsman, M., Ilharco, G., Gordon, C., Schuhmann, C., Schmidt, L., Jitsev, J.: Reproducible scaling laws for contrastive language-image learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2818–2829 (2023)
4. Dai, W., Li, J., Li, D., Tiong, A., Zhao, J., Wang, W., Li, B., Fung, P., Hoi, S.: Instructblip: Towards general-purpose vision-language models with instruction tuning (05 2023)
5. Dawood, M., Branson, K., Rajpoot, N., Minhas, F.u.A.A.: All you need is color: Image based spatial gene expression prediction using neural stain learning (08 2021)

6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021), <https://openreview.net/forum?id=YicbFdNTTy>
7. falcon: Falcon-rw-1b-instruct-openorca. <https://huggingface.co/ericzzz/falcon-rw-1b-instruct-openorca> (2023)
8. Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. pp. 1024–1034 (2017), <https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7e9bea9-Abstract.html>
9. He, B., Bergenstr  hle, L., Stenbeck, L., Abid, A., Andersson, A., Borg, A., Maaskola, J., Lundeberg, J., Zou, J.: Integrating spatial gene expression and breast tumour morphology via deep learning. *Nature Biomedical Engineering* **4**, 1–8 (08 2020). <https://doi.org/10.1038/s41551-020-0578-x>
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. pp. 770–778 (06 2016). <https://doi.org/10.1109/CVPR.2016.90>
11. Intel: Neural-chat-v3-1. <https://huggingface.co/Intel/neural-chat-7b-v3-1> (2023)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. *CoRR* **abs/1609.02907** (2016), <http://arxiv.org/abs/1609.02907>
13. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models (01 2023). <https://doi.org/10.48550/arXiv.2301.12597>
14. open llama: open-llama-3b-v2-instruct. <https://huggingface.co/mediocredev/open-llama-3b-v2-instruct> (2023)
15. Marx, V.: Method of the year: spatially resolved transcriptomics. *Nature Methods* **18**, 9–14 (01 2021). <https://doi.org/10.1038/s41592-020-01033-y>
16. Mej  a, G., C  rdenas, P., Ruiz, D., Castillo, A., Arbel  ez, P.: SEPAL: spatial gene expression prediction from local graphs. *arXiv* (2023)
17. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P., Li, S., Misra, I., Rabbat, M.G., Sharma, V., Synnaeve, G., Xu, H., J  gou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision. *CoRR* **abs/2304.07193** (2023). <https://doi.org/10.48550/ARXIV.2304.07193>, <https://doi.org/10.48550/arXiv.2304.07193>
18. Pourpanah, F., Abdar, M., Luo, Y., Zhou, X., Wang, R., Lim, C.P., Wang, X., Wu, Q.M.J.: A review of generalized zero-shot learning methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(4), 4051–4070 (2023). <https://doi.org/10.1109/TPAMI.2022.3191696>, <https://doi.org/10.1109/TPAMI.2022.3191696>
19. Rahman, S., Khan, S.H., Porikli, F.: A unified approach for conventional zero-shot, generalized zero-shot, and few-shot learning. *IEEE Trans. Image Process.* **27**(11), 5652–5667 (2018). <https://doi.org/10.1109/TIP.2018.2861573>, <https://doi.org/10.1109/TIP.2018.2861573>

20. Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., von Werra, L., Fourier, C., Habib, N., Sarrazin, N., Sanseviero, O., Rush, A.M., Wolf, T.: Zephyr: Direct distillation of LM alignment. CoRR **abs/2310.16944** (2023). <https://doi.org/10.48550/ARXIV.2310.16944>, <https://doi.org/10.48550/arXiv.2310.16944>
21. Wang, W., Zheng, V.W., Yu, H., Miao, C.: A survey of zero-shot learning: Settings, methods, and applications. *ACM Trans. Intell. Syst. Technol.* **10**(2), 13:1–13:37 (2019). <https://doi.org/10.1145/3293318>, <https://doi.org/10.1145/3293318>
22. Yang, Y., Hossain, M., Stone, E., Rahman, S.: Exemplar guided deep neural network for spatial transcriptomics analysis of gene expression prediction (10 2022)
23. Yang, Y., Hossain, M.Z., Stone, E., Rahman, S.: Spatial transcriptomics analysis of gene expression prediction using exemplar guided graph neural network. *Pattern Recognition* **145**, 109966 (2024). <https://doi.org/https://doi.org/10.1016/j.patcog.2023.109966>, <https://www.sciencedirect.com/science/article/pii/S0031320323006647>
24. Zhu, D., Chen, J., Shen, X., Li, X., Elhoseiny, M.: Minigpt-4: Enhancing vision-language understanding with advanced large language models (04 2023)