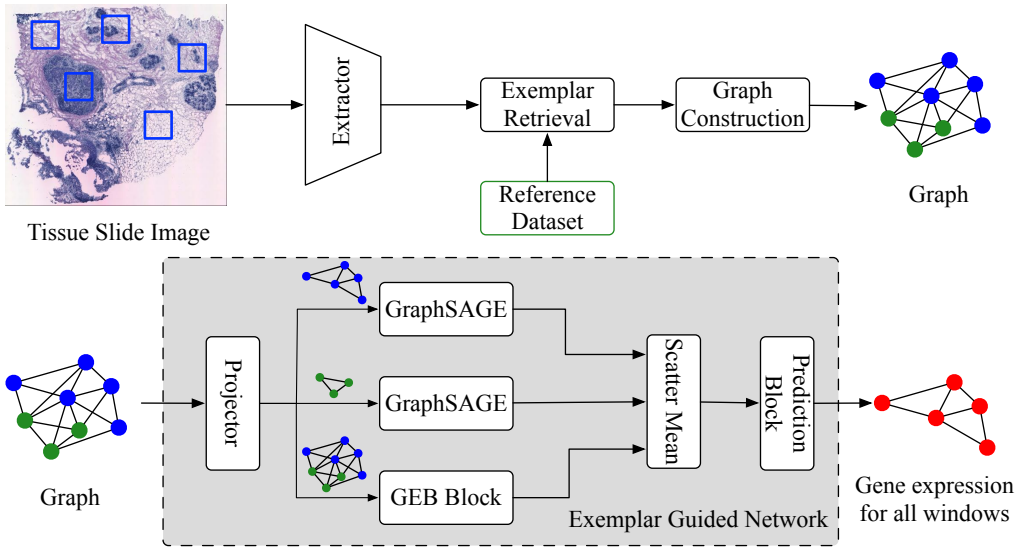
Graphical Abstract

Spatial Transcriptomics Analysis of Gene Expression Prediction using Exemplar Guided Graph Neural Network

Yan Yang, Md Zakir Hossain, Eric Stone, Shafin Rahman



In this paper, we aim to predict gene expression of each window in a tissue slide image. Given a tissue slide image, we encode the windows to feature space, retrieve its exemplars from the reference dataset, construct a graph, and then dynamically predict gene expression of each window with our exemplar guided graph network.

Email addresses: u6169130@anu.edu.au (Yan Yang), zakir.hossain@anu.edu.au (Md Zakir Hossain), eric.stone@anu.edu.au (Eric Stone), shafin.rahman@northsouth.edu (Shafin Rahman)

Highlights

Spatial Transcriptomics Analysis of Gene Expression Prediction using Exemplar Guided Graph Neural Network

Yan Yang, Md Zakir Hossain, Eric Stone, Shafin Rahman

- We propose an exemplar guided graph network to accurately predict gene expression from a slide image window.
- We design a graph construction strategy to connect windows and exemplars for performing exemplar learning of gene expression prediction.
- We propose a graph exemplar bridging block to revise the window feature by using its nearest exemplars.
- Experiments on two standard benchmark datasets demonstrate our superiority when compared with state-of-the-art approaches.

Spatial Transcriptomics Analysis of Gene Expression Prediction using Exemplar Guided Graph Neural Network

Yan Yang[a], Md Zakir Hossain[a,b], Eric Stone[a,*], Shafin Rahman[c]

[a] *Biological Data Science Institute, The Australian National University, Canberra, Australia*
[b] *Optus Centre for AI, Curtin University, Perth, Australia*
[c] *Department of Electrical and Computer Engineering, North South University, Bangladesh*

Abstract

Spatial transcriptomics (ST) is essential for understanding diseases and developing novel treatments. It measures the gene expression of each fine-grained area (i.e., different windows) in the tissue slide with low throughput. This paper proposes an exemplar guided graph network dubbed EGGN to accurately and efficiently predict gene expression from each window of a tissue slide image. We apply exemplar learning to dynamically boost gene expression prediction from nearest/similar exemplars of a given tissue slide image window. Our framework has three main components connected in a sequence: i) an extractor to structure a feature space for exemplar retrievals; ii) a graph construction strategy to connect windows and exemplars as a graph; iii) a graph convolutional network backbone to process window and exemplar fea-

*Corresponding author.
Email addresses: u6169130@anu.edu.au (Yan Yang), zakir.hossain@anu.edu.au (Md Zakir Hossain), eric.stone@anu.edu.au (Eric Stone), shafin.rahman@northsouth.edu (Shafin Rahman)

tures, and a graph exemplar bridging block to adaptively revise the window features using its exemplars. Finally, we complete the gene expression prediction task with a simple attention-based prediction block. Experiments on standard benchmark datasets indicate the superiority of our approach when compared with past state-of-the-art methods.

Keywords: Spatial transcriptomics, Gene expression prediction, Deep learning, Graph convolution, Tissue slide image

1. Introduction

Based on an editorial report of the Natural Methods [1], spatial transcriptomics (ST) is the future of studying disease because of its capabilities in measuring gene expression of fine-grained areas (i.e., different windows) of tissue slides. However, ST is in low throughput due to limitations in concurrent analysis for the candidate windows [2]. To accurately predict gene expression from each window of a tissue slide image (Fig. 1), this paper proposes a solution named exemplar guided graph network (EGGN), allowing efficient and concurrent analysis.

Previous works adopt end-to-end neural networks, namely STNet [3] and NSL [4], to independently establish a mapping between gene expression and the slide image window. STNet is a transfer learning-based approach that finetunes a pretrained DenseNet [5] for the gene expression prediction task. On the contrary, NSL maps the color intensity of the slide image window to gene expression by a single convolution operation. Though amenable to high throughput because of using neural networks, their prediction performance is inferior.

We investigate two important limitations of the existing approaches [3, 4].

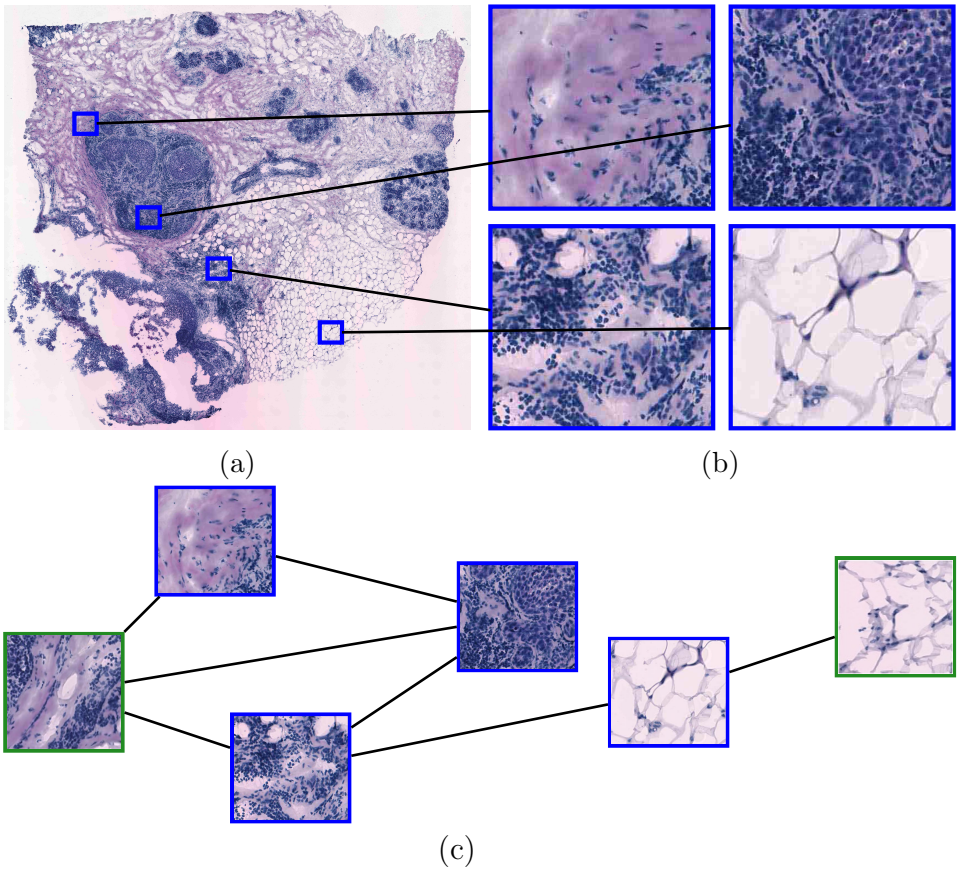


Figure 1: Overview of fields. Each fine-grained area (i.e., window) of (a) a tissue slide image has a distinct expression of the same gene types. For example, we have a tissue slide image with (b) four windows and each of the windows corresponds with the different expression of the same gene types. Our goal is to predict the gene expression of each window. Our key idea is to build (c) a graph connecting windows spatially close with each other and the exemplars for each window, which is used by our proposed framework to dynamically benefit gene expression prediction.

i) Local feature aggregation: gene expression prediction can be considered as individually aggregating and identifying the feature of each gene type for the slide image window. The long-range dependency, i.e., global context, among identified features is needed to reason about complex scenarios [6, 7], as those features are generally non-uniformly distributed across the slide image (see

Sec. 3 for details). STNet (i.e., a pure convolution approach) emphasizes local context during feature aggregation within a limited slide image window. It fails to bring interaction among features that are apart from each other. By experimenting with extensive state-of-the-art (SOTA) network architectures, we show that models with local feature aggregation achieve low performance when compared to models with long-range dependencies on the slide image. ii) Vulnerable assumption: identifying gene expression by directly detecting the color intensity of the slide image window is vulnerable. By experimenting on standard benchmark datasets, we show that NSL only works in extreme cases. For example, in the STNet dataset [3], tumor areas of slide image windows are usually purple, which benefits tumor-related gene expression prediction. This method finds a negative Pearson correlation coefficient (PCC) when evaluating the model with the least reliable gene expression prediction, i.e., PCC@F in Tab. 1.

Furthermore, as shown by [3], windows of a slide image spatially close with each other usually exhibit similar features, suggesting correlated gene expression among them. However, both of the above two approaches independently predict the gene expression of each window and ignore interactions among these windows. Windows distributed over the slide image can be connected to construct a graph for gene expression prediction with a graph convolutional network (GCN) [8]. Compared to independently modeling the window, graph-based modeling of the task allows reasoning dependency among windows for predicting gene expression, considering the local structure/context/spatial relations among windows.

In this paper, we propose an EGGN framework to address the above lim-

itations. EGGN uses GraphSAGE [9] as a GCN backbone and incorporates exemplar learning concepts for the gene expression prediction task. To enable exemplar retrieval of a given slide image window, we use a feature extractor for defining a feature space to calculate the similarity between two slide image windows. Then, we construct a graph, considering the local context (i.e., nearby windows) and global context (i.e., shared exemplars act as anchors for information propagation) for the gene expression prediction task. We use GraphSAGE layers as our backbone to allow interactions among windows and exemplars. Meanwhile, we have a graph exemplar bridging (GEB) block to update window features by the exemplars and the gene expression of exemplars. Allowing dynamic information propagation, the exemplar feature also receives and is updated with the status of the window features. Semantically, the former update corresponds with 'the known gene expression', and the latter corresponds with 'the gene expression the model wants to be known'. Finally, we have an attention-based prediction block to aggregate exemplars of each window and the exemplar-revised window features, for predicting gene expression.

Our contributions are summarised as follows:

- We propose an EGGN framework, a GCN-based exemplar learning approach, to accurately predict gene expression from the slide image window;
- We design a graph construction strategy to connect windows and exemplars for performing exemplar learning of gene expression prediction under GCNs;
- We propose a GEB block to revise the window feature by using its

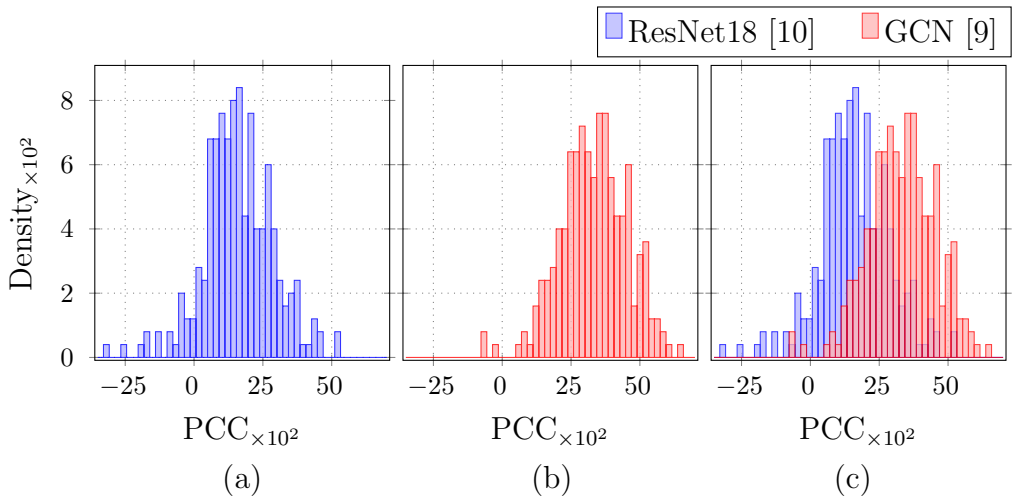


Figure 2: We compare the PCC distribution obtained by using (a) ResNet18 model and (b) GCN model for gene expression prediction, where window features used in the GCN [9] are extracted by the ImageNet-1K pre-trained ResNet18 [10]. The x-axis and y-axis respectively denote the PCC and the density at different PCC. We also combine the plot (a) and plot (b) into the plot (c), for a better comparison. As shown, modeling the spatial relations among windows out-weights feature learning of each window.

nearest exemplars;

- Experiments on two standard benchmark datasets demonstrate our superiority when compared with SOTA approaches.

A preliminary version of this paper has been published previously [11]. However, it considers interactions between exemplars and each window independently, and ignores the exemplar shared by multiple different windows potentially serve as anchors for facilitating information propagation among spatial apart windows of a slide image. In this paper, we extend our previous version as follows: i) we demonstrate that reasoning spatial relations among windows out-weights feature learning in our problem, which motivates us to design a GCN-based exemplar learning framework (Fig. 2); ii) we address the above independent interaction and information propagation bottleneck

issues in our new exemplar learning framework; iii) we analyze the proposed framework with careful ablation designs, *e.g.*, we show that the proposed exemplar learning strategy benefits general GCN frameworks; iv) we explore SOTA GCN-based methods in extensive experimental comparisons.

2. Related Work

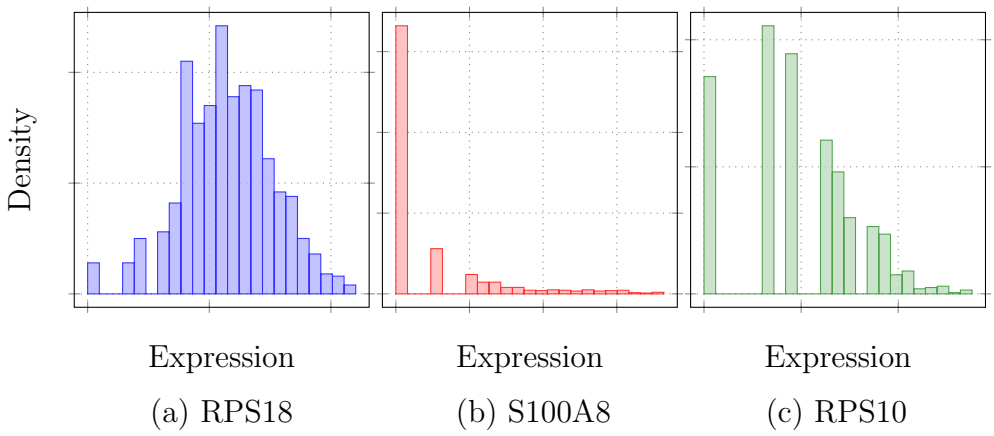
This section first reviews the study of gene expression prediction. Then, we summarise recent exemplar learning achievements in natural language processing and computer vision domains. Finally, we briefly introduce the background of GCNs.

Gene Expression Prediction. Measuring gene expression is a fundamental process in developing novel treatments and monitoring human diseases [12]. Recently, deep learning methods have been introduced to this task. Existing methods predict the gene expression either from DNA sequences [12] or slide images [3, 4, 13]. This paper explores the latter approach which is divided into two streams. First, Schmauch *et al.*[13] employs a multi-stage method including pretrained ResNet feature extractions [10] and a K-Means algorithm to model the Bulk RNA-Seq technique [14]. It measures the gene expression across cells within a large predefined area that is up to $10^5 \times 10^5$ pixels in a corresponding slide image [13]. However, this approach is ineffective for studies that require fine-grained gene expression information, such as tumor heterogeneity [15]. Second, on the contrary, He *et al.* [13, 5] and Dawood *et al.* [4] design a STNet and an NSL framework to predict the gene expression for each window (i.e., fine-grained area) of a slide image. This corresponds with the ST technique [16]. We model ST to predict gene expression, as this

potentially solves the bulk RNA-Seq prediction task simultaneously [3]. For example, aggregation of gene expression predictions for each window across a slide image results in a bulk RNA-Seq prediction.

Exemplar Learning. The K-nearest neighborhood classifier is the most straightforward case of exemplar learning. It classifies the input by considering the class labels of the nearest neighbors. Exemplar learning is a composition of retrieval tasks and learning tasks [17]. It has been widely employed to increase the model capability by bringing in extra knowledge of similar exemplars. The applications of exemplar learning include visual question answering [18, 19, 20, 21], language generation [22, 23, 24, 25], real-time visual object tracking [24], fact-checking [26], fact completion [27, 28], and dialogue [29]. However, most of the exemplar learning approaches do not apply to our task because of the domain shift. This paper investigates an application of exemplar learning in gene expression prediction from slide image windows. As a result, we devise a GEB block to adapt exemplar learning to our task.

Graph Convolutional Network. GCNs are an extension of Convolutional Neural Networks (CNNs) designed for modeling non-Euclidean structured data such as graphs and reasoning the underlying complex relationships [8]. GCNs can be categorized into two main groups: spectral-based and spatial-based approaches. Spectral-based GCNs rely on the graph Laplacian and perform convolutions of spatial domain in the frequency domain [30, 31], while spatial-based GCNs utilize message-passing frameworks to progressively propagate information from source nodes to their neighbors (target nodes) along the edges of the graph. After information propagation, target nodes aggregate all received information, potentially bringing the perception of the graph



(a) RPS18 (b) S100A8 (c) RPS10

Figure 3: Gene expression distributions of STNet dataset [3]. Each gene expression is log-transformed. (a) is the well-distributed expression of gene RPS18. (b) and (c) are the long-tail distributed expression of gene S100A8 and gene RPS10.

structure into final node features [9, 32]. Some works also explore edge attributes during the propagation process [33, 34, 35]. GCNs with different information propagation and aggregation strategies have been proposed for diverse vision tasks [36, 37, 9, 32, 38]. In this paper, we propose a GEB to manage information propagation between windows and exemplars for the gene expression prediction task. The GEB block dynamically revises the window features using the nearest exemplars and updates the exemplars to improve the gene expression prediction.

3. EGGN Framework

Problem Formulation. We have a tissue slide image containing multiple windows; each window is annotated with gene expression. We denote the slide image as pairs of slide image windows $\mathbf{w}_i$ and gene expression $\mathbf{y}_i$, i.e., $\{(\mathbf{w}_i, \mathbf{y}_i)\}_{i=1}^{\mathsf{p}}$, where p is the set size. We aim to train a neural network model to predict $\mathbf{y}_i$ from $\mathbf{w}_i$.

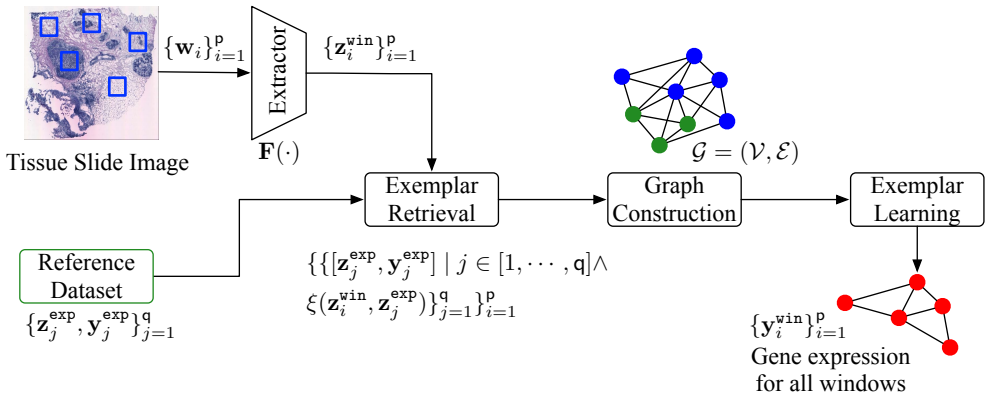


Figure 4: Framework overview. Given a slide images containing $\mathtt{p}$ windows $\mathbf{w}_i$, i.e., $\{\mathbf{w}_i\}_{i=1}^{\mathtt{p}}$, we embed each window $\mathbf{w}_i$ into features by using the feature extractor $\mathbf{F}(\cdot)$. We have $\mathbf{z}_i^{\mathtt{win}} = \mathbf{F}(\mathbf{w}_i)$. Meanwhile, there is a reference database $\{\mathbf{z}_j^{\mathtt{exp}}, \mathbf{y}_j^{\mathtt{exp}}\}_{j=1}^{\mathtt{q}}$ that collects $\mathtt{q}$ pairs of exemplar embedded by the same $\mathbf{F}(\cdot)$ and the gene expression of the exemplar. For each window $\mathbf{w}_i$, we then perform exemplar retrieval from the reference database, resulting in $\{[\mathbf{z}_j^{\mathtt{exp}}, \mathbf{y}_j^{\mathtt{exp}}] \mid j \in [1, \cdots, \mathtt{q}] \wedge \xi(\mathbf{z}_i^{\mathtt{win}}, \mathbf{z}_j^{\mathtt{exp}})\}$, where $\xi(\cdot, \cdot)$ determines if $\mathbf{z}_j^{\mathtt{exp}}$ is the nearest exemplar of $\mathbf{z}_i^{\mathtt{win}}$. We then construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{V}$ is a node set of windows and exemplars, and $\mathcal{E}$ is edges that connect the nodes. We finally perform exemplar learning on the graph $\mathcal{G}$, obtaining gene expression for all window nodes, i.e., $\{\mathbf{y}_i^{\mathtt{win}}\}_{i=1}^{\mathtt{p}}$.

From the ST study [3], two main challenges exist. i) Long-range dependency: gene expression-related features are non-uniformly distributed over the slide image (refer to Figure 3 of [3] for evidence). The interactions among these features are needed to group expression of the same gene type; ii) Skewed gene expression distribution: the expression of some gene types has a skewed distribution, similar to the imbalance class distribution problem. This skewed distribution (see Fig. 3 for an example) poses challenges in predicting the expression of these gene types. In this paper, we attempt to mitigate them by learning from similar exemplars.

Model Overview. With these motivations, we have designed the EGGN framework containing three main modules in sequence. The overview of our frame-

work is shown in Fig. 4. i) Exemplar retrieval (Sec. 3.1): we have a feature extractor $\mathbf{F}(\cdot)$ to embed the slide image window $\mathbf{w}_i$ in to feature $\mathbf{z}_i^{\mathtt{win}}$, i.e., $\mathbf{z}_i^{\mathtt{win}} = \mathbf{F}(\mathbf{w}_i)$. Meanwhile, there is a reference dataset $\{\mathbf{z}_j^{\mathtt{exp}}, \mathbf{y}_j^{\mathtt{exp}}\}_{j=1}^{\mathtt{q}}$ containing $\mathtt{q}$ pairs of exemplars $\mathbf{z}_j^{\mathtt{exp}}$ embeded by the feature extractor $\mathbf{F}(\cdot)$ and the exemplar gene expression $\mathbf{y}_j^{\mathtt{exp}}$. In the feature space of $\mathbf{F}(\cdot)$, we then retrieve the nearest exemplars of $\mathbf{w}_i$ from the reference dataset to form a set $\{[\mathbf{z}_j^{\mathtt{exp}}, \mathbf{y}_j^{\mathtt{exp}}] \mid j \in [1, \cdots, \mathtt{q}] \wedge \xi(\mathbf{z}_i^{\mathtt{win}}, \mathbf{z}_j^{\mathtt{exp}})\}$, where $\xi(\cdot, \cdot)$ is the kNN algorithm used to determine if $\mathbf{z}_j^{\mathtt{exp}}$ is the nearest exemplar of $\mathbf{z}_i^{\mathtt{win}}$. ii) Graph construction (Sec. 3.2): we construct an exemplar-based graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ for the slide image. Each window or each exemplar can be considered as a node, and they form two sets of nodes, a window node set $\mathcal{V}^{\mathtt{win}}$ and an exemplar node set $\mathcal{V}^{\mathtt{exp}}$. The union of $\mathcal{V}^{\mathtt{win}}$ and $\mathcal{V}^{\mathtt{exp}}$ is $\mathcal{V}$, i.e., $\mathcal{V} = \mathcal{V}^{\mathtt{win}} \cup \mathcal{V}^{\mathtt{exp}}$. The edge set is denoted as $\mathcal{E} = \mathcal{E}_{\mathtt{win}\to\mathtt{win}} \cup \mathcal{E}_{\mathtt{exp}\to\mathtt{exp}} \cup \mathcal{E}_{\mathtt{win}\to\mathtt{exp}} \cup \mathcal{E}_{\mathtt{exp}\to\mathtt{win}}$, exploring relations of window to window $\mathcal{E}_{\mathtt{win}\to\mathtt{win}}$, exemplar to exemplar $\mathcal{E}_{\mathtt{exp}\to\mathtt{exp}}$, window to exemplar $\mathcal{E}_{\mathtt{win}\to\mathtt{exp}}$, and exemplar to window $\mathcal{E}_{\mathtt{exp}\to\mathtt{win}}$. In our formulation, $\mathcal{G}$ is a heterogeneous graph. iii) Exemplar learning (Sec. 3.3): we train a model $\mathbf{C}(\cdot, \cdot, \cdot)$ that maps $\mathcal{G}$, $\{\mathbf{z}_i^{\mathtt{win}}\}_{i=1}^{\mathtt{p}}$, and $\{(\mathbf{z}_i^{\mathtt{exp}}, \mathbf{y}^{\mathtt{exp}_\mathtt{i}})\}_{i=1}^{\mathtt{q}}$ to $\{\mathbf{y}_i^{\mathtt{win}}\}_{i=1}^{\mathtt{p}}$ by a single foward pass. Our model uses a GraphSAGE-based backbone. We bring interactions between $\mathbf{e}_i^{\mathtt{win}}$ and $\mathbf{e}_j^{\mathtt{exp}}$ to leverage $\mathbf{y}_j^{\mathtt{exp}}$ with a proposed GEB block, whenever $e_{ji} \in \mathcal{E}^{\mathtt{exp}\to\mathtt{win}}$. Meanwhile, for facilitating their interactions, the $\mathbf{e}_i^{\mathtt{win}}$ is propagated back to revise $\mathbf{e}_j^{\mathtt{exp}}$. With the introduction of exemplars, our framework dynamically benefits when predicting gene expression.

3.1. Exemplar Retrieval

To retrieve the exemplar of a given window $\mathbf{w}_i$, we have an extractor (i.e., an encoder) that is coupled with a distance metric to amount the similar-

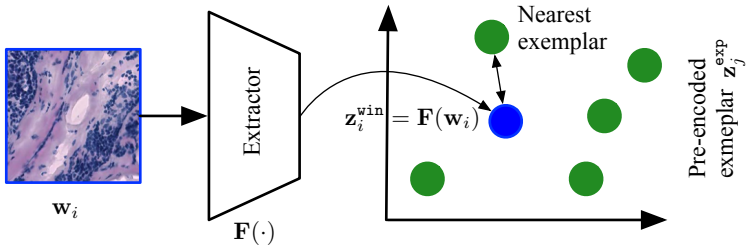


Figure 5: Overview of the exemplar retrieval. The green cycles are pre-embeded exemplar features, *e.g.*, $\mathbf{z}_j^{\texttt{exp}}$. Given a window $\mathbf{w}_i$, we extract $\mathbf{z}_i^{\texttt{win}}$ with $\mathbf{F}(\cdot)$ and retrieve the nearest exemplar.

ity between the given window and exemplar for a reference dataset for the exemplar retrieval.

Feature Extractor. Unlike our previous work [11] that trains a StyleGAN-based autoencoder [39, 40] for retrieving the exemplars, we found that an ImageNet-1K pre-trained ResNet brings a more accurate gene expression prediction in the graph-based setting. We denote $\mathbf{F}(\cdot)$ as a ResNet feature extractor, where the classification layer is removed from a standard ResNet. Though $\mathbf{F}(\cdot)$ is trained from a different domain, we show that $\mathbf{F}(\cdot)$ effectively encodes textures for gene expression prediction, by comparing it with other possible encoders in a later section.

Method. We use the extractor $\mathbf{F}(\cdot)$ for the exemplar retrieval. The window $\mathbf{w}_i$ is embedded into $\mathbf{z}_i^{\texttt{win}} = \mathbf{F}(\mathbf{w}_i)$. In the feature space of $\mathbf{F}(\cdot)$, we measure the similarity between $\mathbf{z}_i^{\texttt{win}}$ and each exemplar $\mathbf{z}_j^{\texttt{exp}}$ from the reference dataset by using the Manhattan distance $\mathcal{L}_1$. We then execute the kNN algorithm $\xi(\cdot, \cdot)$ for retrieving the nearest exemplar set. We empirically verify the optimal number of used exemplars in Sec. 4.2. To generalize the model performance, we restrict that the candidate image pairs are from different patients. The

overall process is presented in Fig. 5. In the experiment section, we compare the proposed exemplar retrieval with alternative retrieval approaches.

3.2. Graph Construction

We construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each window and exemplar is treated as a node, forming a window node set $\mathcal{V}^{\texttt{win}} = \{v_i^{\texttt{win}}\}_{i=1}^{\texttt{p}}$ and an exemplar node set $\mathcal{V}^{\texttt{exp}} = \{v_j^{\texttt{exp}}\}_{j=1}^{\texttt{q}}$. We take an union of the two set, $\mathcal{V} = \mathcal{V}^{\texttt{win}} \cup \mathcal{V}^{\texttt{exp}}$, to obtain a node set of our graph $\mathcal{G}$. As described, we consider four edge types. They are

$$\mathcal{E}_{\texttt{win}\to\texttt{win}} = \{e_{ij} \mid v_i^{\texttt{win}}, v_j^{\texttt{win}} \in \mathcal{V}^{\texttt{win}} \times \mathcal{V}^{\texttt{win}} \wedge \phi(v_i^{\texttt{win}}, v_j^{\texttt{win}})\} \ , \tag{1}$$

$$\mathcal{E}_{\texttt{exp}\to\texttt{exp}} = \{e_{ij} \mid v_i^{\texttt{exp}}, v_j^{\texttt{exp}} \in \mathcal{V}^{\texttt{exp}} \times \mathcal{V}^{\texttt{exp}} \wedge \xi(v_i^{\texttt{exp}}, v_j^{\texttt{exp}})\} \ , \tag{2}$$

$$\mathcal{E}_{\texttt{win}\to\texttt{exp}} = \{e_{ij} \mid v_i^{\texttt{win}}, v_j^{\texttt{exp}} \in \mathcal{V}^{\texttt{win}} \times \mathcal{V}^{\texttt{exp}} \wedge \xi(v_i^{\texttt{win}}, v_j^{\texttt{exp}})\} \ , \tag{3}$$

$$\mathcal{E}_{\texttt{exp}\to\texttt{win}} = \{e_{ij} \mid e_{ji} \in \mathcal{E}_{\texttt{win}\to\texttt{exp}}\} \ . \tag{4}$$

$\phi(\cdot,\cdot)$ explores the spatial relation of windows, determining if the spatial distance of two input windows in the slide is below a given threshold. Note that $\xi(\cdot,\cdot)$ is a kNN-based function in the feature space of $\mathcal{F}(\cdot)$ for connecting k-nearest neighbors. Thus, we have the edge set of the graph $\mathcal{G}$ as $\mathcal{E} = \mathcal{E}_{\texttt{win}\to\texttt{win}} \cup \mathcal{E}_{\texttt{exp}\to\texttt{exp}} \cup \mathcal{E}_{\texttt{win}\to\texttt{exp}} \cup \mathcal{E}_{\texttt{exp}\to\texttt{win}}$.

3.3. Exemplar Learning

Our model $\mathbf{C}(\cdot,\cdot,\cdot)$ is composed of a projector, a L-layer GraphSAGE-based backbone, GEB blocks, and a prediction block, where we use the GEB block in each of the backbone layers. With graph $\mathcal{G}$, our model maps node features ($\{\mathbf{z}_i^{\texttt{win}}\}_{i=1}^{\texttt{p}}$ of a slide image and exemplars $\{\mathbf{z}_j^{\texttt{exp}}, \mathbf{y}_j^{\texttt{exp}}\}_{j=1}^{\texttt{q}}$) to gene expression $\{\mathbf{y}_i^{\texttt{win}}\}_{i=1}^{\texttt{p}}$ of each window. Our model allows interactions between

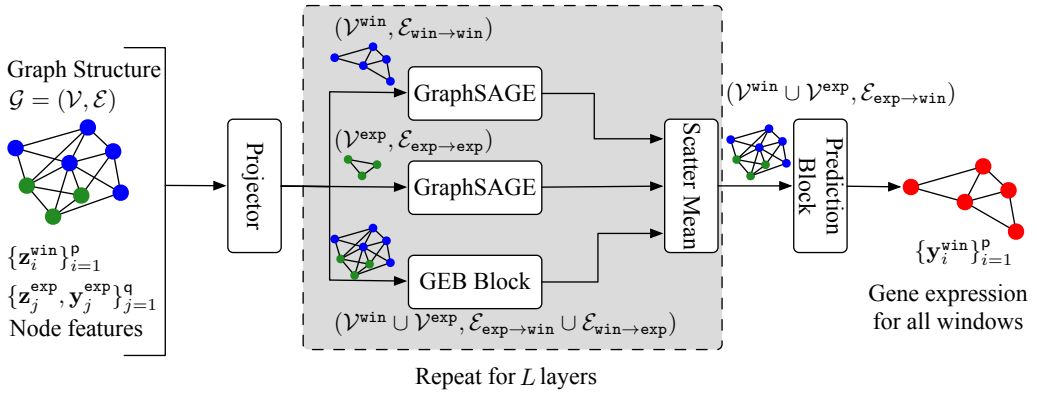


Figure 6: Architectures of our model. We have a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the node set $\mathcal{V} = \mathcal{V}^{\mathtt{win}} \cup \mathcal{V}^{\mathtt{exp}}$ is a union of windows and exemplars, and the edge set $\mathcal{E} = \mathcal{E}_{\mathtt{win}\to\mathtt{win}} \cup \mathcal{E}_{\mathtt{exp}\to\mathtt{exp}} \cup \mathcal{E}_{\mathtt{win}\to\mathtt{exp}} \cup \mathcal{E}_{\mathtt{exp}\to\mathtt{win}}$ contains four types of relations among nodes. We first refine features (i.e., $\mathcal{G}$, $\{\mathbf{z}_i^{\mathtt{win}}\}_{i=1}^{\mathtt{p}}$ and $\{\mathbf{z}_i^{\mathtt{exp}}\}_{i=1}^{\mathtt{q}}$) of each node by using a projector. We then respectively allow interactions among $\mathcal{V}^{\mathtt{win}}$ and $\mathcal{V}^{\mathtt{exp}}$ by two GraphSAGE block [9]. Meanwhile, we have a GEB block that revises the features of nodes contained in $\mathcal{V}^{\mathtt{win}}$ and $\mathcal{V}^{\mathtt{exp}}$ by considering edges between the two node sets. The scatter mean operation [41] average the features computed by the two GraphSAGE blocks and the GEB block for each node. Finally, after repeating the above operations for L layers, there is a prediction block that weights the contribution of each exemplar to a window for the gene expression prediction task. Finally, we have gene expression of all window nodes as $\{\mathbf{y}_i^{\mathtt{win}}\}_{i=1}^{\mathtt{p}}$. Note that the blue window nodes turn red, carrying gene expression prediction at each of the window nodes.

.

$\{\mathbf{z}_i^{\mathtt{win}}\}_{i=1}^{\mathtt{p}}$ and $\{\mathbf{z}_j^{\mathtt{exp}}, \mathbf{y}_j^{\mathtt{exp}}\}_{i=1}^{\mathtt{q}}$ to progressively revise their intermediate features within the backbone for the prediction task. The overall architecture is shown in Fig. 6.

Projector. The features, $\mathbf{z}_i^{\mathtt{win}}$ and $\mathbf{z}_j^{\mathtt{exp}}$, are embeded with a wide range of dataset-dependent attributes. We refine them to concentrate on the gene expression of interest by several multi-layer perceptrons (MLPs). Firstly, each window $\mathbf{z}_i^{\mathtt{win}}$ is projected by $\mathrm{MLP}_h^0(\cdot)$. Secondly, for each $\mathbf{z}_j^{\mathtt{exp}}$, as its associated gene expression $\mathbf{y}_j^{\mathtt{exp}}$, is available, we empower the refinement of $\mathbf{z}_j^{\mathtt{exp}}$ by $\mathbf{y}_j^{\mathtt{exp}}$. We concatenate $\mathbf{z}_j^{\mathtt{exp}}$ and $\mathbf{y}_j^{\mathtt{exp}}$ before feeding to $\mathrm{MLP}_r^0(\cdot)$. We

have

$$\mathbf{h}_i^0 = \mathrm{MLP}_h^0\left(\mathbf{z}_i^{\mathtt{win}}\right) \ , \qquad \forall v_i \in \mathcal{V}^{\mathtt{win}} \ ,$$
$$\mathbf{r}_j^0 = \mathrm{MLP}_r^0\left([\mathbf{z}_j^{\mathtt{exp}} \parallel \mathbf{y}_j^{\mathtt{exp}}]\right) \ , \qquad \forall v_j \in \mathcal{V}^{\mathtt{exp}} \ ,$$

where the superscripts of $\mathbf{h}_i^0$ and $\mathbf{r}_j^0$ denote that they are initial refined feature, and $\cdot \parallel \cdot$ is a concatenation operator. $\mathrm{MLP}_h^0(\cdot)$ and $\mathrm{MLP}_r^0(\cdot)$ are three-layer perceptrons with LeakyReLU activation functions. The bottom two layers of $\mathrm{MLP}_h^0(\cdot)$ and $\mathrm{MLP}_r^0(\cdot)$ share the same parameter, restricting them in the same feature space.

Backbone. We use a sequence of GraphSAGE layers as our backbone [9], to allow interactions respectively among windows and exemplars. These interactions evolve each window feature under their neighborhood structural information and smooth each exemplar with neighbors to provide more accurate features that are used in the following layers. Assuming there is L layer and $l \in [1, \cdots, L]$. At l^{th} layer, we mathematically define

$$\mathbf{h}_i^{l+1} = \mathbf{W}_h^l \left[\mathbf{h}_i^l \,\middle\|\, \frac{1}{|\mathcal{N}_i^{\mathcal{E}_{\mathtt{win}\to\mathtt{win}}}|} \sum_{j \in \mathcal{N}_i^{\mathcal{E}_{\mathtt{win}\to\mathtt{win}}}} \mathbf{h}_j^l\right], \qquad \forall v_i \in \mathcal{V}^{\mathtt{win}} \tag{5}$$

$$\mathbf{r}_j^{l+1} = \mathbf{W}_r^l \left[\mathbf{r}_j^l \,\middle\|\, \frac{1}{|\mathcal{N}_j^{\mathcal{E}_{\mathtt{exp}\to\mathtt{exp}}}|} \sum_{i \in \mathcal{N}_j^{\mathcal{E}_{\mathtt{exp}\to\mathtt{exp}}}} \mathbf{r}_i^l\right], \qquad \forall v_j \in \mathcal{V}^{\mathtt{exp}} \tag{6}$$

where $\mathbf{W}_h^l$ and $\mathbf{W}_r^l$ are linear weight matrices, and $\mathcal{N}_i^{\mathcal{E}_{\mathtt{win}\to\mathtt{win}}} = \{j \mid e_{jk} \in \mathcal{E}_{\mathtt{win}\to\mathtt{win}} \wedge k = i\}$ and $\mathcal{N}_j^{\mathcal{E}_{\mathtt{exp}\to\mathtt{exp}}} = \{i \mid e_{ik} \in \mathcal{E}_{\mathtt{exp}\to\mathtt{exp}} \wedge k = j\}$ contain the index of neighborhood nodes under the given edge set.

GEB Block. This block is concurrent with the GraphSAGE backbone, bringing knowledge about gene expression $\mathbf{y}_j^{\mathtt{exp}}$ of the exemplar to the window

feature $\mathbf{h}_i^l$. For brevity, we do not differentiate between the outputs of the GraphSAGE layer and the GEB block. At l^{th} layer, we project $\mathbf{y}_j^{\texttt{exp}}$ to $\mathbf{s}_j^l$, and have interactions between $\mathbf{h}_i^l$ and $\mathbf{r}_j^l$ to obtain the revised window and exemplar features, i.e., $\mathbf{h}_i^{l+1}$ and $\mathbf{r}_j^{l+1}$.

In detail, the difference between $\mathbf{h}_i^l - \mathbf{r}_j^l$ is passed to $\text{MLP}_m^t(\cdot)$, querying the potential knowledge of their gene expression difference. We chunk the outputs to $\mathbf{m}_{h,j,i}$ and $\mathbf{m}_{r,i,j}$. Then, they are used to retrieve gene expression from $\mathbf{s}_j^l$ (i.e., a projection of $\mathbf{y}_j^{\texttt{exp}}$), and adjust feature significance of $\mathbf{h}_i^l$ and $\mathbf{r}_j^l$. Semantically, $\mathbf{m}_{h,j,i}$ summarises 'the existing knowledge of gene expression', and $\mathbf{m}_{r,j,i}$ tells 'the desired gene expression knowledge'. Mathematically, we have

$$\mathbf{h}_i^{l+1} = \mathbf{h}_i^t + \frac{1}{|\mathcal{N}_i^{\mathcal{E}_{\texttt{exp}\to\texttt{win}}}|} \sum_{j \in \mathcal{N}_i^{\mathcal{E}_{\texttt{exp}\to\texttt{win}}}} \text{MLP}_h^t\left([\mathbf{h}_i^l \parallel \mathbf{r}_j^l \parallel \mathbf{s}_j^l] \odot \mathbf{m}_{h,j,i}\right) \ , \quad \forall v_i \in \mathcal{V}^{\texttt{win}} \ , \tag{7}$$

$$\mathbf{r}_j^{l+1} = \mathbf{r}_j^l + \frac{1}{|\mathcal{N}_j^{\mathcal{E}_{\texttt{win}\to\texttt{exp}}}|} \sum_{j \in \mathcal{N}_j^{\mathcal{E}_{\texttt{win}\to\texttt{exp}}}} \text{MLP}_r^t\left([\mathbf{h}_i^l \parallel \mathbf{r}_j^l \parallel \mathbf{s}_j^l] \odot \mathbf{m}_{h,i,j}\right) \ , \quad \forall v_j \in \mathcal{V}^{\texttt{exp}} \ , \tag{8}$$

$$\mathbf{m}_{h,j,i}, \mathbf{m}_{r,i,j} = \text{Chunk}\left(\sigma\left(\text{MLP}_m^l\left(\mathbf{h}_i^l - \mathbf{r}_j^l\right)\right)\right) \ , \qquad \forall e_{ij} \in \mathcal{E}_{\texttt{win}\to\texttt{exp}} \ , \tag{9}$$

$$\mathbf{s}_j^l = \text{MLP}_s^l\left(\mathbf{y}_j^{\texttt{exp}}\right) \ , \qquad \forall v_j \in \mathcal{V}^{\texttt{exp}} \ , \tag{10}$$

where $\mathcal{N}_i^{\mathcal{E}_{\texttt{exp}\to\texttt{win}}} \coloneqq \{j \mid e_{jk} \in \mathcal{E}_{\texttt{exp}\to\texttt{win}} \wedge k = i\}$, $\mathcal{N}_j^{\mathcal{E}_{\texttt{win}\to\texttt{exp}}} \coloneqq \{i \mid e_{ik} \in \mathcal{E}_{\texttt{win}\to\texttt{exp}} \wedge k = j\}$, $\text{MLP}_h^t(\cdot)$, $\text{MLP}_r^t(\cdot)$, and $\text{MLP}_m^l(\cdot)$ are multi-layer perceptrons, the $\odot$ is element-wise multiplication, $\text{MLP}_s^l(\cdot)$ is a single layer perceptron, the $\text{Chunk}(\cdot)$ operator equally splits the input into two outputs,

and $\sigma(\cdot)$ is a Sigmoid function. By scaling the magnitudes of the features in Eq. 7, we directly inject the knowledge about the gene expression of the exemplars into the window feature, while we also update the exemplar feature to facilitate both interactions with the exemplars and the window feature in following layers. We finally apply a scatter mean operation to aggregate the updated features from the GEB block and the GraphSAGE layers.

Prediction Block. We extend the attention pooling from [42] to consider the contribution of exemplars to a window toward the gene expression prediction, while also encouraging the model to refine exemplars feature in a more window-dependent way. We measure the importance $\hat{\mathbf{a}}_{i,j}$ of each exemplar by using the difference between a window $\mathbf{h}_i^L$ and a linearly projected exemplar $\mathbf{r}_j^L$. Then, the importance score of exemplars for a given window is normalized in the same vein with [43]. The normalized importance score $\mathbf{a}_{i,j}$ is used to weigh the exemplar and aggregate it into the window feature before making the gene expression prediction. Overall, our prediction block is defined as

$$\hat{\mathbf{a}}_{i,j} = \mathbf{h}_i^L - \mathbf{W}_a \mathbf{r}_j^L \ , \qquad \forall e_{ij} \in \mathcal{E}_{\texttt{win}\rightarrow\texttt{exp}} \ , \tag{11}$$

$$\mathbf{a}_{i,j} = \frac{\exp \hat{\mathbf{a}}_{i,j}}{\sum_{k \in \mathcal{N}_i^{\mathcal{E}_{\texttt{exp}\rightarrow\texttt{win}}}} \exp \hat{\mathbf{a}}_{i,k}} \ , \qquad \forall e_{ij} \in \mathcal{E}_{\texttt{win}\rightarrow\texttt{exp}} \ , \tag{12}$$

$$\mathbf{y}_i = \mathbf{W}_{\mathbf{y}} \left(\mathbf{h}_i^L + \sum_{j \in \mathcal{N}_i^{\mathcal{E}_{\texttt{exp}\rightarrow\texttt{win}}}} \mathbf{a}_{i,j} \odot \mathbf{r}_j^L \right) \ , \qquad \forall v_i \in \mathcal{V}^{\texttt{win}} \tag{13}$$

where $\mathbf{W}_a$ and $\mathbf{W}_{\mathbf{y}}$ are linear weight matrix. Note that Eq. 12 performs computations element-wisely.

Objective. Our model is optimized with mean squared loss (i.e., the Euclidean distance) $\mathcal{L}_2$ and batch-wise PCC $\mathcal{L}_{\text{pcc}}$. We have

$$\mathcal{L}_{\text{total}} = \mathcal{L}_2 + \mathcal{L}_{\text{pcc}}$$

4. Experiments

Datasets. We perform experiments in the publicly available STNet dataset [3] and 10xProteomic datasets[1]. The STNet dataset contains roughly 30,612 pairs of the slide image window and the gene expression. This dataset covers 68 slide images from 23 patients. Following [3], we target predicting expression of 250 gene types that have the largest mean across the dataset. The 10xProteomic dataset has 24,263 slide image windows and gene expression pairs from 6 slide images. We select target gene types in the same way as the STNet dataset. We apply log transformation and min-max normalization to the target gene expression. Our normalization method is different from [3] (they use log transformation and $\mathcal{L}_1$ normalization, i.e., log-transforming the division of the expression of each gene type by the sum of expression of all gene types, for each slide image window). Our normalization method allows independent analysis of expression prediction of each gene type.

4.1. Experimental Set-up

Baseline Methods. We compare with extensive SOTA methods in domains of gene expression prediction, ImageNet classification benchmarks, exemplar learning, and GCNs.

[1] `https://www.10xgenomics.com/resources/datasets`

- STNet [3], NSL [4], and EGN [11]. They are the SOTA methods in gene expression prediction.
- ViT [43], MPViT [44] and CycleMLP [45]. We use the SOTA ImageNet classification methods in our task. They are strong baselines in our task. Specifically, we use ViT-B, MPViT-Base, and CycleMLP-B2. Please refer to [43, 44, 45] for details.
- Retro [22] and ViTExp. We explore the SOTA exemplar learning methods, where the input window is represented in image $\mathbf{w}_i$. However, Retro is originally developed for natural language processing. We adapt it by providing the feature extractor output as the exemplar features. ViTExp directly concatenates the exemplar features to the ViT patch representation. The exemplar features are added with an embedding to be differentiated from the patch representation of the slide image window. Both Retro and ViTExp are based on the ViT-B architecture [43].
- GraphSAGE [9], GATv2Conv [38], and TransformerConv [35]. In the same vein as the last categories, we adapt the SOTA GCNs for exemplar learning, by allowing communications from windows to windows, exemplars to exemplars, windows to exemplars, and exemplars to windows.

Evaluation Metrics. We evaluate the proposed methods and alternative baselines with PCC, mean squared error (MSE), and mean absolute error (MAE). We use PCC@F, PCC@S, and PCC@M to denote the first quantile, median, and mean of the PCC. The PCC@F verifies the least performed model predictions. The PCC@S and PCC@M measure the median and mean of

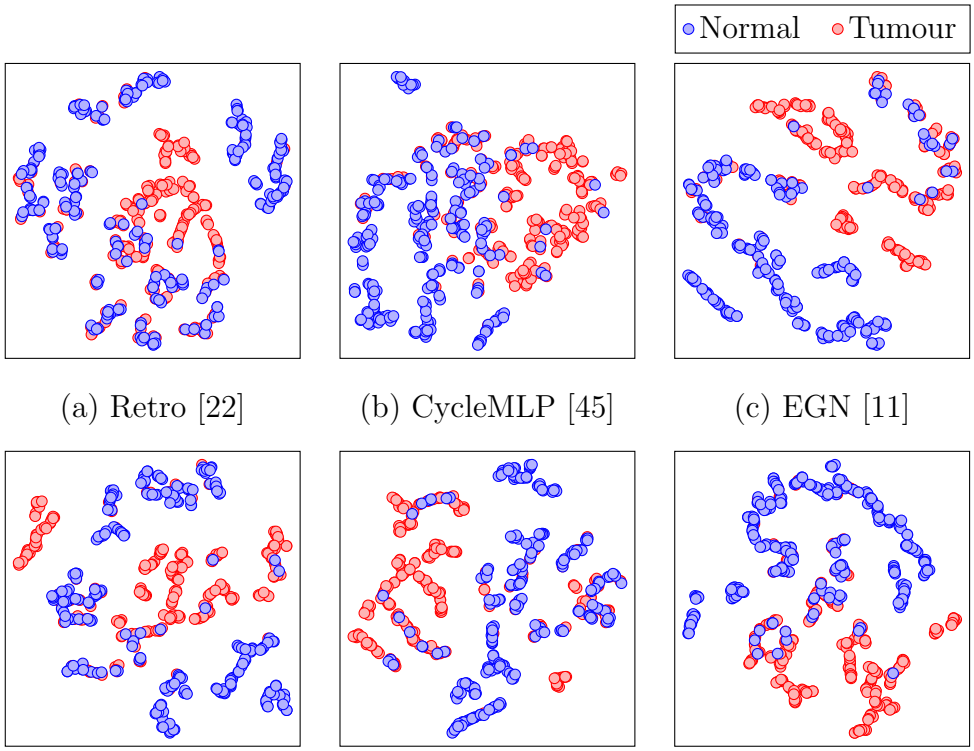
correlations for each gene type, given predictions and ground truth for all of the slide image windows. Meanwhile, the MSE and MAE measure the sample-wise deviation between predictions and ground truth of each slide image window for each gene type. For PCC@F, PCC@S, and PCC@M, the higher value indicates better performance. In contrast, for MSE and MAE, the lower value means better performance.

Implementation Details. We use the ResNet18 [10] that is officially pre-trained on the ImageNet-1K dataset as our feature extractor. We implement EGGN by using the *Pytorch* [46] and *PyTorch Geometric* [47] frameworks. EGGN is respectively trained from scratch for 50 epochs and 300 epochs on the STNet dataset and 10xProteomic dataset. We use batch size 1. All windows of a slide image composite a batch with size 1 in our case. We set the learning rate to 5×10^{-4}. Our weight decay is 1×10^{-4}. We use a GraphSAGE backbone with hidden dimensions 512 and layers 4. All the experiments are conducted with NVIDIA Tesla P100 GPUs.

Table 1: Quantitative gene expression prediction comparisons with SOTA methods on STNet dataset and 10xProteomic dataset. We bold the best results. We use '-' to denote unavailable results. Models are evaluated by four-fold cross-validation and three-fold cross-validation on the above datasets. Our proposed EGGN framework consistently outperforms the SOTA methods in $\text{PCC@F}_{\times 10^1}$, $\text{PCC@S}_{\times 10^1}$ and $\text{PCC@M}_{\times 10^1}$ for both datasets. GraphSAGE finds the best $\text{MSE}_{\times 10^2}$ and $\text{MAE}_{\times 10^1}$ on the STNet dataset.

Method	STNet Dataset					10xProteomic Dataset				
	$\text{MSE}_{\times 10^2}$	$\text{MAE}_{\times 10^1}$	$\text{PCC@F}_{\times 10^1}$	$\text{PCC@S}_{\times 10^1}$	$\text{PCC@M}_{\times 10^1}$	$\text{MSE}_{\times 10^2}$	$\text{MAE}_{\times 10^1}$	$\text{PCC@F}_{\times 10^1}$	$\text{PCC@S}_{\times 10^1}$	$\text{PCC@M}_{\times 10^1}$
STNet [3]	4.52	1.70	0.05	0.92	0.93	12.40	2.64	1.25	2.26	2.15
NSL [4]	-	-	-0.71	0.25	0.11	-	-	-3.73	1.84	0.25
ViT [43]	4.28	1.67	0.97	1.86	1.82	7.54	2.27	4.64	5.11	4.90
CycleMLP [45]	4.41	1.68	1.11	1.95	1.91	4.69	1.55	5.88	6.60	6.32
MPViT [44]	4.49	1.70	0.91	1.54	1.69	5.45	1.56	6.40	7.15	6.84
Retro [22]	4.53	1.71	0.99	1.74	1.79	5.25	1.65	5.46	6.35	6.04
ViTExp	4.46	1.69	0.87	1.72	1.74	5.04	1.66	5.59	6.36	6.00
EGN [11]	4.10	1.61	1.51	2.25	2.02	5.49	1.55	6.78	7.21	7.07
GraphSAGE [9]	**3.79**	**1.57**	1.98	2.90	2.83	4.98	1.65	6.82	7.42	7.25
GATv2Conv [38]	4.03	1.62	2.00	2.85	2.77	4.35	1.49	6.79	7.25	7.11
TransformerConv [35]	4.14	1.64	1.97	2.78	2.69	4.71	1.53	6.81	7.39	7.27
Ours	3.94	1.61	**2.12**	**3.05**	**2.92**	**3.52**	**1.31**	**7.06**	**7.60**	**7.44**

Quantitative Evaluation. We compare our EGGN framework with the baselines on the STNet dataset and the 10xProteomic dataset (Tab. 1). As the gene expression prediction task emphasizes capturing the relativity variation, we bias on the PCC-related evaluation metrics, i.e., PCC@F, PCC@S, and PCC@M. Our EGGN consistently achieves the best performance in PCC@F, PCC@S, and PCC@M. Our findings are as follows: i) it's worth noting that GCN-based approaches achieve higher PCC-related evaluation metrics than the approaches that predict gene expression individually from each window, *e.g.*, EGN, the past SOTA method. GCN-based approaches are capable of modeling spatial relations among windows, capturing relative changes among window features which improve the PCC-related evaluation metrics; ii) GraphSAGE, GATv2Conv, and TransformerConv, the SOTA GCN methods, lead to the second-best performance in the STNet dataset and 10xProteomic dataset in PCC-related evaluations. Meanwhile, GraphSAGE finds the best MSE and MAE on the STNet dataset. These GCN models all outperform their counterparts, the SOTA methods in the ImageNet-1K classification task (*e.g.*, CycleMLP and MPViT), evidencing our claims that reasoning spatial relations among windows out-weight feature learning in our gene expression prediction problem (Sec. 1); iii) the PCC@F of our model significantly outperforms the baseline methods. This metric evaluates the worst model capability by calculating the first quantile of PCC across all gene types. The majority of gene types covered by the first quantile have skewed expression distributions, which is the most challenging part of the prediction task. Our method has 0.012 - 0.024 higher than the second-best performance from other methods in PCC@F; iv) STNet and

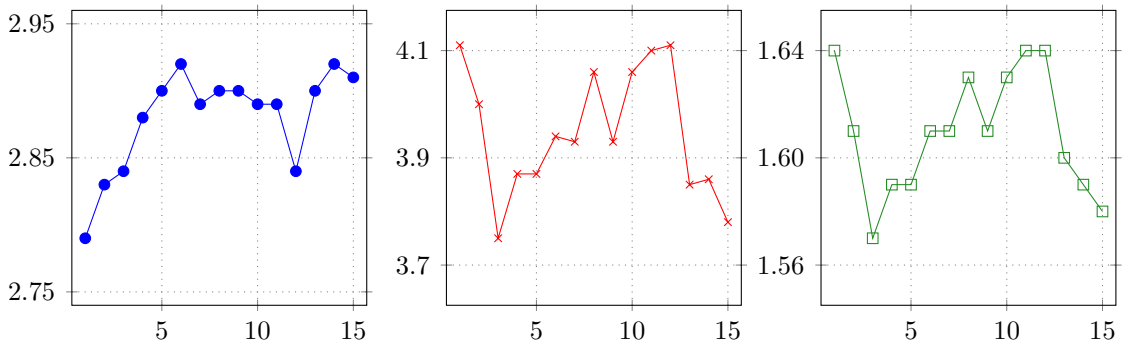


(a) Retro [22] (b) CycleMLP [45] (c) EGN [11]

(d) GraphSAGE [9] (e) GATV2Conv [38] (f) Ours

Figure 7: Quantitative evaluation of the top performed models from Tab. 1. We employ t-SNE [48] for feature dimension reduction. We use the extra labels (i.e., tumour and normal) from the STNet dataset for annotations.

NSL fail to achieve good performance. Again, the gene expression-related features are usually non-uniformly distributed across the slide image. They predict the gene expression of each window independently, failing to capture feature dependency among the slide image. Moreover, NSL shows a negative correlation with PCC@F. This validates our claims that predicting gene expression directly from the color intensity is vulnerable, and it is only feasible in extreme cases, *e.g.*, the example of tumor-related gene expression in Sec. 1.



(a) PCC@M$_{\times 10^1}$ (b) MSE$_{\times 10^2}$ (c) MAE$_{\times 10^1}$

Figure 8: Ablation study on the number of exemplars used in our model. The number is varied from 1 to 15, and we respectively present PCC@M, MSE, and MAE in (a), (b), and (c).

Quantitative Evaluation. We present the latent space visualization (Fig. 7), by considering the top performed models from Tab. 1. Fig. 7 (a, b, c) are models that predict gene expression of each window individually, and Fig. 7 (d, e, f) are best performed GCN-based methods. To enable a clean visualization, we randomly sample 256 slide image window features for each label, i.e., tumour and normal. Our method sufficiently separates the tumour features from the normal features.

4.2. Ablation study

We study the capability of each model component by conducting a detailed ablation study on the STNet dataset.

Number of Exemplars. We present PCC@M (Fig. 8 (a)), MSE (Fig. 8 (b)), and MAE (Fig. 8 (c)), by varying the number of exemplars used in our model from 1 to 15. Having 6 exemplars finds the best PCC@M, and we have the best MSE and MAE by using 3 exemplars. Again, our task emphasizes

Table 2: Ablation study on exemplar retrieval. We compare features from StyleEncoder, AlexNet and ResNet18 for exemplar retrieval. We use the ResNet18 with $\mathcal{L}_1$ for exemplar retrieval in our EGGN framework.

Feature Space	Distance Metrics	$\text{MSE}_{\times 10^2}$	$\text{MAE}_{\times 10^1}$	$\text{PCC@M}_{\times 10^1}$
StyleEncoder	*cosine*	3.87	1.59	2.61
StyleEncoder	$\mathcal{L}_2$	4.05	1.62	2.60
StyleEncoder	$\mathcal{L}_1$	4.02	1.62	2.62
AlexNet	LPIPS	4.21	1.65	2.64
ResNet18	*cosine*	4.00	1.62	2.88
ResNet18	$\mathcal{L}_2$	4.03	1.62	2.86
ResNet18	$\mathcal{L}_1$	3.94	1.61	2.92

capturing relative gene expression changes. Thus, our final recommendation is to use 6 exemplars.

Exemplar Retrieval. We explore alternative approaches for retrieving exemplars (Tab. 2). We study exemplar retrieval strategy by considering StyleEncoder [11], AlexNet [49], and ResNet18 [10]. Among the three backbone networks, StyleEncoder is learned in an unsupervised manner with slide images, while AlexNet and ResNet18 are trained with supervision for ImageNet-1K (i.e., natural RGB images) classification task. We explore diverse distance matrices including LPIPS [50], $\mathcal{L}_2$, $\mathcal{L}_1$, and *cosine* (i.e., cosine similarity). We have the following findings: i) LPIPS distance retrieved exemplars lead to bad model performance. The distance is trained based on the human perception of natural RGB images. These natural RGB images are different from the slide images, leading to bad exemplar retrieves, and damaging the model performance; ii) the StyleEncoder is optimized for reconstructing slide images. However, as shown in Fig. 3, gene expression-related knowledge is not

Table 3: Ablation study on model architectures.

Settings	$MSE_{\times 10^2}$	$MAE_{\times 10^1}$	$PCC@M_{\times 10^1}$
Backbone only	4.22	1.66	2.76
W/o GEB	4.18	1.65	2.85
W/o Projector	4.10	1.63	2.84
Ours	3.94	1.61	2.92

balanced across the dataset, imposing biased knowledge to the StyleEncoder. When performing interactions among windows and exemplars in our model, such biases are potentially amplified and then decrease the model performance; iii) using ResNet18 with $\mathcal{L}_1$ distance achieves the best performance, though the pre-trained ResNet18 lacks gene expression-related knowledge. This retrieval method is an expert at encoding knowledge of image textures, benefiting from the large-scale training dataset, ImageNet-1K for our task. This allows modeling the relation among windows and exemplars in a more accurate way.

Model Architectures. We study the performance of three baseline settings, 'Backbone only', 'w/o EB block', and 'w/o projector', in Tab. 3. The 'Backbone only' setting uses the GraphSAGE backbone architecture, considering the windows from the graph only. The 'w/o GEB block' setting removes the GEB block, and replaces our prediction block with a single linear layer for gene expression prediction. The 'w/o projector' setting replaces the projector with a linear layer to unify the dimension. Our findings are as follows: i) the 'Backbone only' setting achieves the worst performance because of the absence of extra knowledge from the exemplar; ii) the 'W/o GEB' block has the second-worst performance because of a similar reason. However, it has a

Table 4: Ablation study on graph construction. The performance of our method in the 'w/o exemplars' setting is omitted, as our method is designed for exemplar learning only.

Settings	Method	$MSE_{\times 10^2}$	$MAE_{\times 10^1}$	$PCC@M_{\times 10^1}$
w/o exemplars	GraphSAGE	4.22	1.66	2.76
	GATv2Conv	4.34	1.68	2.75
	TransformerConv	4.45	1.70	2.62
w/ exemplars	GraphSAGE	3.79	1.57	2.83
	GATv2Conv	4.03	1.62	2.77
	TransformerConv	4.14	1.64	2.69
	Ours	3.94	1.61	2.92

projector to refine the ResNet18 feature to gene expression-related features; iii) with all proposed components, we have the best performance.

Graph Construction. We study if constructing the exemplar nodes in our graph can generally benefit other baseline GCN frameworks, i.e., GraphSAGE, GATv2Conv, and TransformerConv (Tab. 4). As shown, the performance of these models has consistently been improved, by leveraging extra knowledge from the exemplars. With our GEB blocks, we have the best performance.

5. Limitation and Future Work

Limitation. Our method is capable to predict the gene expression of multiple windows in a single forward pass. However, computation costs have to be wasted, when a user is only interested in the gene expression of one window in a slide image. With a single window, our model is equivalent to sequential stacking linear layers, lacking interactions among windows and exemplars, and decreasing the model performance. Thus, dummy windows from the

slide image are needed to be sampled for forming a graph. The sampling and prediction processes consume extra computation resources.

Future Work. Though exemplars could be used as anchors for allowing information propagation for windows distributed in a slide image, a more explicit solution for facilitating information propagation needs to be explored. In our future work, we will study downsampling and upsampling operations for our exemplar and window-based graphs to build an encoder and decoder-based architecture to allow broader information propagation in the slide image.

6. Conclusion and Broader Impact

This paper proposes an EGGN framework to accurately predict gene expression from each fine-grained area of tissue slide image, i.e., different windows. EGGN uses the GraphSAGE as a backbone while integrating with exemplar learning concepts. We first have an extractor to retrieve the exemplars of the given tissue slide image window. Then, we construct a graph to connect windows within the same slide image and their corresponding exemplars, and propose a GEB block to progressively revise the intermediate GraphSAGE feature by reciprocating with the nearest exemplars. With extensive experiments, we demonstrate the superiority of the EGGN framework over the SOTA methods. EGGN is promising to facilitate studies on diseases and novel treatments with accurate gene expression prediction.

Acknowledgement. The authors would like to thank Machine Learning & Artificial Intelligence Future Science Platforms, CSIRO for computation resource funding.

References

[1] V. Marx, Method of the year: spatially resolved transcriptomics, Nature Methods 18 (2021) 9–14. doi:10.1038/s41592-020-01033-y.

[2] M. Asp, J. Bergenstrhle, J. Lundeberg, Spatially resolved transcriptomesnext generation tools for tissue exploration, BioEssays 42 (2020) 1900221. doi:10.1002/bies.201900221.

[3] B. He, L. Bergenstrhle, L. Stenbeck, A. Abid, A. Andersson, A. Borg, J. Maaskola, J. Lundeberg, J. Zou, Integrating spatial gene expression and breast tumour morphology via deep learning, Nature Biomedical Engineering 4 (2020) 1–8. doi:10.1038/s41551-020-0578-x.

[4] M. Dawood, K. Branson, N. Rajpoot, F. u. A. A. Minhas, All you need is color: Image based spatial gene expression prediction using neural stain learning (08 2021).

[5] G. Huang, Z. Liu, L. van der Maaten, K. Weinberger, Densely connected convolutional networks, 2017. doi:10.1109/CVPR.2017.243.

[6] S. Yang, H. Su, W. H. Hsu, W. Chen, Class-agnostic few-shot object counting, in: IEEE Winter Conference on Applications of Computer Vision, WACV 2021, Waikoloa, HI, USA, January 3-8, 2021, IEEE, 2021, pp. 869–877. doi:10.1109/WACV48630.2021.00091.
URL https://doi.org/10.1109/WACV48630.2021.00091

[7] H. Lin, Z. Ma, R. Ji, Y. Wang, X. Hong, Boosting crowd counting via multifaceted attention, CoRR abs/2203.02636 (2022). arXiv:2203.

02636, doi:10.48550/arXiv.2203.02636.
URL https://doi.org/10.48550/arXiv.2203.02636

[8] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, CoRR abs/1609.02907 (2016). arXiv:1609.02907.
URL http://arxiv.org/abs/1609.02907

[9] W. L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 1024–1034.
URL https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html

[10] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.

[11] Y. Yang, M. Hossain, E. Stone, S. Rahman, Exemplar guided deep neural network for spatial transcriptomics analysis of gene expression prediction (10 2022).

[12] . Avsec, V. Agarwal, D. Visentin, J. Ledsam, A. Grabska-Barwinska, K. Taylor, Y. Assael, J. Jumper, P. Kohli, D. Kelley, Effective gene expression prediction from sequence by integrating long-range interactions, Nature Methods 18 (2021) 1196–1203. doi:10.1038/s41592-021-01252-x.

[13] B. Schmauch, A. Romagnoni, E. Pronier, C. Saillard, P. Maill, J. Calderaro, A. Kamoun, M. Sefta, S. Toldo, M. Zaslavskiy, T. Clozel, M. Moarii, P. Courtiol, G. Wainrib, A deep learning model to predict rna-seq expression of tumours from whole slide images, Nature Communications 11 (08 2020). doi:10.1038/s41467-020-17678-4.

[14] X. Li, C.-Y. Wang, From bulk, single-cell to spatial rna sequencing, International Journal of Oral Science 13 (12 2021). doi:10.1038/s41368-021-00146-0.

[15] M. Gerlinger, A. Rowan, S. Horswell, J. Larkin, D. Endesfelder, E. Gronroos, P. Martinez, N. Matthews, A. Stewart, P. Tarpey, I. Varela, B. Phillimore, S. Begum, N. Mcdonald, A. Butler, D. Jones, K. Raine, C. Latimer, C. Santos, C. Swanton, Intratumor heterogeneity and branched evolution revealed by multiregion sequencing, The New England journal of medicine 366 (2012) 883–92. doi:10.1056/NEJMoa1113205.

[16] V. Marx, Method of the year: spatially resolved transcriptomics, Nature Methods 18 (2021) 9–14. doi:10.1038/s41592-020-01033-y.

[17] M. Bautista, A. Sanakoyeu, E. Sutter, B. Ommer, Cliquecnn: Deep unsupervised exemplar learning (08 2016).

[18] B. Patro, V. Namboodiri, Deep exemplar networks for vqa and vqg (12 2019).

[19] Z. Chen, J. Chen, Y. Geng, J. Pan, Z. Yuan, H. Chen, Zero-Shot Visual

Question Answering Using Knowledge Graph, 2021, pp. 146–162. doi:10.1007/978-3-030-88361-4_9.

[20] D. Teney, A. Hengel, Zero-shot visual question answering (11 2016).

[21] M. Farazi, S. Khan, N. Barnes, From known to the unknown: Transferring knowledge to answer questions about novel visual and semantic concepts, Image and Vision Computing 103 (2020) 103985. doi:10.1016/j.imavis.2020.103985.

[22] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. Driessche, J.-B. Lespiau, B. Damoc, A. Clark, D. Casas, A. Guy, J. Menick, R. Ring, T. Hennigan, S. Huang, L. Maggiore, C. Jones, A. Cassirer, L. Sifre, Improving language models by retrieving from trillions of tokens (12 2021).

[23] Y. Wu, M. Rabe, D. Hutchins, C. Szegedy, Memorizing transformers (03 2022).

[24] P. Blatter, M. Kanakis, M. Danelljan, L. Gool, Efficient visual tracking with exemplar transformers (12 2021).

[25] K. Guu, K. Lee, Z. Tung, P. Pasupat, M.-W. Chang, Realm: Retrieval-augmented language model pre-training (02 2020).

[26] J. Thorne, A. Vlachos, C. Christodoulopoulos, A. Mittal, Fever: a large-scale dataset for fact extraction and verification, 2018. doi:10.18653/v1/N18-1074.

[27] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kttler, M. Lewis, W.-t. Yih, T. Rocktschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive nlp tasks (05 2020).

[28] F. Petroni, P. Lewis, A. Piktus, T. Rocktschel, Y. Wu, A. Miller, S. Riedel, How context affects language models' factual predictions, 2020.

[29] N. Moghe, S. Arora, S. Banerjee, M. Khapra, Towards exploiting background knowledge for building conversation systems (09 2018).

[30] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, 2016, pp. 3837–3845.
URL https://proceedings.neurips.cc/paper/2016/hash/04df4d434d481c5bb723be1b6df1ee65-Abstract.html

[31] H. Zhu, P. Koniusz, Simple spectral graph convolution, in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, OpenReview.net, 2021.
URL https://openreview.net/forum?id=CYO5T-YjWZV

[32] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, International Conference on Learning Rep-

resentations (2018).
URL https://openreview.net/forum?id=rJXMpikCZ

[33] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, M. M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model cnns, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, IEEE Computer Society, 2017, pp. 5425–5434. doi:10.1109/CVPR.2017.576.
URL https://doi.org/10.1109/CVPR.2017.576

[34] M. Fey, J. E. Lenssen, F. Weichert, H. Müller, Splinecnn: Fast geometric deep learning with continuous b-spline kernels, in: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, Computer Vision Foundation / IEEE Computer Society, 2018, pp. 869–877. doi:10.1109/CVPR.2018.00097.
URL http://openaccess.thecvf.com/content_cvpr_2018/html/Fey_SplineCNN_Fast_Geometric_CVPR_2018_paper.html

[35] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, Y. Sun, Masked label prediction: Unified message passing model for semi-supervised classification, in: Z. Zhou (Ed.), Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021, ijcai.org, 2021, pp. 1548–1554. doi:10.24963/ijcai.2021/214.
URL https://doi.org/10.24963/ijcai.2021/214

[36] X. Fan, M. Gong, Y. Xie, F. Jiang, H. Li, Structured self-attention

architecture for graph-level representation learning, Pattern Recognit. 100 (2020) 107084. doi:10.1016/j.patcog.2019.107084.
URL https://doi.org/10.1016/j.patcog.2019.107084

[37] B. Yang, Y. Kang, L. Zhang, H. Li, GGAC: multi-relational image gated GCN with attention convolutional binary neural tree for identifying disease with chest x-rays, Pattern Recognit. 120 (2021) 108113. doi:10.1016/j.patcog.2021.108113.
URL https://doi.org/10.1016/j.patcog.2021.108113

[38] S. Brody, U. Alon, E. Yahav, How attentive are graph attention networks?, in: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022, OpenReview.net, 2022.
URL https://openreview.net/forum?id=F72ximsx7C1

[39] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, T. Aila, Analyzing and improving the image quality of stylegan, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, Computer Vision Foundation / IEEE, 2020, pp. 8107–8116. doi:10.1109/CVPR42600.2020.00813.
URL https://openaccess.thecvf.com/content_CVPR_2020/html/Karras_Analyzing_and_Improving_the_Image_Quality_of_StyleGAN_CVPR_2020_paper.html

[40] Y. Yang, M. Z. Hossain, T. Gedeon, S. Rahman, S2FGAN: semantically aware interactive sketch-to-face translation, in: IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022,

Waikoloa, HI, USA, January 3-8, 2022, IEEE, 2022, pp. 3162–3171. doi:10.1109/WACV51458.2022.00322.
URL https://doi.org/10.1109/WACV51458.2022.00322

[41] M. Fey, J. Lenssen, Fast graph representation learning with pytorch geometric (03 2019).

[42] K. Zhu, J. Wu, Residual attention: A simple but effective method for multi-label recognition (10 2021). doi:10.1109/ICCV48922.2021.00025.

[43] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, 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.
URL https://openreview.net/forum?id=YicbFdNTTy

[44] Y. Lee, J. Kim, J. Willette, S. J. Hwang, Mpvit: Multi-path vision transformer for dense prediction, CoRR abs/2112.11010 (2021). arXiv:2112.11010.
URL https://arxiv.org/abs/2112.11010

[45] S. Chen, E. Xie, C. Ge, D. Liang, P. Luo, Cyclemlp: A mlp-like architecture for dense prediction, CoRR abs/2107.10224 (2021). arXiv:2107.10224.
URL https://arxiv.org/abs/2107.10224

[46] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, 2019, pp. 8024–8035.
URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html

[47] M. Fey, J. E. Lenssen, Fast graph representation learning with pytorch geometric, CoRR abs/1903.02428 (2019). arXiv:1903.02428.
URL http://arxiv.org/abs/1903.02428

[48] P. E. Rauber, A. X. Falcão, A. C. Telea, Visualizing time-dependent data using dynamic t-sne, in: E. Bertini, N. Elmqvist, T. Wischgoll (Eds.), 18th Eurographics Conference on Visualization, EuroVis 2016 - Short Papers, Groningen, The Netherlands, June 6-10, 2016, Eurographics Association, 2016, pp. 73–77. doi:10.2312/eurovisshort.20161164.
URL https://doi.org/10.2312/eurovisshort.20161164

[49] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–

90. doi:10.1145/3065386.
URL http://doi.acm.org/10.1145/3065386

[50] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, O. Wang, The unreasonable effectiveness of deep features as a perceptual metric, in: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, Computer Vision Foundation / IEEE Computer Society, 2018, pp. 586–595. doi:10.1109/CVPR.2018.00068.
URL http://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html