Self-supervised edge features for improved Graph Neural Network training

Arijit Sehanobish[*]
Internal Medicine (Cardiology) and Computer Science
Yale University
arijit.sehanobish@yale.edu
Neal G. Ravindra[*]
Internal Medicine (Cardiology) and Computer Science
Yale University
neal.ravindra@yale.edu
David van Dijk
Internal Medicine (Cardiology) and Computer Science
Yale University
david.vandijk@yale.edu

Abstract

Graph Neural Networks (GNN) have been extensively used to extract meaningful representations from graph structured data and to perform predictive tasks such as node classification and link prediction. In recent years, there has been a lot of work incorporating edge features along with node features for prediction tasks. One of the main difficulties in using edge features is that they are often handcrafted, hard to get, specific to a particular domain, and may contain redundant information. In this work, we present a framework for creating new edge features, applicable to any domain, via a combination of self-supervised and unsupervised learning. In addition to this, we use Forman-Ricci curvature as an additional edge feature to encapsulate the local geometry of the graph. We then encode our edge features via a Set Transformer and combine them with node features extracted from popular GNN architectures for node classification in an end-to-end training scheme. We validate our work on three biological datasets comprising of single-cell RNA sequencing data of neurological disease, *in vitro* SARS-CoV-2 infection, and human COVID-19 patients. We demonstrate that our method achieves better performance on node classification tasks over baseline Graph Attention Network (GAT) and Graph Convolutional Network (GCN) models. Furthermore, given the attention mechanism on edge and node features, we are able to interpret the cell types and genes that determine the course and severity of COVID-19, contributing to a growing list of potential disease biomarkers and therapeutic targets.

1 Introduction

Graph neural networks (GNN) have been widely used and developed for predictive tasks such as node classification and link prediction [1] and have been shown to learn from any sparse and discrete relational structure in data [2]. In particular, the use of similarity metrics to construct graphs from feature matrices expands the scope of GNN applications to domains where graph structured data is not readily available [3]. GNNs typically use message passing, or recursive neighborhood aggregation, to construct a new feature vector for a particular node by collecting its neighbor's feature vectors [4, 5].

[*] Equal contribution

Preprint. Under review.

However, most GNN schemes do not use edge features in learning new representations of graphical data.

Recently, edge features have been incorporated into GNNs to harness information describing different aspects of the relationships between nodes [6, 7]. However, there are very few frameworks for creating *de novo* edge feature vectors in a domain agnostic manner.

In this article, using Graph Attention Networks, we propose a self-supervised learning framework that is applicable to any graphical data, in which the learned edge attention coefficients become a set of edge features. We show that this novel approach improves GNN performance in downstream node classification tasks and improves training. Our framework is broad in the sense that any available metadata associated with a particular node can be fed into a self-supervised learning framework in this manner to extract edge features.

Given the devastating impact of the coronavirus disease of 2019 (COVID-19) caused by infection of SARS-CoV-2 and the gap in our understanding of the molecular mechanisms of the disease, we sought to focus our efforts on COVID-19 datasets that can generate hypotheses related to these gaps [8, 9]. Our focus on single-cell transcriptomic data relating to COVID-19 was motivated by recent work showing that Graph Attention Networks are effective at predicting disease states on an individual cell-to-cell basis [10]. Single-cell RNA sequencing (scRNA-seq) is a technology that yields large datasets comprising many thousands of cells' gene expression in a variety of conditions [11, 12, 13]. However, identifying factors important for determining an individual cell's pathophysiological trajectory or response to viral insult remains a challenge as single-cell data is noisy, sparse, and multi-dimensional [14, 15]. We reasoned that our framework's improved performance could extract useful insights into the genes and cell types that might be important determinants of COVID-19 severity and SARS-CoV-2 infection.

Our paper makes the following contributions:

- **Creation of edge features using semi-supervised learning** We propose a new approach to obtaining edge features that does not require supervision. From a feature matrix, we use unsupervised Louvain clustering in the graph domain to obtain community labels per node [16]. Then, using these cluster labels, we train a GAT model and use the learned edge attention coefficients as self-supervised edge features. We show that this self-supervised learning framework improves performance and training with two popular GNN architectures.
- **Use of Forman-Ricci curvature as an edge feature** We apply a novel way of describing the local structure of a graph by weighting an edge according to the common edges that pass through two nodes in the connection. This unsupervised learning of the graph structure can be applied to any graph and may enable archetypal or prototypical analysis of nodes that have high curvature.
- **Use of a Set Transformer to ingest edge features and enhance interpretability** In our self-supervised framework, we use a Set Transformer to combine edge features with message passing GNN layers in an end-to-end node classification task.
- **Identification of potential genes and cell types important to SARS-CoV-2 infection and COVID-19 severity** We use attention mechanisms in the GAT layer and the Set Transformer to propose genes and cells that can be important in determining the temporal dynamics of infection and disease severity in COVID-19 patient samples. These genes and cell types may provide potential therapeutic targets or markers of disease severity.

2 Related works

There is a wealth of research on Graph Neural Networks. A significant amount of work has been focused on graph embedding techniques, representation learning and various predictive analyses using node features. There has been recent interest in using edge features, in addition to node features, to improve the performance of Graph Neural Networks [6, 17, 18]. Many real-world graphs already have edge features, such as common keywords between nodes in citation networks or interaction affinities in protein-protein interaction networks. Even for graphs that do not *prima facie* have weighted edges or directions, one can construct edge features that describe the interaction between two nodes using Jaccard similarity, common neighbor metrics such as Adamic Adar or a variety of similarity metrics after embedding the graph. However in this work we use an unique multi–tasking approach in creating these edge features. We hope that this multi–tasking pre–training regime will further the research in meta–learning in the graph domain.

Since the primary focus of our work is in biological applications, it becomes necessary to be able to interpret the results of our network to inform further study of biology and medicine. One of the most common and popular ways to interpret machine learning models is via Shapley values [19] and it's various generalizations [20]. However Shapley values require independence of features which is hard to guarantee in general. Thus we follow the approach of [10, 21] in using attention mechanisms for interpretability. Thus even though set2set [22] is a popular mechanism to encode sets and has been previously used in the graph domain [23, 24], our view is that it is hard to interpret the hidden state of a LSTM. The transformer model [25, 26], on the other hand, allows us to interpret the results by looking at their attention heads.

GNNs have been used in biomedical research to predict medications, diagnoses, and outcomes from graphical representations of electronic health records [27]. GNNs have also been used to predict protein-protein and drug-protein interactions and molecular activity [28, 29, 30, 31]. However, fewer works attempt to predict pathophysiological state on an individual cell basis. One recent work uses GAT models to predict the disease state of individual cells derived from clinical samples [10]. However, their work ignores edge features, which may contain important information regarding cell type and the source from which a particular cell is derived. They do not consider multiple disease states or severity nor do they account for the confounding bias of batch effects, which may allow the network to learn a label for an individual cell based on its origin [13]. Here, we use the information contained within the dataset and a graph-structure that balances the batches, thus reducing the bias of cell source while preserving biological variation [32]. As such, we believe interpreting our model will provide information that is more biologically relevant than proposed in the previous works. To the best of our knowledge, this is the first attempt to apply a Deep Learning model to gain insight into multiple pathophysiological states, merging time-points, severity, and disease-state prediction into a multi-label node classification task from single-cell data using edge features.

3 Methods

Our work consists of three parts: (1) creating new edge features via self-supervised learning and local curvature; (2) using the edge features for downstream node classification tasks in an inductive setting by encoding edge features via a Set Transformer and node features via message passing GNN layers; (3) using attention coefficients to interpret our results and provide insights into our datasets.

3.1 Creating edge features via self-supervised learning

We use a combination of semi-supervised and unsupervised learning, and local graph geometry to create new edge features. We then concatenate these vectors to create a new edge feature vector.

3.1.1 Using semi-supervised learning to create new edge features

We use semi-supervised learning to create two types of edge features.
Using labels from unsupervised clustering in the spectral domain: Our first type of edge features rely on community detection by optimizing modularity. We use the Louvain clustering algorithm to assign nodes to different communities, as it has been widely used in single-cell data analysis, though it can be applied to any graph [16, 15]. We then use a GAT model to predict the community labels per node. Finally, we extract the edge attention coefficients from the first layer using equation 13. In this way we get an h dimensional vector, where h is the number of heads, which becomes h edge features.
Using other node metadata: Some graphs may have node labels that are not of interest for a particular task. For example, in single-cell data, cells from different patients, experiments, or sources are referred to as "batches" and are pooled into one dataset. In our datasets, we use the batch label for an individual cell as additional input for self-supervised learning. Using the same method as before, we construct an h-dimensional vector from the GAT model after batch label prediction.

3.1.2 Forman-Ricci curvature

We now use the internal geometry of the graph to create our next edge feature. We use the discrete version of the Ricci curvature as introduced by Forman [33] and discussed in [34].

$$Ric_F(e) := \omega(e)\Bigg(\frac{\omega(v_1)}{\omega(e)} + \frac{\omega(v_2)}{\omega(e)} - \sum_{e_{v_1}\sim v_1, e_{v_2}\sim v_2}\Bigg[\frac{\omega(v_1)}{\sqrt{\omega(e)\omega(e_{v_1})}} + \frac{\omega(v_2)}{\sqrt{\omega(e)\omega(e_{v_2})}}\Bigg]\Bigg) \tag{1}$$

where $e \in \mathbb{E}$ connecting nodes v_1 and v_2, $\omega(e)$ is the weight of the edge e which we take to the Euclidean distance in the PCA space, $\omega(v_i)$ is the weight of the node which we take to be 1 for simplicity and $e_{v_i} \sim v_i$ is the set of all edges connected to v_i and *excluding* e. This is an intrinsic invariant that captures the local geometry of the graph and relates to the global property of the graph via a Gauss-Bonnet style result [35].

3.1.3 Create edge features via node2vec

We use a popular embedding method called node2vec [36] to embed the nodes in a d dimensional space. We then calculate the dot product between these node embeddings as a measure of similarity. However to be consistent with our other methods we only compute the dot product between the nodes which share an edge. node2vec embeddings preserve the local community structure of a graph, which we expect should provide information to enable enhanced discriminability between nodes, as previously suggested [37].

3.2 Models

In this subsection we describe our model, which consists of two components: (1) A Set Transformer and (2) Message passing GNNs, like GCN or GAT layers.
Thus our model is quite general and is readily applicable with a wide variety of architectures. Since we use a GAT model to do feature extraction to create the edge features, we will describe the GAT model in detail below.

3.2.1 Set Transformer

We use a Set Transformer as in [26]. The Set Transformer is permutation invariant so it is an ideal architecture to encode sets. The building block of our Set Transformer is the multi-head attention, as in [25]. Given n query vectors Q of dimension d_q, a key-value pair matrix $K \in \mathbb{R}^{n_v \times d_q}$ and a value matrix $V \in \mathbb{R}^{n_v \times d_v}$ and, assuming for simplicity that $d_q = d_v = d$, then the attention mechanism is a function given by the following formula:

$$\text{att}(Q, K, V) := \text{softmax}(\frac{QK^T}{\sqrt{d}})V \tag{2}$$

This multihead attention is computed by first projecting Q, K, V onto h different d^h_q, d^h_q, d^h_v dimensional vectors where, for simplicity, we take $d^h_q = d^h_v = \frac{d}{h}$

$$\text{Multihead}(Q, K, V) := \text{concat}(O_1, \cdots, O_h)W^O \tag{3}$$

where

$$O_j = \text{att}(QW^Q_j, KW^K_j, VW^V_j) \tag{4}$$

and W^Q_j, W^K_j, W^V_j are projection operators of dimensions $\mathbb{R}^{d_q \times d^h_q}, \mathbb{R}^{d_q \times d^h_q}$ and $\mathbb{R}^{d_v \times d^h_v}$ respectively and W^O is a linear operator of dimension $d \times d$. Now, given a set S, the Set Transformer Block (STB) is given the following formula:

$$STB(S) := \text{LayerNorm}(X + rFF(X)) \tag{5}$$

where

$$X = \text{LayerNorm}(S + \text{Multihead}(S, S, S)) \tag{6}$$

rFF is a row-wise feedforward layer and LayerNorm is layer normalization [38].
A Set Transformer takes as input a 3d tensor of the form [batch, seq-length, input-dim] and outputs 3d tensor of the form [batch, seq-length, output-dim], i.e. it outputs sets of the same size as the input

sets. If, for a batch b_i, the set transformer outputs a set of the form $\{w_{i1}, ...w_{ij}\}$, we modify the output of the transformer to a fixed length vector

$$w_i := \sum_j \lambda_j w_{ij} \tag{7}$$

where λ_j are learnable weights. This step is necessary for us because our downstream tasks require vectors of fixed length.

3.2.2 Graph Attention Network

We use the popular Graph Attention Network (GAT) for extracting features from our auxiliary tasks. We follow the exposition in [31]. The input to a GAT layer are the node features, $\mathbf{h} = \{h_1, h_2, ..., h_N\}$, where $h_i \in \mathbb{R}^F$, N is the number of nodes, and F is the number of features in each node. The layer produces a new set of node features (of possibly different dimension F') as its output, $\mathbf{h}' = \{h'_1, h'_2, ...h'_N\}$ where $h'_i \in \mathbb{R}^{F'}$. The heart of this layer is multi-head self-attention like in [25, 31]. Self-attention is computed on the nodes

$$a^l : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R} \tag{8}$$

that computes attention coefficients, where a is a feedforward network.

$$e^l_{ij} = a^l(\mathbb{W}^l h_i, \mathbb{W}^l h_j) \tag{9}$$

where $\mathbb{W}^l$ is a linear transformation and also called the weight matrix for the head l. We then normalize these attention coefficients.

$$\alpha^l_{ij} = \text{softmax}_j(e^l_{ij}) = \frac{\exp(e^l_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e^l_{ik})} \tag{10}$$

where $\mathcal{N}_i$ is a 1-neighborhood of the node i. The normalized attention coefficients are then used to compute a linear combination of the features corresponding to them, to serve as the final output features for every node (after applying a nonlinearity, σ):

$$h^l_i = \sigma\Bigg(\sum_{j \in \mathcal{N}_i} \alpha^l_{ij} \mathbb{W}^l h_j \Bigg). \tag{11}$$

We concatenate the features of these heads to produce a new node feature, $h'_i := || \, h^l_i$.
However, for the final prediction layer, we average the representations over the heads and apply a logistic sigmoid non-linearity. Thus the equation for the final layer is:

$$h'_i = \sigma\Bigg(\frac{1}{K} \sum_{l=1}^{K} \sum_{j \in \mathcal{N}_i} \alpha^l_{ij} \mathbb{W}^l h_j \Bigg). \tag{12}$$

where K is the number of heads.
Our new edge features Λ_{ij} for the node e_{ij} are created by concatenating the α^l_{ij} across all heads, i.e.

$$\Lambda_{ij} := ||_{l=1}^{K} \alpha^l_{ij} \tag{13}$$

Thus we end up with a K-dimensional edge feature by this method.

3.2.3 Our model

In this section we will describe our model that combines edge features, obtained as described above, with node features for our main node classification tasks. We use a message passing networks to encode the node features. For example, for all of our experiments, we use either two GCN or two GAT layers. In the case of the GAT layers, we concatenate the representations obtained by different heads. For each node i, we construct a set $S_i := \{e_{ij} : j \in N_i\}$, where e_{ij} is the vector representing the edge features of the edge connecting nodes i and j. We then encode this set, S_i, which we call the edge feature set attached to the node i via our modified Set Transformer. This fixed length vector is concatenated with the node representation obtained after the second GCN or GAT layer. We call this new representation an enhanced node feature vector. This enhanced node feature vector is then passed through a dense layer with a logistic sigmoid non-linearity for classification. Figure 1A describes our model architecture which uses edge features, connectivity, and node features for node classification in an inductive setting.

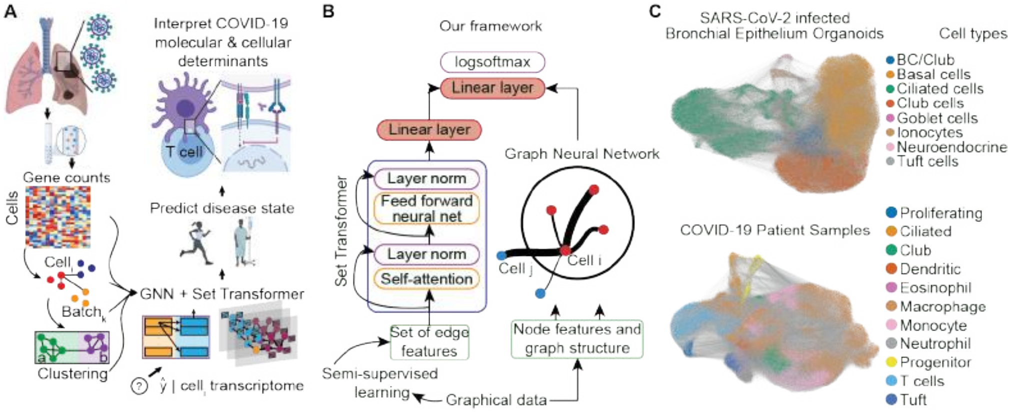

Figure 1: Our framework and datasets of interest. (**A**) Overview of our approach with respect to gaining molecular and cellular insights into COVID-19. (**B**) Our framework and models' architecture, integrating edge features with GNNs via a Set Transformer. (**C**) Graphical data used, showing cell types for each cell and edges in a node-feature, dimension-reduced embedding.

Table 1: Dataset description showing train/val/test splits.

Datasets	SARS-CoV-2 infected organoids	COVID-19 patients	Multiple sclerosis patients
# Nodes	54353/11646/11648	63486/13604/13605	53280/19980/11988
# Node features	24714	25626	22005
# Edges	1041226/230429/228630	2746280/703217/707529	6871820/2635746/1602662
# Edge features	18	18	18
# Classes	7	3	2

4 Experiments

We validate our model on the following scRNA-seq datasets:

- 4 human bronchial epithelial cell cultures or "organoids" that were inoculated with SARS-CoV-2 and co-cultured for 1, 2, and 3 days post-infection [39].
- Bronchoalveolar lavage fluid samples from 12 patients enrolled in a study at Shenzen Third People's Hospital in Guangdong Province, China of whom 3 were healthy controls, 3 had a mild or moderate form of COVID-19 and 6 had a severe or critical COVID-19 illness [40].
- Blood and CSF samples from 13 patients of whom 6 were healthy controls and 7 had multiple sclerosis, a neurological disease [10].

Table 2: Experimental tasks

Task	SARS-CoV-2 infected organoids	COVID-19 patients	Multiple sclerosis patients
Louvain cluster ID	Cell type	Cell type	Cell type
Batch or node metadata	Culture sample ID	Patient ID	Patient ID + sample type
Inductive prediction	Timepoint and infection	No, Mild, or Severe Disease	MS or Healthy

See table 1 for a summary of our datasets. For all the datasets we create a Batch-Balanced kNN graph to remove the confounding bias of experimental or sequencing differences between samples [41]. For more details about data pre-processing and graph construction from single cell data, please refer to the supplementary material. Table 2 details all the tasks that we perform on our datasets.

Auxiliary tasks : We first describe our auxiliary tasks which we devise to create new edge features. We cluster our datasets using Louvain clustering [16], and annotate these clusters as "cell types," as commonly done in single-cell analysis [15]. Then, we use a 2-layer GAT with 8 attention heads in each layer to predict the cell type label. We extract the edge attention coefficients from the first layer of our trained model as edge features. Thus we get an 8-dimensional edge feature vector by equation 13. All of our biological datasets have a batch ID associated to it, i.e. some metadata that keeps track of the origin of the cell. We use the same method as before to create another 8-dimensional edge feature vector. More details and results about the auxiliary tasks can be found in the supplementary material.

Main tasks : Our final task is node label prediction in an inductive setting, as shown in 2. All the results shown are from the test set and our model's performance is reported in table 3. Our model outperforms the baseline models by a significant margin. We note that our results on the MS dataset differ from the results as reported in [10] since we use a different graph kernel (BB-kNN kernel [32]), which reduces bias due to technical measurement artifacts of the data. We also calculated the p-value (Welch's t-test) between our model and the baseline GAT and GCN models. The p-value was $< .001$ for all the experiments showing that our models are a significant improvement over the baseline.

Table 3: Results of inductive tasks on single-cell datasets showing accuracy and 95% confidence intervals.

Models	SARS-CoV-2 infected organoids	COVID-19 patients	Multiple sclerosis patients	P (Welch's t-test)
GCN	65.43 (65.21-65.65)	89.26 (89.06-89.47)	73.23 (72.93-73.53)	-
GCN + Edge Features (Ours)	**81.61 (79.34-83.87)**	**92.84 (91.95-93.74)**	**85.06 (83.85-86.28)**	< 0.001
GAT	73.10 (70.93-75.27)	92.25 (91.27-93.24)	73.03 (72.22-73.83)	-
GAT + Edge Features (Ours)	**82.95 (81.75-84.15)**	**95.12 (94.02-96.22)**	**85.03 (84.34-85.72)**	< 0.001

Other than improved performance we found that our model trains faster than the baseline GCN and GAT models. We compared the loss per epoch for baseline GCN and GAT models versus our models. Broadly, our model achieves significant and consistently lower loss per epoch and requires fewer epochs to train, as shown in Figure 2.

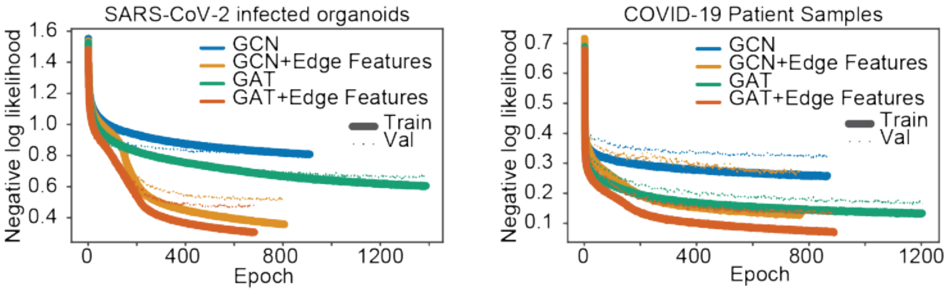

Figure 2: Negative log likelihood losses per epoch for training models for our COVID-19 datasets.

5 Model Interpretability

In addition, we extract the learned weights from the GAT layer to investigate our model's feature saliency with respect to gene importance in predicting SARS-CoV-2 infection and COVID-19 severity. In predicting COVID-19 severity from patient samples, our model gives high weight to genes involved in the innate immune system response to type I interferon (CCL2, CCL7, IFITM1), regulation of signaling (NUPR1, TAOK1, MTRNR2L12), a component of the major histocompatibility complex II (HLA-DQA2), which is important for developing immunity to infection, and a marker of eosinophil cells, which are involved in fighting parasites (RETN). In predicting SARS-CoV-2 infection, our model finds saliency in counts of viral transcript, which is encouraging. In addition, to predicting SARS-CoV-2, genes involved in inflammatory response and cell death (NFKBIA) and interferon signaling (IFI27) appear to be important, as does signaling, which may provide clues as to the dynamic response to SARS-CoV-2 infection in the lung's airways (IFI27, HCLS1, NDRG1, NR1D1, TF).

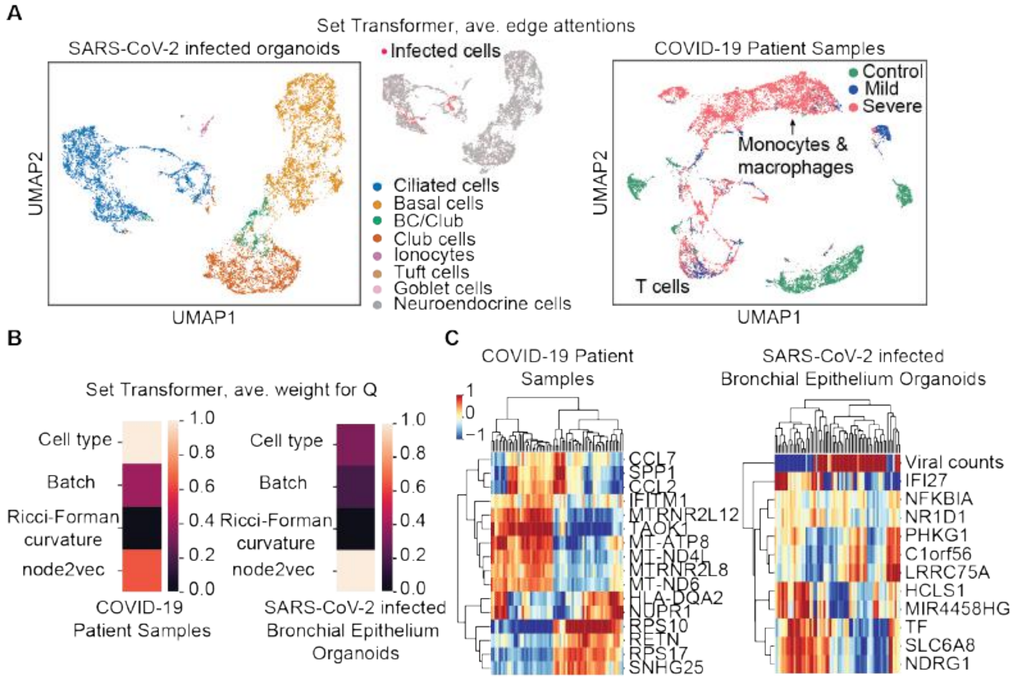

Figure 3: Model interpretability to generate hypotheses for genes and cells important to COVID-19 severity. (**A**) Embedding learned from graphs extracted from average edge attentions across Set Transformer output dimensions, showing cell type or condition per cell. (**B**) Relative importance of crafted edge features in disease state prediction tasks, averaged across the query layer from the Set Transformer. (**C**) Top 5 important gene features for each GAT head, colored by normalized, learned weights.

Using the edge attentions from the Set Transformer, we construct a new graph and perform unsupervised clustering and manifold learning [42, 16]. We obtain distinct cell clusters of SARS-CoV-2 infected cells which are also segregated by cell type. These cells may have unique behaviors that warrant further analysis. The learned embedding for the organoids dataset highlights that our model segregates infected ciliated cells, which is the reported SARS-CoV-2 cell tropism, validating our models' interpretability [39]. In predicting COVID-19 severity, it is interesting that our model learns to mix macrophages and monocytes in a predominantly severe patient cell cluster while cells derived from mild and severe patients are mixed in a T cell cluster. Monocytes derived from macrophages are thought to be enriched in severe COVID-19 cases and T cells are proposed targets for immune checkpoint therapy of COVID-19, despite lack of understanding as to the transcriptional differences between mild and severe COVID-19 illness [43, 40, 44]. Lastly, our models find that genes involved in type I interferon signaling are important in predicting both COVID-19 severity and susceptibility to SARS-CoV-2 infection. Interferon signaling is involved in pro-inflammatory immune responses and it is suspected that type I interferon signaling may cause immunopathology during SARS-CoV-2 infection leading to critical illness [39, 44].

6 Discussion

We achieved significant performance enhancements using self-supervised edge features when comparing two popular GNN architectures, GCN and GAT models, to our architecture that builds on those models. This suggests that using edge features derived from self-supervised learning and local graph information, with no requirement for hand-crafted edge features, can improve graph neural network performance on challenging node classification tasks. Our models are simple, easy to train and can be used with various graph neural architectures and our edge creation framework is applicable to any graphical data. This flexibility may benefit training and performance, as we show with three biological datasets, but will also expand interpretability to local geometry of the graph using Forman-Ricci curvature. We anticipate that in the future, this metric will help with local explanations of decision boundaries in GNNs. Finally as a future direction we hope to pursue our multi–tasking approach for meta-learning in the graph domain.

Our model allows us to gain insights into the cell tropism of SARS-CoV-2 and to elucidate the genes and cell types that are important for predicting COVID-19 severity. We are encouraged to find that genes involved in regulating the immune system are important for predicting SARS-CoV-2 infection and COVID-19 severity. Given the inclusion of edge features and cell types in our model, we are also encouraged that we identified clusters of cells that may be involved in immunopathology [44, 39]. Further study into the interaction partners and the subtle transcriptional differences between the cells and cell types we identified may provide complementary hypotheses or avenues for therapeutic intervention to mitigate the impacts of COVID-19.

7 Broader Impact

The impact of the COVID-19 pandemic is tragic and its extent is still unknown. Here, we attempt to bring accurate disease state prediction to a molecular and cellular scale so that we can identify the cells and genes that are important for determining susceptibility and resistance to SARS-Cov-2 infection and severe COVID-19 illness via interpreting our models. To the best of our knowledge, no deep learning method can perform as well as we have on predicting multiple disease-states for a single-cell sample. Typically, biologists rely on identifying cells by clustering and dimensionality reduction and compare their differential gene expression to identify molecular determinants of disease. However, this is often done without checking if the differences are meaningful or *predictive*, which we do here. In addition, identifying the cells, cell types, and genes that are important for SARS-CoV-2 infection and COVID-19 severity contributes a long list of potential biomarkers for disease state diagnosis and therapeutic targets for further investigation. However, there are many caveats to our study. While we achieve good performance with our models, model interpretability in artificial neural networks does not have a strong theoretical basis, and any proposed features should merely be thought of as putative hypotheses into the mechanisms of viral insult. In addition, cells in the COVID-19 patient and MS patient datasets are derived from a relatively small patient population, albeit large for single-cell or clinical studies. While we, for the first time, attempt to limit this bias by using a batch-balanced kNN graph, which we also think makes it more likely that our framework learns from biological variability, we remain vulnerable to the idiosyncrasies of the samples. Thus, any potential feature that identified as important for prediction should only be considered meaningful after extensive experimental validation. We are not medical professionals so we do *NOT* claim that interpretation of our model will bear any fruit. Rather, we hope that the approach of seeking excellent and state-of-the-art predictive results on disease states at single-cell resolution will enhance study of biology and medicine and potentially accelerate our understanding of critical diseases during crises like the COVID-19 pandemic.

Acknowledgements

We acknowledge the Yale Center for Research Computing for our use of their High Performance Computing infrastructure. We thank Mia Madel Alfajaro and Craig B. Wilen for generating the SARS-CoV-2 infected organoids dataset and sharing the data with us. We also thank Jenna L. Pappalardo and David A. Hafler for generating the MS patients dataset and sharing the data with us.

References

[1] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2020. arXiv: 1901.00596.

[2] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning Discrete Structures for Graph Neural Networks. *arXiv:1903.11960*, May 2019.

[3] J. B. Tenenbaum. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 290(5500):2319–2323, December 2000.

[4] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation Learning on Graphs with Jumping Knowledge Networks. *International Conference on Machine Learning (ICML)*, pages 5453–5462, 2018.

[5] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[6] Liyu Gong and Qiang Cheng. Exploiting edge features in graph neural networks, 2018.

[7] Zheng Gao, Gang Fu, Chunping Ouyang, Satoshi Tsutsui, Xiaozhong Liu, Jeremy Yang, Christopher Gessner, Brian Foote, David Wild, Qi Yu, and Ying Ding. edge2vec: Representation learning using edge semantics for biomedical knowledge discovery, 2018.

[8] Li Yan, Hai-Tao Zhang, Jorge Goncalves, Yang Xiao, Maolin Wang, Yuqi Guo, Chuan Sun, Xiuchuan Tang, Liang Jing, Mingyang Zhang, Xiang Huang, Ying Xiao, Haosen Cao, Yanyan Chen, Tongxin Ren, Fang Wang, Yaru Xiao, Sufang Huang, Xi Tan, Niannian Huang, Bo Jiao, Cheng Cheng, Yong Zhang, Ailin Luo, Laurent Mombaerts, Junyang Jin, Zhiguo Cao, Shusheng Li, Hui Xu, and Ye Yuan. An interpretable mortality prediction model for COVID-19 patients. *Nature Machine Intelligence*, 2(5):283–288, 2020.

[9] Jixin Zhong, Jungen Tang, Cong Ye, and Lingli Dong. The immunology of COVID-19: is immune modulation an option for treatment? *The Lancet Rheumatology*, 2020.

[10] Neal Ravindra, Arijit Sehanobish, Jenna L. Pappalardo, David A. Hafler, and David van Dijk. Disease state prediction from single-cell data using graph attention networks. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, CHIL '20, page 121–130, New York, NY, USA, 2020. Association for Computing Machinery.

[11] Grace X. Y. Zheng, Jessica M. Terry, Phillip Belgrader, Paul Ryvkin, Zachary W. Bent, Ryan Wilson, Solongo B. Ziraldo, Tobias D. Wheeler, Geoff P. McDermott, Junjie Zhu, Mark T. Gregory, Joe Shuga, Luz Montesclaros, Jason G. Underwood, Donald A. Masquelier, Stefanie Y. Nishimura, Michael Schnall-Levin, Paul W. Wyatt, Christopher M. Hindson, Rajiv Bharadwaj, Alexander Wong, Kevin D. Ness, Lan W. Beppu, H. Joachim Deeg, Christopher McFarland, Keith R. Loeb, William J. Valente, Nolan G. Ericson, Emily A. Stevens, Jerald P. Radich, Tarjei S. Mikkelsen, Benjamin J. Hindson, and Jason H. Bielas. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 8(1):14049, April 2017.

[12] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. Single-cell RNA sequencing technologies and bioinformatics pipelines. *Experimental and Molecular Medicine*, 50, 2018.

[13] Tim Stuart and Rahul Satija. Integrative single-cell analysis. *Nature Reviews Genetics*, 20(5):257–272, MAY 2019.

[14] Geng Chen, Baitang Ning, and Tieliu Shi. Single-cell rna-seq technologies and related computational data analysis. *Frontiers in Genetics*, 10:317, 2019.

[15] Vladimir Yu Kiselev, Tallulah S. Andrews, and Martin Hemberg. Challenges in unsupervised clustering of single-cell rna-seq data. *Nature Reviews Genetics*, 20(5):273–282, May 2019.

[16] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, Oct 2008.

[17] Pengfei Chen, Weiwen Liu, Chang-Yu Hsieh, Guangyong Chen, and Shengyu Zhang. Utilizing edge features in graph neural networks via variational information maximization, 2019.

[18] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. Learning edge representations via low-rank asymmetric projections. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, Nov 2017.

[19] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[20] T. P. Michalak, K. V. Aadithya, P. L. Szczepanski, B. Ravindran, and N. R. Jennings. Efficient computation of the shapley value for game-theoretic network centrality. *Journal of Artificial Intelligence Research*, 46:607–650, Apr 2013.

[21] Ahmed M. Alaa and Mihaela van der Schaar. Attentive state-space modeling of disease progression. In *Advances in Neural Information Processing Systems 32*, pages 11338–11348. Curran Associates, Inc., 2019.

[22] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets, 2015.

[23] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1263–1272. JMLR.org, 2017.

[24] Han Altae-Tran, Bharath Ramsundar, Aneesh S. Pappu, and Vijay Pande. Low data drug discovery with one-shot learning, 2016.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[26] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks, 2018.

[27] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F. Stewart, and Jimeng Sun. Gram: Graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 787–795, New York, NY, USA, 2017. Association for Computing Machinery.

[28] Thin Nguyen, Hang Le, Thomas P. Quinn, Thuc Le, and Svetha Venkatesh. GraphDTA: Predicting drug–target binding affinity with graph neural networks. *bioRxiv*, 2019.

[29] H.C. Stephen Chan, Hanbin Shan, Thamani Dahoun, Horst Vogel, and Shuguang Yuan. Advancing Drug Discovery via Artificial Intelligence. *Trends in Pharmacological Sciences*, 40(8):592–604, 2019.

[30] Haripriya Harikumar, Thomas P. Quinn, Santu Rana, Sunil Gupta, and Svetha Venkatesh. A random walk down personalized single-cell networks: predicting the response of any gene to any drug for any patient, 2019.

[31] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[32] Md Luecken, M Büttner, K Chaichoompu, A Danese, M Interlandi, Mf Mueller, Dc Strobl, L Zappia, M Dugas, M Colomé-Tatché, and Fj Theis. Benchmarking atlas-level data integration in single-cell genomics. *bioRxiv*, 2020.

[33] Robin Forman. Bochner's method for cell complexes and combinatorial ricci curvature. *Discrete and Computational Geometry*, 29:323–374, 2003.

[34] Areejit Samal, R. P. Sreejith, Jiao Gu, Shiping Liu, Emil Saucan, and Jürgen Jost. Comparative analysis of two discretizations of ricci curvature for complex networks. *Scientific Reports*, 8(1), Jun 2018.

[35] Kazuyoshi Watanabe. Combinatorial ricci curvature on cell-complex and gauss-bonnnet theorem, 2017.

[36] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks, 2016.

[37] Megha Khosla, Vinay Setty, and Avishek Anand. A Comparative Study for Unsupervised Network Representation Learning. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2019. arXiv: 1903.07902.

[38] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[39] Neal G. Ravindra, Mia Madel Alfajaro, Victor Gasque, Jin Wei, Renata B. Filler, Nicholas C. Huston, Han Wan, Klara Szigeti-Buck, Bao Wang, Ruth R. Montgomery, Stephanie C. Eisenbarth, Adam Williams, Anna Marie Pyle, Akiko Iwasaki, Tamas L. Horvath, Ellen F. Foxman, David van Dijk, and Craig B. Wilen. Single-cell longitudinal analysis of sars-cov-2 infection in human bronchial epithelial cells. *bioRxiv*, 2020.

[40] Mingfeng Liao, Yang Liu, Jing Yuan, Yanling Wen, Gang Xu, Juanjuan Zhao, Lin Cheng, Jinxiu Li, Xin Wang, Fuxiang Wang, Lei Liu, Ido Amit, Shuye Zhang, and Zheng Zhang. Single-cell landscape of bronchoalveolar immune cells in patients with COVID-19. *Nature Medicine*, 2020.

[41] Krzysztof Polański, Matthew D Young, Zhichao Miao, Kerstin B Meyer, Sarah A Teichmann, and Jong-Eun Park. BBKNN: fast batch alignment of single cell transcriptomes. *Bioinformatics*, 36(3):964–965, 2019.

[42] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018.

[43] Melissa Bersanelli. Controversies about covid-19 and anticancer treatment with immune checkpoint inhibitors. *Immunotherapy*, 12(5):269–273, 2020.

[44] Benjamin Israelow, Eric Song, Tianyang Mao, Peiwen Lu, Amit Meir, Feimei Liu, Mia Madel Alfajaro, Jin Wei, Huiping Dong, Robert J Homer, Aaron Ring, Craig B Wilen, and Akiko Iwasaki. Mouse model of sars-cov-2 reveals inflammatory role of type i interferon signaling. *bioRxiv*, 2020.

[45] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks, 2010.

[46] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus), 2016.

[47] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc.

[48] Jose Alquicira-Hernandez, Anuja Sathe, Hanlee P Ji, Quan Nguyen, and Joseph E Powell. scPred: accurate supervised method for cell-type classification from single-cell RNA-seq data. *Genome Biology*, 20(1):264, 2019.

[49] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks, 2019.

[50] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings, 2016.

[51] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[52] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks, 2018.

[53] John Boaz Lee, Ryan A. Rossi, Sungchul Kim, Nesreen K. Ahmed, and Eunyee Koh. Attention models in graphs: A survey. *ACM Trans. Knowl. Discov. Data*, 13(6), November 2019.

[54] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks? *arXiv:1810.00826 [cs, stat]*, February 2019.

[55] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems (NIPS)*, pages 1025–1035, 2017.

[56] Yang Ye and Shihao Ji. Sparse Graph Attention Networks. *arXiv:1912.00552 [cs, stat]*, 2019. arXiv: 1912.00552.

[57] F. Alexander Wolf, Philipp Angerer, and Theis Fabian J. Scanpy: large-scale single-cell gene expression data analysis. *Genome Biology*, 19(15):e–print, 2018.

[58] Maren Büttner, Zhichao Miao, F. Alexander Wolf, Sarah A. Teichmann, and Fabian J. Theis. A test metric for assessing single-cell RNA-seq batch correction. *Nature Methods*, 16(1):43–49, Jan 2019.

[59] Carlos Torroja and Fatima Sanchez-Cabo. Digitaldlsorter: Deep-learning on scrna-seq to deconvolute gene expression data. *Frontiers in Genetics*, 10:978, 2019.

[60] Cédric Arisdakessian, Olivier Poirion, Breck Yunits, Xun Zhu, and Lana X Garmire. Deepimpute: an accurate, fast, and scalable deep neural network method to impute single-cell rna-seq data. *Genome Biology*, 20(1):211, 2019.

[61] Matthew Amodio, David van Dijk, Krishnan Srinivasan, William S Chen, Hussein Mohsen, Kevin R Moon, Allison Campbell, Yujiao Zhao, Xiaomei Wang, Manjunatha Venkataswamy, Anita Desai, V Ravi, Priti Kumar, Ruth Montgomery, Guy Wolf, and Smita Krishnaswamy. Exploring single-cell data with deep multitasking neural networks. *Nature Methods*, 16(11):1139–1145, 2019.

[62] Jingshu Wang, Divyansh Agarwal, Mo Huang, Gang Hu, Zilu Zhou, Chengzhong Ye, and Nancy R. Zhang. Data denoising with transfer learning in single-cell transcriptomics. *Nature Methods*, 16(9):875+, SEP 2019.

[63] Xiang Yue, Zhen Wang, Jingong Huang, Srinivasan Parthasarathy, Soheil Moosavinasab, Yungui Huang, Simon M Lin, Wen Zhang, Ping Zhang, and Huan Sun. Graph embedding on biomedical networks: methods, applications and evaluations. *Bioinformatics*, 36(4):1241–1251, Oct 2019. btz718.

[64] Danuta R Gawel, Jordi Serra-Musach, Sandra Lilja, Jesper Aagesen, Alex Arenas, Bengt Asking, Malin Bengnér, Janne Björkander, Sophie Biggs, Jan Ernerudh, Henrik Hjortswang, Jan-Erik Karlsson, Mattias Köpsen, Eun Jung Lee, Antonio Lentini, Xinxiu Li, Mattias Magnusson, David Martínez-Enguita, Andreas Matussek, Colm E Nestor, Samuel Schäfer, Oliver Seifert, Ceylan Sonmez, Henrik Stjernman, Andreas Tjärnberg, Simon Wu, Karin Åkesson, Alex K Shalek, Margaretha Stenmarker, Huan Zhang, Mika Gustafsson, and Mikael Benson. A validated single-cell-based strategy to identify diagnostic and therapeutic targets in complex diseases. *Genome Medicine*, 11(1):47, 2019.

[65] Tao Zeng and Hao Dai. Single-cell rna sequencing-based computational analysis to describe disease heterogeneity. *Frontiers in Genetics*, 10:629, 2019.

[66] Mohammad Lotfollahi, F Alexander Wolf, and Fabian J Theis. scGen predicts single-cell perturbation responses. *Nature Methods*, 16(8):715–721, 2019.

[67] Jie Zheng and Ke Wang. Emerging deep learning methods for single-cell RNA-seq data analysis. *Quantitative Biology*, 7(4):247–254, 2019.

[68] Adrian Haimovich, Neal G Ravindra, Stoytcho Stoytchev, H Patrick Young, Francis P Wilson, David van Dijk, Wade L Schulz, and Richard Andrew Taylor. Development and validation of the covid-19 severity index (csi): a prognostic tool for early respiratory decompensation. *medRxiv*, 2020.

[69] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *Advances in Neural Information Processing Systems 32*, pages 15663–15674. Curran Associates, Inc., 2019.

[70] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Jonathan Reynolds, Alexander Melnikov, Natalia Lunova, and Orion Reblitz-Richardson. Pytorch captum. `https://github.com/pytorch/captum`, 2019.

[71] Rahul Satija, Jeffrey A Farrell, David Gennert, Alexander F Schier, and Aviv Regev. Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology*, 33(5):495–502, 2015.

[72] Xinxin Zhang, Yujia Lan, Jinyuan Xu, Fei Quan, Erjie Zhao, Chunyu Deng, Tao Luo, Liwen Xu, Gaoming Liao, Min Yan, Yanyan Ping, Feng Li, Aiai Shi, Jing Bai, Tingting Zhao, Xia Li, and Yun Xiao. CellMarker: a manually curated resource of cell markers in human and mouse. *Nucleic Acids Research*, 47(D1):D721–D728, 10 2018.

A Data pre-processing

A.1 Feature matrix preparation

Prior to graph creation, all samples were processed with the standard single-cell RNA-sequencing pre-processing recipe using Scanpy [57, 71]. Cells and genes for the MS dataset were pre-processed as described in [10]. For the SARS-CoV-2 infected organoids and COVID-19 patients datasets, genes expressed in fewer than 3 cells and cells expressing fewer than 200 genes were removed but, to allow for characterization of stress response and cell death, cells expressing a high percentage of mitochondrial genes were not removed. For all single-cell datasets, transcript or ”gene” counts per cell were normalized by library size and square-root transformed.

A.2 Graph creation

To create graphs from a cell by gene counts feature matrix, we used a batch-balanced kNN graph [41]. BB-kNN constructs a kNN graph that identifies the top k nearest neighbors in each ”batch”, effectively aligning batches to remove bias in cell source while preserving biological variability [32]. We used annoy's method of approximate neighbor finding with a Euclidean distance metric in 50-dimensional PCA space. Per ”batch” we find $k = 3$ top nearest neighbors. An example BB-kNN graph is schematized in main text, Figure 1A.

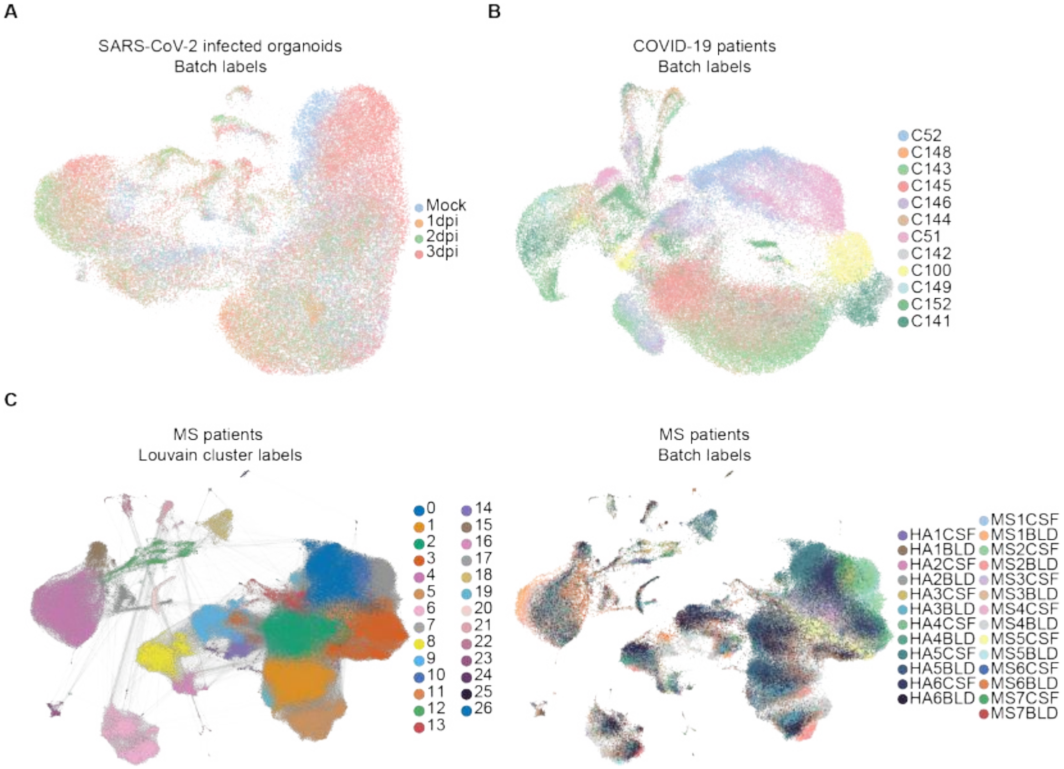

Figure 4: UMAP embeddings of individual cells colored by labels for auxiliary tasks. (**A**) Batch labels for SARS-Cov-2 infected organoids dataset. (**B**) Batch labels for COVID-19 patients, for patient IDs described in [40]. (**C**) Graph used for the MS dataset with Louvain cluster labels to represent cell types (left) and batch labels per patient sample (right).

B Hyperparameters and Training details

For auxiliary tasks and for training our models, we break our graph into 5000 subgraphs using the ClusterData function in PyTorch Geometric library and then minibatched the subgraphs using the ClusterData function. These algorithms are originally introduced in [49]. We used a single

Table 4: Default hyperparameters used in the experiments

	Graph Attention Network	Graph Convolution Network
Number of layers	2	2
Hidden_size	8	256
Attention Heads	8	N/A
Optimizer	Adagrad	Adagrad
weight_decay	.0005	.0005
Batch size	256	256
Dropout	.5	.4
Slope in LeakyRelu	.2	.2
Training Epochs	1000	1000
Early stopping	100	100

block of Set Transformer with input dimension 18, output dimension 8 and 2 heads. The rest of the hyperparamaters of GAT and GCN can be found in table 4.

For our auxiliary tasks and for baseline experiments we used an 8GB Nvidia RTX2080 GPU and for our main tasks we used an Intel E5-2660 v3 CPU with 121GB RAM.

C Auxiliary task

In this section we describe our auxiliary tasks. Table 5 gives details about the number of labels for the auxiliary tasks. We first predict the cell types as given by the Louvain clustering [16]. In figure 4 we show Louvain community ID or cluster labels for the MS patients dataset, which can be annotated as cell types as in [10]. In the main text, we used [72] to obtain cell type markers and annotate the Louvain cluster labels as ”cell types” explicitly.

Next we predict the batch ID of each node, i.e. which patient or from where the cell is obtained. Table 6 shows our results for these auxiliary tasks. In single-cell RNA-sequencing, variability between batches can explain more of the transcriptomic variability than variability in the biological process of interest; these ”batch effects” can complicate model inference [15]. Our novel use of BB-kNN graphs for the tasks described in this paper limits this ”batch effect” bias.

Table 5: Number of labels for auxiliary tasks

Task	SARS-CoV-2 infected organoids	COVID-19 patients	Multiple sclerosis blood & CSF
Cell type	8	10	27
Batch	4	12	26

Table 6: Results on auxiliary tasks

Prediction	SARS-CoV-2 infected organoids	COVID-19 patients	Multiple sclerosis blood & CSF
Cell type	93.84	82.03	75.90
Batch	76.16	64.08	36.21

D Code and Data Availability

The processed data for SARS-CoV-2 infected organoids samples and the COVID-19 patient samples can be found at this link. We do not have the permission to share the MS patient data, as it belongs to a third party. All code used to reproduce these results are available in the associated supplementary material and will be published on GitHub post-review.