STAT 946 Deep Learning

Project Title:

Cancer Classification using DeepSets and Graph Neural Network

Group Members:

Surname, First Name	Student ID	Your Department
Liu, Siqi	20428295	Computer Science
Wang, Zibai	20819005	Computer Science

Your project falls into one of the following categories. Check the boxes which describe your project the best.

☑ **Suggested projects** Our project is one of the suggested projects (Covid 19 or Cancer classification)

□ **New algorithm.** We developed a new algorithm and demonstrated (theoretically and/or empirically) why our technique is better (or worse) than other algorithms. (Note: A negative result does not lose marks, as long as you followed proper theoretical and/or experimental techniques).

□ **Application.** We applied known algorithm(s) to some domain.

- □ We applied the algorithm(s) to our own research problem.
  - □ We used an existing implementation of the algorithm(s).
  - □ We implemented the algorithm(s) ourself.
- □ We tried to reproduce results of someone else's paper.
  - □ We used an existing implementation of the algorithm(s).
  - □ We implemented the algorithm(s) ourself.

Our most significant contributions are (List at most three):

1. Propose a graph neural network approach, enhanced by DeepSets, for representing a WSI using a set of patches
2. Illustrate the impacts of class imbalance and small training set size on a graph neural network
3. .

Cancer Classification using DeepSets and Graph Neural Network

Siqi Liu
Department of Computer Science
University of Waterloo
Waterloo, N2L 3G1
sq2liu@uwaterloo.ca

Zibai Wang
Department of Computer Science
University of Waterloo
Waterloo, N2L 3G1
zibai.wang@uwaterloo.ca

Abstract

The objective of this project is to design an algorithm to distinguish three sub-types of lung cancers, Lung Adenocarcinoma (LUAD), Lung Squamous Cell Carcinoma(LUSC) and Malignant Mesothelioma (MESO), by using the pre-collected relevant patches from Whole Slide Images (WSIs). We design and compare three approaches, including a single patch classification model, a DeepSet-based model and an enhanced DeepSet model with a graph neural network. We show that the DeepSet and graph neural network approach yields the highest accuracy (72%) on our test set. We provide evidence that class imbalance has a profound impact on model performance, and more training data for minority classes are needed for generalizability.

1 Introduction

Whole slide images (WSI) are created by scanning a complete microscope slide to build a single high-resolution digital file.[1] Past efforts have been made to use deep learning to aid pathologists in inspecting WSIs and diagnosing cancer. [10] One of the main challenges is that each WSI consists of billions of pixels, a substantial amount of data for any network to process [2]. To apply a deep learning algorithm, WSIs have to be divided into smaller patches. Other research [2] has shown promising results by using multiple instances learning and leveraging all patches information to learn a representation for a WSI. In this paper, we develop three approaches to classify a given WSI into one of three cancer types (LUAD, LUSC and MESO) using a set of patches. Our results show that by leveraging DeepSets, a form of permutation invariant learning, and graph neural network, we can derive meaningful representations for our WSIs and achieve an accuracy score of 72% on our test set.

We organize the rest of the paper as follows. We begin by providing some background information on the graph neural network and our preprocessing method. Next, we provide the details of our three approaches and showcase their results. Lastly, we conclude and state our findings.

2 Background

2.1 ChebNet

The Chebyshev spectral graph convolutional operator was introduced by Defferrard et al. [4]. Spectral convolutions on graphs are defined as the multiplication of a signal (node features) by a filter, similar to convolutional operations on images, where pixel values are multiplied by filter values [9].

In ChebNet, the filter is represented by this equation.

[1] https://www.mbfbioscience.com/whole-slide-imaging

$$g_\theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k T_k \tilde{\Lambda} \tag{1}$$

As Tong [9] explained: g_θ is a kernel (θ represents the vector of Chebyshev coefficients) applied to Λ. Λ is the diagonal matrix of Laplacian eigenvalues and $\tilde{\Lambda}$ stands for the diagonal matrix of scaled Laplacian eigenvalues. k represents the smallest order neighbourhood and K represents the largest order neighbourhood. Finally, T represents the Chebyshev polynomials of the k_{th} order. We are using an implementation from PyTorch-geometric [2].

3 Data

3.1 Preprocessing

The patches we have are of size 512 x 512. To accommodate our methodologies which rely on a large number of patches per each WSI, we use ImageMagick[3] to divide each patch into four equal-sized patches of size 256 x 256. Table 1 shows the resulting distribution for both the training and test sets. Note that after the division, around 95% of all WSIs have at least 30 patches.

Table 1: Distribution of WSI and 256 x 256 Patches

Dataset	Cancer Type	# WSI	# Patches			
			min	5%tile	50%tile	max
Train	LUAD	72	20	32	200	1,132
	LUSC	156	8	44	408	1,128
	MESO	9	8	28	96	588
Test	LUAD	38	24	32	320	916
	LUSC	41	8	36	420	1,188
	MESO	5	168	228	304	904

4 Approach

4.1 Single Patch Classification

We begin by predicting each patch individually and select the majority class as the prediction for the WSI. To introduce sample diversity, we apply data augmentation techniques such as random flipping and rotation to our training data. We create a classification model based on the pre-trained MobileNetV2. We replace the original classification head with a global average pooling layer, a dropout layer and a classification linear layer. We unfreeze parameters in the pre-trained model from layers 100 and fine-tune it using a custom class weight of [LUAD: 6, LUSC: 1, MESO: 10]. We achieve an accuracy score of 0.53 on the test set.

4.2 DeepSets

We use the DeepSets architecture [2] to form a permutation invariant representation of each WSI, shown in Figure 1. For any given WSI, we select 30 patches and extract their features using DenseNet [6] and global max pooling. We choose 30 to be the maximum number of patches from each WSI because 95% of all training WSIs, except for MESO, have at least 30 patches, as shown in Table 1. The resulting feature representation for each patch is of size 1024. Before passing the set and its features into DeepSets, we apply a linear layer with a Tanh activation function to reduce the dimensionality from 1024 to 512. In cases where the number of patches available in a WSI is less than 30, we apply zero paddings to the stacked features and use binary masks to inform the model which rows are patches and which rows are pads. We feed the stacked features into the DeepSets

[2] https://pytorch-geometric.readthedocs.io/en/latest/modules/nn.html

[3] https://imagemagick.org/

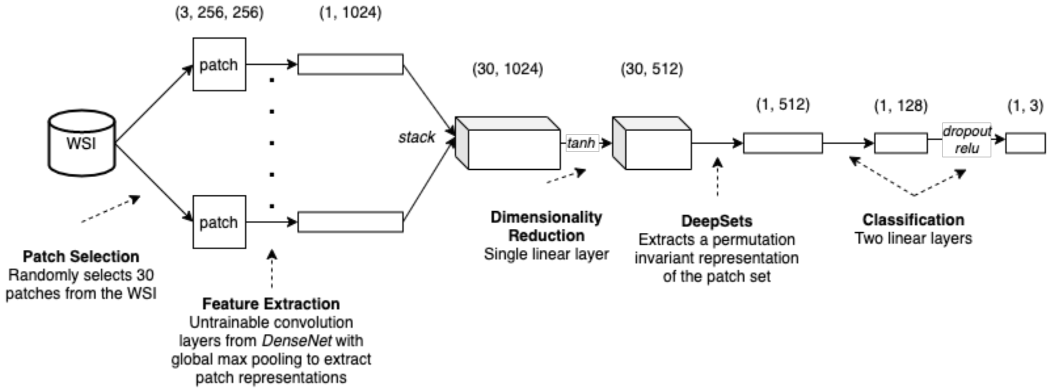


Figure 1: Proposed architecture with DeepSets

layer, forming a permutation invariant representation of the patch set. After experimenting with various reduction methods (e.g., mean, max, sum) for DeepSets, we find the mean reduction method to be the most effective. Finally, the output of the DeepSets layer gets passed into two linear layers for classification.

We train our model using the 256 x 256 dataset. To mitigate class imbalance, we apply the SMOTE algorithm [7] to upsample data points in the minority classes (i.e., LUAD and MESO). Since MESO only has 9 WSIs in the training set, some WSIs repeat as many as 18 times to match the count in the majority class. We train our model using AdamW optimizer [8] with a learning rate of 1e-6 and cross-entropy as our loss function. We achieve an accuracy score of 0.81 on the training set.

For evaluation, we draw multiple sets of random patches for each WSI in the test set and form our final prediction for the WSI based on a simple majority vote. This is to reduce the variance in our results caused by randomness in patch sampling. The number of sets drawn is dynamically determined based on the number of patches in the WSI (shown in Equation 2). In our test set, the average number of patches in a WSI is 294, resulting in an average number of 25 sets drawn per WSI. We obtain an accuracy score of 0.64 on our test set.

$$\#Sets = min(50, max(1, log_{10}(\binom{\#Patches}{30}))) \tag{2}$$

4.3 DeepSets and Graph Neural Network

Inspired by the approach taken by Adnan et al. [1], we enhance our DeepSets approach with a graph neural network. Our proposed architecture is shown in Figure 2. We start with the same feature extraction steps as in the DeepSets approach with an additional linear layer to reduce the output feature space from 1024 to 512. We lower the number of patches for each WSI from 30 to 15 to reduce the size of the fully connected graphs. We feed the sampled patch features through a DeepSets layer with mean reduction to obtain a permutation invariant representation of size 256 for the set, which we then append to each patch feature. Next, we model our patches as nodes in a fully connected graph. Connections between the nodes are modelled as an adjacency matrix of size 15 x 15. We use a graph convolutional network to learn the representation of the graph and use a graph pooling layer to get a single feature representation for the WSI. Adnan et al. [1] used ChebConv and GraphSAGE graph convolutional networks. After experimenting with other types of graph convolutional networks such as GCNConv, GraphConv and SGConv, we find ChebConv to yield the best performance. Finally, we pass the single feature representation into two linear layers for classification.

We train our model using the upsampled training set and evaluate our model on the test set using dynamically drawn sets and majority voting, identical to our previous approach. We achieve improved accuracy scores of 0.84 and 0.72 on the training and test sets respectively. Figure 3a and Figure 3b show the resulting confusion matrices. As we can see, our model still overfits the minority classes even after upsampling. Figure 4 shows a t-SNE visualization [11] of the upsampled training set features before the classification layers. Based on the visualization, our model can differentiate

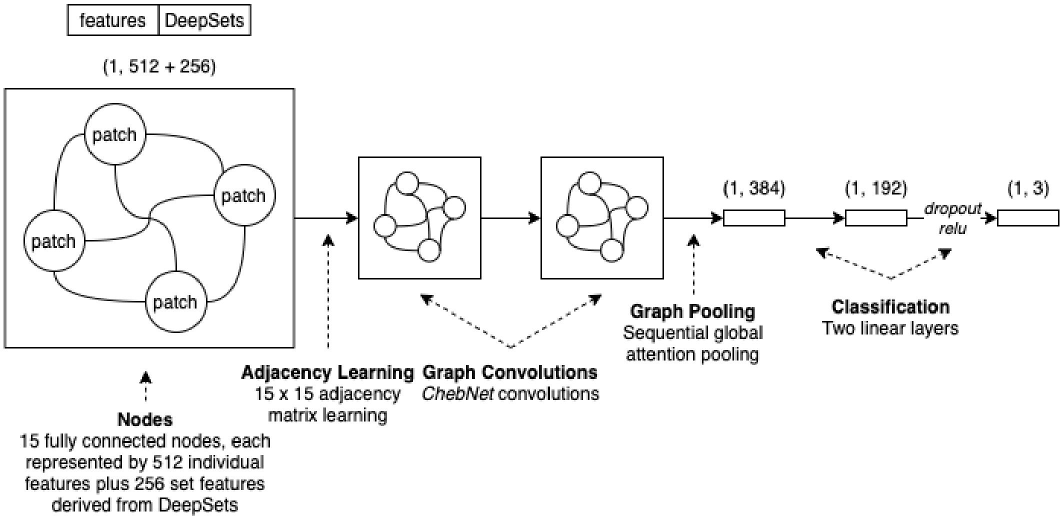
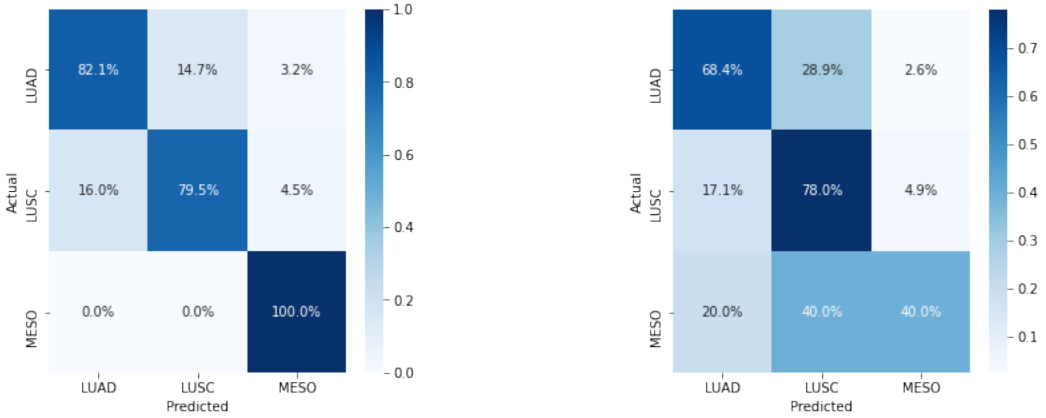


Figure 2: Proposed architecture with DeepSets and Graph Neural Network



(a) Training set (upsampled)

(b) Test set

Figure 3: Confusion matrices, values shown as % of actual

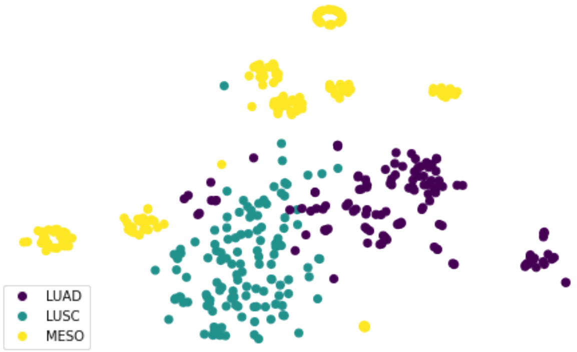


Figure 4: t-SNE visualization of training set features after graph pooling

LUAD and LUSC and separate them into two clusters effectively. However, the upsampled MESO data points are grouped into individual clusters, most likely one for each WSI. This indicates that even though our model can learn similarities between patches within a MESO WSI, it has difficulty learning a generalizable representation for MESO as a whole, which explains the drop in model performance on the test set for MESO. We believe that more training data for MESO is crucial in increasing model performance and establishing better generalizability for MESO.

5 Conclusion and Discussion

In this paper, we compare three different approaches to classifying a WSI using a set of patches. Our first approach attempts to classify each patch individually and form prediction on the whole WSI based on simple majority votes. This naive approach performs poorly on the test set with an accuracy score of 0.53. Our second approach involves modelling a permutation invariant representation of each WSI using the DeepSets framework [3]. Under this approach, we see a drastic increase in model performance on the test set and our accuracy score increases to 0.64. Finally, we enhance our previous approach with a graph neural network, and the accuracy score increases to 0.72. We show that our final approach with DeepSets and graph neural network is effective in classifying LUAD and LUSC. However, it experiences significant difficulty in learning generalizable representations for MESO due to a lack of training data, despite our best effort at upsampling. We are confident that with more training data, our proposed approach can form meaningful representations for all three classes.

6 Acknowledgement

We would like to thank Prof. Ali Ghodsi for delivering an informative and interesting course. We also want to express deep gratitude to our beloved TA, Aref Jafari, for preparing the dataset and inspiring us with great ideas.

References

[1] Adnan, Mohammed, Shivam Kalra, and Hamid R. Tizhoosh. "Representation Learning of Histopathology Images using Graph Neural Networks." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020.

[2] "Representation Learning of Histopathology Images Using Graph Neural Networks." Kimia Lab, kimialab.uwaterloo.ca/kimia/index.php/representation-learning-of-histopathology-images-using-graph-neural-networks/.

[3] Zaheer, Manzil, et al. "Deep sets." arXiv preprint arXiv:1703.06114 (2017).

[4] Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. "Convolutional neural networks on graphs with fast localized spectral filtering.", arXiv preprint arXiv:1606.09375 (2016).

[5] Hemati, Sobhan, et al. "CNN and Deep Sets for End-to-End Whole Slide Image Representation Learning." OpenReview, 17 Mar. 2021, openreview.net/forum?id=BX0kKB1zB1Q.

[6] Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[7] Chawla, Nitesh V., et al. "SMOTE: synthetic minority over-sampling technique." Journal of artificial intelligence research 16 (2002): 321-357.

[8] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[9] "Graph Convolutional Networks for Geometric Deep Learning", Flawnson Tong. https://towardsdatascience.com/graph-convolutional-networks-for-geometric-deep-learning-1faf17dee008.

[10] Iizuka, O., Kanavati, F., Kato, K. et al. Deep Learning Models for Histopathological Classification of Gastric and Colonic Epithelial Tumours. Sci Rep 10, 1504 (2020). https://doi.org/10.1038/s41598-020-58467-9

[11] Van der Maaten, Laurens, and Geoffrey Hinton. "Visualizing data using t-SNE." Journal of machine learning research 9.11 (2008).