Unsupervised Image Segmentation with GNNs

- Abdelnour Fellah
- Adel Abdelkader Mokadem
- Meriem Mekki
- Yacine Lazreg Benyamina

ESI-SBA | Artificial Intelligence and Data Science

Introduction

This project is part of the **Network Sciences** course unit and focuses on exploring the application of **Graph Neural Networks (GNNs) in image segmentation**. The aim is to delve into some research papers that address how GNNs can enhance image segmentation tasks, highlighting innovative approaches and their impact on this field **(UNSEGGNET, UnSeGArmaNet, and DeepCut).**

Source code is available in this GitHub Repo.

What are GNNs ?

Graph neural networks (GNNs) are neural models that capture the dependence of graphs via **message passing** between the nodes of graphs.
In recent years, variants of GNNs such as graph convolutional network **(GCN)**, graph attention network **(GAT)**, graph recurrent network **(GRN)** have demonstrated ground-breaking performances on many deep learning tasks.

What is Image Segmentation ?

Image segmentation is an end-to-end image analysis process that divides a digital image into multiple segments and classifies the information contained in each region. There are three kinds of image segmentation tasks: semantic, instance and panoptic segmentation. This project addresses **semantic segmentation** which labels every region of an image by its semantic class, classifying them by using information like color, contrast, placement within the image and other attributes.

Why is Semantic Image Segmentation Important ?

Semantic segmentation helps machines **distinguish** the different object classes and background regions in an image. It plays an important role in training computers to **recognize important context** in digital images such as landscapes, photos of people, medical images, and much more. In the medical field, segmentation proves invaluable for tasks such as identifying anatomical structures, diagnosing diseases, planning treatments, and monitoring disease progression.

Supervised vs. Unsupervised Learning

Supervised Learning

- Relies on labeled data, where the model learns from input-output pairs (e.g., images with corresponding labels). It is widely used for tasks like classification and regression.

Unsupervised Learning

- Works with unlabeled data, aiming to identify patterns, structures, or clusters in the data without explicit supervision.

Why Unsupervised Segmentation?

Limited Availability of Labeled Data

Collecting and labeling data is often expensive, time-consuming, and requires domain expertise.

Generalization Challenges

Supervised models, trained on a specific dataset, may struggle to generalize to new or diverse datasets due to variations in imaging protocols, equipment, and patient populations.

Modality Imbalance

Supervised learning can suffer from modality imbalance, where certain structures or pathologies are underrepresented in the dataset, leading to biased segmentation models that perform suboptimally for minority classes.

Scalability

Unsupervised learning leverages vast amounts of unlabeled data, addressing these challenges while scaling efficiently.

Image Segmentation as a Graph Clustering Problem

Image segmentation can be framed as a **graph clustering** or partitioning problem, where **an image is treated as a graph**. The approach of the explored research papers (DeepCut, UnSeGArmaNet, and UNSEGGNET) treats **each patch of the image as a node**, with **features extracted from each patch**. The edges represent the relationships or similarities between adjacent patches. **The goal is to partition the graph into distinct clusters** or subgraphs, where each cluster corresponds to a homogeneous region of the image, such as an object or background.

Graph Convolutional Networks (GCN)

Core Idea:
GCNs extend convolution operations from grid-like data (e.g., images) to graph-structured data. They operate on nodes, aggregating features from their neighbors to generate new node embeddings.

How it Works:
Each node aggregates feature information from its neighbors. Aggregation is weighted by the graph's adjacency matrix and normalized to account for varying node degrees.

Update Rule: $H^{(l+1)} = \sigma(\tilde{A} * H^{(l)} * W^{(l)})$

Where:

- H (l) : Node features at layer l
- Â : Normalized adjacency matrix
- W(l): Trainable weights
- σ: Activation function

Graph Attention Networks (GAT)

Core Idea:
GATs introduce attention mechanisms to GNNs, allowing the model to assign different importance to neighbors during feature aggregation.

How it Works:
Computes an attention coefficient a(i,j) for each edge between nodes i,j

and: $softmax(LeakyReLU(\alpha^T[W.h_i||W.h_j]))$

Where:

- α is a learnable weight vector, W transforms the node features, and || represents concatenation.
- Node features are updated as a weighted sum of neighbors' features based on attention coefficients.

ARMA Graph Convolutional Networks (ARMA)

Core Idea:
ARMA layers are inspired by Autoregressive Moving Average (ARMA) filters from signal processing. They aim to improve the stability and expressiveness of graph convolutions over multiple layers.

How it Works:

- Each ARMA layer uses stacked parallel filters to approximate a smoother graph convolution operator.
- These filters consist of learnable parameters to refine feature propagation over the graph.
- The model uses K-layer depth to propagate information, maintaining stability and reducing oversmoothing.

Update Rule: $H^{(l+1)} = \frac{1}{K} \sum_{k=1}^{K} \sigma(\tilde{A}^k * H^{(l)} * W_k^{(l)})$

Unsupervised Image Segmentation with GNNs Pipeline

1 **Feature Extraction:**

- All the explored research papers (DeepCut and UNSEGGNET) leverage the DINO-ViT model to extract visual features for every patch of the input image.
- These feature vectors represent the high-level characteristics of each patch, capturing essential visual information.

2 **Graph Construction:**

- For every image, a graph is constructed based on the visual features of its patches.
- Nodes in the graph represent the feature vectors of individual patches.
- Edges are created based on similarity or spatial relationships between the patches.

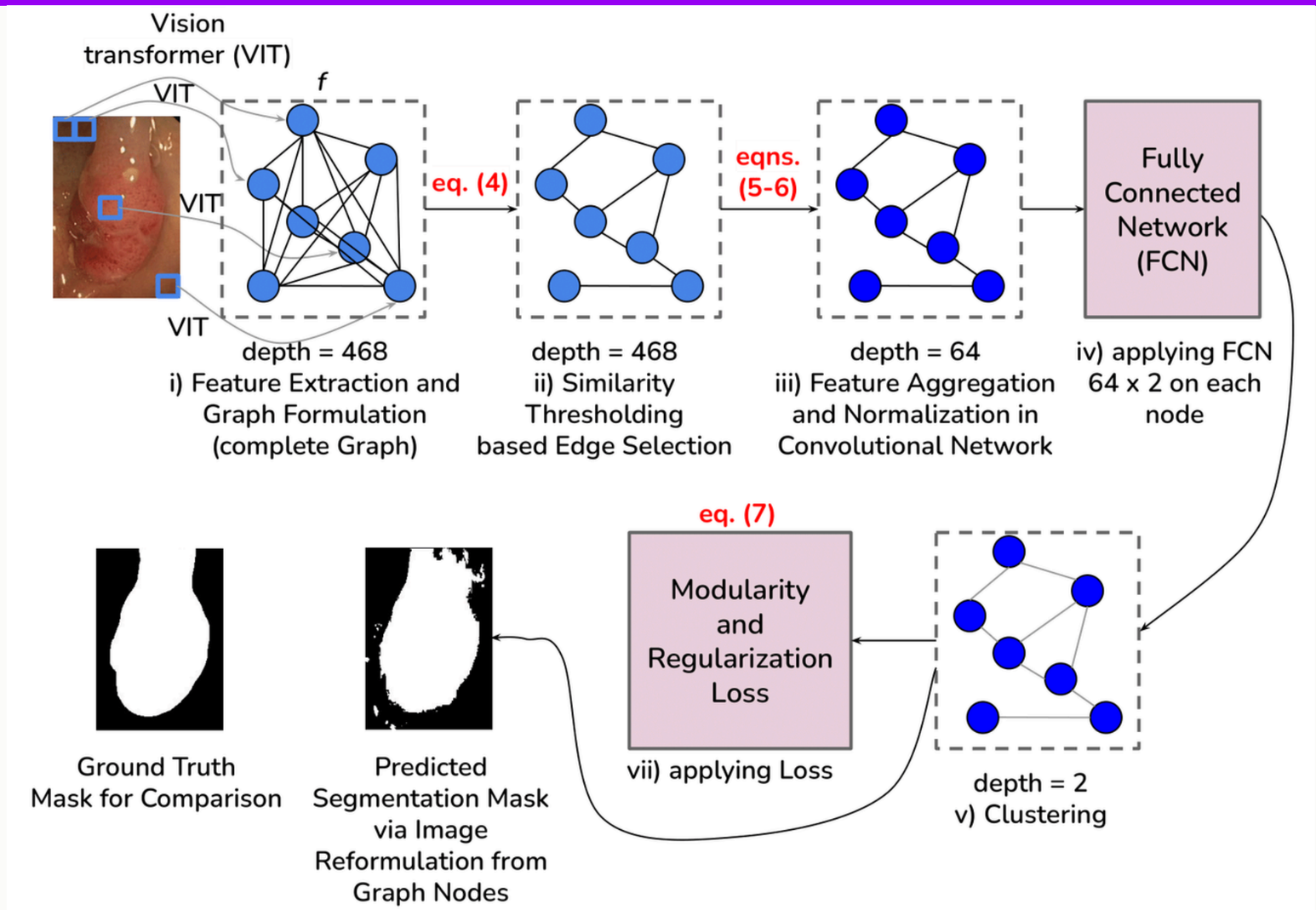
3 **GNN Model Training:**

- A Graph Neural Network is trained to capture relationships between nodes characterizing the image patches and assign each node to a cluster.
- The training process involves optimizing a specific loss function that promotes accurate clustering of similar nodes.

4 **Post-Processing:**

- After clustering, a post-processing step refines the segmentation results to ensure coherence and smoothness in the segmented regions.

Unsupervised Image Segmentation with GNNs Pipeline



Feature Extraction With DINO

Overview

Self-**di**stillation with **no** labels is a technique used to train Vision Transformers (ViTs) in a self supervised manner without the need of labeled data making possible to make use of the vast amount of unlabeled images in the internet to pre-train models that can be used later for feature extraction or fine-tuned on a specific task.

Feature Extraction With DINO

Overview

The framework makes use of data augmentation techniques to create two versions of the same image that are fed to two models : Student model & a teacher model. where the student model is updated using back-propagation on the following loss function :

$$L = -softmax\left(\frac{p_t - C}{tmp_t}\right) \times log\left(softmax\left(\frac{p_s}{tmp_s}\right)\right)$$

Feature Extraction With DINO

Overview

Where :

- p_t : Teacher logits.
- p_s : Student logits.
- tmp_t : Teacher's temperature.
- tmp_s : Teacher's temperature.
- C : Centering parameter.

Feature Extraction With DINO

Training Algorithm

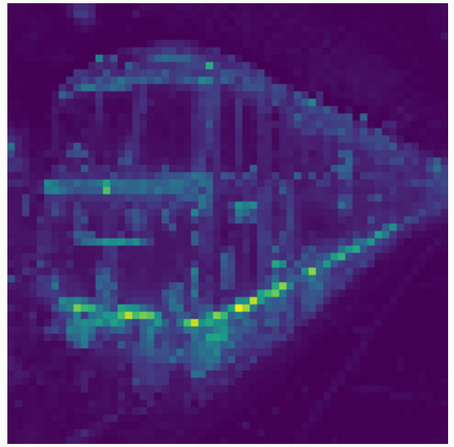
```
- Initialize Student & Teacher model
- copy(teacher.params, student.params)
- For x in dataset:
  - x1, x2 = augment(x), augment(x)
  - t1, t2 = teacher(x1), teacher(x2)
  - s1, s2 = student(x1), student(x2)
  - loss = 0.5 * L(t1, s2) + 0.5 * L(t2, s1)
  - backprop(student, loss)
  - update(student)
  - teacher.params = l * teacher.params + (1 - l) * student.params
  - C = m * C + (1-m) * mean(cat(t1, t2))
```

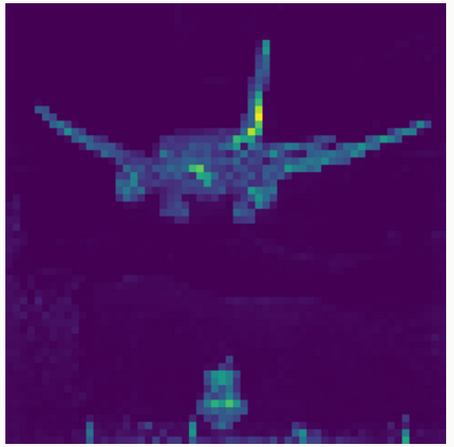
- Where m and l are two hyper-parameters chosen between 0 and 1.

Feature Extraction With DINO

Why DINO ?

A particularity that makes **DINO** specifically convenient to use as a feature extractor in segmentation tasks is the correspondence between the attention maps and the relevant objects in the image.





Feature Extraction With DINO

How ?

For feature extraction we use a pre trained DINO model on **ImageNet21K** that uses a patch size of 8, without any further **fine-tuning** and the necessary preprocessing is applied to the images including resizing and normalization and then the key vectors are extracted for each patch in the images.

Methods

For the graph creation we use three different methods each with its own associated loss functions,**NCUT** & **CC** methods were proposed in the **DeepCut** paper while the **DMON** method was proposed in the **UnSegGNet** paper.

Methods

DeepCut : NCUT

Graph Construction

The weight matrix representing the graph is computed using the matrix product of the features matrix with itself,eliminating the negative values given that the objective function only accepts positive values then normalized so that the maximum value is one.

$$W = \frac{max(0, X.\,X^T)}{\max_{i,j} X.\,X^T}$$

- X : Represents the image features.

Methods

DeepCut : NCUT

Objective Function

the loss function that make use of soft assignments used during back-propagation is defined as follows :

$$L\left(S,W\right)=\frac{Tr\left(S^{T}WS\right)}{Tr\left(S^{T}DS\right)}+\|\frac{S^{T}S}{\|S^{T}S\|_{F}}-\frac{I_{k}}{\sqrt{K}}\|_{F}$$

Where :

- Tr : Trace of a matrix
- S : Model output / soft cluster assignments.
- W : Graph’s weight matrix.
- D : A diagonal matrix representing row wise sum of W
- k : Number of clusters.

Methods

DeepCut : Correlation Clustering (CC)

Graph Construction

Unlike **NCUT**,Correlation clustering allows for negative values in the weight matrix

$$W = X.X^T - \frac{max\left(X.X^T\right)}{\alpha}$$

- X : Image's DINO Features.
- α : A hyper-parameter that controls the number of clusters.

Methods

DeepCut : Correlation Clustering (CC)

Objective function

The corresponding loss function used during back propagation to update the model have the following equation :

$$L(S, W) = -Tr(W.S.S^T)$$

Methods

UnSegGNet : DMON

Graph Construction

DMON introduces a new a hyper-parameter which is a threshold τ to generate a binary, sparse adjacency matrix :

$$W_{i,j} = 1 \text{ if } cos(x_i, x_j) > \tau \text{ else } 0$$

Methods

UnSegGNet : DMON

Objective function

$$L(S,W) = -\frac{1}{2m}Tr(S^T.B.S) + \frac{\sqrt{k}}{n} \mid\mid \sum_{i=1}^{N} S_i \mid\mid_F -1$$

Where :

- $B = W - \frac{d.d^T}{2m}$
- $d = \sum_j W_{i,j}$
- n : Number of nodes.
- k : Number of clusters.
- $m = \sum_{i,j} W_{i,j}$

Experiments and Results

Datasets

ISIC 2016

International Skin Imaging Collaboration (ISIC) dataset is a collection of images that represents skin lesions with their corresponding masks,only the training set was used which contains 900 data points.

EMD6

Environmental Microorganism Image Dataset Sixth Version (EMD6) is a collection of 840 environmental microorganisms divided into 21 subtypes with their corresponding mask often used to evaluate computer vision models on various tasks including image segmentation.

Experiments and Results

Metrics

For the task of object detection and image segmentation two metrics are common to assess the model's performance are **IoU** (Intersection over union) and **Dice** Coefficient.

$$IoU = \frac{|\ A \cap B\ |}{|\ A \cup B\ |}$$

$$DICE = \frac{2*|\ A \cap B\ |}{|\ A\ | + |\ B\ |} = \frac{2 * IoU}{IoU + 1}$$

Experiments and Results

Experiments

For each **NCUT** and **DMON** losses we additionally experimented with three different **GNN** layers (**GCN**,**GAT** and **ARMA**),for CC method we've only tried **GCN** and **GAT** since **ARMA** doesn't accept negative values due to its normalization method (thus it has to be noted that normalization had to be deactivated in the case of **GCN**)

Additionally the models were evaluated every 10 epochs for a maximum of 100 epochs to assess the impact of the number of epochs on the segmentation quality.

Experiments and Results

Experiments

In all cases we used the same model structure a single **GNN** layer of a given type followed by a **SiLU** activation function followed by a linear layer with **ELU** activation function then a finally softmax layer to output cluster probabilities.

In all expermiments we set $\alpha = 5$ and $\tau = 0.7$.

Experiments and Results

Results : ISIC 2016

Method	GNN Layer	IoU	Dice	Best Epoch
NCUT	GCN	0.622	0.708	90
NCUT	GAT	0.388	0.409	100
NCUT	ARMA	**0.623**	**0.714**	30
CC	GCN (NO Normalization)	0.434	0.539	20
CC	GAT	0.158	0.219	80

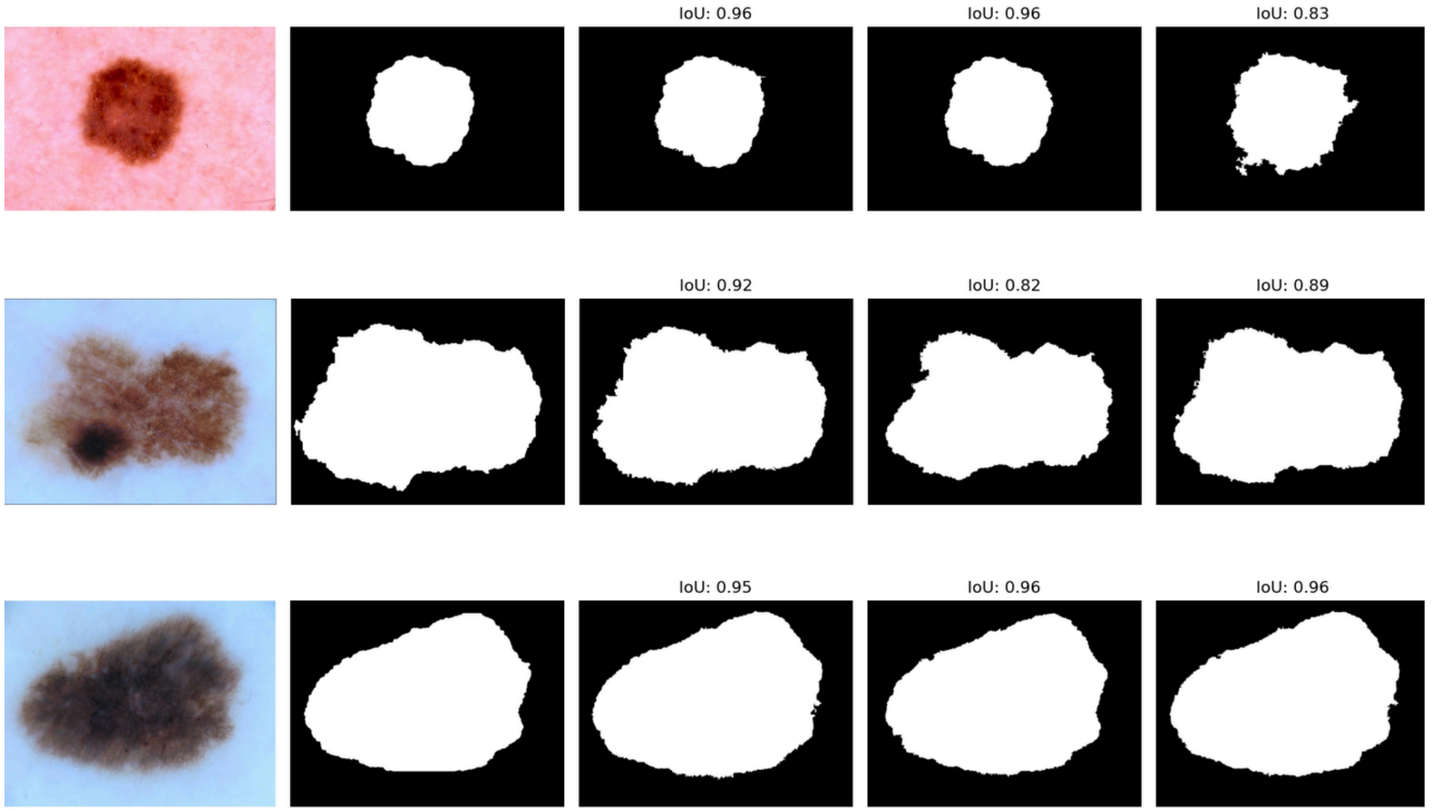
Experiments and Results

Results : ISIC 2016

Method	GNN Layer	IoU	Dice	Best Epoch
DMON	GCN	0.363	0.483	100
DMON	GAT	0.378	0.498	90
DMON	ARMA	0.500	0.606	80

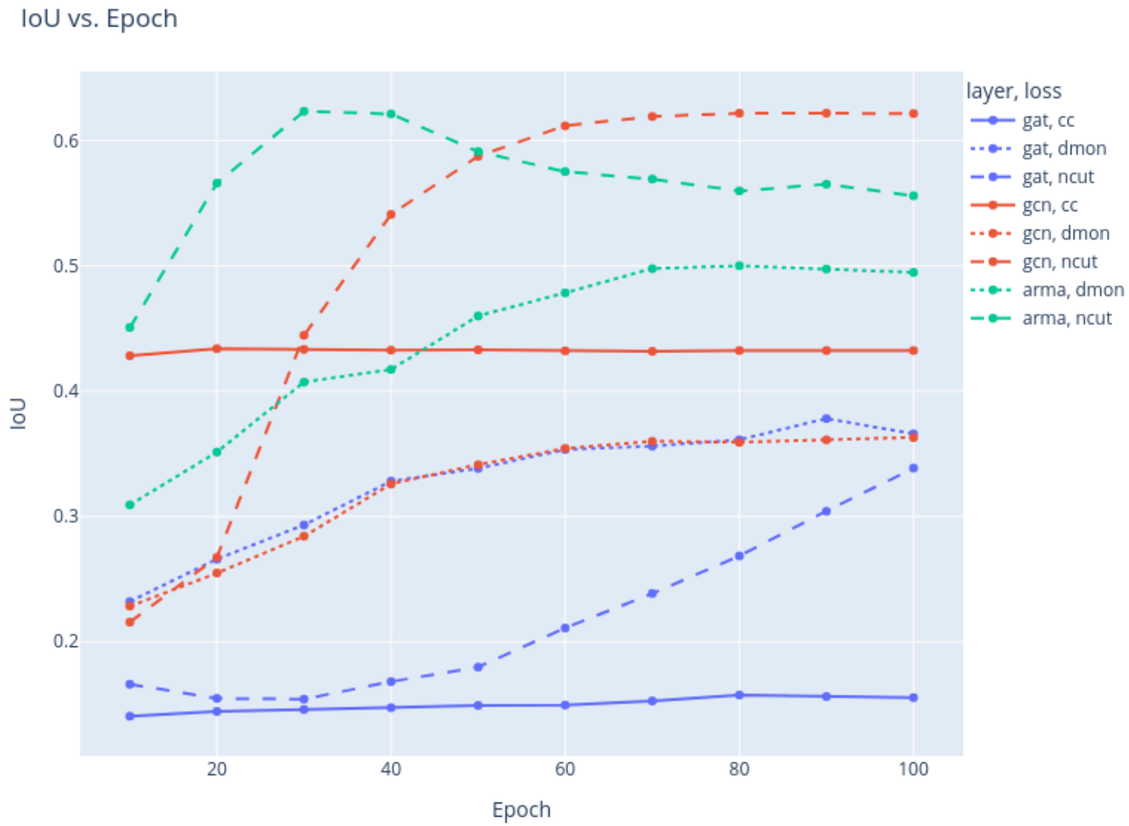
Experiments and Results

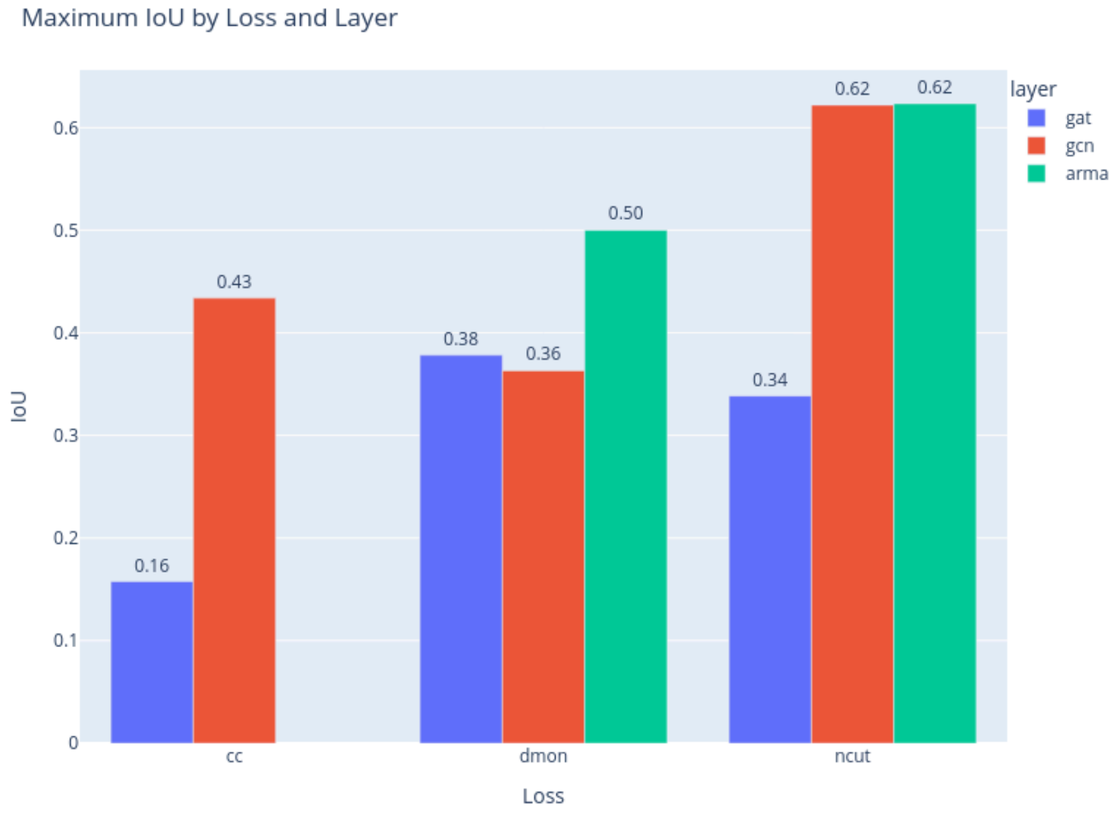
Results : ISIC 2016



Experiments and Results

Results : ISIC 2016





Experiments and Results

Results : EMD6

Method	GNN Layer	IoU	Dice	Best Epoch
NCUT	GCN	0.642	0.723	50
NCUT	GAT	0.519	0.599	100
NCUT	ARMA	**0.649**	**0.731**	30
CC	GCN (NO Normalization)	0.543	0.636	40
CC	GAT	0.157	0.222	100

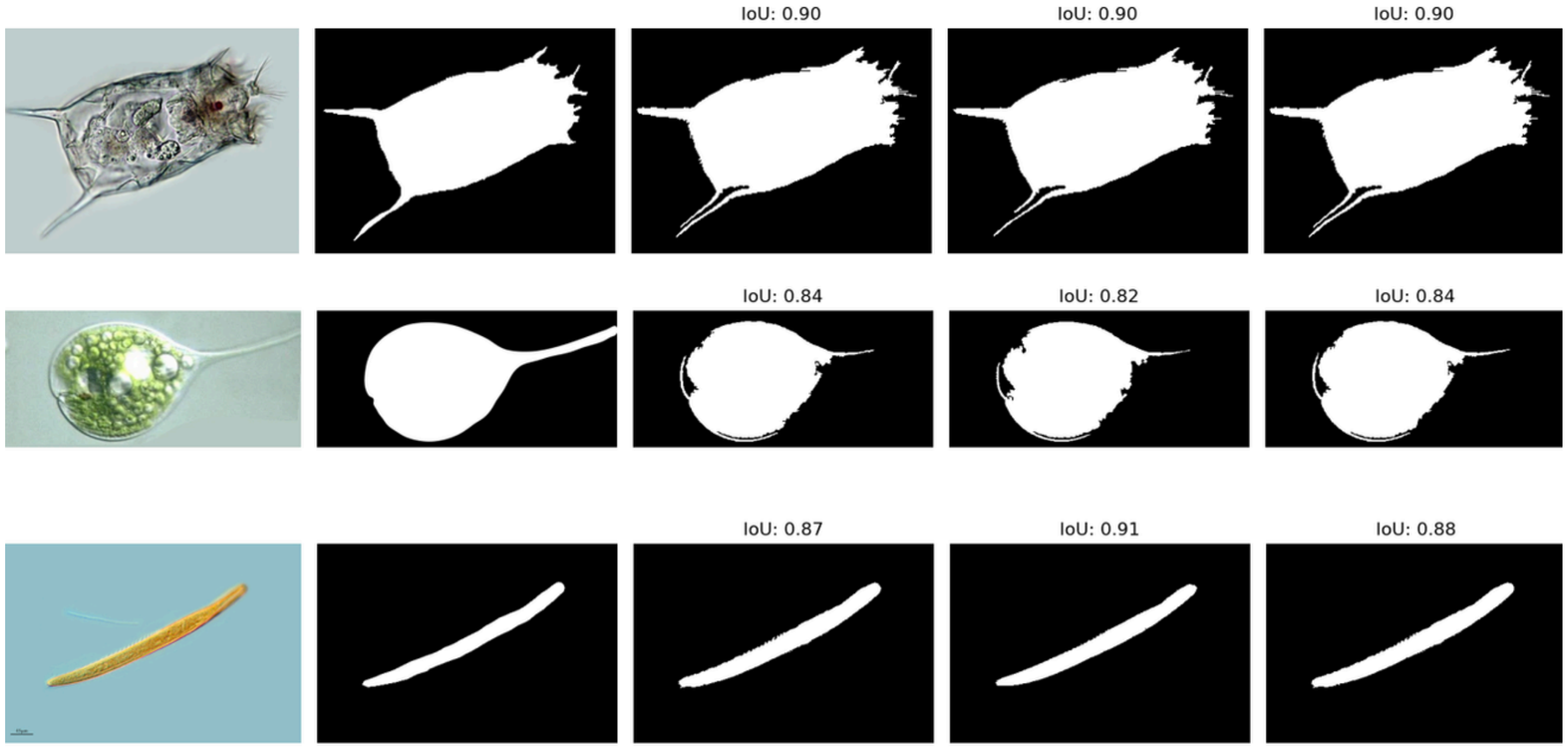
Experiments and Results

Results : EMD6

Method	GNN Layer	IoU	Dice	Best Epoch
DMON	GCN	0.269	0.483	70
DMON	GAT	0.273	0.498	90
DMON	ARMA	0.453	0.558	80

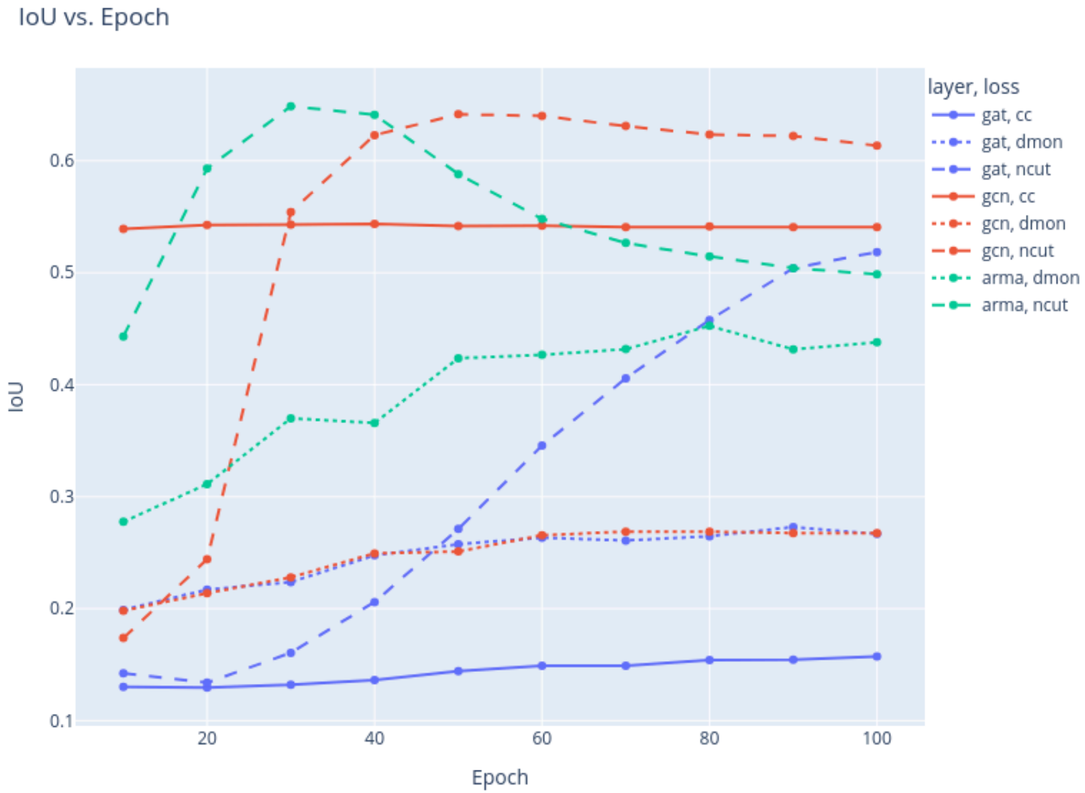
Experiments and Results

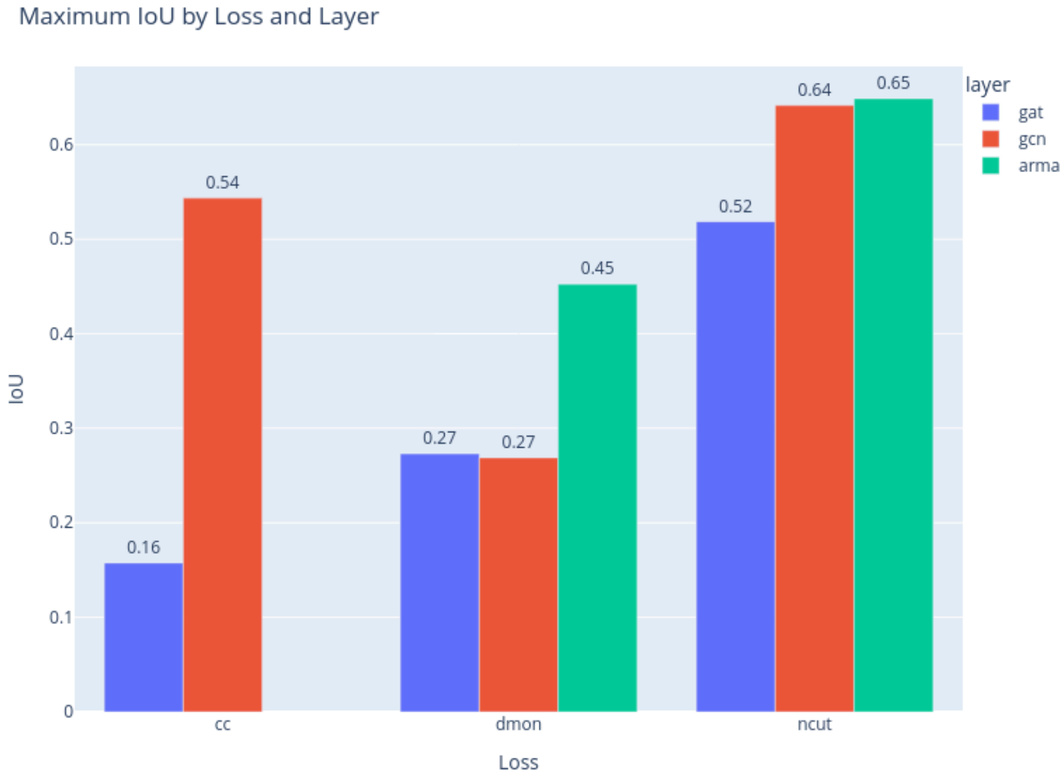
Results : EMD6



Experiments and Results

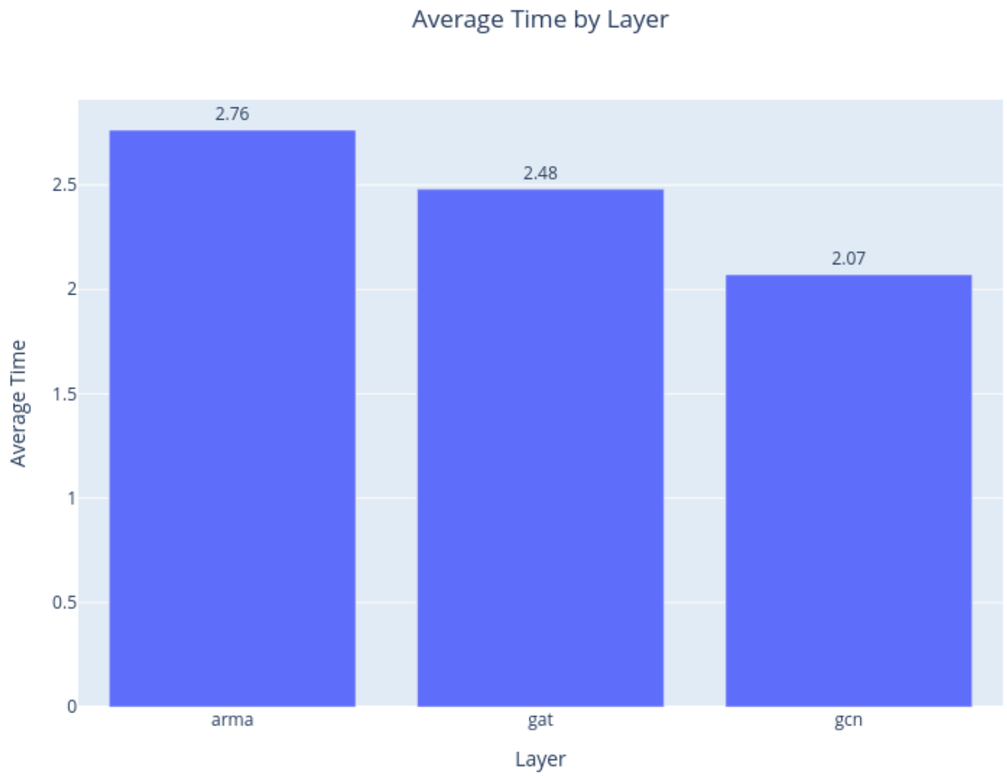
Results : EMD6





Experiments and Results

Average Convergence Time (100 Epochs)



Discussion

From the results and the graphs of both datasets
we can conclude that :

- **ARMA** layers perform the best across the different loss types followed by **GCN** layers with **GAT** coming last.
- In General we can say that NCUT loss provides better segmentation quality compared to the other loss types.
- **GCN** and **GAT** preform almost the same when used with **DMON** loss.
- **GCN** and **ARMA** results are similar when used with **NCUT** loss

Discussion

- We can see that the number of epochs is an important hyper-parameter in the case of **NCUT** and **DMON** as it can heavily affect the quality of segmentation and we can see that the more epochs we have doesn’t necessarily mean better segmentation.
- For the **CC** loss we notice that it is robust to the number of epochs as the graphs for both types of layers are almost stable.
- **ARMA** layers while offering better performance are more computationally expensive,however it should be noted that it requires less epochs than **GNCs** to achieve optimal performance when combined with **NCUT** Loss.

Conclusion

While often underperforming compared to supervised methods, research in unsupervised image segmentation techniques remains important for fields with limited labeled data. Progress is being made in this area. In this project, we explored how combining modern Graph Neural Networks with objective functions derived from classical graph theory metrics can unlock significant potential in the field of unsupervised image segmentation.

Thank You