Inverse function, from visualisation to graph toplogy

Abel Kahsay Gebreslassie
PhD student BKB team
University of Bordeaux

What we aim to achieve?

Given a graph visualization Y of data X with algorithm A

$$Y = A(X)$$

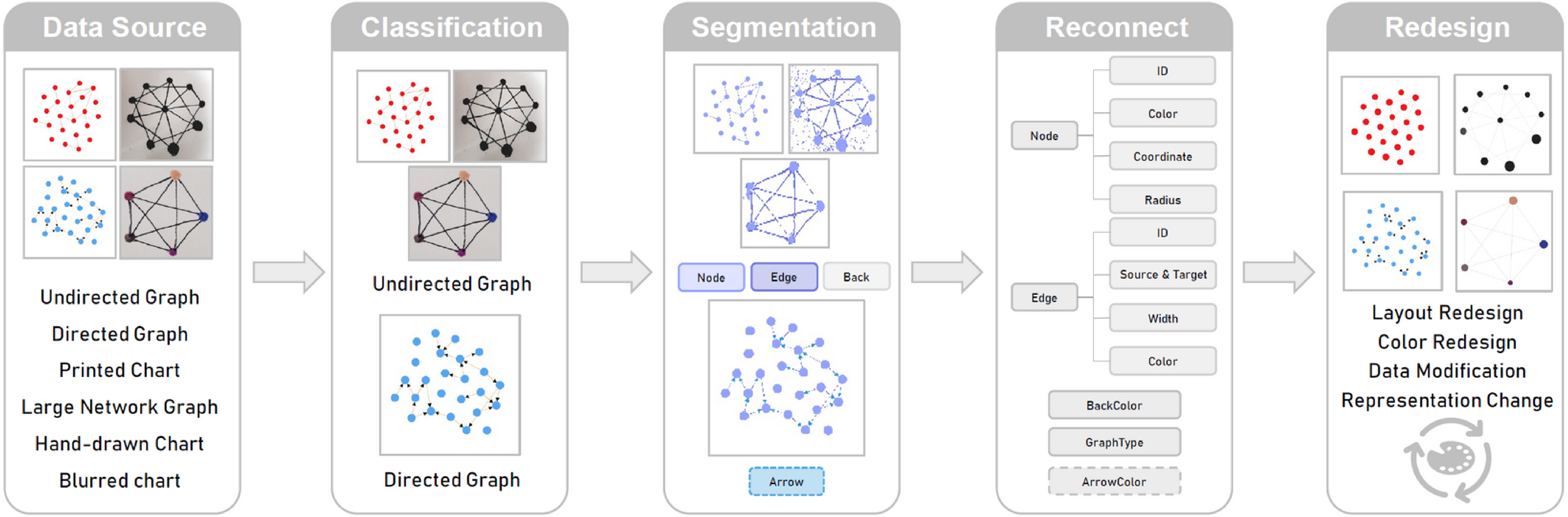
Automatically evaluate how "good" A is based on $\tilde{A}^{-1}$

$$X' = \tilde{A}^{-1}(Y)$$

- **utilize computer vision and deep learning to get inverse reconstruction**
- **Utilize metrics to evaluate reconstruction**

Vivid graph

- Vis to graph for graph redesign
- We are interested in the inverse reconstruction part. Hence reimplemented it



Song, S., Li, C., Sun, Y. and Wang, C., 2022. Vividgraph: Learning to extract and redesign network graphs from visualization images. IEEE Transactions on Visualization and Computer Graphics.

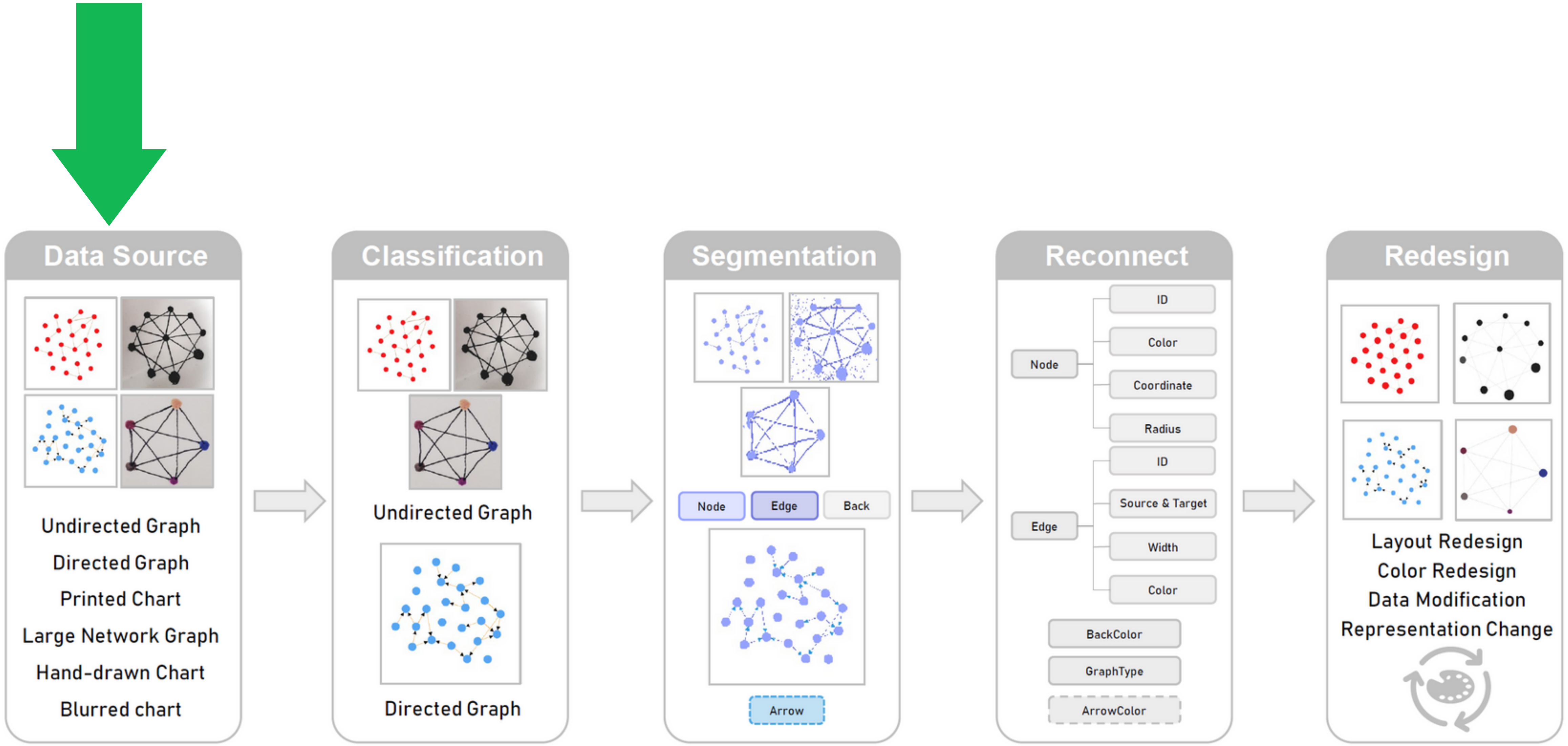
Data source

Graph assumptions

- Has no text
- Nodes are circular
- Nodes don't overlap
- Edges are straight

Data generation

- Directed and undirected graph
- With semantic labels



Song, S., Li, C., Sun, Y. and Wang, C., 2022. Vividgraph: Learning to extract and redesign network graphs from visualization images. IEEE Transactions on Visualization and Computer Graphics.

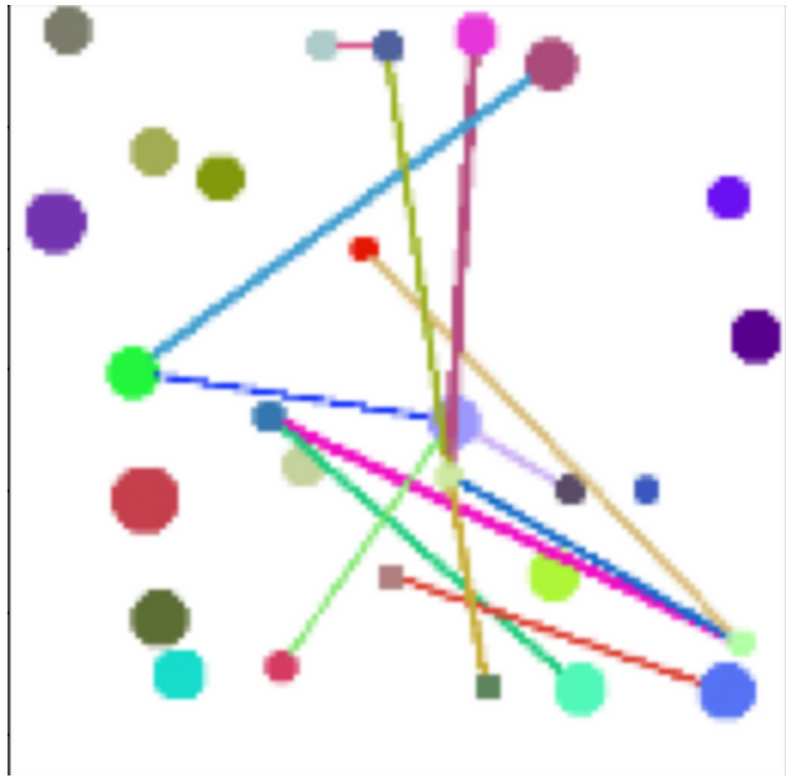
Data Generation

Data generation

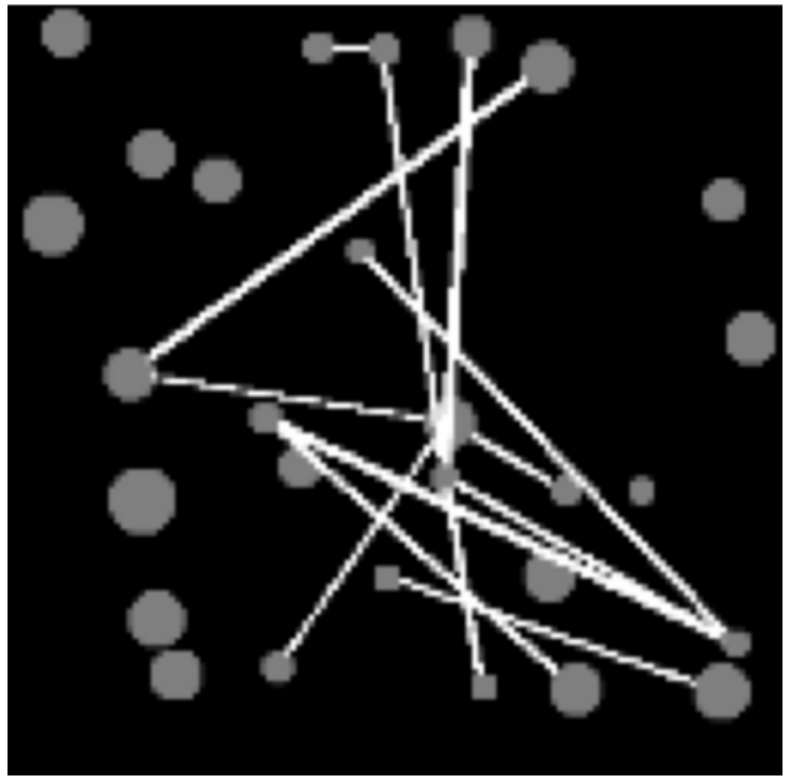
- Directed and undirected graph
- We only generated undirected graphs
- With semantic labels

image dimension: 320x320px
Number of nodes: 0-49
Node size(radius): 6-15px
edge width: 1-6 px

Node placement: random and non ovelapping
with existing nodes
Background/node/edge color: random
[(0,0,0)-(255,255,255)]



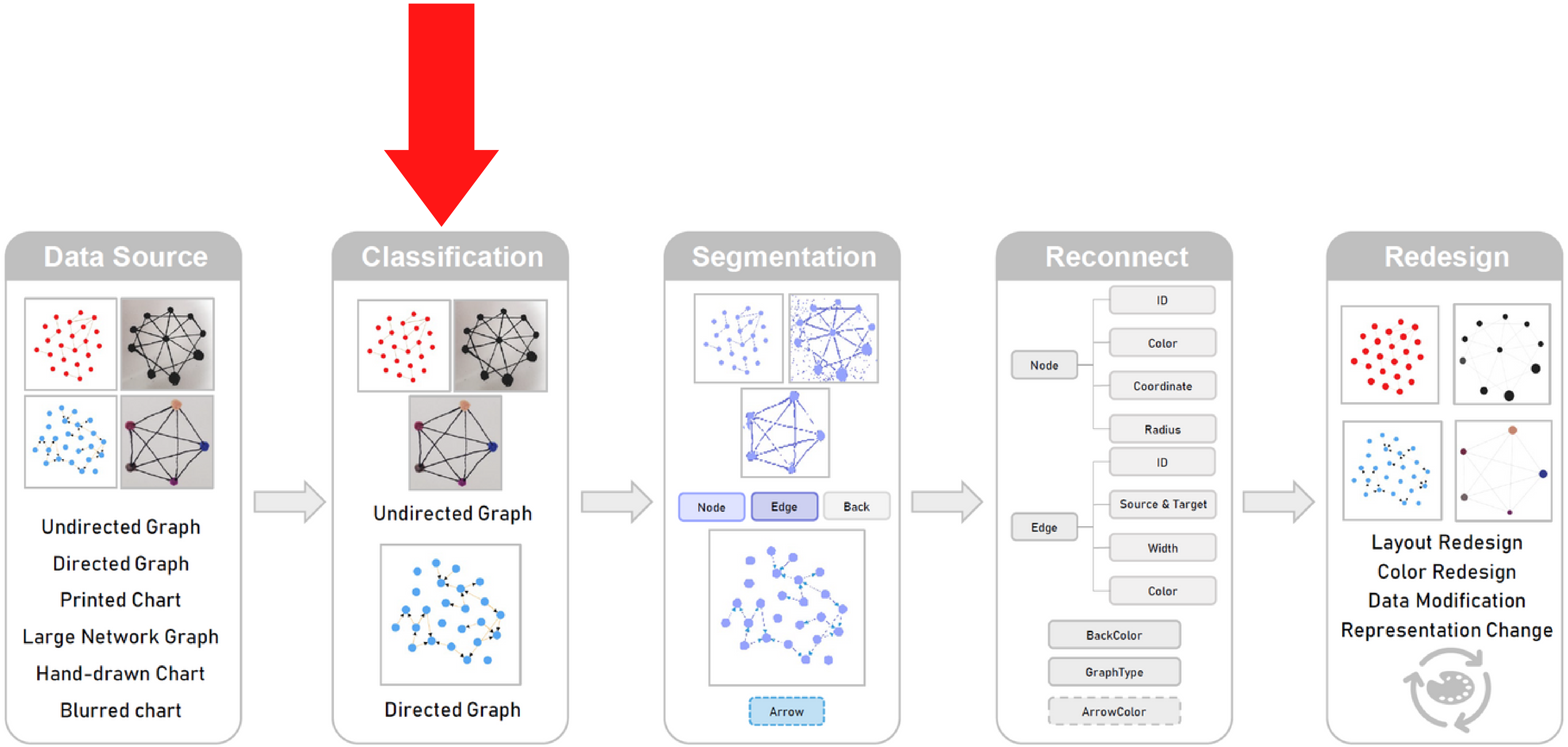
Sample generated graph



Semantic segmentation label for the graph

Classification

- Classify directed and undirected graphs
- Currently generating undirected graphs only, hence not included



Song, S., Li, C., Sun, Y. and Wang, C., 2022. Vividgraph: Learning to extract and redesign network graphs from visualization images. IEEE Transactions on Visualization and Computer Graphics.

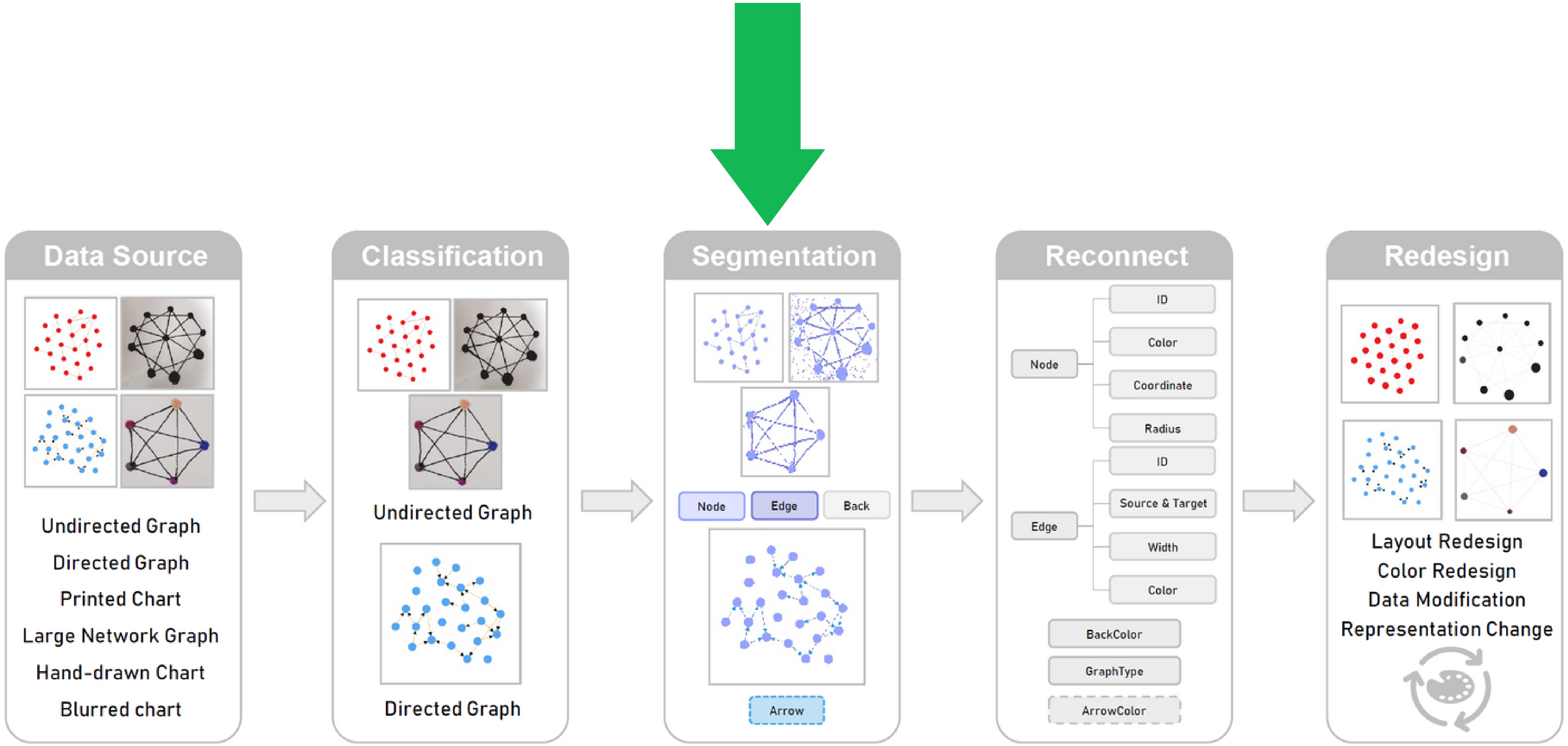
Segmentation

Model used

- U-Net with VGG-16 backbone [input_dim: 320x320]

Current implementation

- U-Net with MobileNetV2 backbones [input_dim: 128x128]



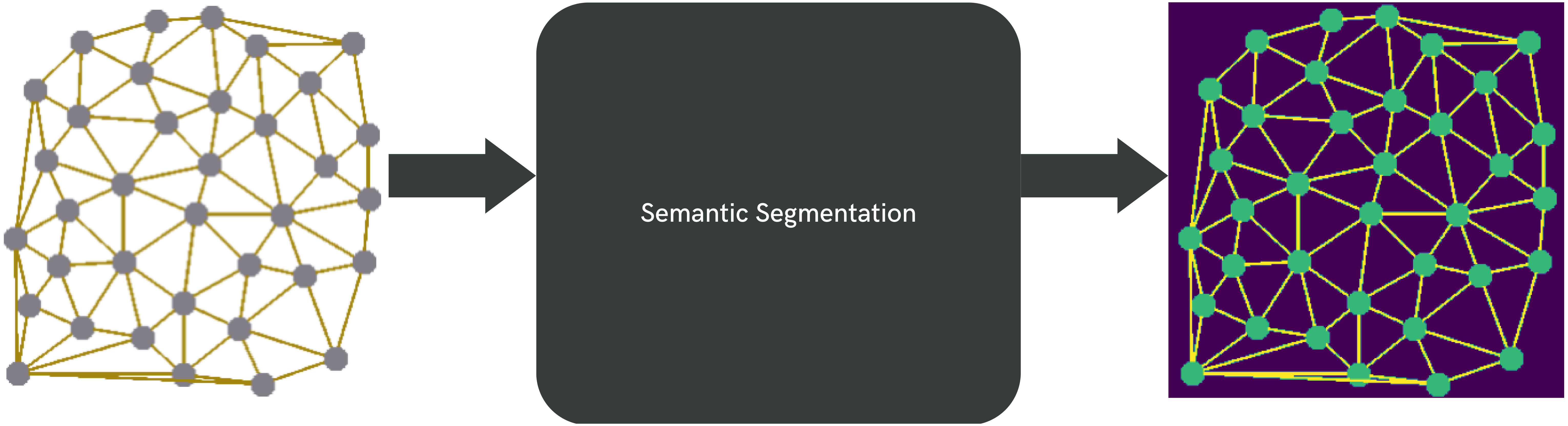
Song, S., Li, C., Sun, Y. and Wang, C., 2022. Vividgraph: Learning to extract and redesign network graphs from visualization images. IEEE Transactions on Visualization and Computer Graphics.

Semantic Segmentation

- Supervised learning task
- Densel labelling task
- every pixel assigned class label

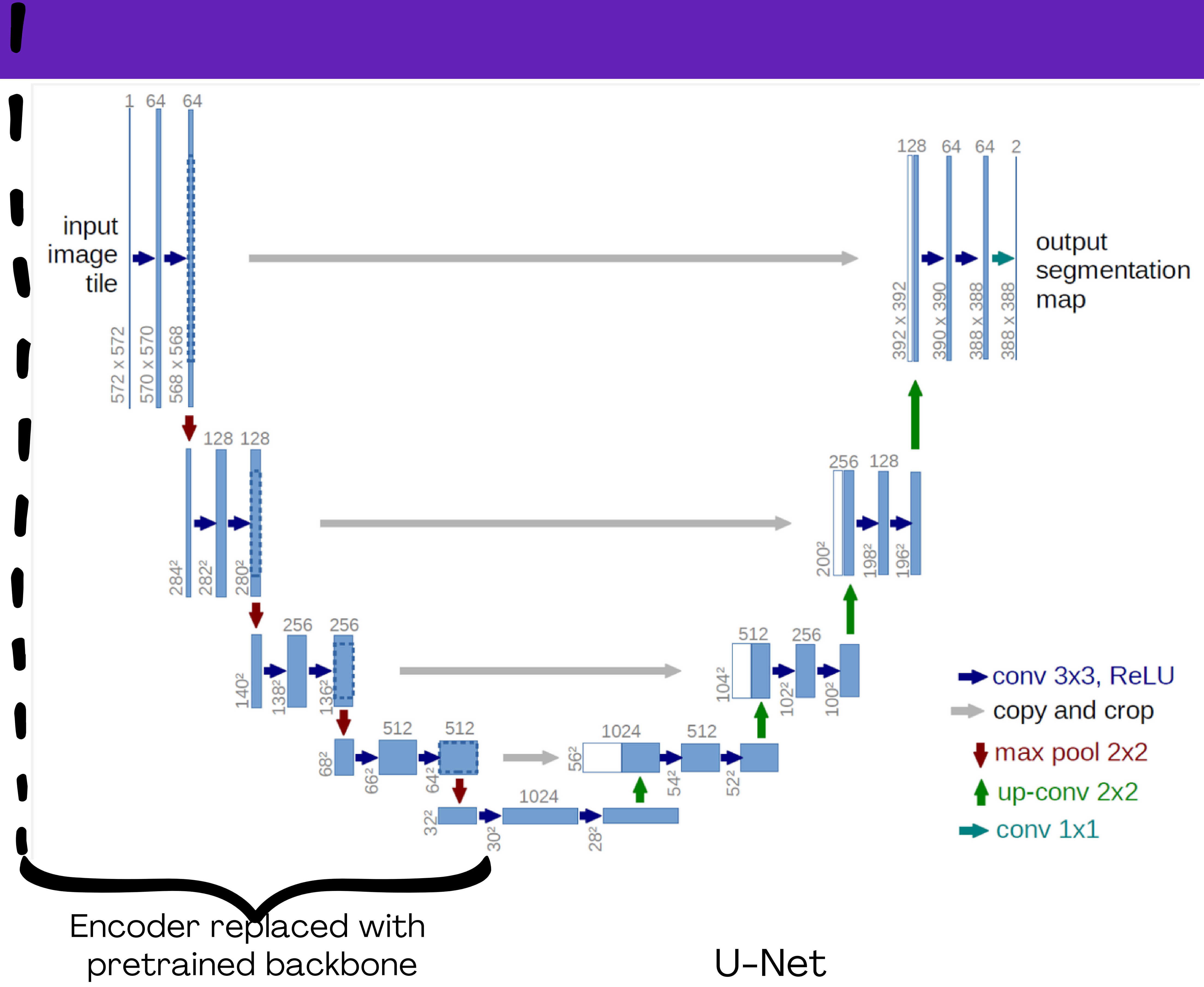
Semantic segmentation

Background: 0
node pixel: 1
edge pixel: 2



U-Net model

- U-Net with MobileNetV2 backbones [input_dim: 128x128]
- Pretrained backbone is a better feature extractor than training a new encoder
- MobileNet trained on ImageNet



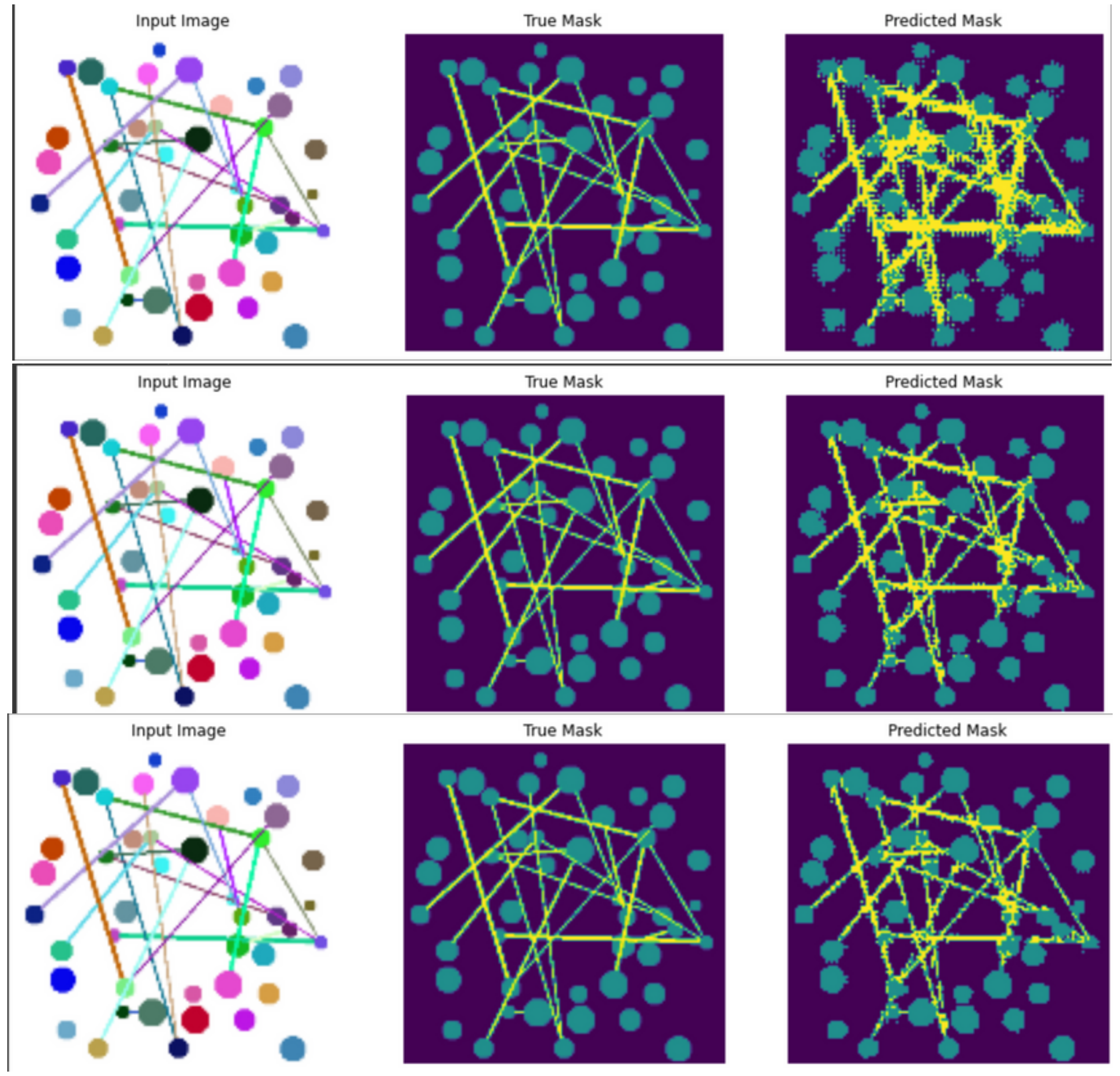
Model training

Epoch

1

2

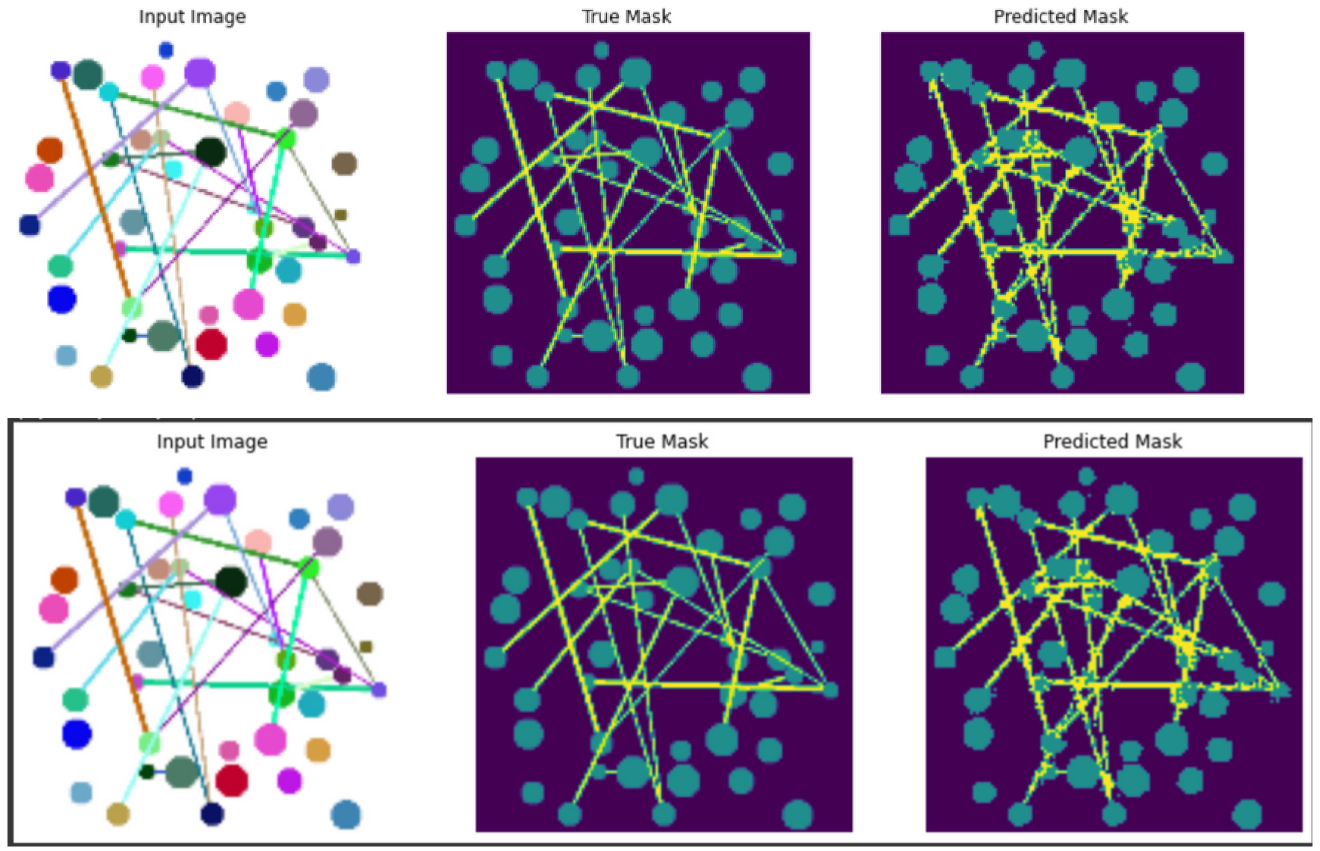
3



Epoch

4

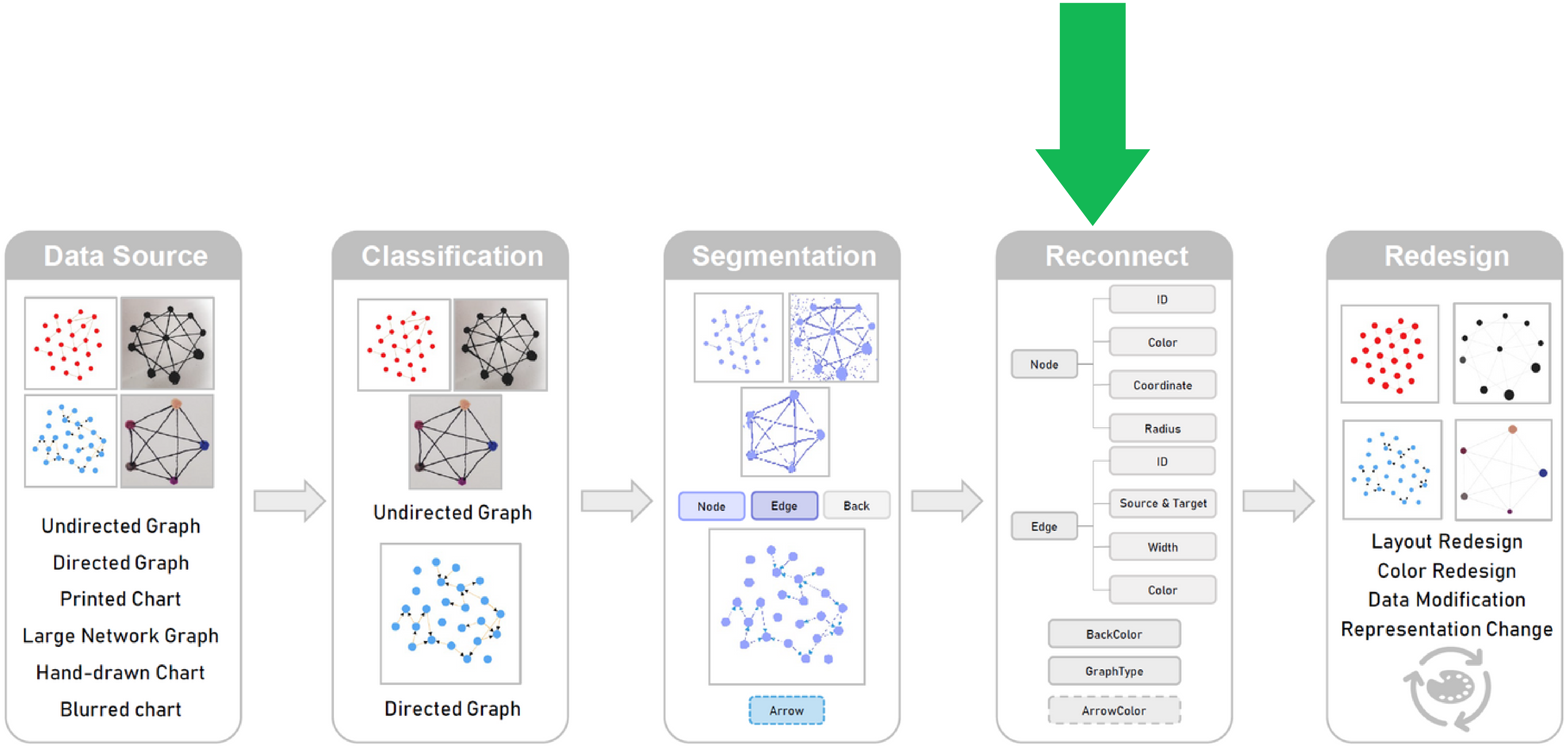
5



U-Net with MobileNetV2 backbone training

Reconnect

- Morphological operation to enhance segmentation results
- CC to find nodes
- For every nodes i and j draw a straight line connecting them and use the ratio of pixels classified as edge to determine if there is an edge between the nodes



Reconnect

- Morphological operation to enhance segmentation results
- CC to find nodes
- For every nodes i and j draw a straight line connecting them and use the ratio of pixels classified as edge to determine if there is an edge between the nodes

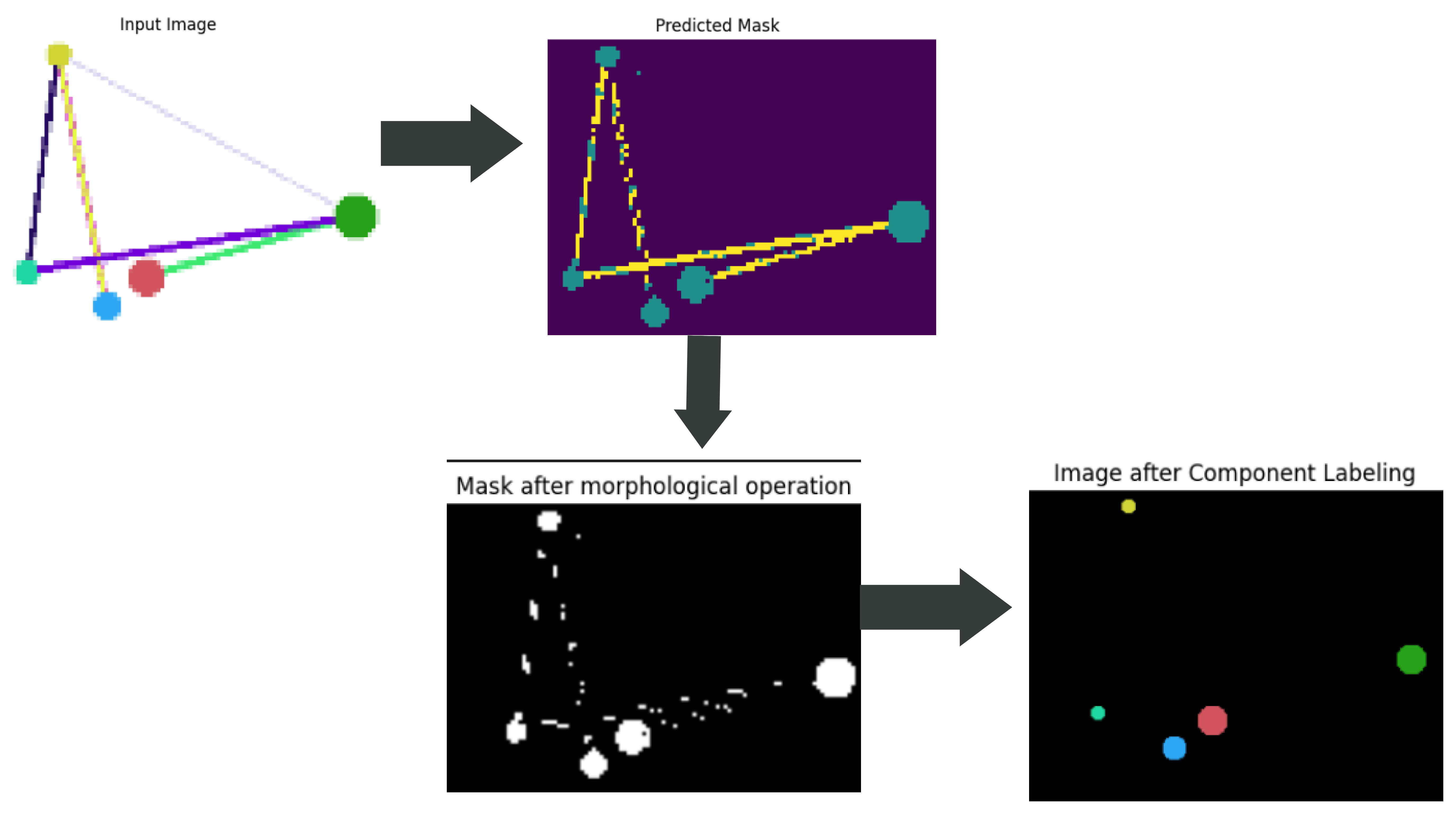
Algorithm 1 Node Reconnect Algorithm

Input: $\{C_{x,y} | x \in [0, W], y \in [0, H]\}$,
$\{Label_{x,y} | x \in [0, W], y \in [0, H]\}$
Output: O_i, R_i, $C^j_{i_1,i_2}$
Extract the CC of $Label_{x,y} = 1$
for all CC **do**
$k = \frac{1}{3} \times \sqrt{Area_i}$
Use (k, k) size kernel to perform morphological opening on CC
end for
Extract CC again
for all CC **do**
O_i =The coordinates of the center pixel of CC
$R_i = \frac{1}{2} \times \sqrt{Area_i}$
end for

Song, S., Li, C., Sun, Y. and Wang, C., 2022. Vividgraph: Learning to extract and redesign network graphs from visualization images. IEEE Transactions on Visualization and Computer Graphics.

Reconnect

- Morphological operation to enhance segmentation results
- CC to find nodes
- For every nodes i and j draw a straight line connecting them and use the ratio of pixels classified as edge to determine if there is an edge between the nodes



Reconnect

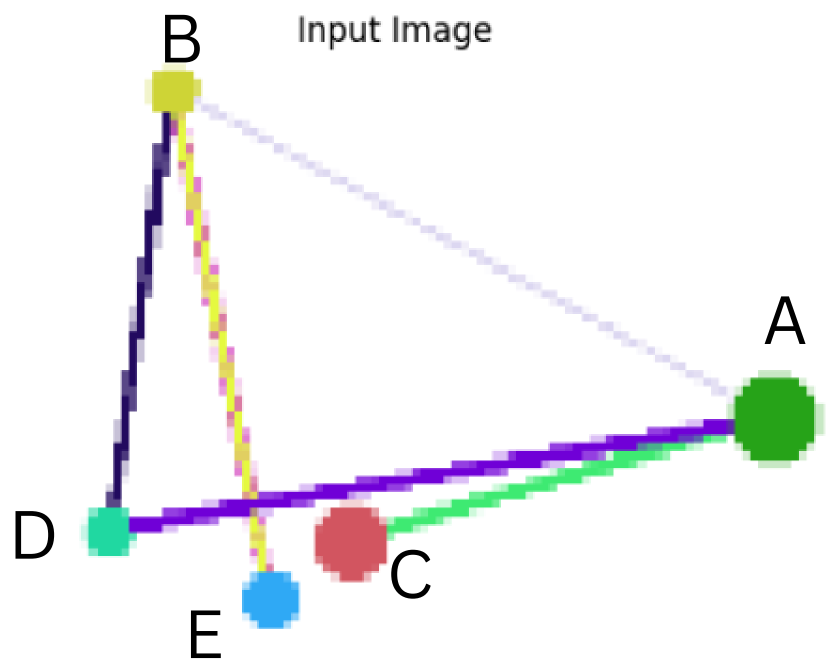
- Predicted colors for nodes edges and background are average color of pixels segmented as such
- Edge width estimation [didn't understand it well]

for each Node i_1 and Node i_2,$i_1 \neq i_2$ **do**
    Draw an line connecting Node i_1 and Node i_2
    Check $A = \{(x, y)|Label_{x,y} = 2, (x, y) \in line\}$
    Perform dilation operation
    Set $\gamma \propto length_{line}$
    if $|A| > \gamma$ **then**
        Node i_1 and Node i_2 are connected
        $C^j_{i_1,i_2} = \bar{C}_{x,y}, (x, y) \in A$
    end if
end for
return O_i, R_i, $C^j_{i_1,i_2}$;

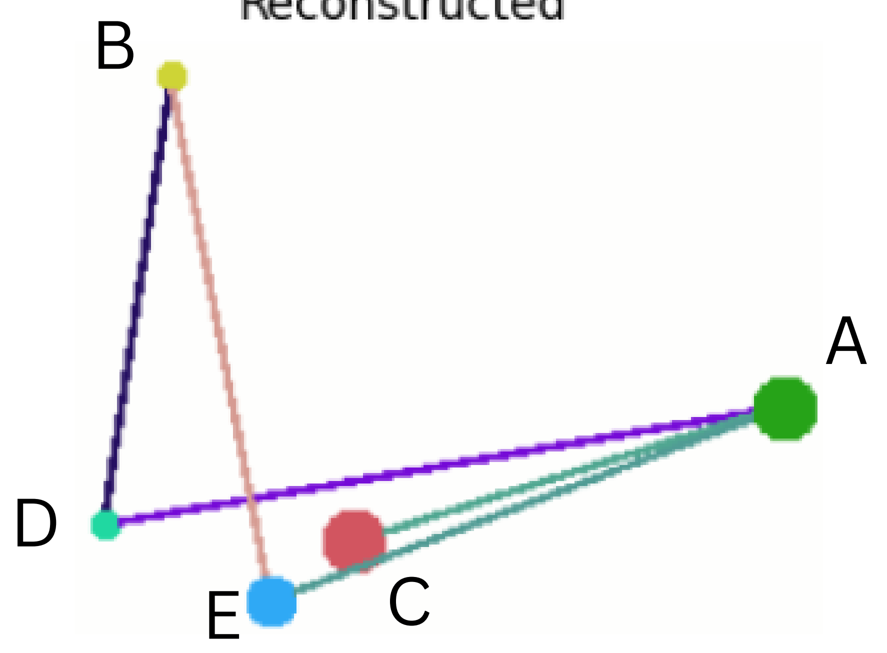
Song, S., Li, C., Sun, Y. and Wang, C., 2022. Vividgraph: Learning to extract and redesign network graphs from visualization images. IEEE Transactions on Visualization and Computer Graphics.

Reconnect

- Map reconstructed nodes to original nodes based on distance
- Reconstruction might have equal/less/more number of nodes compared to original graph
- Compare the two matrix based on similarity (distance between frobinus norms)

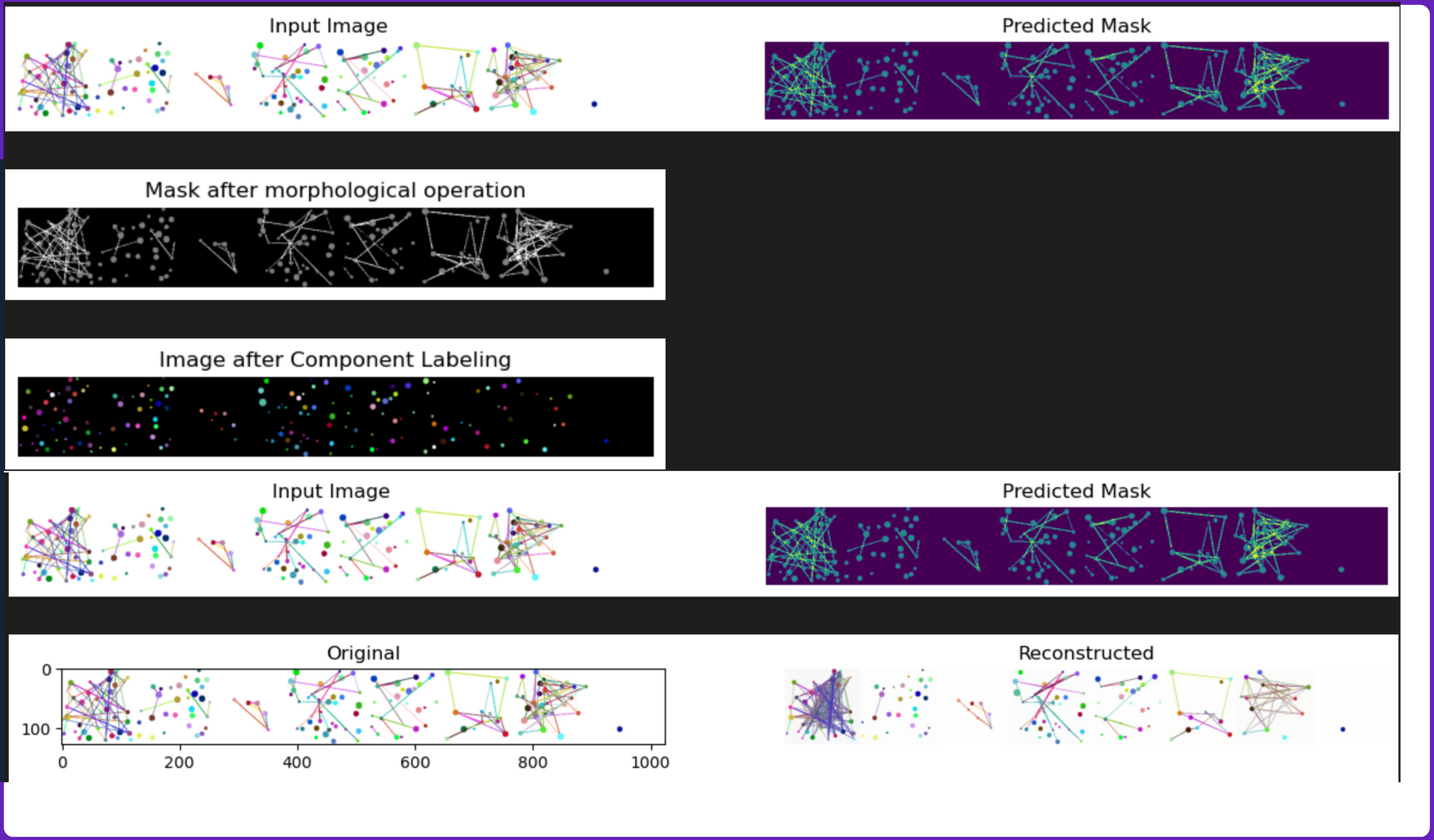


	A	B	C	D	E
A	3	-1	-1	-1	0
B	-1	3	0	-1	-1
C	-1	0	1	0	0
D	-1	-1	0	2	0
E	0	-1	0	0	1



	A	B	C	D	E
A	3	0	-1	-1	-1
B	0	2	0	-1	-1
C	-1	0	1	0	0
D	-1	-1	0	2	0
E	-1	-1	0	0	2

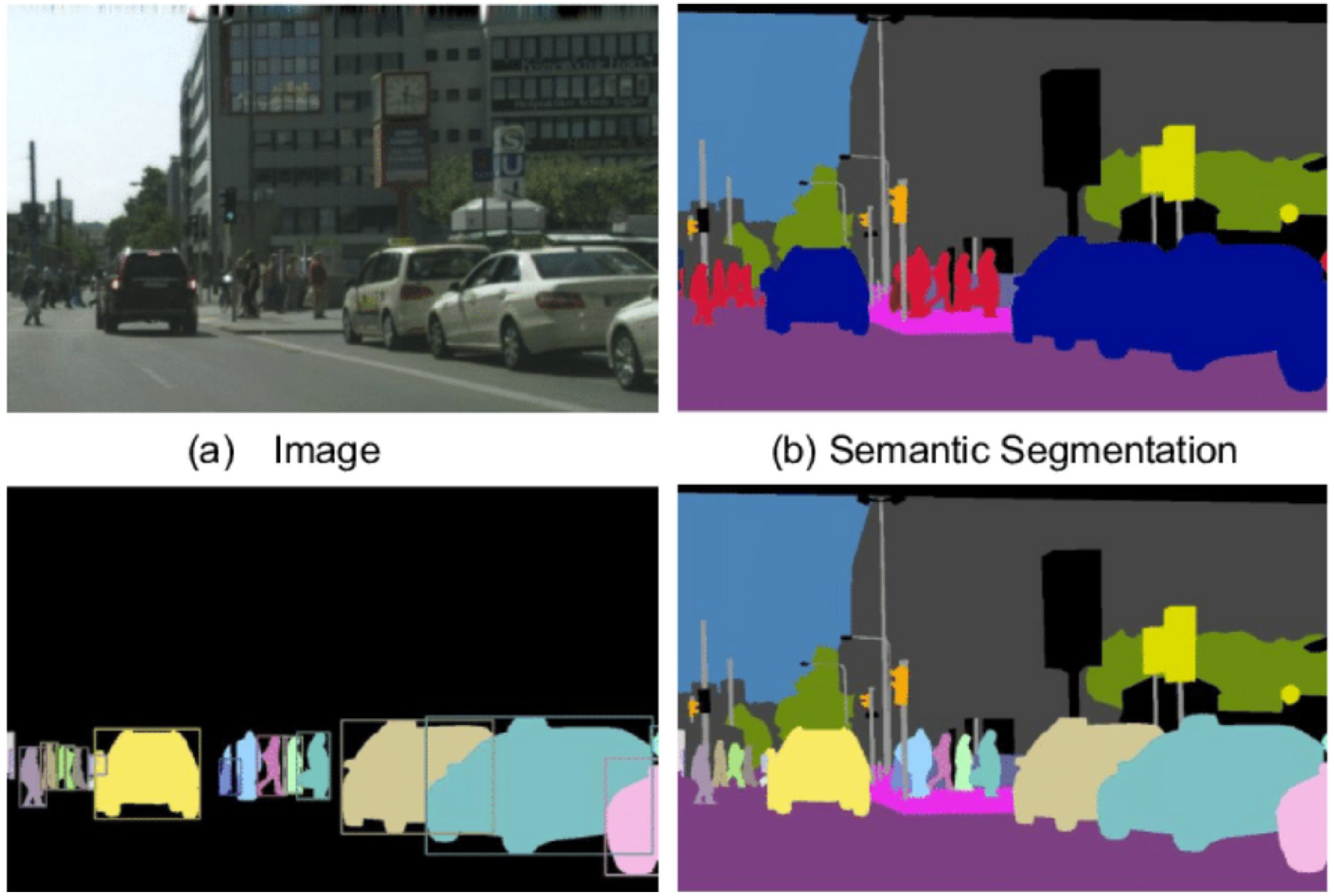
Reconstruction result on batch



Panoptic segmentation

Panoptic segmentation

- In addition to semantic label generate instance id label
- Segmentation result includes to which instance pixel belongs



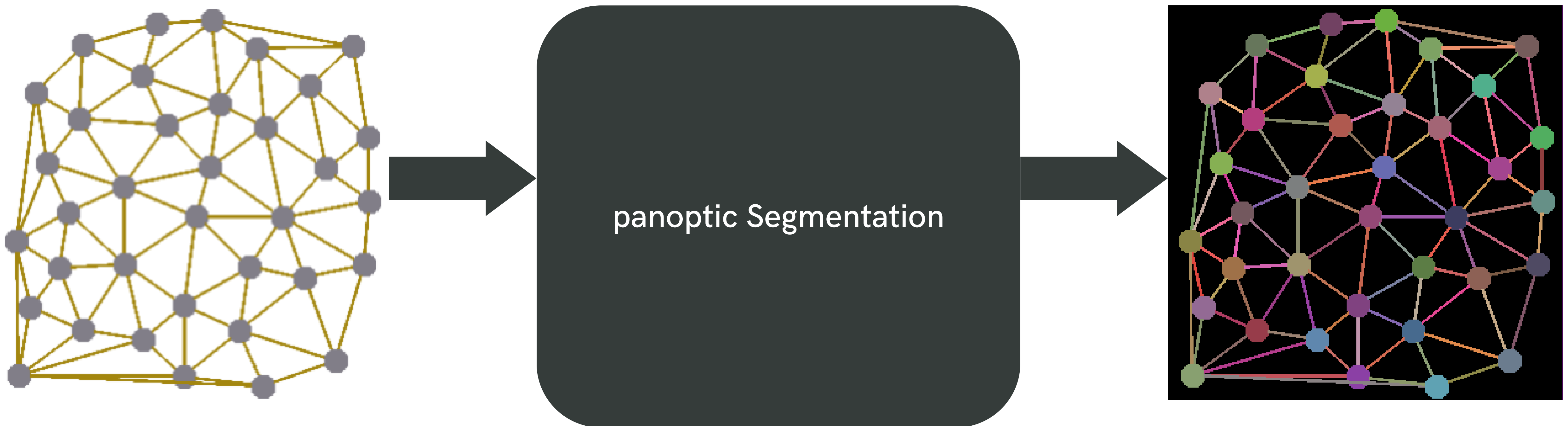
https://www.researchgate.net/publication/342409316/figure/fig5/AS:905798087630850@1592970501809/b-semantic-segmentation-c-instance-segmentation-and-d-panoptic-segmentation-for.png

Panoptic segmentation

Why use panoptic segmentation instead

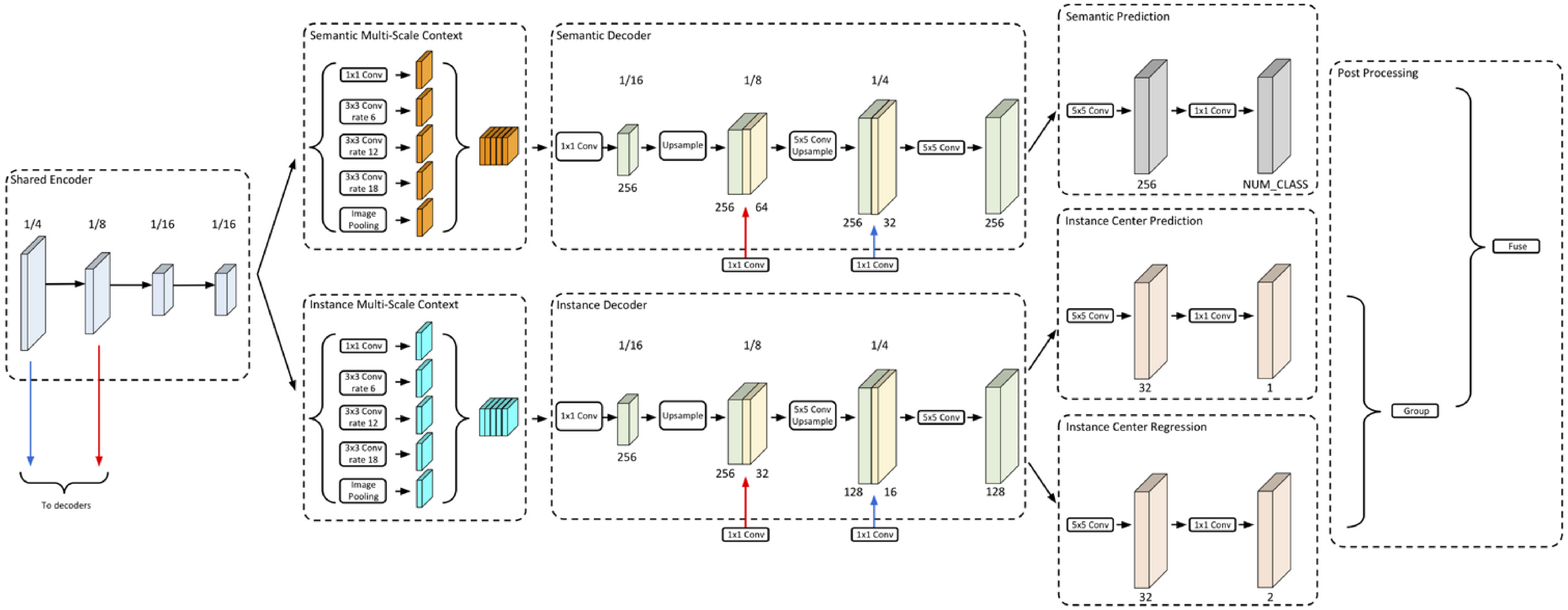
- To better resolve challenges with CC when nodes are close to each other
- To extract edges without having to go through all possible edges, which is O(n^2)

Background: 0
node pixel: [1*LD, floor(I_id/256), I_id%256]
edge pixel: [2*LD, floor(I_id/256), I_id%256]



Panoptic deeplab

- Shared backbone encoder (ResNet50 in our case)
- Dual decoder module
- Instance segmentation module is class agnostic
    - Center of mass to represent objests
    - instance center regression to predict instance id

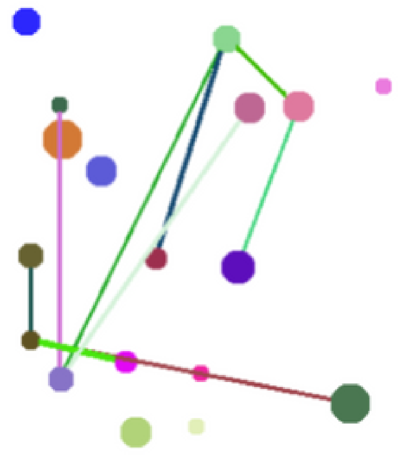


Cheng, B., Collins, M.D., Zhu, Y., Liu, T., Huang, T.S., Adam, H. and Chen, L.C., 2020. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 12475-12485).

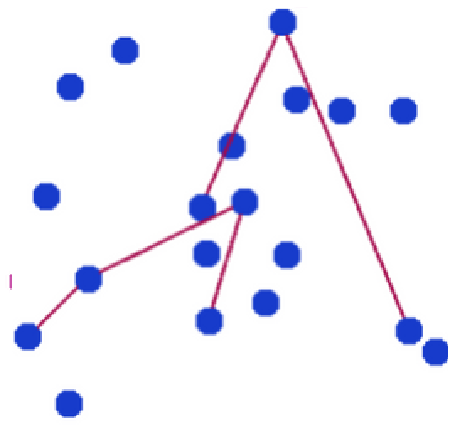
Generated datasets

- Generated different panoptic graph datasets
- 15,000 train images
- 4000 validation images
- 1000 test images

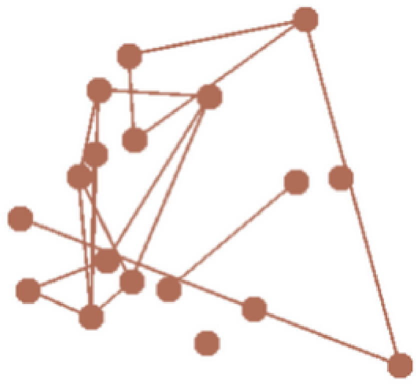
Random node and edge color



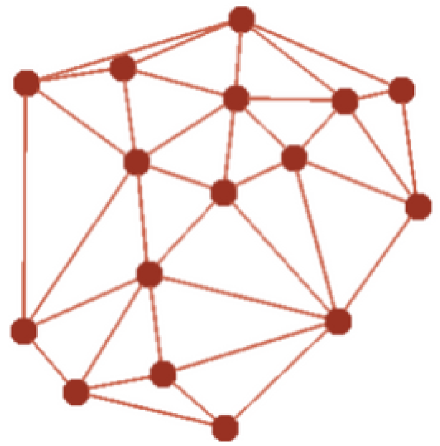
Color A for nodes and color B for edges



Single color for whole graph



delaunay triangulation planar graph

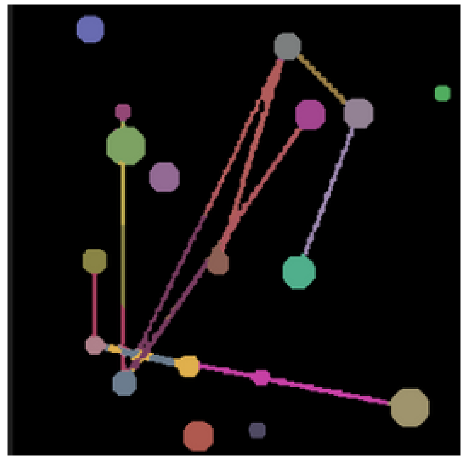


Panoptic segmentation results

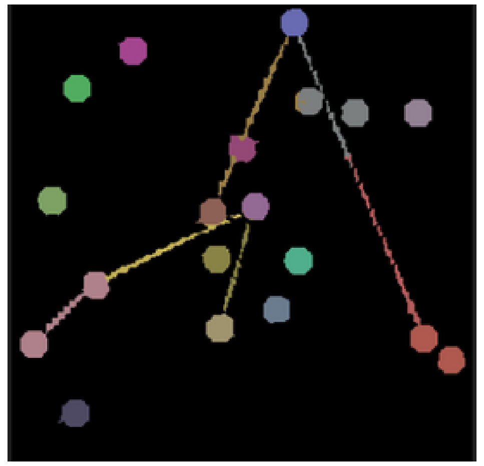
Challenges

- Instance id of node spills to edges
- Some edges are assigned multiple instance ids

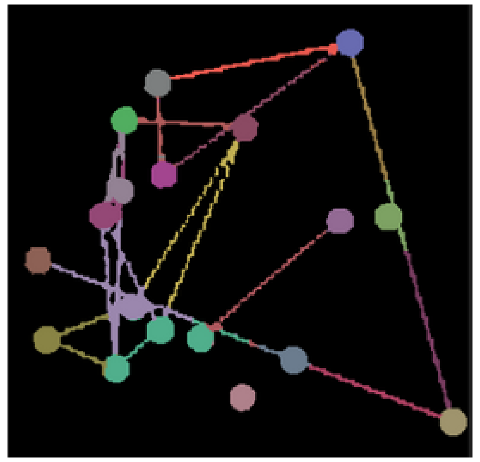
Random node and edge color



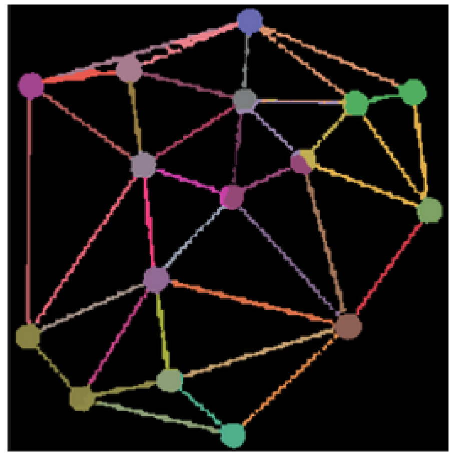
Color A for nodes and color B for edges



Single color for whole graph



delaunay triangulation planar graph



Panoptic segmentation results

LIkely due to

- Distance based instance id regression (edges are often long)
- class agnostic instance segmentation

Ways to explore

- Two pass through the network (masking nodes and edges alternately on the second pass)
- Change instance segmentation method (with YOLO)