Good Morning all, I'm Xu Ma, a second-year Ph.D. student from Northeastern University. Here, we will present our recent work, Towards Layer-wise Image Vectorization, collaborating with UIUC, Adobe, and Picsart AI Research.

Generally, given an input raster image, image vectorization requires converting the raster image to a vector graphics format—for example, scalable vector graphics or SVGs in short. There are some advantages of converting traditional raster images to SVGs. Compared to raster image, vector graph always enjoys smaller file size and more important, unlimited resizing without degradation of image quality. For example, from this image, if we increase the resolution by a factor of 5, the local region of the raster image gets unclear, while the vector graph still has good quality.

Two outstanding works in this field are DiffVG and Im2Vec. DiffVG considers an optimization-based method to optimize the randomly initialized bezier paths and use the L2 loss to guide the optimization. With the help of differentiable rendering, DiffVG can easily optimize the paths and generate an SVG. In Im2Vec, authors introduce a new neural network that can generate complex vector graphics with varying topologies and requires no input SVG guidance. While these two methods show outstanding achievements in image vectorization, they are not without drawbacks.

For the Im2Vec method, it cannot generalize well to other domain images since it is a learning-based method and is trained on a small dataset. The left image shows the results of applying Im2Vec to a new image which is not included in the training set. Clearly, the vectorization and reconstruction results are not satisfying.

For the famous DiffVG method, while it can generalize well to all images, DiffVG still lacks a layer-wise representation. This would introduce lots of unnecessary paths, which are meaningless, just as the right figure shows. The reconstruction result is good, but many small partial components combine it.

Considering the advantages and drawbacks of the above methods, we are motivated to consider the following objectives. First, we should have a better exploration of the layer-wise topology and fundamental semantics. Also, generalizing well to out-of-distribution images is required. Last, using fewer paths is always good.

In this work, we propose Layer-wise Image Vectorization, namely LIVE, to convert raster images to SVGs and maintain their image topology at the same time.

The figure shows the results: vectorized images should be explicitly vectorized in a layer-wise fashion, that is, one component by one component. For example, the smiling face can be divided into face, eyes, and other components. Each component is one bezier path.

Here we show the algorithm of our LIVE. It is simple and straightforward. We optimize the image one layer by one layer. In each layer, we initialize new paths according to our component-wise path initialization. Then we combine them with previous paths and render into images. We employ two novel losses, the Unsigned Distance guided Focal loss and self-interaction loss, to guide the optimization. Combining all layers, we will give a final SVG file in the end.

Three contributions make our LIVE framework to be possible. First, component-wise path initialization identifies the most suitable initial location based on the color and size of each connected area. Unsigned Distance guided Focal loss preserves the color of the target shape. Also, a Xing loss is introduced to mitigate the self-interaction problem.

LIVE shows great results for image vectorization. Compared with DiffVG and Im2Vec, LIVE can achieve perfect results using only four paths, and more paths would not degrade the performance.

Also, LIVE explicitly vectorized each visual concept without any redundancy for simple images, like emojis.

For complex images like photos and natural images, the topological clues are relatively hard to model. However, LIVE still exhibits a gratifying ability of the coarse-to-fine learning style.

Thanks for your listening. Please let me know if you have any questions or concerns.