Extracting Binary Trees from Hand-drawn Images

Chami Lamelas Ryan Polhemus

November 15, 2022

1 Introduction

For our project, we will be building a model for constructing the binary trees that correspond to hand drawn trees in images. One application of such a model would be to automatically grade questions that ask a student to draw a binary tree. For example, suppose a student was asked to draw an example of a binary search tree. The model would read in the image the student submits and produce a unique mathematical representation of the tree. This representation could be passed to another program that would determine if the tree is indeed a binary search tree. The problem we will be considering is simplified hence our particular model would not be able to be used in this capacity. However, we believe it serves as a potential proof of concept.

To make both the data collection and model construction processes easier, we will be considering drawings that are a subset of possible binary trees. We will only be considering drawings of complete binary trees where the values are zero through nine placed in a strictly increasing level order traversal. In other words, we will only consider binary minimum heaps where its values are unique and increasing from left to right, top to bottom. Furthermore, we assume that nodes are indicated by circled data values (digits) and edges are indicated by lines connecting nodes. Lastly, the image should only contain a single binary tree and nothing else. An example tree image is shown in Figure 1.

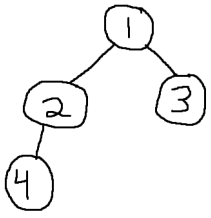

Figure 1: Sample Hand Drawn Binary Tree

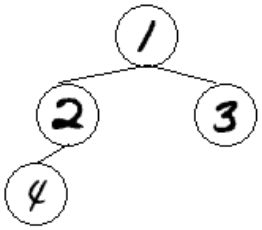

Figure 2: Sample Generated Binary Tree

As we are unable to find any existing dataset of hand drawings of binary trees, we will generate our own dataset of drawings. Drawings of node values will be taken from the MNIST dataset [1], hence the limitation on possible node values. Once MNIST drawings are placed onto the image,

circles and lines will be filled in. Figure 2 shows an example of a generated binary tree image. This is a particularly neat drawing of a tree, but as part of the generation process, we will add some variance in the drawings to make it more like a human drawing. This means we will vary aspects of the image such as the size of the node circles and edge line angles and length.

For our model, we will restrict the number of nodes in a tree image to be at most some N. This allows us to uniquely represent one of the above described trees with a $N(N-1)/2$ element binary vector v. To explain how this vector is constructed, consider a simple case where $N = 3$. v_1 indicates whether there exists an edge from a node with value one to a node with value two. v_2 indicates whether an edge exists from a node with value one to one with value three. v_3 indicates whether an edge exists from a node with value two to one with value three. In general we have an entry in v for each potential directed edge between nodes in the tree. Since we have ten possible node values, the outputs generated as part of our dataset will be length forty five binary vectors, which means our problem is a multiclass binary classification problem.

Initially, we planned to design this model as a combination of object detection, image classification, and modified nearest neighbor approaches. That is, we would first use object detection to distinguish between nodes and edges in the image. As described in chapter 14 of Dive into Deep Learning [5], Faster R-CNN and Mask R-CNN are popular choices. The second step would be to use an image classification method to classify bounding boxes identified as nodes to be one of the digits. As MNIST is a well studied dataset, a variety of CNN-based architectures would be appropriate here such as one of the ResNets [5]. The third step would need to match edges with the nodes that connect them. This would involve some kind of nearest neighbor strategy to do the matching.

The process above would require a mixture of methods as well as some degree of engineering in the nearest neighbor stage. After discussion with Professor Liu, we decided to go with an end-to-end model. The transformer model is particularly of interest due to its ability to encode images of varying objects and gather context from throughout the image. Given that an image of a binary tree is difficult to think of as a fixed length input, having a model that learns a fixed length encoding is beneficial. We hope that the encoder component of a transformer model will be able to achieve this as transformers have become increasingly popular in computer vision [3]. As described above, our problem is binary multiclass classification, we can use binary cross entropy loss to train our model to learn to extract the binary tree in a given image.

A transformer model will predict a vector of forty five probabilities for each image. We initially planned to use a simple threshold strategy to use the maximum class probability to assign a class (zero or one) for each of the forty five elements of the desired tree representation vector. However, Professor Liu proposed a three stage strategy that would reduce the number of incorrect predictions made by the model. In the first stage, we pass an image into a transformer to obtain a forty five element probability vector. Next, we pass the image into a second transformer that outputs a ten element binary vector w where w_i indicates if the the digit $i-1$ is present as a node in the image. We expect this model to perform quite well again due to the digit images being drawn from the MNIST dataset. In the third stage, we construct a complete acyclic undirected graph from all the nodes that are determined to be in the image from the second stage. The weights of the graph are set to be one minus the probabilities calculated in the first stage. Then, we run Prim's algorithm to determine the minimum spanning tree of the graph. The resulting tree is the prediction for the image. This helps to reduce false positives in edge detection.

Given that we do not know of any existing work on this problem, we will evaluate the success of

the project by computing the accuracy of our model prediction process described above on held out validation and test sets of generated images. We hope to achieve good results, especially on node value identification. We do not expect any tree structures to be more difficult to learn than others. We will try to maximize the accuracy of our model on the validation set before a final evaluation.

2 Related Work

We were unable to find any existing work on problems related to extracting binary trees or any similarly drawn data structures from images. Furthermore, based on the recent survey [3] by Khan et. al. on Transformers in vision, there is not a particular network that is suitable to this type of object detection task. For object detection, they discuss a variety of hybrid CNN and transformer architectures. However, we plan to use YOLOS [2] which is an end-to-end transformer that will directly predict class labels. Determining layout and structure of a scene is an active field in object detection, and we feel that we could adapt a simplified version of the layout codebook suggested in [4] to fit the use case of this project; the set of possible trees is small enough that an entry in the codebook for each layout is not out of the question.

References

[1] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[2] Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *Advances in Neural Information Processing Systems*, 34:26183–26197, 2021.

[3] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

[4] Tao Wang, Xuming He, Yuanzheng Cai, and Guobao Xiao. Learning a layout transfer network for context aware object detection. *IEEE Transactions on Intelligent Transportation Systems*, 21(10):4209–4224, 2020.

[5] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342,* 2021.