Introduction to Artificial Intelligence
MNIST and CIFAR10 with Boundary Trees

Jay Ricco, David Van Chu

August 5, 2017

1 Boundary Trees

Features are the attributes of an input, or example. With the MNIST data set, each example is composed of 784 pixels, each one of which are considered the features.

The Boundary Tree is trained by querying the existing tree using the example as an input, looking for the closest node (using a Euclidean distance as the distance function). When it finds the closest node, if the example is not the same as the closest node, it will add a new node to the tree containing the example.

Testing the Boundary Tree involves simply querying the existing tree using an example from the test set. The Boundary Tree determines the class of the example by the class associated with the closest node.

2 MNIST

Changing the maximum branching factor, k, played a large role in both accuracy as well as the speed of training and querying. As the branching factor grows, it takes longer to train and query, although the accuracy is higher, so it may be worth it depending on the use case. For example:

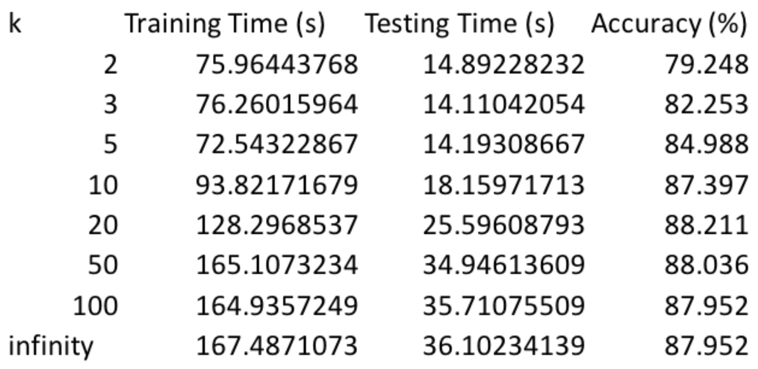
1. with branching factor = ∞, averaged over 10 runs
   - accuracy = 88%
   - training = 167.48 seconds
   - testing = 36.10 seconds
2. with branching factor = 5, averaged over 10 runs
   - accuracy = 85%
   - training = 72.54 seconds
   - testing = 14.19 seconds

After running the tests, we noticed that setting the branching factor to 5-10 seemed to be the best if both time and accuracy is a concern, as anything less than 5 will result in lower accuracy while taking about the same amount of time to train and test. With branching factors larger than 10, training and testing time grew quite a bit with only a small increase in accuracy.

Another interesting point is that with a larger branching factor, the training time itself varies by quite a bit, ranging from 140.47 seconds to 223.79 seconds, whereas tests ran with a lower branching factor resulted in consistent training times.

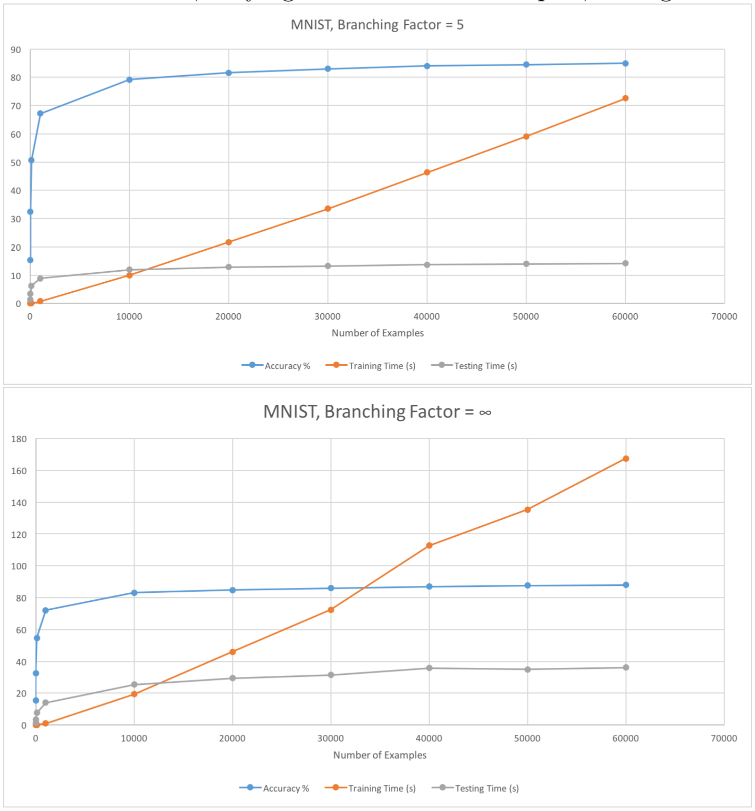
We also tests varying the number of examples we used to train the Boundary Tree. While accuracy and testing time plateaued, the time it took to train grew linearly with more examples.

Figure 1: MNIST dataset, varying the branching factor, averaged over 10 runs.



k	Training Time (s)	Testing Time (s)	Accuracy (%)
2	75.96443768	14.89228232	79.248
3	76.26015964	14.11042054	82.253
5	72.54322867	14.19308667	84.988
10	93.82171679	18.15971713	87.397
20	128.2968537	25.59608793	88.211
50	165.1073234	34.94613609	88.036
100	164.9357249	35.71075509	87.952
infinity	167.4871073	36.10234139	87.952

Figure 2: MNIST dataset, varying the number of examples, averaged over 10 runs.



3 CIFAR10

We see similar results when using Boundary Trees on the CIFAR10 dataset, although it never achieves over 27.6% accuracy. Using a smaller branching factor quickens the training time significantly compared to the MNIST dataset results.

1. with branching factor = ∞, averaged over 10 runs

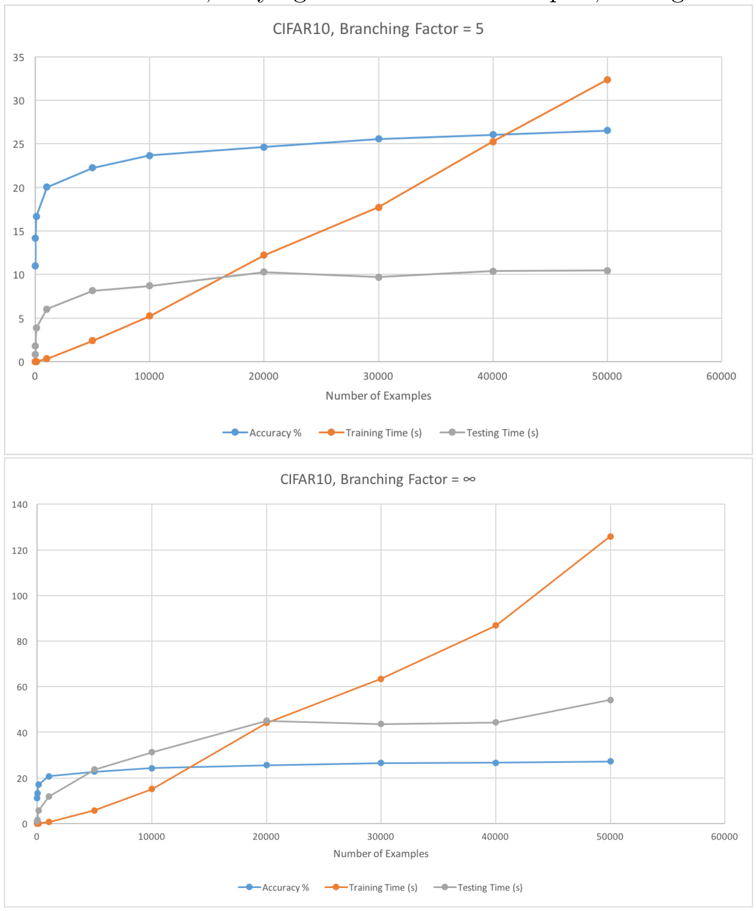
   - accuracy = 27.3%
   - training = 125.91 seconds

- testing = 52.22 seconds

2. with branching factor = 5, averaged over 10 runs

   - accuracy = 26.5%
   - training = 32.36 seconds
   - testing = 10.46 seconds

Figure 3: CIFAR10 dataset, varying the number of examples, averaged over 10 runs.


CIFAR10, Branching Factor = 5
Number of Examples
Accuracy %
Training Time (s)
Testing Time (s)
CIFAR10, Branching Factor = ∞
Number of Examples
Accuracy %
Training Time (s)
Testing Time (s)