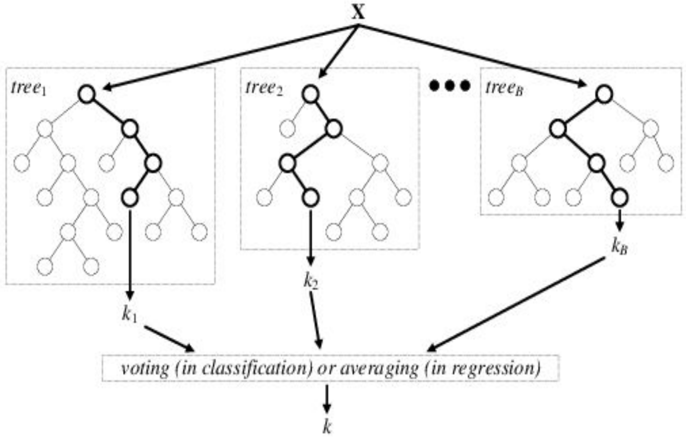
Neural Decision trees

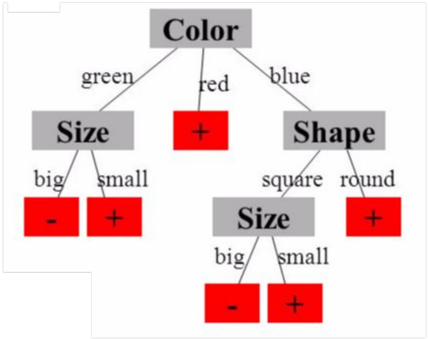
Decision trees & Random Forests

- The original idea by Tim Kam Ho to implement stochastic discrimination (1995)
- Ensemble of decision trees: Wait! Decision trees are deterministic though!
- Main oracle: Bagging
    - Bagging trees
    - Random subspace method : feature bagging
    - Variance reduced without affecting bias much
- Extra trees
- Widely used in practice:
    - Efficient human pose estimation from single depth images (Shotton, CVPR 2012)
    - Oriented Edge Forests for Boundary Detection, (Hallman, Fowlkes, CVPR 2015)



DTs: Training and testing

➔ For each tree in ensemble:
  - Select a subset of samples and subset of features
  - Create a tree with only 1 node
  - Until stopping condition is achieved:
    - Make a split according to the criterion
  - For each leaf node
    - Discrete: $p(c|v)$ $\quad \arg\max_c p(c|v)$
    - Continuous: mean or predefined func.



- Still, no use of differentiability. How can we convert DT learning to a differentiable one?

Pros and cons

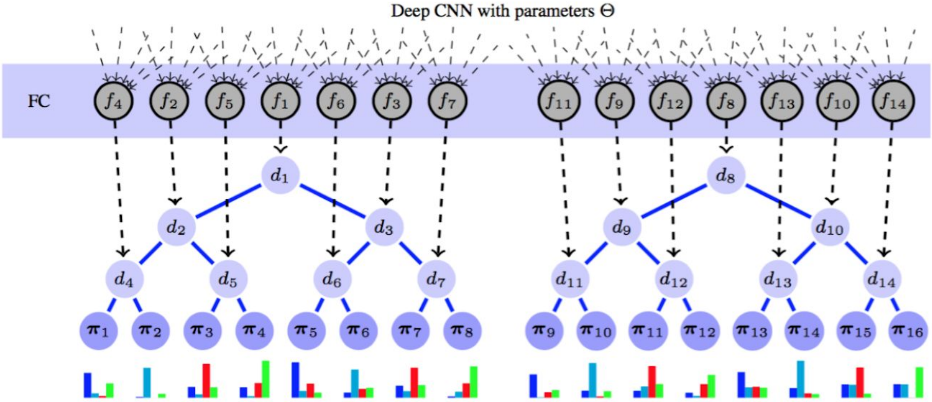
	Decision trees	Neural Networks
Easily interpretable	✓	✗
Model functions diversity	Only axis parallel splits	Arbitrary functions, Complex structures
Time complexity	Reasonably fast	Comparably slow, long training
Online learning	✗	✓
Model parameters	Only a few	Up to millions (hidden layers, number of units)
Layout	Deterministic splits	Differentiable, stochastic, back-propagation compatible

Neural Decision trees

- Instead of using weak learners as base classifiers, we can make use of the features learned by neural network.

Neural Decision trees

- The input data is fed to the neural network.
- The outputs of FC layer represent routing probabilities in each of the trees of the ensemble.
- The assignment is random



$$d_i(x) = g(f_j(x))$$

$$p(x, \pi_i) = \prod_{n \in \{d_1, \ldots, \pi_i\}} p_{route}(n)$$

Neural Decision trees

$$\mathbb{P}_T[y|\boldsymbol{x},\Theta,\boldsymbol{\pi}] = \sum_{\ell\in\mathcal{L}} \pi_{\ell y}\mu_\ell(\boldsymbol{x}|\Theta)$$

$$\mu_\ell(\boldsymbol{x}|\Theta) = \prod_{n\in\mathcal{N}} d_n(\boldsymbol{x};\Theta)^{\mathbb{1}_{\ell\swarrow n}}\bar{d}_n(\boldsymbol{x};\Theta)^{\mathbb{1}_{n\searrow\ell}}$$

$$d_n(\boldsymbol{x};\Theta) = \sigma(f_n(\boldsymbol{x};\Theta))$$

$$\mathbb{P}_{\mathcal{F}}[y|\boldsymbol{x}] = \frac{1}{\mathsf{k}}\sum_{h=1}^{\mathsf{k}} \mathbb{P}_{T_h}[y|\boldsymbol{x}]\,,$$

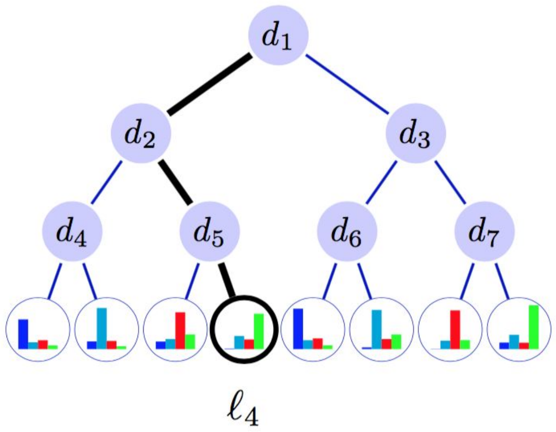


Figure 1. Each node $n \in \mathcal{N}$ of the tree performs routing decisions via function $d_n(\cdot)$ (we omit the parametrization Θ). The black path shows an exemplary routing of a sample $\boldsymbol{x}$ along a tree to reach leaf ℓ_4, which has probability $\mu_{\ell_4} = d_1(\boldsymbol{x})\bar{d}_2(\boldsymbol{x})\bar{d}_5(\boldsymbol{x})$.

Neural Decision trees

- The loss over decision nodes should be converted to a differentiable one

$$L(\Theta, \boldsymbol{\pi}; \boldsymbol{x}, y) = -\log(\mathbb{P}_T[y|\boldsymbol{x}, \Theta, \boldsymbol{\pi}])$$

- Leaf nodes outputs are $\pi_{\ell y}^{(t+1)} = \frac{1}{Z_\ell^{(t)}} \sum_{(\boldsymbol{x}, y') \in \mathcal{T}} \frac{\mathbb{1}_{y=y'} \, \pi_{\ell y}^{(t)} \mu_\ell(\boldsymbol{x}|\Theta)}{\mathbb{P}_T[y|\boldsymbol{x}, \Theta, \boldsymbol{\pi}^{(t)}]}$

Neural Decision trees training

Algorithm 1 Learning trees by back-propagation

Require: $\mathcal{T}$: training set, nEpochs

random initialization of Θ
for all $i \in \{1, \dots, \text{nEpochs}\}$ **do**
  Compute $\boldsymbol{\pi}$ by iterating (11)
  break $\mathcal{T}$ into a set of random mini-batches
  for all $\mathcal{B}$: mini-batch from $\mathcal{T}$ **do**
    Update $\boldsymbol{\Theta}$ by SGD step in (7)
  end for
end for

- 2-step optimization process

Characteristics of models

	Decision Forests	Neural Networks	NDT's
Easy to parallelize	✓	✗	✗
Feature learning	✗	✓	✓
GD applicable	✗	✓	✓
Loss	NP hard to grow optimal tree	not convex	not convex

Comparison and results

- The loss over decision nodes should be converted to a differentiable one

$$L(\Theta, \boldsymbol{\pi}; \boldsymbol{x}, y) = -\log(\mathbb{P}_T[y|\boldsymbol{x}, \Theta, \boldsymbol{\pi}])$$

- Leaf nodes outputs are $\pi_{\ell y}^{(t+1)} = \frac{1}{Z_\ell^{(t)}} \sum_{(\boldsymbol{x}, y') \in \mathcal{T}} \frac{\mathbb{1}_{y=y'} \, \pi_{\ell y}^{(t)} \mu_\ell(\boldsymbol{x}|\Theta)}{\mathbb{P}_T[y|\boldsymbol{x}, \Theta, \boldsymbol{\pi}^{(t)}]}$

References

- http://ect.bell-labs.com/who/tkh/publications/papers/odt.pdf
- https://www.ncbi.nlm.nih.gov/pubmed/27120604
- https://github.com/chrischoy