NEUROSYMBOLIC MODELS BASED ON HYBRIDS OF CONVOLUTIONAL NEURAL NETWORKS AND DECISION TREES

Rasul Kairgeldin and **Miguel Á. Carreira-Perpiñán**
Dept. Computer Science & Engineering, UC Merced

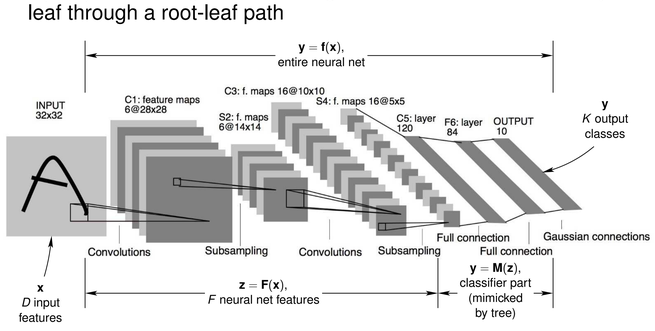
1 Abstract

Building on previous work, we propose a specific form of neurosymbolic model consisting of the composition of convolutional neural network layers with a sparse oblique classification tree (having hyperplane splits using few features). This can be seen as a neural feature extraction that finds a more suitable representation of the input space followed by a form of rule-based reasoning to arrive at a decision that can be explained. We show how to control the sparsity across the different decision nodes of the tree and its effect on the explanations produced. We demonstrate this on image classification tasks and show, among other things, that relatively small subsets of neurons are entirely responsible for the classification into specific classes, and that the neurons' receptive fields focus on areas of the image that provide best discrimination.

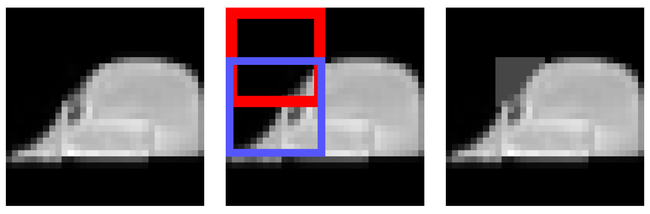
Work supported by NSF award IIS–2007147

2 The neural net / tree hybrid model

- Each decision node $i \in \mathcal{D}$ has a decision function $g_i(\mathbf{x}; \theta_i)$: "if $\mathbf{w}_i^T\mathbf{x} + w_{i0} \geq 0$ then $g_i(\mathbf{x}) = \texttt{right}_i$, otherwise $g_i(\mathbf{x}) = \texttt{left}_i$"
- Each leaf $j \in \mathcal{L}$ contains a constant label classifier that outputs a single class $c_j \in \{1, \dots, K\}$
- $\mathbf{T}(\mathbf{x}; \Theta)$ is a routing function that guides an instance $\mathbf{x}$ to exactly one leaf through a root-leaf path



- CNN of the form $\mathbf{y} = \mathbf{M}(\mathbf{F}(\mathbf{x}))$, where $\mathbf{x}$ is the pixel image, $\mathbf{y}$ the predicted label (or label distribution), $\mathbf{F}$ the convolutional layers and $\mathbf{M}$ the fully-connected layers (i.e., an MLP)
- We replace $\mathbf{M}$ with a sparse oblique classification tree $\mathbf{T}$ trained to replicate the behavior of $\mathbf{M}$. That is, $\mathbf{T}$ is trained to map each neural net feature vector $\mathbf{F}(\mathbf{x}_n)$ to the NN output $\mathbf{M}(\mathbf{F}(\mathbf{x}_n))$. For each data point $(\mathbf{x}_n, \mathbf{y}_n)$ we use $(\mathbf{z}_n, \mathbf{y}'_n)$, where $\mathbf{z}_n = \mathbf{F}(\mathbf{x}_n)$ and $\mathbf{y}'_n = \mathbf{M}(\mathbf{z}_n)$. If a tree $\mathbf{T}$ fits $\mathbf{y}'_n$ well, then $\mathbf{T}$ (and $\mathbf{T} \circ \mathbf{F}$) is approximately functionally equivalent to $\mathbf{M}$ (and $\mathbf{M} \circ \mathbf{F}$). Our neural net / tree hybrid is then $\mathbf{T}(\mathbf{F}(\mathbf{x}))$. We can interpret T.



Sample of class 8 misclassified as 7 (Left). Receptive field of neurons from the last convolutional layer of Lenet with largest positive (red) and negative (blue) weights in oblique decision node J (Middle). Small changes in the intersection of two regions fixed the misclassification error (Right).

3 Learning algorithm

Given a oblique tree $\mathbf{T}(\mathbf{x}, \Theta)$ of a fixed structure (e.g. a complete tree of depth Δ) and initial parameters (e.g. random), we use Tree Alternating Optimization (**TAO**) to minimize the following objective:

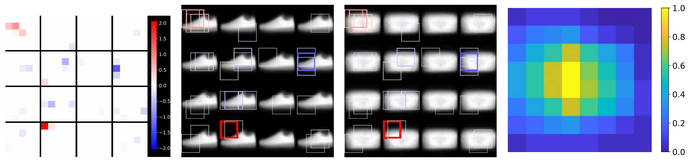
$$E(\Theta) = \sum_{n=1}^{N} L(\mathbf{y}_n, \mathbf{T}(\mathbf{F}(\mathbf{x}_n); \Theta)) + \lambda \sum_{i \in \mathcal{D}} h_\alpha(|\mathcal{R}_i|) \, \|\mathbf{w}_i\|_1$$

$$h_\alpha(t) = \begin{cases} 1, & t = 0 \\ t^\alpha, & t > 0 \end{cases}$$

where $L(\cdot, \cdot)$ is the loss, $\Theta = \{(\mathbf{w}_i, w_{i0})\}_{i \in \mathcal{D}} \cup \{\mathbf{c}_j\}_{j \in \mathcal{L}}$ are the set of all learnable model parameters, ℓ_1 is penalty over the weight vectors to promote sparsity via hyperparameters $\lambda \geq 0$, $\mathcal{R}_i$ is the reduced set of node i and $|\mathcal{R}_i|$ its cardinality.

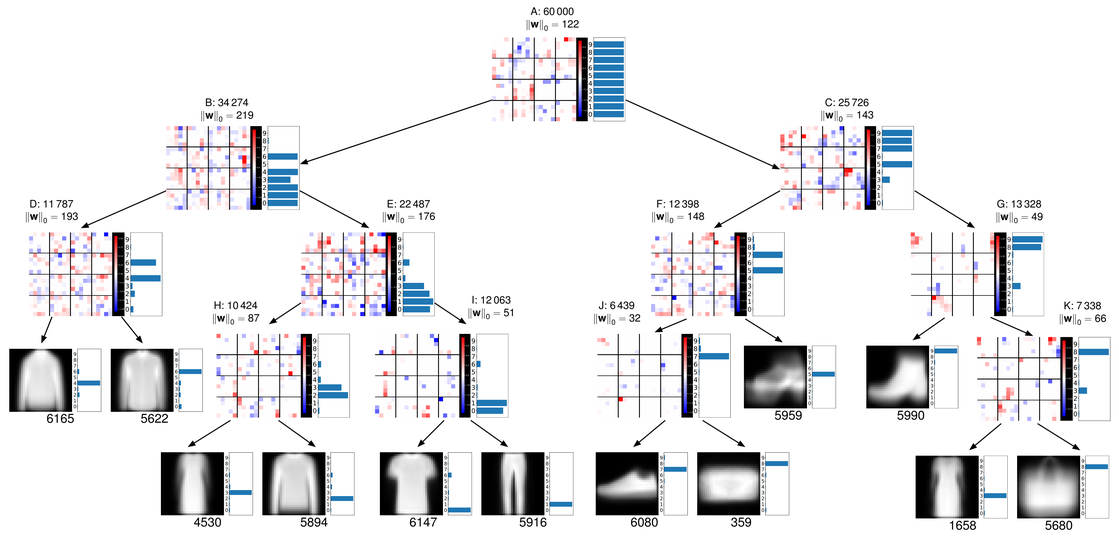
input training set; initial tree $\mathbf{T}(\cdot; \Theta)$ of depth Δ
$\mathcal{N}_0, \dots, \mathcal{N}_\Delta \leftarrow$ nodes at depth $0, \dots, \Delta$, respectively
generate $\mathcal{R}_1 \leftarrow \{1, \dots, N\}$ using initial tree
repeat
  for $d = \Delta$ **down to** 0
    parfor $i \in \mathcal{N}_d$
      if i is a leaf **then**
        $\theta_i \leftarrow$ fit a leaf predictor
          $\mathbf{g}_i$ on reduced set $\mathcal{R}_i$
      else
        generate pseudolabels $\overline{y}_n$ for each point $n \in \mathcal{R}_i$
        $\theta_i \leftarrow$ minimizer of the reduced problem:
        $\sum_{n \in \mathcal{R}_i} \overline{L}(\overline{y}_n, g_i(\mathbf{x}_n; \theta_i)) + \lambda \, h_\alpha(|\mathcal{R}_i|) \, \|\mathbf{w}_i\|_1$
  update $\mathcal{R}_i$ for each node
until stop
prune dead subtrees of **T**
return T

- We can rewrite RP objective as "avg-loss + λ' reg", with $\lambda' = \lambda N_i^{\alpha-1}$, avg-loss is the loss per instance in node i; reg = $|\mathbf{w}_i|_1$ (an *effective sparsity* hyperparameter)
- $\alpha < 1$: large RS penalized less $\rightarrow$ the root is denser
- $\alpha = 1$: all nodes penalized equally
- $\alpha > 1$: large RS penalized more $\rightarrow$ the root is sparser
- Can produce trees that are sparser and more accurate than regular TAO (i.e., $\alpha = 0$)

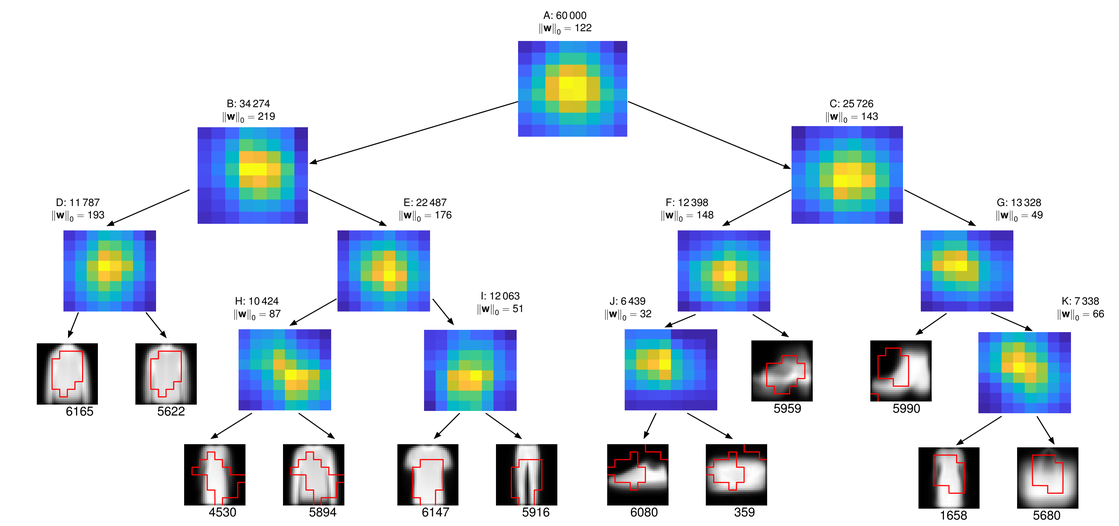


Where is decision node looking in the image? Col. 1 shows sparse weights of decision node J for each CNN feature map in the last layer of Lenet ($16 \times 5 \times 5$). Col. 2-3 show receptive field produced of neurons with non-zero decision node weights on the mean image from left and right leaf. Receptive field follow the order of CNN outputs left to right and top to bottom. Color and thickness of receptive fields correspond to weights of decision node. Col. 4 is heatmap of the "density" of the receptive fields. Tree hyperparameters are $\lambda = 0.001, \alpha = 1$.

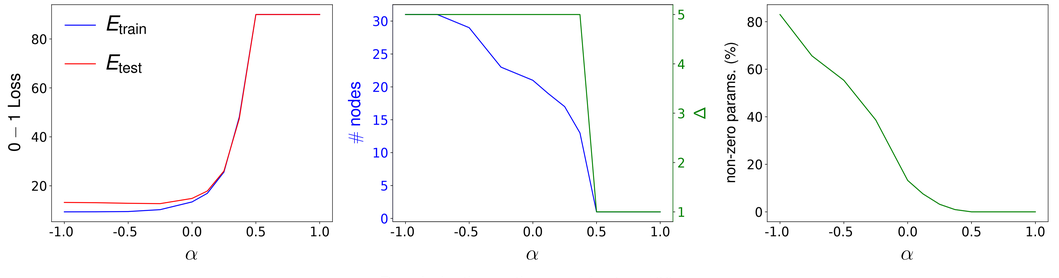
4 Experiments Results



Tree trained on LeNet embeddings on Fashion MNIST dataset $\lambda = 0.001, \alpha = 1$. $E_{\text{train}} = 5.4\%$, $E_{\text{test}} = 11.7\%$ and # non-zero params is 1298



Tree trained on LeNet embeddings on Fashion MNIST dataset $\lambda = 0.001, \alpha = 1$. Similar to fig. 2 each decision node contains "density" map of the receptive fields. Each leaf node contains contour produced by the parent decision node weights and density map.



Regularization path over α for $\lambda = 100$.