Ensemble Tree Model With Deep Architecture

Yuyang Zhang

2017-07-29

1 Related Work

1.1 Introduction of Tree Model

- 周志华,2016,机器学习,北京,清华大学出版社,425pp
- Zhou Z H. Ensemble methods: foundations and algorithms[M]. CRC press, 2012.
- Zhou Z H. Ensemble learning[J]. Encyclopedia of biometrics, 2015: 411-416.

1.2 Paper List

- Kontschieder P, Fiterau M, Criminisi A, et al. Deep neural decision forests[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 1467-1475.
- Zhou Z H, Feng J. Deep forest: Towards an alternative to deep neural networks[J]. arXiv preprint arXiv:1702.08835, 2017.

2 Deep Neural Decision Forests

这篇论文提出一种基于概率的树模型，同时这种模型可以利用BP进行训练，这应该是它的一大特点。

定义输入为$\mathcal{X}$和输出$\mathcal{Y}$，决策树的内部决策节点为$n \in \mathcal{N}$，叶子节点为预测节点，记为$\ell \in \mathcal{L}$，对于每一个内部节点都有一个决策函数

$$d_n(x;\theta) : \mathcal{X} \to [0,1]$$

在标准决策树中，每个决策节点都是二分的，而且决策标准是确定的，但本篇论文提出的模型中，决策节点的输出是一个二项分布的随机变量，该随机变量的均值为$d_n(x;\theta)$，因此叶节点的预测值为到达该叶节点的样本概率值，我们记为

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

其中$\pi = (\pi_\ell)_{\ell \in \mathcal{L}}$，$\pi_{\ell y}$表示拥有$y$标签的样本到达叶子节点$\ell$的概率，$\mu_\ell(x|\theta)$是选路函数(routing function)，表明样本x到达叶子节点ℓ的概率，其满足

$$\sum_{\ell} \mu_\ell(x|\theta) = 1$$

我们对选路函数作详细的定义，$\ell \swarrow n$和$n \searrow \ell$分别表示在决策节点表示叶子节点ℓ属于节点n的左子树和右子树，所以选路函数可以写作

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

其中$\bar{d}_n(x;\theta) = 1 - d_n(x;\theta)$，$\mathbb{I}_P$是关于条件$P$的指示函数。

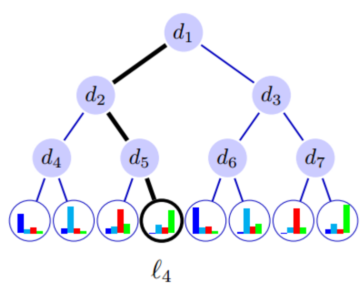

Figure 1. Each node $n \in \mathcal{N}$ of the tree performs routing decisions via function $d_n(\cdot)$ (we omit the parametrization Θ). The black path shows an exemplary routing of a sample $\boldsymbol{x}$ along a tree to reach leaf ℓ_4, which has probability $\mu_{\ell_4} = d_1(\boldsymbol{x})\bar{d}_2(\boldsymbol{x})\bar{d}_5(\boldsymbol{x})$.

图 1: 树结构示意图

如图1所示，$\mu_{\ell_4} = d_1(x)\bar{d}_2(x)\bar{d}_5(x)$。此处注意，$\mathbb{I}_{\ell \swarrow n}$和$\mathbb{I}_{n \searrow \ell}$ 可能同时为0。

对于决策节点，定义

$$d_n(x;\theta) = \sigma(f_n(x;\theta))$$

其中$\sigma(x) = (1+e^{-x})^{-1}$，并且要求$f_n(x;\theta) : \mathcal{X} \to \mathbb{R}$是一个实数的映射，可以认为这是神经网络单元的一种，把f_n定义为为一个线性单元，也可以做一些别的定义。这样的一棵树也可以做集成学习，定义森林为$\mathcal{F} = T_1, ..., T_k$是一个森林学习器，则它的预测输出为

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

这样一个树模型，可以通过BP的方式进行学习，把每个树的输出看作概率输出，定义数据集为$\mathcal{T} \subset \mathcal{X} \times \mathcal{Y}$，求解其极大似然估计，即优化对数损失，则误差为

$$\mathcal{R}(\theta,\pi;\mathcal{T}) = \frac{1}{|\mathcal{T}|}\sum_{(x,y)\in\mathcal{T}} L(\theta,\pi;x,y)$$

其中损失函数为

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

按照论文中的说法： All decision functions depend on a common parameter θ, which in turn parametrizes each function f_n. 所有的f_n使用相同的参数，我感觉此处不是十分合理，但别人论文里就是这么写的，可能效果好吧，或者为了减少参数计算量。优化该目标函数同样采用和神经网络一样的SGD，我们分为两部分：优化θ和π。对于θ有

$$\begin{aligned}\theta^{(t+1)} &= \theta^{(t)} - \eta\frac{\partial}{\partial\theta}\mathcal{R}(\theta^{(t)},\pi;\mathcal{B}) \\ &= \theta^{(t)} - \frac{\eta}{|\mathcal{B}|}\sum_{(x,y)\in\mathcal{B}}\frac{\partial}{\partial\theta}L(\theta^{(t)},\pi;x,y)\end{aligned}$$

其中$\eta > 0$为学习率，$\mathcal{B}$为一个数据随机子集(a mini-batch)。损失L的梯度可以表示为

$$\frac{\partial}{\partial\theta}L(\theta,\pi;x,y) = \sum_{n\in\mathcal{N}}\frac{\partial L(\theta,\pi;x,y)}{\partial f_n(x;\theta)}\frac{\partial f_n(x;\theta)}{\partial\theta}$$

我们重点关注一下前面那项：

$$\begin{aligned}\frac{\partial L(\theta,\pi;x,y)}{\partial f_n(x;\theta)} &= \frac{\partial}{\partial f_n(x;\theta)}\{-\log(\mathbb{P}_T[y|x,\theta,\pi])\} \\ &= -\frac{1}{\mathbb{P}_T[y|x,\theta,\pi])}\frac{\partial \mathbb{P}_T[y|x,\theta,\pi])}{\partial f_n(x;\theta)} \\ &= -\frac{1}{\mathbb{P}_T[y|x,\theta,\pi])}\frac{\partial \sum_{\ell\in\mathcal{L}}\pi_{\ell y}\mu_\ell(x|\theta)}{\partial f_n(x;\theta)}\end{aligned}$$

定义$\mathcal{L}_m \subset \mathcal{L}$表示以节点$m$为根节点的子树，则有

$$\begin{aligned}\frac{\partial L(\theta,\pi;x,y)}{\partial f_n(x;\theta)} &= -\frac{\sum_{\ell\in\mathcal{L}_n}\pi_{\ell y}}{\mathbb{P}_T[y|x,\theta,\pi])}\frac{\partial \mu_\ell(x|\theta)}{\partial f_n(x;\theta)} \\ &= -\frac{\sum_{\ell\in\mathcal{L}_n}\pi_{\ell y}}{\mathbb{P}_T[y|x,\theta,\pi])}\frac{\partial \prod_{m\in\mathcal{N}} d_m(x;\theta)^{\mathbb{I}_{\ell\swarrow m}}\bar{d}_m(x;\theta)^{\mathbb{I}_{m\searrow\ell}}}{\partial f_n(x;\theta)} \\ &= \frac{\sum_{\ell\in\mathcal{L}_n}\pi_{\ell y}\prod_{m\in\bar{\mathcal{N}}_n} d_m(x;\theta)^{\mathbb{I}_{\ell\swarrow m}}\bar{d}_m(x;\theta)^{\mathbb{I}_{m\searrow\ell}}}{\mathbb{P}_T[y|x,\theta,\pi])}\frac{\partial d_n(x;\theta)^{\mathbb{I}_{\ell\swarrow n}}\bar{d}_n(x;\theta)^{\mathbb{I}_{n\searrow\ell}}}{\partial f_n(x;\theta)}\end{aligned}$$

我们令$f_n(x;\theta)=z_n$，同时只关心最后一项时，有

$$\begin{aligned}\frac{\partial d_n(x;\theta)^{\mathbb{I}_{\ell\swarrow n}}\bar{d}_n(x;\theta)^{\mathbb{I}_{n\searrow\ell}}}{\partial f_n(x;\theta)} &= \frac{\partial}{\partial z_n}d_n(z_n)^{\mathbb{I}_{\ell\swarrow n}}\bar{d}_n(z_n)^{\mathbb{I}_{n\searrow\ell}} \\ &= \frac{\partial}{\partial z_n}[\sigma(z_n)^{\mathbb{I}_{\ell\swarrow n}}(1-\sigma(z_n))^{\mathbb{I}_{n\searrow\ell}}]\end{aligned}$$

其中当$\mathbb{I}_{\ell\swarrow n}=1$时

$$\frac{\partial}{\partial z_n}[\sigma(z_n)^{\mathbb{I}_{\ell\swarrow n}}(1-\sigma(z_n))^{\mathbb{I}_{n\searrow\ell}}] = \sigma(z_n)(1-\sigma(z_n))$$

当$\mathbb{I}_{n\searrow\ell}=1$时

$$\frac{\partial}{\partial z_n}[\sigma(z_n)^{\mathbb{I}_{\ell\swarrow n}}(1-\sigma(z_n))^{\mathbb{I}_{n\searrow\ell}}] = -\sigma(z_n)(1-\sigma(z_n))$$

我们可以发现，原式基本没变，所以将其带回，我们有

$$\frac{\partial L(\theta,\pi;x,y)}{\partial f_n(x;\theta)} = \frac{\sum_{\ell\in\mathcal{L}_n}\pi_{\ell y}\mu_\ell(x|\theta)}{\mathbb{P}_T[y|x,\theta,\pi])}d_n(x;\theta)^{\mathbb{I}_{n\searrow\ell}}(-\bar{d}_n(x;\theta))^{\mathbb{I}_{\ell\swarrow n}}$$

在推导验证此处公式的时候，不要忘记log前面的负号，同时真的也让我理解了Sigmoid函数有多方便...

对于预测的叶子节点概率分布π的优化也很简单，该问题可以写作

$$min_\pi \mathcal{R}(\theta,\pi;\mathcal{T})$$

迭代更新公式为

$$\pi_{\ell y}^{(t+1)} = \frac{1}{Z_\ell^{(t)}} \sum_{(x,y')\in\mathcal{T}} \frac{\mathbb{I}_{y=y'}\pi_{(\ell y)}^{(t)}\mu_\ell(x|\theta)}{\mathbb{P}_T[y|x,\theta,\pi^{(t)}])}$$

其中$Z_\ell^{(t)}$为归一化因子。

同时，我们也可以同时训练多棵树做集成学习，文中的说法是所有树可以共享θ参数，同时每棵树也可以选择不同的f_n和π。初始化时比较随意，均匀分布或者别的随机数都可以，具体的训练算法如下图所示。

Algorithm 1 Learning trees by back-propagation

Require: $\mathcal{T}$: training set, nEpochs
random initialization of Θ
for all $i \in \{1, \ldots, \text{nEpochs}\}$ **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

图 2: 概率树的BP训练算法

简单来讲就是初始化参数后先计算分布π，再通过训练数据迭代更新θ，重复n次。在训练时候，初始化和具体训练有些小技巧，可以去看原论文。这篇论文中的实验是做的图像分类，与CNN等模型做了对比。

总的来说，我个人感觉这个模型的优点有以下：

- 训练参数少，迭代次数少，训练开销小
- 思路好，传统的tree model都是在节点上确定分类，最后输出，本文在树的节点上就比较具有随机性，应该泛化能力会比较强
- 提出一种基于BP训练的tree model，让人眼前一亮，虽然内部还是用了方便求导的激活函数，还是有点神经网络模型的意思

说完好的，再说说我质疑的

- 模型效果应该不会特别特别好，相比于其它深度学习模型，较少的参数决定了模型性能的上限
- 我感觉这个东西训练不会收敛啊，这么做模型感觉没什么理由，完全像是一个工程上的做法，或者一种突发奇想的尝试，不过我也没自己实现，只是有点怀疑

总的来说，这篇论文发表于2015年，是深度学习比较火热的时候，也算是对有深度的集成学习模型的尝试吧，而且做了一种能用BP训练的树模型，这点的确有点牛逼。

3 Deep Forest: Towards an Alternative to Deep Neural Networks

这篇文章解读分析的人就比较多了，各种大神也都说过了，具体详情转步知乎搜索周志华的gcForest，我这里就不细说了，主要是以下两张图：

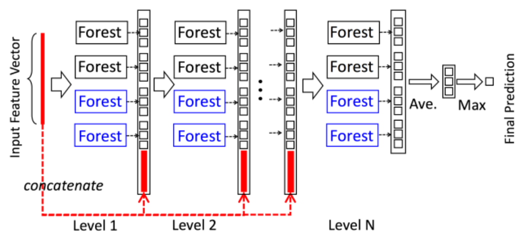

Figure 1: Illustration of the cascade forest structure. Suppose each level of the cascade consists of two random forests (black) and two completely-random tree forests (blue). Suppose there are three classes to predict; thus, each forest will output a three-dimensional class vector, which is then concatenated for re-representation of the original input.

图 3: gcForest模型结构

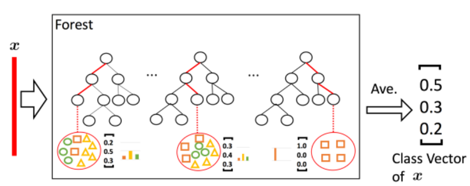

Figure 2: Illustration of class vector generation. Different marks in leaf nodes imply different classes.

图 4: gcForest结果生成

图一表明了gcForest模型的训练结构，层联级的结构，每层都是多个集成学习模型，随机森林这种，然后每一层的输出作为下一层的输入，外加

最初始的输入一并进入下一层，然后模型可以通过验证自己决定层数深度，当生成的模型效果不够好的时候会终止生长，这也是集成学习的核心思想之一，好而不同的模型集成。

这篇论文是周老师今年的新作之一，应该已经酝酿已久了，说说我感觉的优点吧：

- 在“深度学习的时代”做出了一个深度的集成学习模型，而且某种程度上来讲，是有一定“深度”的
- 给我感觉是一种boosting和bagging的结合，既有串行的层联级结构，也有并行训练的并行结构，结合了集成学习的两个大方向
- 深度学习时代不忘初心的表现吧，既能吸取深度学习的精华，也不忘传统集成学习的模型
- 给大家提供了另一种思路，深度不一定都需要BP来训练，交叉验证也可以作为训练的依据
- 自增长的结构，通过模型表现控制模型深度，控制模型的规模，适应不同大小的数据

再说说质疑的点：

- 首先还是模型能力的问题，我记得该论文的一个作者说过，训练这样的模型还是比较浪费资源的，相当于每层都要计算所有的数据，然后还要把结果都保留下来，与深度学习的BP相比，还是有一定劣势，当把足够大的数据喂给模型的时候，是否能取得足够好的表现，还是要打个问号的
- 无法使用BP，给我一种很难有全局优化的感觉，只是通过不断的训练来学习数据的表达而已，而且这种表达，并不是通过全局优化学习到的，有可能很多种表示到最后都可以达到最优，但是层与层之间学习到的方向不一样，这样对模型最后的效果应该会打折扣
- 我没有仔细去看源码，但我感觉可以考虑多加每层的模型数量，减少一下深度，这样可以多一些并行的训练，效率应该是会高一点

4 Summary

总结一下，这两篇论文都是在深度学习火热的情况，提出的具有“深度”的树模型，或者说集成学习模型，虽然可能在表现上比不过深度学习的某些模型，但是作为研究，都是属于那种让人眼前一亮的论文，新的思路总是能给人很多启发。