Label Distribution Learning Forests

Wei Shen[1,2], Kai Zhao[1], Yilu Guo[1], Alan Yuille[2]
[1]School of Communication and Information Engineering, Shanghai University
[2] Department of Computer Science, Johns Hopkins University

NIPS 2017

Background

- What is label distribution learning (LDL)?
  - a sample: $\mathbf{x} \in \mathbb{R}^m$,
  - a set of disjoint labels $\mathcal{Y} = \{y_1, y_2, \ldots, y_C\}$.
  - $\mathbf{x}$ is associated with a label distribution $\mathbf{d} = (d_{\mathbf{x}}^{y_1}, d_{\mathbf{x}}^{y_2}, \ldots, d_{\mathbf{x}}^{y_C})$,
  - $d_{\mathbf{x}}^{y_c}$ expresses the probability of the sample $\mathbf{x}$ having the c-th label y_c, $d_{\mathbf{x}}^{y_c} \in [0,1], \sum_{c=1}^{C} d_x^{y_c} = 1$.
- LDL is suitable for many real-world problems
  - facial age estimation
  - expression recognition
  - hand orientation estimation
  - crowd opinion prediction on movies

} **label ambiguity!**

Target is a distribution!

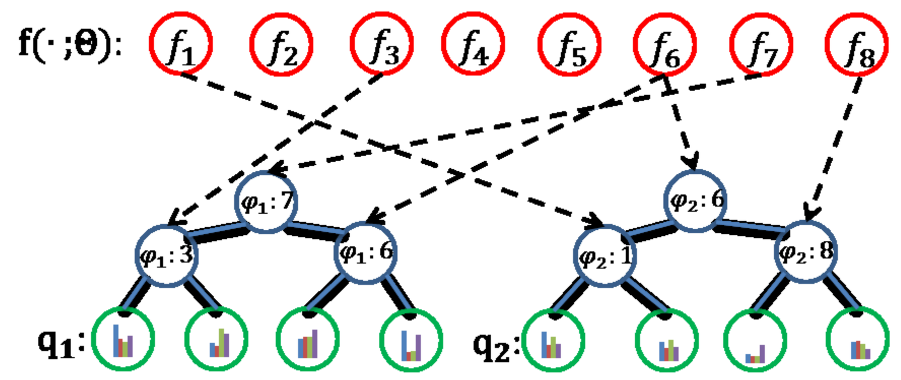
- How to address an LDL problem?
  - $d_{\mathbf{x}}^{y_c} = \frac{e^{f(\mathbf{x}, \mathbf{w}_c)}}{\sum_{j=1}^{C} e^{f(\mathbf{x}, \mathbf{w}_j)}}$, represent label distribution by a maximum entropy model
    ----limited generality, have difficulty in representing mixture distribution
  - extend the existing learning algorithms to deal with label distribution
    ----have limitations in representation learning

Our Intuition

- We propose label distribution learning forests (LDLFs)
  - decision trees have the potential to model any general form of label distributions by mixture of the leaf node predictions
  - differentiable decision trees [1] enable the combination of tree learning and representation learning in an end-to-end manner

Method

- Problem Formulation
  - Our goal is to learn a mapping function $g: \mathbf{x} \to \mathbf{d}$ between an input sample $\mathbf{x}$ and its corresponding label distribution $\mathbf{d}$.
  - We want to learn the mapping function $g(\mathbf{x})$ by a decision tree based model $\mathcal{T}$



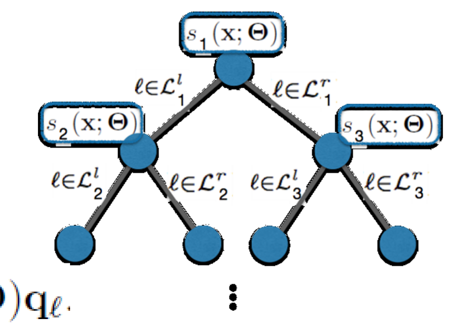
$\varphi(\cdot)$is an index function to bring the $\varphi(n)$-th output of function $\mathbf{f}(\mathrm{x}, \mathbf{\Theta})$ in correspondence with split node n

$\mathbf{f}: \mathrm{x} \to \mathbb{R}^M$is a real-valued feature learning function

split nodes $n \in \mathcal{N}$ each n defines a split function $s_n(\cdot, \mathbf{\Theta}): \mathrm{x} \to [0,1]$, $s_n(\mathbf{x}; \mathbf{\Theta}) = \sigma(f_{\varphi(n)}(\mathbf{x}; \mathbf{\Theta}))$

leaf nodes $\ell \in \mathcal{L}$ each ℓ holds a distribution $\mathbf{q}_\ell = (q_{\ell_1}, q_{\ell_2}, \ldots, q_{\ell_C})^\top$

- Learning a tree



  - The probability of the sample $\mathbf{x}$ falling into leaf node ℓ is given by

$$p(\ell|\mathbf{x};\mathbf{\Theta}) = \prod_{n\in\mathcal{N}} s_n(\mathbf{x};\mathbf{\Theta})^{\mathbf{1}(\ell\in\mathcal{L}_n^l)}(1 - s_n(\mathbf{x};\mathbf{\Theta}))^{\mathbf{1}(\ell\in\mathcal{L}_n^r)}$$

  - The output of the tree $\mathcal{T}$ w.r.t. $\mathbf{x}$, is defined by $\mathbf{g}(\mathbf{x};\mathbf{\Theta},\mathcal{T}) = \sum_{\ell\in\mathcal{L}} p(\ell|\mathbf{x};\mathbf{\Theta})\mathbf{q}_\ell$.
  - Loss function:

$$R(\mathbf{q},\mathbf{\Theta};\mathcal{S}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} d_{\mathbf{x}_i}^{y_c} \log(g_c(\mathbf{x}_i;\mathbf{\Theta},\mathcal{T})) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} d_{\mathbf{x}_i}^{y_c} \log\Big(\sum_{\ell\in\mathcal{L}} p(\ell|\mathbf{x}_i;\mathbf{\Theta})q_{\ell_c}\Big)$$

  - An alternating optimization strategy to address $(\mathbf{\Theta}^*, \mathbf{q}^*) = \arg\min_{\mathbf{\Theta},\mathbf{q}} R(\mathbf{q},\mathbf{\Theta};\mathcal{S})$
    - **Learning Split Nodes**

$$\frac{\partial R(\mathbf{q},\mathbf{\Theta};\mathcal{S})}{\partial\mathbf{\Theta}} = \sum_{i=1}^{N}\sum_{n\in\mathcal{N}} \frac{\partial R(\mathbf{q},\mathbf{\Theta};\mathcal{S})}{\partial f_{\varphi(n)}(\mathbf{x}_i;\mathbf{\Theta})}\frac{\partial f_{\varphi(n)}(\mathbf{x}_i;\mathbf{\Theta})}{\partial\mathbf{\Theta}},$$

$$\frac{\partial R(\mathbf{q},\mathbf{\Theta};\mathcal{S})}{\partial f_{\varphi(n)}(\mathbf{x}_i;\mathbf{\Theta})} = \frac{1}{N}\sum_{c=1}^{C} d_{\mathbf{x}_i}^{y_c}\Big(s_n(\mathbf{x}_i;\mathbf{\Theta})\frac{g_c(\mathbf{x}_i;\mathbf{\Theta},\mathcal{T}_n^r)}{g_c(\mathbf{x}_i;\mathbf{\Theta},\mathcal{T})} - (1 - s_n(\mathbf{x}_i;\mathbf{\Theta}))\frac{g_c(\mathbf{x}_i;\mathbf{\Theta},\mathcal{T}_n^l)}{g_c(\mathbf{x}_i;\mathbf{\Theta},\mathcal{T})}\Big)$$

    - **Learning Leaf Nodes**

$$\min_{\mathbf{q}} R(\mathbf{q},\mathbf{\Theta};\mathcal{S}), \mathbf{s.t.}, \forall\ell, \sum_{c=1}^{C} q_{\ell_c} = 1$$ **Constrained Convex Optimization Problem!**

  - We propose to address this optimization problem by Variational Bounding (VB) [2]

$$R(\mathbf{q},\mathbf{\Theta};\mathcal{S}) \le -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} d_{\mathbf{x}_i}^{y_c}\sum_{\ell\in\mathcal{L}} \xi_\ell(\bar{q}_{\ell_c},\mathbf{x}_i)\log\Big(\frac{p(\ell|\mathbf{x}_i;\mathbf{\Theta})q_{\ell_c}}{\xi_\ell(\bar{q}_{\ell_c},\mathbf{x}_i)}\Big), \quad \xi_\ell(q_{\ell_c},\mathbf{x}_i) = \frac{p(\ell|\mathbf{x}_i;\mathbf{\Theta})q_{\ell_c}}{g_c(\mathbf{x}_i;\mathbf{\Theta},\mathcal{T})}$$

  - Define

$$\phi(\mathbf{q},\bar{\mathbf{q}}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} d_{\mathbf{x}_i}^{y_c}\sum_{\ell\in\mathcal{L}} \xi_\ell(\bar{q}_{\ell_c},\mathbf{x}_i)\log\Big(\frac{p(\ell|\mathbf{x}_i;\mathbf{\Theta})q_{\ell_c}}{\xi_\ell(\bar{q}_{\ell_c},\mathbf{x}_i)}\Big)$$

$\phi(\mathbf{q},\bar{\mathbf{q}}) \ge \phi(\mathbf{q},\mathbf{q}) = R(\mathbf{q},\mathbf{\Theta};\mathcal{S})$ and $\phi(\bar{\mathbf{q}},\bar{\mathbf{q}}) = R(\bar{\mathbf{q}},\mathbf{\Theta};\mathcal{S})$ hold the conditions for VB

$$\mathbf{q}^{(t+1)} = \arg\min_{\mathbf{q}} \phi(\mathbf{q},\mathbf{q}^{(t)}), \mathbf{s.t.}, \forall\ell, \sum_{c=1}^{C} q_{\ell_c} = 1 \Rightarrow \varphi(\mathbf{q},\mathbf{q}^{(t)}) = \phi(\mathbf{q},\mathbf{q}^{(t)}) + \sum_{\ell\in\mathcal{L}}\lambda_\ell(\sum_{c=1}^{C} q_{\ell_c} - 1)$$ **Lagrangian!**

$$\lambda_\ell = \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} d_{\mathbf{x}_i}^{y_c}\xi_\ell(q_{\ell_c}^{(t)},\mathbf{x}_i) \text{ and } q_{\ell_c}^{(t+1)} = \frac{\sum_{i=1}^{N} d_{\mathbf{x}_i}^{y_c}\xi_\ell(q_{\ell_c}^{(t)},\mathbf{x}_i)}{\sum_{c=1}^{C}\sum_{i=1}^{N} d_{\mathbf{x}_i}^{y_c}\xi_\ell(q_{\ell_c}^{(t)},\mathbf{x}_i)}$$

- Learning a forest $\mathcal{F} = \{\mathcal{T}_1,\ldots,\mathcal{T}_K\}$ $\mathbf{g}(\mathbf{x};\mathbf{\Theta},\mathcal{F}) = \frac{1}{K}\sum_{k=1}^{K}\mathbf{g}(\mathbf{x};\mathbf{\Theta},\mathcal{T}_k)$

$$R_{\mathcal{F}} = \frac{1}{K}\sum_{k=1}^{K} R_{\mathcal{T}_k} \qquad \frac{\partial R_{\mathcal{F}}}{\partial\mathbf{\Theta}} = \frac{1}{K}\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{n\in\mathcal{N}_k}\frac{\partial R_{\mathcal{T}_k}}{\partial f_{\varphi_k(n)}(\mathbf{x}_i;\mathbf{\Theta})}\frac{\partial f_{\varphi_k(n)}(\mathbf{x}_i;\mathbf{\Theta})}{\partial\mathbf{\Theta}}$$

Experimental Results

We can either use LDLF as a shallow stand-alone model (sLDLFs) or integrate it with any deep networks (dLDLFs)

- Comparison of sLDLFs to Stand-alone LDL Methods

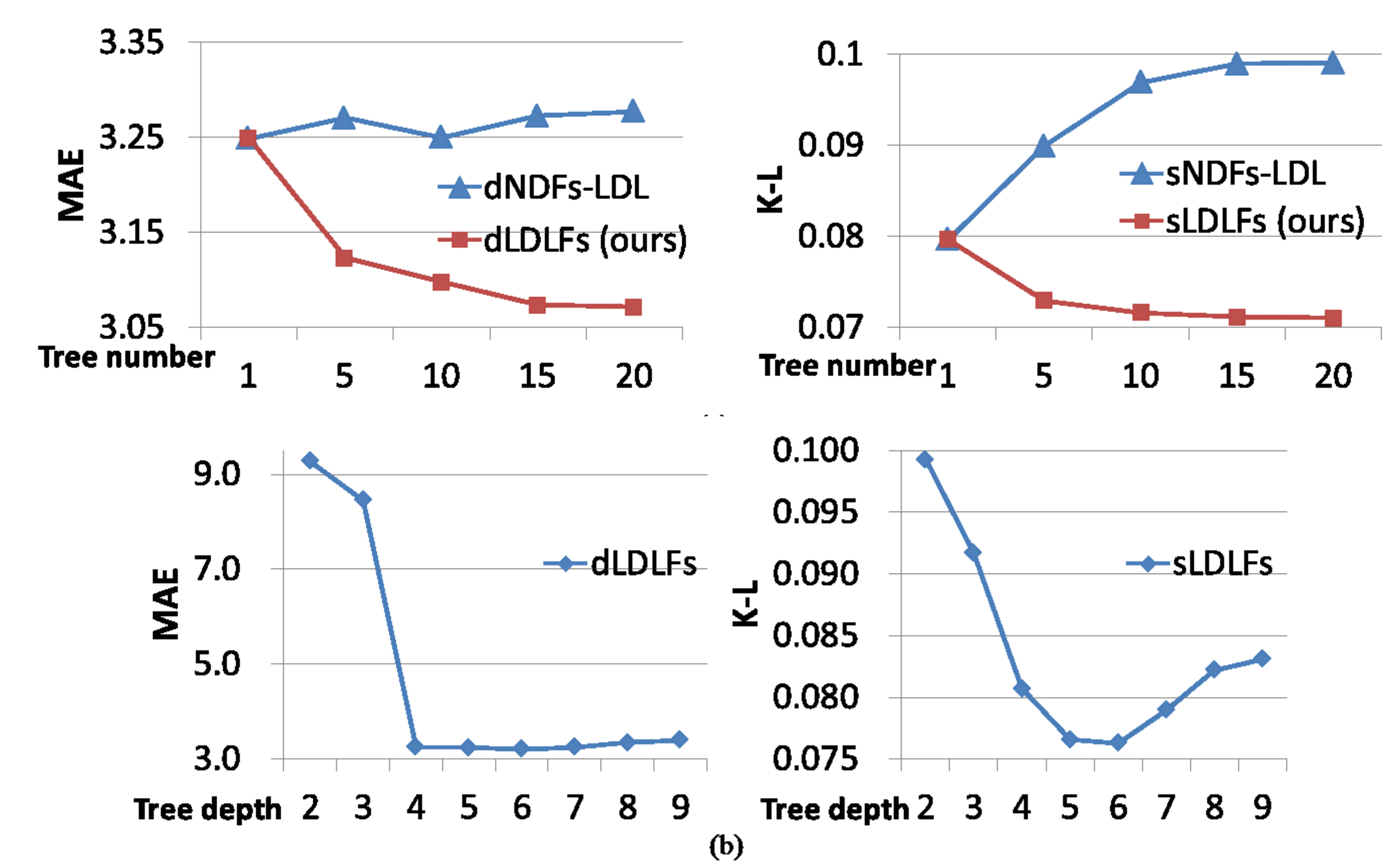
Dataset	Method	K-L ↓	Euclidean↓	Intersection↑
Moive	**sLDLF (ours)**	**0.073±0.005**	**0.133±0.003**	**0.870±0.003**
	AOSO-LDLogitBoost	0.086±0.004	0.155±0.003	0.848±0.003
	LDLogitBoost	0.090±0.004	0.159±0.003	0.845±0.003
	LDSVR	0.092±0.005	0.158±0.004	0.844±0.004
	BFGS-LDL	0.099±0.004	0.167±0.004	0.836±0.003
	IIS-LDL	0.129±0.007	0.187±0.004	0.817±0.004
Human Gene	**sLDLF (ours)**	**0.228±0.006**	**0.085±0.002**	**0.788±0.002**
	LDSVR	0.245±0.019	0.099±0.005	0.771±0.015
	BFGS-LDL	0.231±0.021	0.076±0.006	0.769±0.012
	IIS-LDL	0.239±0.018	0.089±0.006	0.747±0.009
Natural Scene	**sLDLF (ours)**	**0.534±0.013**	**0.317±0.014**	**0.664±0.010**
	LDSVR	0.852±0.023	0.511±0.021	0.509±0.016
	BFGS-LDL	0.856±0.061	0.475±0.029	0.492±0.026
	IIS-LDL	0.879±0.023	0.458±0.014	0.461±0.011

- Evaluation of dLDLFs on Facial Age Estimation

Method	IIS-LDL	CPNN	BFGS-LDL	DLDL+VGG-Face	dLDLF+VGG-Face (ours)
MAE	5.67±0.15	4.87±0.31	3.94±0.05	2.42±0.01	**2.24±0.02**

MAE of age estimation comparison on the Morph Dataset

- Parameter Discussion



Reference

[1] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò. Deep neural decision forests. In Proc. ICCV, 2015.
[2] M. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. Machine Learning, 37(2):183–233, 1999.