Semi-Supervised Learning with Decision Trees: Graph Laplacian Tree Alternating Optimization

Arman Zharmagambetov and **Miguel Á. Carreira-Perpiñán**
Dept. of Computer Science & Engineering, University of California, Merced, USA

1 Introduction and Motivation

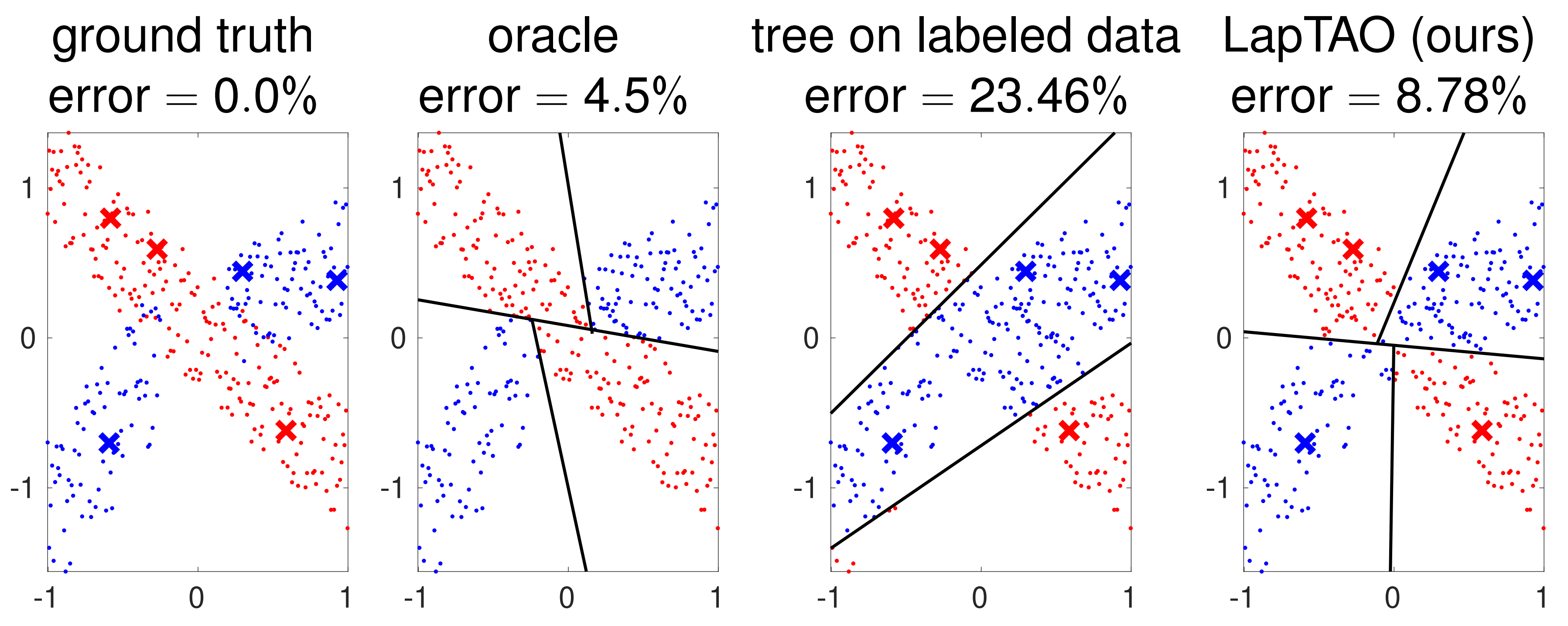
- ML use is rapidly growing $\rightarrow$ as is the need for data labeling;
- . . . but manually labeling data is expensive!
- Unlabeled data are usually cheap and easy to get;
- Unlabeled data contain useful information that can improve a model;
- **Semi-supervised learning** (SSL) seeks to train a machine learning model by leveraging a small percentage of labeled data and much larger sample of unlabeled data.

Why Decision Trees?

- **Widespread usage** – successfully used as a standalone predictor or as a building block for popular ML frameworks: XGBoost, Random Forest, etc.
- **Interpretability** – input follows a unique root-to-leaf path
- DT + SSL $\rightarrow$ *Interpretable semi-supervised learning?*

Motivational example: Binary classification on 2D using *oblique* trees. The leftmost plot shows the original data and corresponding class labels. *Crossed markers (six in total)* indicate the labeled points that we provide to any given SSL algorithm.

- "oracle" (plot 2): the decision boundary obtained by using all available labeled data, i.e., fully supervised learning;
- "tree on labeled data" (plot 3): the model uses only six labeled points to train a tree;
- "LapTAO" (rightmost): our proposed SSL framework which clearly improves over the naive baseline (plot 3) by leveraging both labeled and unlabeled data.



Work supported by NSF awards IIS–1423515 and IIS–2007147

2 LapTAO: semi-supervised learning for decision trees

How to apply SSL for Decision Trees? The most widespread approaches use a **graph prior**. Consider dataset $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$: $\mathcal{D}_l = \{\mathbf{x}_n, y_n\}_{n=1}^{l}$ labeled data, $\mathcal{D}_u = \{\mathbf{x}_n\}_{n=l+1}^{N}$ unlabeled data, and $l << N$. T: $\mathbb{R}^D \rightarrow \mathbb{R}$ – tree predictive mapping with parameters $\Theta = \{\boldsymbol{\theta}_i\}_{\text{nodes}}$. Then, our goal is to minimize the following regularized loss:

$$E(\Theta) = \sum_{n=1}^{l} (T(\mathbf{x}_n; \Theta) - y_n)^2 + \alpha\, \phi(\Theta) + \gamma \sum_{n,m=1}^{N} w_{nm} (T(\mathbf{x}_n; \Theta) - T(\mathbf{x}_m; \Theta))^2 \quad (1)$$

- w_{nm} are the elements of the similarity matrix **W** (affinity matrix) which encourages similar instances to have similar predictions;
- $\phi(\cdot)$ is a penalty on Θ such as $\|\cdot\|_1$;
- How to solve? The loss is **non-differentiable and non-convex** due to $T(\cdot)$!

Introduce a new variable z for each data point and consider the constrained problem:

$$\min_{z_1,\dots,z_N,\Theta} \sum_{n=1}^{l} (z_n - y_n)^2 + \alpha\, \phi(\Theta) + \gamma \sum_{n,m=1}^{N} w_{nm}(z_n - z_m)^2$$
$$\text{s.t.} \quad z_n = T(\mathbf{x}_n; \Theta) \quad n = 1, \dots, N.$$

Let's rewrite this in matrix form:

$$\min_{\mathbf{z},\Theta} (\mathbf{z} - \mathbf{y})^T \mathbf{J}\, (\mathbf{z} - \mathbf{y}) + \alpha\, \phi(\Theta) + \gamma\, \mathbf{z}^T \mathbf{L}\, \mathbf{z}$$
$$\text{s.t.} \quad \mathbf{z} = \mathbf{t}(\mathbf{X}; \Theta),$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian and $\mathbf{J} = \text{diag}(1, \dots, 1, 0, \dots, 0) \in \mathbb{R}^{N \times N}$ is diagonal matrix with first l entries equal to 1 and the rest are 0. Next, we apply the *augmented Lagrangian*. This defines a new, unconstrained optimization problem parametrized by μ and $\boldsymbol{\lambda}$:

$$\min_{\mathbf{z},\Theta} (\mathbf{z} - \mathbf{y})^T \mathbf{J}\, (\mathbf{z} - \mathbf{y}) + \alpha\, \phi(\Theta) + \gamma\, \mathbf{z}^T \mathbf{L}\, \mathbf{z} - \boldsymbol{\lambda}^T(\mathbf{z} - \mathbf{t}(\mathbf{X}; \Theta)) + \mu\|\mathbf{z} - \mathbf{t}(\mathbf{X}; \Theta)\|^2$$

Finally, we apply alternating optimization to minimize the above objective over: $\mathbf{z}$ a.k.a "label-step" and $\mathbf{t}(\mathbf{X}; \Theta)$ a.k.a "tree-step":

- **Label-step**. The objective is a quadratic function and a minimizer is obtained by solving the **sparse** linear system:

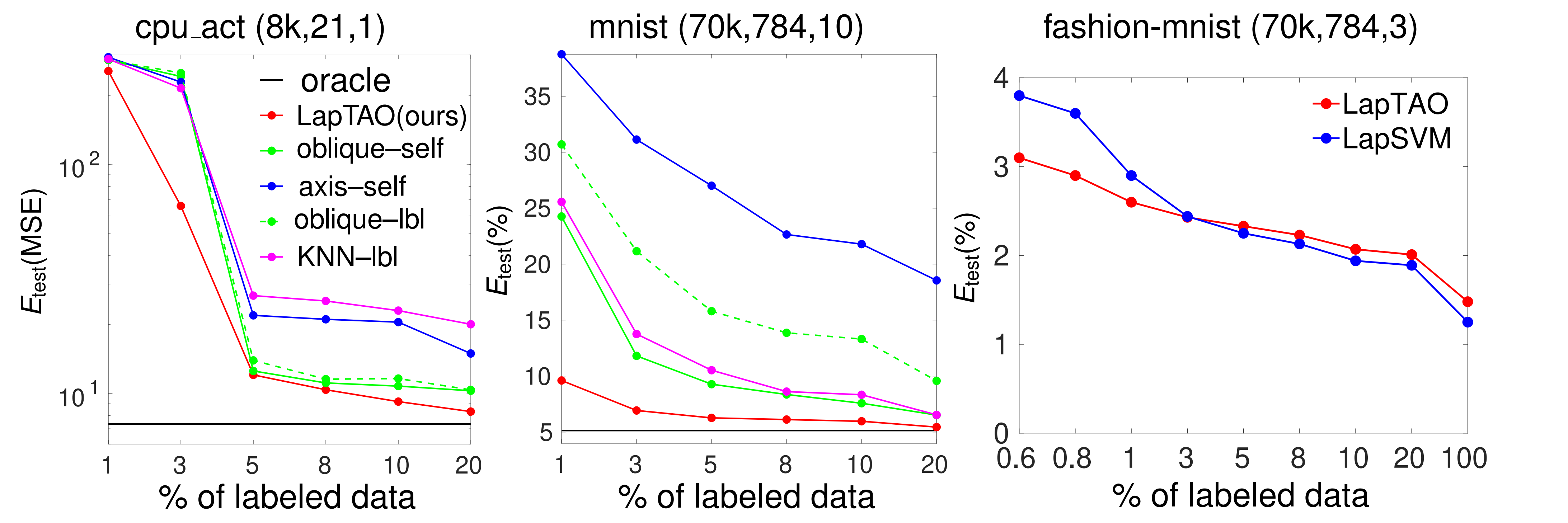
  $$\min_{\mathbf{z}} (\mathbf{z} - \mathbf{y})^T \mathbf{J}\, (\mathbf{z} - \mathbf{y}) + \gamma\, \mathbf{z}^T \mathbf{L}\, \mathbf{z} - \boldsymbol{\lambda}^T(\mathbf{z} - \mathbf{t}(\mathbf{X}; \Theta)) + \mu\|\mathbf{z} - \mathbf{t}(\mathbf{X}; \Theta)\|^2 \Rightarrow$$
  $$\mathbf{A}\mathbf{z} = \mathbf{J}\mathbf{y} + \mu\mathbf{t}(\mathbf{X}; \Theta) + \frac{1}{2}\boldsymbol{\lambda}$$

  where $\mathbf{A} = \mathbf{J} + \mu\mathbf{I} + \gamma\mathbf{L}$ is a positive definite and sparse matrix. Intuitively, this step can be interpreted as "approximating" the labels (for $\mathcal{D}_u$) using the graph Laplacian and predictions obtained from the current tree (i.e., label smoothing).
- **Tree-step**. The problem reduces to a regression fit of a tree:

  $$\min_{\Theta}\ \mu\|\mathbf{z} - \mathbf{t}(\mathbf{X}; \Theta)\|^2 + \alpha\, \phi(\Theta) - \boldsymbol{\lambda}^T(\mathbf{z} - \mathbf{t}(\mathbf{X}; \Theta)) \Leftrightarrow$$
  $$\min_{\Theta} \left\| \left(\mathbf{z} - \frac{1}{2\mu}\boldsymbol{\lambda}\right) - \mathbf{t}(\mathbf{X}; \Theta) \right\|^2 + \frac{\alpha}{\mu}\phi(\Theta).$$

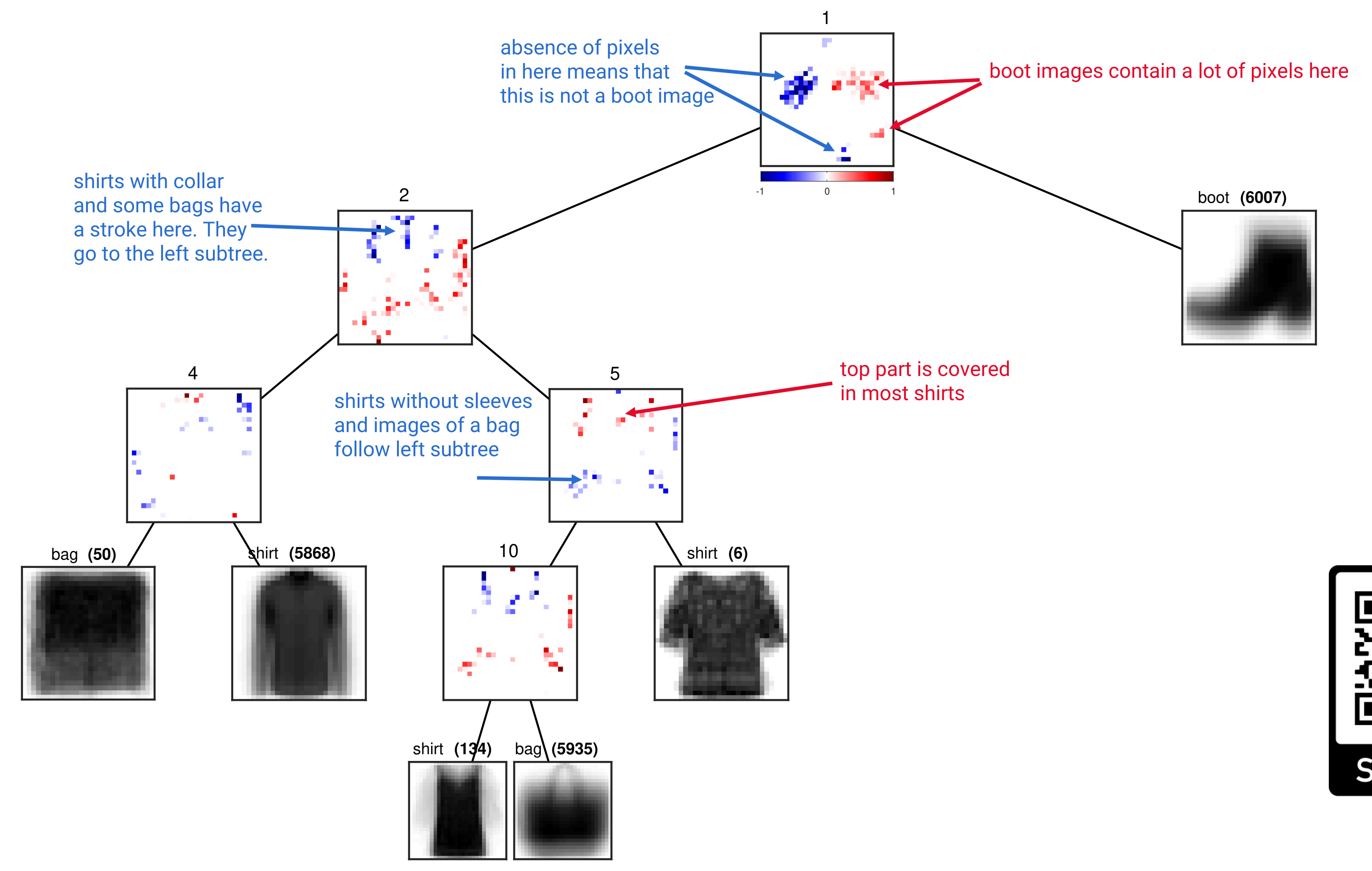
  Although any tree learning algorithm can be used, we solve this step using Tree Alternating Optimization (TAO) algorithm. Intuitively, this step can be understood as fitting a tree with the current estimates of labels.

3 Experiments



Results on regression (cpu_act) and classification (mnist, 3 classes of fashion-mnist) datasets:

- We use an oblique tree (makes hyperplane-based split) as an underlying model for LapTAO.
- **Left and Middle:** comparison against (mostly) tree-based methods. "Oracle" is an oblique tree trained by providing 100% of labeled data, "*–self" is self-training baseline, "*–lbl" uses only given % of labeled data (see x–axis), "oblique/axis" means an oblique or axis-aligned tree, respectively.
- *LapTAO consistently improves over all other SSL methods*, and quickly approaches the fully supervised baseline.
- **Right:** comparison against Laplacian SVM with RBF kernel. It uses the same formulation as in eq. (1) but for SVMs. The error gap between LapTAO and LapSVM narrows as we introduce more label scarcity and eventually we start to outperform when % of labeled data $\leq$ 3%.
- Below is the visualization of an example tree obtained by LapTAO on fashion-mnist (3 classes):


1
absence of pixels in here means that this is not a boot image
boot images contain a lot of pixels here
-1
0
1
2
shirts with collar and some bags have a stroke here. They go to the left subtree.
boot (6007)
4
5
shirts without sleeves and images of a bag follow left subtree
top part is covered in most shirts
bag (50)
shirt (5868)
10
shirt (6)
shirt (134)
bag (5935)

SCAN ME