Joint Leaf-Refinement and Ensemble Pruning Through L_1 Regularization

Sebastian Buschjäger and Katharina Morik

Lamarr Institute for Machine Learning and Artificial Intelligence, D-44227 Dortmund, Germany

LAMARR INSTITUTE FOR MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Motivation

Apple's Product Environmental Report shows that we must use an iPhone for around ten years so that CO_2 production for manufacturing the devices roughly matches the CO_2 production for its energy consumption. Hence, our goal must be to develop algorithms and models that run on hardware that is roughly ten years old!

iPhone-14	1 Year [%]	3 Years [%]	10 Years [%]
Production	90.8	79.0	56.0
Transport	2.3	1.9	1.4
Usage	6.9	18.0	42.5

(Percentages may not total 100 due to rounding. See `apple.com/environment/`)

Tree ensembles such as Random Forests are among the most-used classifiers in practice and improve the accuracy over a single tree by a large margin while still having manageable computational costs. Unfortunately, tree ensembles have the tendency to become large in practice and thereby use a lot of memory:

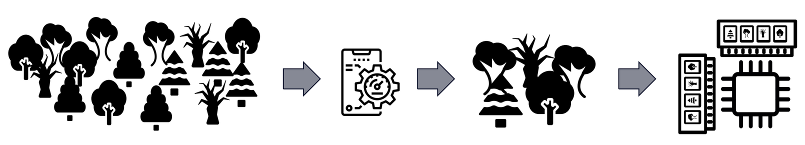
	adult	avila	bank	eeg	elec	mnist
accuracy [%]	86.78	98.58	90.39	93.42	88.98	96.53
model size [MB]	24.99	32.85	24.99	14.95	24.99	56.99

(5-fold cross-validated accuracy and model size of a Random Forest.)

Can we compute a small *and* accurate tree ensemble?

Ensemble Pruning

Ensemble Pruning is a standard technique to reduce the size of an already trained ensemble by removing unnecessary members.



More formally, given a large forest with M trees, the goal is to find a small sub-ensemble

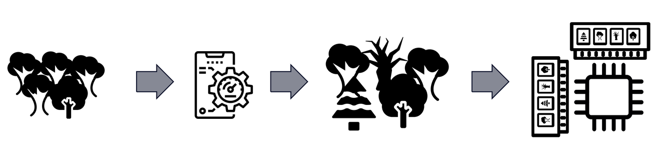
$$f_w(x) = \frac{1}{K}\sum_{i=1}^{M} w_i h_i(x)$$

by solving

$$\arg\min_{w\in\{0,1\}^M} \sum_{(x,y)\in\mathcal{S}} \ell\left(f_w(x), y\right) \text{ s.t. } \|w\|_0 = K \ll M$$

Leaf Refinement

Leaf Refinement is a technique specifically suited for tree ensembles. Instead of training and pruning a large forest, it first trains a small forest and then refines the probability estimates in the leaf nodes using a joint loss to capture interactions between the trees.



More formally, given the tree

$$h_i(x) = \sum_{j=1}^{L_i} y_j \pi_j(x),\ \ \pi_j(x) = 1 \text{ if x in leaf i else } 0$$

with leaves $\theta_i = (y_{i,1}, \dots, y_{i,L_i})$, $\theta = [\theta_1, \dots, \theta_M]$ refine the ensemble by solving

$$\arg\min_{\theta\in\mathbb{R}^{M\cdot L_1 \dots L_M}} \sum_{(x,y)\in\mathcal{S}} \ell\left(f_\theta(x), y\right)$$

Pruning and Refinement

Can we combine both approaches to find even smaller and better ensembles? Yes we can! By using L_1 regularization to enforce pruning while performing leaf-refinement:

$$\arg\min_{\substack{w\in[0,1]^M \\ \theta\in\mathbb{R}^{M\cdot L_1\dots L_M}}} \sum_{(x,y)\in\mathcal{S}} \ell\left(f_{w,\theta}(x), y\right) + \lambda\|w\|_1$$

Relaxed Constraints   Enforce pruning

Optimize both parameters

To optimize this objective, we use proximal gradient descent, which first performs a regular gradient descent step and then applies the `prox` operator to project the new solution onto the feasible set:

$$\beta^{t+1} \leftarrow \mathcal{P}_{R,\lambda}\left(\beta^t - \alpha^t \frac{1}{\|\nabla_{\beta^t} g_{\mathcal{B}}(\beta^t)\|}\nabla_{\beta^t} g_{\mathcal{B}}(\beta^t)\right)$$

$$\mathcal{P}_{\|\cdot\|_1,\lambda}(\beta)_i = \text{sgn}(\beta_i) max(0, |\beta_i| - \lambda)$$

Putting it all together

function PRUNE_AND_REFINE($\mathcal{T}, h_1, \dots, h_M$)
  $\theta_1, \dots, \theta_M \leftarrow$ `get_leafs`$(h_1, \dots, h_M)$ ▷ Load leafs
  $w_1, \dots, w_M \leftarrow$ `get_weights`$(h_1, \dots, h_M)$ ▷ Load weights
  for epoch $1, \dots, E$ **do** ▷ Perform PSGD for E epochs
    for next batch $\mathcal{B}$ in epoch **do**
      $w \leftarrow w - \alpha g_{\mathcal{B}}(w)$ ▷ Update weights
      $\theta \leftarrow \theta - \alpha g_{\mathcal{B}}(\theta)$ ▷ Update leafs
      $w \leftarrow \mathcal{P}_{\lambda,\|\cdot\|_1}(w)$ ▷ Apply the `prox` operator
  $H \leftarrow \emptyset, W \leftarrow \emptyset$
  for $i = 1, \dots, M$ **do**
    if $w_i \neq 0$ **then**
      h_i.`update_leafs`(θ_i) ▷ Copy new leaf probs.
      $H \leftarrow H \cup \{h_i\}$
      $W \leftarrow W \cup \{w_i\}$
  return H, W

This new pruning algorithm takes *any* tree ensembles as input and refines the probability estimates in the leaf nodes while removing unnecessary trees from the ensemble. λ controls the trade-off between pruning and refinement of the joint loss.

PyPruning Framework

To validate our method, we propose the `PyPruning` framework that implements 15 different Ensemble Pruning methods and which allows for easy extension of existing methods:

- **Ranking** Assign a score to each tree and select the top-k trees
- **Clustering** Cluster trees and then select a representative from each cluster
- **MQIP** Construct Mixed Quadratic Integer Program to select trees
- **Ordering** Order the trees according to their overall contribution and select the first K trees

```python
def error(i, ensemble_proba, selected_models, target):
    iproba = ensemble_proba[i,:,:]
    sub_proba = ensemble_proba[selected_models, :, :]
    pred = 1.0 / (1 + len(sub_proba)) * (sub_proba.sum(axis=0) + iproba)
    return (pred.argmax(axis=1) != target).mean()

n_base = 128
n_prune = 8
model = RandomForestClassifier(n_estimators=n_base)
model.fit(XTP, ytp)
pred = model.predict(Xtest)

pruned_model = GreedyPruningClassifier(n_prune, single_metric = error)
pruned_model.prune(Xtrain, ytrain, model.estimators_)
pred = pruned_model.predict(Xtest)
```

Conclusion

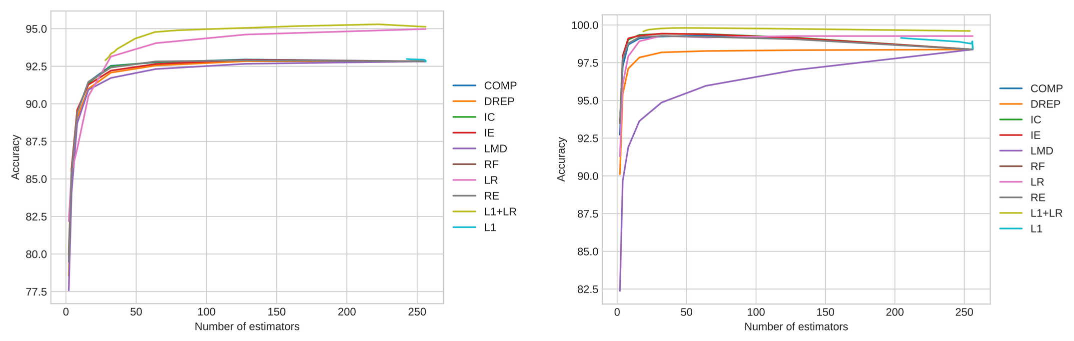
In a direct comparison with Random Forests, our novel Leaf-Refinement and Pruning algorithm offers much better accuracy on a variety of different datasets while using up to magnitudes less memory!

		adult	avila	bank	eeg	elec	mnist
RF	accuracy [%]	86.78	98.58	90.39	93.42	88.98	96.53
	model size [MB]	24.99	32.85	24.99	14.95	24.99	56.99
LR+L1	accuracy [%]	87.25	99.78	90.5	95.55	92.49	98.05
	model size [MB]	0.06	3.52	0.07	5.88	14.37	28.49

(5-fold cross-validated accuracy and model size of LR+L1 compared to RF.)

Detailed Analysis

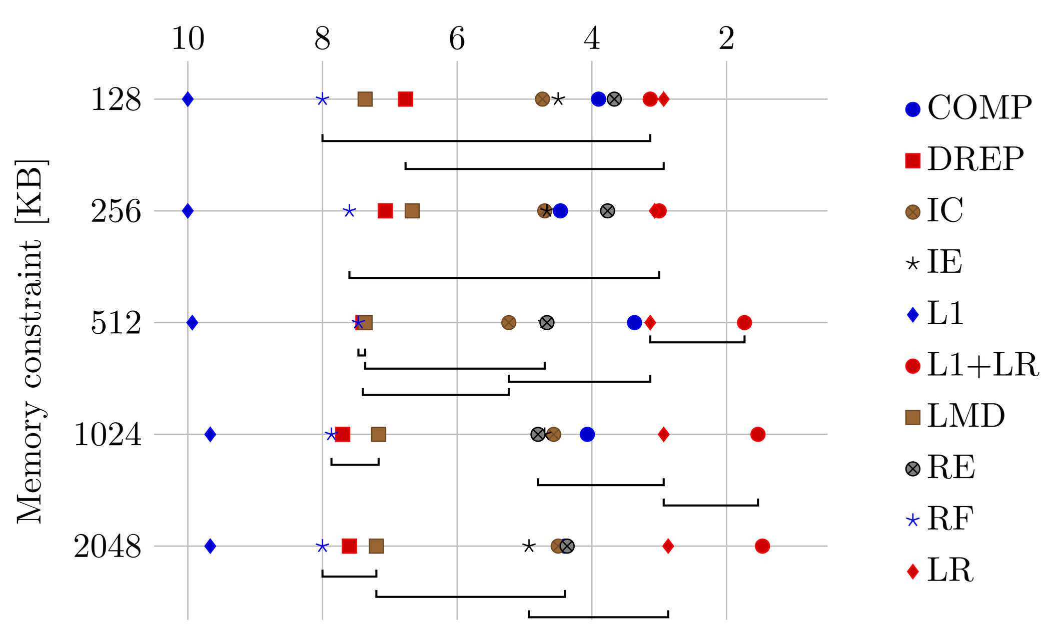
A more detailed analysis of our methods shows two distinct behaviors: Leaf-Refinement manages to outperform existing methods by a large margin (left plot). Second, Leaf-Refinement performs similar to existing methods but manages to keep its performance advantage for a large set of estimators in the pruned ensemble (right plot).



(Accuracy over the number of estimators on the EEG (left) and Avila (right) dataset.)

Systematic Study

A systematic study over 15 datasets with 10 different methods shows that our novel method outperforms existing methods on a large variety of datasets under various memory constraints. One can see that Leaf-Refinement generally performs best, and an additional L_1 regularization further improves the performance once more than 128 KB is available.



Check Out Our Code

sebastian.buschjaeger@tu-dortmund.de

github.com/sbuschjaeger/Pypruning/

github.com/sbuschjaeger/leaf-refinement-experiments

SCAN ME

technische universität dortmund · Fraunhofer IAIS · UNIVERSITÄT BONN · Fraunhofer IML

Funded by the Federal Ministry of Education and Research of Germany and the state of North Rhine-Westphalia as part of the Lamarr-Institute for Machine Learning and Artificial Intelligence.