Shrub Ensembles for Online Classification

Sebastian Buschjäger, Sibylle Hess and Katharina Morik
Thirty-Sixth AAAI Conference on Artificial Intelligence 2022

Artificial Intelligence Group@TU Dortmund University - Data Mining Group@ TU Eindhoven TU/e - Collaborative Research Center 876
Online Learning on the Edge

In 2018 Roughly 22 Billion IoT Devices world wide
By 2025 ≈ 38 Billion IoT Devices world wide\(^1\)

\(^1\)https://findstack.com/internet-of-things-statistics/
In 2018 Roughly 22 Billion IoT Devices world wide
By 2025 $\approx$ 38 Billion IoT Devices world wide\(^1\)

Clear benefits

+ Privacy and independence of data analysis
+ Reduced communication infrastructure costs
+ Faster response times

\(^1\)https://findstack.com/internet-of-things-statistics/
Shrub Ensembles for Online Classification by Sebastian Buschjäger, Sibylle Hess and Katharina Morik
Requirements for Online Learning on the Edge

**Computational efficiency**

The algorithm must process examples at least as fast as new examples arrive.
Requirements for Online Learning on the Edge

Computational efficiency
The algorithm must process examples at least as fast as new examples arrive.

Memory efficiency
The algorithm has only a limited budget of memory and fails if more memory is required.
Requirements for Online Learning on the Edge

**Computational efficiency**
The algorithm must process examples at least as fast as new examples arrive.

**Memory efficiency**
The algorithm has only a limited budget of memory and fails if more memory is required.

**Evolving data streams**
The algorithm must adapt to changes in the new data distribution and preserve its performance.
Online Decision Tree Ensemble Learning
Online Learning on the Edge

Online Decision Tree Ensemble Learning

Gradient-based Learners

+ constant memory consumption
+ joint optimization of all trees
- backpropagation through entire tree
Online Decision Tree Ensemble Learning

**Gradient-based Learners**

- constant memory consumption
- joint optimization of all trees
  - backpropagation through entire tree

**Incremental Learners**

- fast
- proven in practice
  - nodes are not removed
Can we design simple and small online ensembles?
We can use a batch algorithm to train $h(x)$

- We quickly adapt to concept drift
- No long-term learning possible
Revisiting Sliding Windows

\[ \ldots, (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6), (x_7, ?), (x_8, ?), (x_9, ?), \ldots \]

- We can use a batch algorithm to train \( h(x) \)
- We quickly adapt to concept drift
- No long-term learning possible
We can use a batch algorithm to train $h(x)$

+ We quickly adapt to concept drift

- No long-term learning possible
Long-Term Learning With Sliding Windows

\[ \ldots (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, ?) \ldots, (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, ?) \ldots, \]

**Often** Local Patterns repeat over long running processes

**Hence** The two classifiers \( h_1 \) and \( h_2 \) are redundant. We should use \( h_1 \)
Long-Term Learning With Sliding Windows

\[ \ldots \left( x_1, y_1 \right), \left( x_2, y_2 \right), \left( x_3, y_3 \right), \left( x_4, y_4 \right), \left( x_5, y_5 \right), \left( x_6, ? \right) \ldots, \left( x_1, y_1 \right), \left( x_2, y_2 \right), \left( x_3, y_3 \right), \left( x_4, y_4 \right), \left( x_5, y_5 \right), \left( x_6, ? \right) \ldots \]

\[ \text{train} \rightarrow h_1(x) \]

\[ \text{train} \rightarrow h_2(x) \]

**Often** Local Patterns repeat over long running processes

**Hence** The two classifiers \( h_1 \) and \( h_2 \) are redundant. We should use \( h_1 \)

**Idea** Learn an ensemble of local patterns from sliding windows

\[ f(x) = \sum_{i=1}^{M} w_i h_i(x) \]
Formally Let $\mathcal{H} = \{h_1, h_2, \ldots, h_K\}$ be the set of trees learned on local patterns:

$$\arg \min_{w \in \mathbb{R}^K} \sum_{t=1}^{T} \ell \left( f_{S[0:t-1]}(x_t), y_t \right) \text{ s.t. } \|w\|_0 \leq M, w_i \geq 0, \sum_{i=1}^{K} w_i = 1$$

with

- $f_{S[0:t-1]} : \mathbb{R}^d \rightarrow \mathbb{R}^C$ is the model at time $t$
- $\ell : \mathbb{R}^C \times \mathcal{Y} \rightarrow \mathbb{R}_+$ is the loss
Formally, let $\mathcal{H} = \{h_1, h_2, \ldots, h_K\}$ be the set of trees learned on local patterns:

$$\arg\min_{w \in \mathbb{R}^K} \sum_{t=1}^{T} \ell \left( f_{S[0:t-1]}(x_t), y_t \right) \text{ s.t. } \|w\|_0 \leq M, \ w_i \geq 0, \sum_{i=1}^{K} w_i = 1$$

with

- $f_{S[0:t-1]} : \mathbb{R}^d \to \mathbb{R}^C$ is the model at time $t$
- $\ell : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}_+$ is the loss
- $M \geq 1$ is the maximum number of ensemble members
Formally Let $\mathcal{H} = \{h_1, h_2, \ldots, h_K\}$ be the set of trees learned on local patterns:

$$\arg\min_{w \in \mathbb{R}^K} \sum_{t=1}^{T} \ell \left( f_{S[0:t-1]}(x_t), y_t \right) \text{ s.t. } \|w\|_0 \leq M, w_i \geq 0, \sum_{i=1}^{K} w_i = 1$$

with

- $f_{S[0:t-1]} : \mathbb{R}^d \rightarrow \mathbb{R}^C$ is the model at time $t$
- $\ell : \mathbb{R}^C \times \mathcal{Y} \rightarrow \mathbb{R}_+$ is the loss
- $M \geq 1$ is the maximum number of ensemble members
- $\|w\|_0 = \sum_{i=1}^{K} \mathbb{1}\{w_i \neq 0\}$ is the 0-norm
Formally Let $\mathcal{H} = \{h_1, h_2, \ldots, h_K\}$ be the set of trees learned on local patterns:

$$\arg\min_{w \in \mathbb{R}^K} \sum_{t=1}^{T} \ell \left( f_{S[0:t-1]}(x_t), y_t \right) \text{ s.t. } \|w\|_0 \leq M, w_i \geq 0, \sum_{i=1}^{K} w_i = 1$$

with

- $f_{S[0:t-1]} : \mathbb{R}^d \to \mathbb{R}^C$ is the model at time $t$
- $\ell : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}_+$ is the loss
- $M \geq 1$ is the maximum number of ensemble members
- $\|w\|_0 = \sum_{i=1}^{K} 1\{w_i \neq 0\}$ is the 0-norm
Formally Let $\mathcal{H} = \{h_1, h_2, \ldots, h_K\}$ be the set of trees learned on local patterns:

$$\arg \min_{w \in \mathbb{R}^K} \sum_{t=1}^{T} \ell \left( f_{S[0:t-1]}(x_t), y_t \right) \text{ s.t. } \|w\|_0 \leq M, w_i \geq 0, \sum_{i=1}^{K} w_i = 1$$

with

- $f_{S[0:t-1]} : \mathbb{R}^d \rightarrow \mathbb{R}^C$ is the model at time $t$
- $\ell : \mathbb{R}^C \times \mathcal{Y} \rightarrow \mathbb{R}_+$ is the loss
- $M \geq 1$ is the maximum number of ensemble members
- $\|w\|_0 = \sum_{i=1}^{K} 1\{w_i \neq 0\}$ is the 0-norm

For this talk Focus on the MSE loss

$$\ell(f_{S[0:t-1]}(x_t), y_t) = \frac{1}{C} \|f_{S[0:t-1]}(x_t) - y_t\|^2$$
Proximal Gradient Descent

\[ w \leftarrow \mathcal{P} \left( w - \alpha \nabla_w L_B(w) \right), \]

where

- \( B \) is the current window with \( |B| = B \) examples
**Proximal Gradient Descent**

\[ w \leftarrow P\left(w - \alpha \nabla_w L_B(w)\right), \]

where

- \( B \) is the current window with \(|B| = B\) examples
- \( \nabla_w L_B(w) \) is the gradient on the current window
Proximal Gradient Descent

\[ w \leftarrow \mathcal{P} \left( w - \alpha \nabla_w L_B(w) \right), \]

where

- \( \mathcal{B} \) is the current window with \( |\mathcal{B}| = B \) examples
- \( \nabla_w L_B(w) \) is the gradient on the current window
- \( \alpha \in \mathbb{R}_+ \) is the step-size
Online Constraint Optimization

Proximal Gradient Descent

\[ w \leftarrow P \left(w - \alpha \nabla_w L_B(w)\right), \]

where

- \( B \) is the current window with \(|B| = B\) examples
- \( \nabla_w L_B(w) \) is the gradient on the current window
- \( \alpha \in \mathbb{R}_+ \) is the step-size
- \( P(w) \) is the prox-operator for the feasible set \( \Delta = \left\{ w \in \mathbb{R}_+^K \mid \sum_{i=1}^K w_i = 1, \|w\|_0 = M \right\} \)
Online Constraint Optimization

Proximal Gradient Descent

\[ w \leftarrow \mathcal{P} \left( w - \alpha \nabla_w L_B(w) \right), \]

where

- \( B \) is the current window with \(|B| = B\) examples
- \( \nabla_w L_B(w) \) is the gradient on the current window
- \( \alpha \in \mathbb{R}_+ \) is the step-size
- \( \mathcal{P}(w) \) is the prox-operator for the feasible set \( \Delta = \{ w \in \mathbb{R}_+^K \mid \sum_{i=1}^K w_i = 1, \| w \|_0 = M \} \)

Our paper / Kyrillidis et al. 2013 Details for prox-operator
Idea If a tree is helpful to the ensemble, then it should have a large weight

---

Algorithm 1: Shrub Ensembles.

1: \(w \leftarrow (0); \mathcal{B} \leftarrow []; \mathcal{H} \leftarrow []\) \(\triangleright\) Init.
2: \(\textbf{for} \ \text{next item } (x, y) \ \textbf{do}\)
3: \(\textbf{if } |\mathcal{B}| = B \ \textbf{then}\) \(\triangleright\) Update batch
4: \(\mathcal{B}.\text{pop}_\text{first}()\)
5: \(\mathcal{B}.\text{append}((x, y))\)
6: \(h_{new} \leftarrow \text{train}(\mathcal{B})\) \(\triangleright\) Add new classifier
7: \(\mathcal{H}.\text{append}(h_{new})\)
8: \(w \leftarrow (w_1, \ldots, w_M, 0)\) \(\triangleright\) Initialize weight
9: \(w \leftarrow w - \alpha \nabla_w L_B(w)\) \(\triangleright\) Gradient step
10: \(w, \mathcal{H} \leftarrow \text{sorted}(w, \mathcal{H})\) \(\triangleright\) Sort decreasing order
11: \(w \leftarrow \mathcal{P}(w)\) \(\triangleright\) Project on feasible set
12: \(w, \mathcal{H} \leftarrow \text{prune}(w, \mathcal{H})\) \(\triangleright\) Remove zero weights
Theoretical Insights

**Runtime** $O\left(dB^2 \log B + \log M\right)$ per example with d features
Theoretical Insights

**Runtime** \( \mathcal{O}(dB^2 \log B + \log M) \) per example with \( d \) features

CART → shrubs

Shrub Ensembles for Online Classification by Sebastian Buschjäger, Sibylle Hess and Katharina Morik
Theoretical Insights

Runtime $O\left(dB^2 \log B + \log M\right)$ per example with $d$ features

CART   prox operator

Let $m \leq M$ be the number of models in the ensemble and let $\forall j = 1, \ldots, m$: $h_j(x_B) \neq y_B$.

If SE trains fully-grown trees such that $\forall i = 1, \ldots, B$: $h(x_i) = y_i$ it holds for $\alpha > B C^2 / 4 m$ that:

1. If $m < M$, then $h$ is added to the ensemble
2. If $m = M$, then $h$ replaces the tree with the smallest weight from the ensemble
Theoretical Insights

**Runtime** $O\left(\frac{d^2 B^2 \log B}{2} + \log M\right)$ per example with $d$ features

**Memory** $O\left(d B + 2 \cdot B \cdot (M + 1)\right)$ per example with $d$ features

---

CART $\xrightarrow{\text{prox operator}}$ trees $\rightarrow$ shrubs
Theoretical Insights

**Runtime** $O \left( dB^2 \log B + \log M \right)$ per example with $d$ features

**Memory** $O \left( dE + 2 \cdot B \cdot (M + 1) \right)$ per example with $d$ features
Theoretical Insights

**Runtime** $O(dB^2 \log B + \log M)$ per example with d features

CART → prox operator

**Memory** $O(dE + 2 \cdot B \cdot (M + 1))$ per example with d features

window → trees → shrubs
Theoretical Insights

**Runtime** $\mathcal{O}(d B^2 \log B + \log M)$ per example with $d$ features

**Memory** $\mathcal{O}(d E + 2 \cdot B \cdot (M + 1))$ per example with $d$ features

**Behaviour** Let $m \leq M$ be the number of models in the ensemble and let $\forall j = 1, \ldots, m: h_j(x_B) \neq y_B$. If SE trains fully-grown trees such that $\forall i = 1, \ldots, B: h(x_i) = y_i$ it holds for $\alpha > \frac{BC}{4m}$ that:

- (1) If $m < M$, then $h$ is added to the ensemble
- (2) If $m = M$, then $h$ replaces the tree with the smallest weight from the ensemble
**Goal** Compare Accuracy-Memory Trade-off of different configurations

1) Plot Pareto Front of best performing configurations against accuracy
2) Compute Area Under the Pareto Front to measure accuracy-memory trade-off
3) Rank each method according to its trade-off and plot a CD-diagram
**Goal** Compare Accuracy-Memory Trade-off of different configurations

1) Plot Pareto Front of best performing configurations against accuracy
2) Compute Area Under the Pareto Front to measure accuracy-memory trade-off
3) Rank each method according to its trade-off and plot a CD-diagram

- SE: Shrub Ensembles
- SRP: Streaming Random Patches
- ARF: Adaptive Random Forest
- HTT: HoeffdingAnyTree
- SB: (Online) Smooth Boost
- Bag: (Online) Bagging
- HT: HoeffdingTrees
- NB: (Online) NaiveBayes
- SDT: Soft Decision Trees
Experimental Insights: Qualitative Analysis (1)

Shrub Ensembles for Online Classification by Sebastian Buschjäger, Sibylle Hess and Katharina Morik
Experimental Insights: Qualitative Analysis (2)

Shrub Ensembles for Online Classification by Sebastian Buschjäger, Sibylle Hess and Katharina Morik
Edge learning becomes more important every year

- Computational efficiency
- Memory efficiency
- Evolving data streams
Shrub Ensembles for Online Classification

**Edge learning becomes more important every year**

- Computational efficiency
- Memory efficiency
- Evolving data streams

**Current approaches** Unbounded memory or costly gradients
Edge learning becomes more important every year

- Computational efficiency
- Memory efficiency
- Evolving data streams

**Current approaches** Unbounded memory or costly gradients

**Shrub Ensembles** Bounded memory and simple gradients

- Train small trees (→ Shrubs) on sliding window
- Maintain tree weights via proximal gradient descent
- Prune unimportant trees

**Results** Better accuracy-memory trade-off than existing methods!