Mondrian Forests:
Efficient Online Random Forests

Balaji Lakshminarayanan (Gatsby Unit, UCL)
Daniel M. Roy (Cambridge → Toronto)
Yee Whye Teh (Oxford)

September 4, 2014
Outline

Background and Motivation

Mondrian Forests
  Mondrian process distribution over $\mathcal{T}$
  Online learning

Experiments

Conclusion
Outline

Background and Motivation

Mondrian Forests
  Mondrian process distribution over $\mathcal{T}$
  Online learning

Experiments

Conclusion
Introduction

• **Input**: attributes $X = \{ x_i \}_{i=1}^N$, labels $Y = \{ y_i \}_{i=1}^N$ (i.i.d)
• $y_i \in \{1, \ldots, K\}$ (classification) or $y_i \in \mathbb{R}$ (regression)
• **Goal**: Predict $y_*$ for test data $x_*$
Introduction

- **Input:** attributes \( X = \{x_i\}_{i=1}^N \), labels \( Y = \{y_i\}_{i=1}^N \) \( \text{(i.i.d)} \)
- \( y_i \in \{1, \ldots, K\} \) \( \text{(classification)} \) or \( y_i \in \mathbb{R} \) \( \text{(regression)} \)
- **Goal:** Predict \( y_\ast \) for test data \( x_\ast \)
- **Recipe for prediction:** Use a ‘random forest’
  - Ensemble of randomized decision trees
  - State-of-the-art for lots of real world prediction tasks
  - ‘Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning’ [Criminisi et al., 2012]
Example: Classification tree

- Hierarchical axis-aligned binary partitioning of input space
- Rule for predicting label within each block

\[ x_1 > 0.37, \quad x_2 > 0.5 \]

\[ B_j \]: list of nodes, feature-id + location of splits for non-leaf nodes
\[ \theta \]: Multinomial parameters at leaf nodes
Random forest (RF)

- Averaged over iid randomized decision trees $\mathcal{T}_1, \ldots, \mathcal{T}_M$ conditioned on $X$ and $Y$.
  \[
  p(y_*|x_*) = \frac{1}{M} \sum_{m} p(y_*|x_*, \mathcal{T}_m, X, Y)
  \]

- Combining multiple decision trees significantly improves predictive performance over single trees.
- Technique for variance reduction, not bias reduction.
- Model combination, not Bayesian model averaging.
Random forest (RF)

- **Breiman’s Random Forest** [Breiman, 2001]: Bagging + Randomly subsample features and choose best split amongst subsampled features, optimising over all split locations.
- **Extremely Randomized Trees** [Geurts et al., 2006] (ERT-\(k\)): Randomly sample \(k\) (feature-id, location) pairs and choose the best split amongst this subset
  - no bagging
  - ERT-1 does not use labels \(Y\) to guide splits!
Pros and Cons

• Advantages of RF
  – Excellent predictive performance (test accuracy)
  – Fast to train (in batch setting) and test
  – Trees can be trained in parallel
  – No overfitting
Pros and Cons

- **Advantages of RF**
  - Excellent predictive performance (test accuracy)
  - Fast to train (in batch setting) and test
  - Trees can be trained in parallel
  - No overfitting

- **Not possible to train incrementally**
  - Re-training batch version periodically is slow $O(N^2 \log N)$ and requires access to past data
  - Existing online RF variants [Saffari et al., 2009, Denil et al., 2013] require
    - lots of memory / computation (impractical) or
    - need lots of training data before they can deliver good test accuracy (data inefficient)
Pros and Cons

- **Advantages of RF**
  - Excellent predictive performance (test accuracy)
  - Fast to train (in batch setting) and test
  - Trees can be trained in parallel
  - No overfitting

- **Not possible to train incrementally**
  - Re-training batch version periodically is slow $O(N^2 \log N)$ and requires access to past data
  - Existing online RF variants [Saffari et al., 2009, Denil et al., 2013] require
    - lots of memory / computation (impractical) or
    - need lots of training data before they can deliver good test accuracy (data inefficient)

**Mondrian forests** = Mondrian process + Random forests

- Can operate in either batch mode or online mode
- Online speed $O(N \log N)$
- Data efficient (predictive performance of online mode equals that of batch mode!)
Outline

Background and Motivation

Mondrian Forests
  Mondrian process distribution over $T$
  Online learning

Experiments

Conclusion
Mondrian process

Figure: Mondrian Composition II in Red, Blue and Yellow (Source: Wikipedia)

- A stochastic process over binary hierarchical axis-aligned partitions of $\mathbb{R}^d$ [Roy and Teh, 2009].
Generative process: $\mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$

1. Draw $\Delta_\epsilon$ from exponential with rate $u_1 - \ell_1 + u_2 - \ell_2$
2. IF $\Delta_\epsilon > \lambda$ stop,
Generative process: $\mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$

1. Draw $\Delta_\epsilon$ from exponential with rate $u_1 - \ell_1 + u_2 - \ell_2$
2. **IF** $\Delta_\epsilon > \lambda$ stop,
3. **ELSE**, sample a split
   - Split dimension: choose dimension $j$ with prob $\propto u_j - \ell_j$
   - Split location: choose cut location uniformly from $[\ell_j, u_j]$
Generative process: $\mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$

1. Draw $\Delta_\epsilon$ from exponential with rate $u_1 - \ell_1 + u_2 - \ell_2$
2. **IF** $\Delta_\epsilon > \lambda$ stop,
3. **ELSE**, sample a split
   - Split dimension: choose dimension $j$ with prob $\propto u_j - \ell_j$
   - Split location: choose cut location uniformly from $[\ell_j, u_j]$
   - Recurse on left and right subtrees with parameter $\lambda - \Delta_\epsilon$
Self-consistency of Mondrian process

- Simulate $\mathcal{T} \sim \mathcal{MP}(\lambda, [l_1, u_1], [l_2, u_2])$
Self-consistency of Mondrian process

• Simulate $\mathcal{T} \sim \mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$
• Restrict $\mathcal{T}$ to a smaller rectangle $[\ell'_1, u'_1] \times [\ell'_2, u'_2]$
Self-consistency of Mondrian process

- Simulate $\mathcal{T} \sim \mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$
- Restrict $\mathcal{T}$ to a smaller rectangle $[\ell'_1, u'_1] \times [\ell'_2, u'_2]$

$\mathcal{MP}$ is the restriction to $[\ell_1, u_1] \times [\ell_2, u_2]$. Such that $\mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$ is the restriction to $[\ell_1, u_1] \times [\ell_2, u_2]$. 

- Restriction has distribution $\mathcal{MP}(\lambda, [\ell'_1, u'_1], [\ell'_2, u'_2])$!
Self-consistency of Mondrian process

- Simulate $\mathcal{T} \sim \mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$
- Restrict $\mathcal{T}$ to a smaller rectangle $[\ell_1', u_1'] \times [\ell_2', u_2']$

Restriction has distribution $\mathcal{MP}(\lambda, [\ell_1', u_1'], [\ell_2', u_2'])$!

Well-defined extension to $\mathcal{MP}(\lambda, \mathbb{R}, \mathbb{R})$, such that $\mathcal{MP}(\lambda, [\ell_1, u_1], [\ell_2, u_2])$ is the restriction to $[\ell_1, u_1] \times [\ell_2, u_2]$. 
Mondrian trees

- Use $\mathcal{MP}(\lambda, [\ell_1, u_1], \ldots, [\ell_d, u_d])$ as prior over decision trees $p(T|X)$, where the range is given by $X$. 
Mondrian trees

- Use $\mathcal{MP}(\lambda, [\ell_1, u_1], \ldots, [\ell_d, u_d])$ as prior over decision trees $p(\mathcal{T} | X)$, where the range is given by $X$.
- Self-consistency:
  - Equivalent to a prior over trees defined on $\mathbb{R}^d$ and independent of $X$.
  - $p(\mathcal{T} | X)$ is simply the restriction to range of $X$. 

Online learning:
- As dataset grows, we simply unveil $\mathcal{T}$ on a larger range.
- We can enlarge the visible range by simulating from a conditional Mondrian process.
- Distribution of trees in offline and online modes are the same!
- Order of the data points does not matter.
Mondrian trees

• Use $\mathcal{MP}(\lambda, [\ell_1, u_1], \ldots, [\ell_d, u_d])$ as prior over decision trees $p(T|X)$, where the range is given by $X$.

• Self-consistency:
  – Equivalent to a prior over trees defined on $\mathbb{R}^d$ and independent of $X$.
  – $p(T|X)$ is simply the restriction to range of $X$.

• Online learning:
  – As dataset grows, we simply unveil $T$ on a larger range.
  – We can enlarge the visible range by simulating from a conditional Mondrian process.
  – Distribution of trees in offline and online modes are the same!
  – Order of the data points does not matter.
Start with data points $a$ and $b$
Online learning cartoon

Adding new data point \( c \): update range

\[
x_2 > 0.23
\]

\( x_2 \)
Adding new data point $c$: introduce new split above existing one
Online learning cartoon

Adding new data point $d$: traverse to left child and update range

$x_1 > 0.75$

$x_2 > 0.23$

$0 < x_1 < 1$

$0 < x_2 < 1$

$1 < x_2$
Online learning cartoon

Adding new data point \(d\): extend the existing split to new range

\[
x_2 > 0.23 \\
x_1 > 0.75
\]
Online learning cartoon

Adding new data point $d$: split leaf further

$x_2 > 0.23$

$x_1 > 0.47$

$x_1 > 0.75$
Key differences between Mondrian forests and existing online random forests

• Splits not extended to unseen regions
• New split can be introduced *anywhere* in the tree (as long as it is consistent with current tree)
• The size and lifetime of a node control probability of new splits being introduced
• Self-consistent hierarchical Bayesian prior on the leaf parameters (not discussed).
Outline

Background and Motivation

Mondrian Forests
  Mondrian process distribution over $\mathcal{T}$
  Online learning

Experiments

Conclusion
Experimental setup

- Datasets:

<table>
<thead>
<tr>
<th>Name</th>
<th>D</th>
<th>#Classes</th>
<th>#Train</th>
<th>#Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satellite images</td>
<td>36</td>
<td>6</td>
<td>3104</td>
<td>2000</td>
</tr>
<tr>
<td>Letter</td>
<td>16</td>
<td>26</td>
<td>15000</td>
<td>5000</td>
</tr>
<tr>
<td>USPS</td>
<td>256</td>
<td>10</td>
<td>7291</td>
<td>2007</td>
</tr>
<tr>
<td>DNA</td>
<td>180</td>
<td>3</td>
<td>1400</td>
<td>1186</td>
</tr>
</tbody>
</table>

- Training data split into 100 mini batches (unfair to MF)
- Number of trees = 100
- Existing randomised decision trees:
  - Periodically retrained offline methods RF, ERT-1, ERT-k.
  - Online RF [Saffari et al., 2009]
Data efficiency: Online MF very close to offline Breiman’s RF and ERT, and significantly outperforms ORF-Saffari.

Speed: MF much faster than periodically re-trained offline RF and ERT, as well as online RF.
Figure: Test accuracy
Satellite Images

![Fraction of training data vs Test accuracy graph]

- MF
- ERT-\(k\)
- ERT-1
- ORF-Saffari
- Breiman-RF*

Figure: Test accuracy
DNA

Figure: Test accuracy

- **Irrelevant features**: Choosing splits independent of labels (MF, ERT-1) harmful in presence of irrelevant features
- Removing irrelevant features (use only the 60 most relevant features\(^1\)) improves test accuracy (MF\(^\dagger\), ERT-1\(^\dagger\))

\(^1\)https://www.sgi.com/tech/mlc/db/DNA.names
Outline

Background and Motivation

Mondrian Forests
  Mondrian process distribution over $\mathcal{T}$
  Online learning

Experiments

Conclusion
Conclusion

- MF: Alternative to RF that supports incremental learning
- Computationally faster compared to existing online RF and periodically re-trained Breiman-RF, ERT
- Future work:
  - Mondrian forests for high dimensional data with lots of irrelevant features.
  - Use labels to guide splits in MF (e.g. using ERT-$k$ ideas)
Thank you!


code: http://www.gatsby.ucl.ac.uk/~balaji/mondrianforest/

Questions?


In *Computer Vision Workshops (ICCV Workshops)*. IEEE.

A hierarchical Bayesian language model based on Pitman–Yor processes.
Hierarchical prior over $\theta$

- $G_j$ parametrizes $p(y|x)$ in $B_j^x$
- Normalized stable process (NSP): special case of PYP where concentration = 0
- $d_j \in (0, 1)$ is discount for node $j$
- $G_\epsilon | H \sim NSP(d_\epsilon, H)$,
  $G_{j0} | G_j \sim NSP(d_{j0}, G_j)$,
  $G_{j1} | G_j \sim NSP(d_{j1}, G_j)$
- $\mathbb{E}[G_\epsilon(s)] = H(s)$
- $\text{Var}[G_\epsilon(s)] = (1 - d_H)H(s)(1 - H(s))$
- Closed under Marginalization: $G_0 | H \sim NSP(d_\epsilon d_0, H)$
- $d_j = e^{-\gamma \Delta j}$ where $\Delta j$ is the lifetime of node $j$
Posterior inference for NSP

- Special case of approximate inference for PYP [Teh, 2006]
- Chinese restaurant process representation
- **Interpolated Kneser-Ney smoothing**
  - fast approximation
  - Restrict number of tables serving a dish to at most 1
  - IKN popular smoothing technique in language modeling
Prediction

• Extend Mondrian to range of test data (similar to training)
  – Test data point can potentially branch off and form separate leaf node of its own (unlike conventional decision trees)
  – If test point is in its own node, prediction is made from the (hierarchical) prior
  – Points far away from range of training data are more likely to lie in their own node
  – We analytically average over every possible extension (unlike training where we sample an extension)
  – Computational complexity linear in tree depth $\approx \log(N)$

• Prediction interpolates between observed labels and prior depending on how close test data point is to training data