LassoNet: A Neural Network with Feature Sparsity

Ismael Lemhadri
Joint work with Feng Ruan and Rob Tibshirani

Stanford University

July 9, 2020

Talk Materials at: https://tinyurl.com/lassonet
Modern Machine Learning

- Large, complex models
- Massive amounts of data
The ILSVRC Competition

Error Rate in Image Classification (%)

Human Performance Zone

Neural Network Architecture

Deep learning: applications

- Healthcare
- Drug discovery
- Recommender systems
Deep learning: applications

- healthcare
- drug discovery
- recommender systems

Also: gene sequencing, advertisement, speech recognition ...
Deep learning: applications

healthcare  

drug discovery  

recommender systems

Also: gene sequencing, advertisement, speech recognition ...

Deep learning pervades data-rich problems
Benefits of feature selection

▶ reduces overfitting
▶ improves accuracy
▶ helps overcome the curse of dimensionality
▶ allows shorter training time
▶ aids with interpretability
Mice Protein Data

Find proteins that are discriminant between healthy and trisomic mice. 1080 measurements, 77 proteins.[Higuera et al., 2015]

Best six proteins: AKT, NR2B, TIAM1, nNOS, RRP1, GluR3
Prior art

- Filter and wrapper methods
- Embedded methods
Prior art

- **Filter and wrapper methods**
  - Individual scores [Fisher score, Laplacian Score, Trace Ratio]
  - Kernel based methods
  - Mutual information based methods [HSIC-Lasso (Yamada et al., 2014), Conditional covariance minimization (Jordan et al., 2018)]

- **Embedded methods**
  - L1-regularization [Lasso (Tibshirani, 1996) and variants]
Desiderata

- Capture *arbitrary* nonlinearity [nonparametric approach]
- Achieve *adaptive* feature selection
Desiderata

- Capture *arbitrary* nonlinearity [nonparametric approach]
- Achieve *adaptive* feature selection

Today’s proposal:
- An embedded method
- Optimizes over a large function class
- Obeys a natural hierarchy principle
Demonstrating LassoNet on MNIST. Simultaneously selecting informative pixels and classifying digit 5 vs. digit 6. **Top:** The classification accuracy by number of selected features. **Bottom:** A sample from the model with 160, 220 and 300 active features out of the 784.
The hierarchy principle

"Large component main effects are more likely to lead to appreciable interactions than small components. Also, the interactions corresponding to larger main effects may be in some sense of more practical importance."

David Cox, 1980

Photo: General Motors Cancer Research Foundation
The hierarchy principle

"Large component main effects are more likely to lead to appreciable interactions than small components. Also, the interactions corresponding to larger main effects may be in some sense of more practical importance."

David Cox, 1980

Photo: General Motors Cancer Research Foundation

More recently: Lasso for hierarchical interactions (Bien et al., 2013), reluctant interaction modelling (R.J. Tibshirani, 2019)
Our approach

▶ An embedded method

▶ Large function class: residual feedforward neural networks

\[ F = \left\{ f : f(x) = \theta^T x + f_W(x) \right\} \]
LassoNet

- **Objective function:**

\[
\begin{align*}
\text{minimize} & \quad L(\theta, W) + \lambda \|\theta\|_1 \\
\text{subject to} & \quad \|W^{(0)}\|_{j_\infty} \leq M|\theta_j|, \ j = 1, \ldots, d.
\end{align*}
\]

where \(W^{(0)}\) denotes the network’s input layer.
LassoNet

Objective function:

\[
\min_{\theta, W} \ L(\theta, W) + \lambda \|\theta\|_1
\]

subject to \( \|W^{(0)}\|_\infty \leq M|\theta_j|, \ j = 1, \ldots, d. \)

where \( W^{(0)} \) denotes the network’s input layer.

In particular, \( W_j = 0 \) as soon as \( \theta_j = 0. \)
LassoNet

- **Objective function:**

  \[
  \min_{\theta, W} L(\theta, W) + \lambda \|\theta\|_1
  \]

  subject to \( \|W^{(0)}\|_{j\infty} \leq M|\theta_j|, j = 1, \ldots, d. \)

  where \( W^{(0)} \) denotes the network’s input layer.

  In particular, \( W_j = 0 \) as soon as \( \theta_j = 0. \)

- **Hyper-parameters:**
  - \( \ell_1 \) penalty, \( \lambda \). Higher values of \( \lambda \) encourage sparser models
  - Hierarchy parameter, \( M \). Controls the relative strength of the linear and nonlinear components.
LassoNet Training Loop

Algorithm 1 Training LassoNet

1: \textbf{Input:} training dataset $X \in \mathbb{R}^{n \times d}$, training labels $Y$, feed-forward neural network $f_W(\cdot)$, number of epochs $B$, hierarchy multiplier $M$, path multiplier $\epsilon$, learning rate $\alpha$
2: Initialize and train the feed-forward network on the loss $L(X, Y; \theta, W)$
3: Initialize the penalty, $\lambda = \epsilon$, and the number of active features, $k = d$
4: \textbf{while} $k > 0$ \textbf{do}
5: \quad Update $\lambda \leftarrow (1 + \epsilon)\lambda$
6: \quad \textbf{for} $b \in \{1 \ldots B\}$ \textbf{do}
7: \quad \quad Compute gradient of the loss w.r.t to $\theta$ and $W$ using backpropagation
8: \quad \quad Update $\theta \leftarrow \theta - \alpha \nabla \theta L$ and $W \leftarrow W - \alpha \nabla_w L$
9: \quad \quad Update $(\theta, W^{(0)}) = \text{HIER-PROX}(\theta, W^{(0)}, \lambda, M)$
10: \quad \textbf{end for}
11: \quad Update $k$ to be the number of non-zero coordinates of $\theta$
12: \textbf{end while}
Classification accuracies for LassoNet on a hold-out test-set.

Results on the MICE protein dataset where $n = 864$, $d = 77$. 
Algorithm 1 Training LassoNet

1: Input: training dataset $X \in \mathbb{R}^{n \times d}$, training labels $Y$, feed-forward neural network $f_W(\cdot)$, number of epochs $B$, hierarchy multiplier $M$, path multiplier $\epsilon$, learning rate $\alpha$

2: Initialize and train the feed-forward network on the loss $L(X, Y; \theta, W)$

3: Initialize the penalty, $\lambda = \epsilon$, and the number of active features, $k = d$

4: while $k > 0$ do
5:    Update $\lambda \leftarrow (1 + \epsilon)\lambda$
6:    for $b \in \{1 \ldots B\}$ do
7:        Compute gradient of the loss w.r.t to $\theta$ and $W$ using backpropagation
8:        Update $\theta \leftarrow \theta - \alpha \nabla_{\theta} L$ and $W \leftarrow W - \alpha \nabla_{\omega} L$
9:        Update $(\theta, W^{(0)}) = \text{HIER-PROX}(\theta, W^{(0)}, \lambda, M)$
10:       Apply early-stopping criterion
11:    end for
12:    Update $k$ to be the number of non-zero coordinates of $\theta$
13: end while
The power of warm starts

- The sparse to dense optimization along the path efficiently explores the nonconvex landscape.
- Training combines warm starts and early stopping.
- The bulk of the computational cost goes to training the dense model.
- This is effectively pruning.
The power of warm starts

- Training combines warm starts and early stopping
- The bulk of the computational cost goes to training the dense model.
- This is effectively pruning

The sparse to dense optimization along the path efficiently explores the nonconvex landscape.
The power of warm starts

- The sparse to dense optimization along the path efficiently explores the nonconvex landscape.
- Training combines warm starts and early stopping.
The power of warm starts

- The sparse to dense optimization along the path efficiently explores the nonconvex landscape.
- Training combines warm starts and early stopping.
- The bulk of the computational cost goes to training the dense model.
- This is effectively pruning.
The Hier-Prox algorithm

- The hierarchy constraint is separable over the features.
- Objective can be optimized by constrained proximal GD
The **Hier-Prox** algorithm

- The hierarchy constraint is *separable* over the features.
- Objective can be optimized by constrained proximal GD
- At its core, LassoNet solves $d$ problems of the form

  $$
  \underset{\beta \in \mathbb{R}, W \in \mathbb{R}^K}{\text{minimize}} \frac{1}{2}(v - \beta)^2 + \frac{1}{2}\|u - W\|^2 + \lambda\|\beta\|_1 \\
  \text{subject to } \|W\|_\infty \leq M \cdot |\beta|
  $$

- **Hier-Prox**: an efficient hierarchical proximal operator
The **Hier-Prox** operator

- At its core, LassoNet solves $d$ problems of the form

  \[
  \min_{\beta \in \mathbb{R}, W \in \mathbb{R}^K} \frac{1}{2}(v - \beta)^2 + \frac{1}{2}\|u - W\|^2 + \lambda\|\beta\|_1
  \]

  subject to \(\|W\|_\infty \leq M \cdot |\beta|\)

- The **Hier-Prox** operator provides the **global** solution of this **nonconvex** minimization problem.
The **Hier-Prox** operator

- At its core, LassoNet solves $d$ problems of the form

  $$\text{minimize}_{\beta \in \mathbb{R}, W \in \mathbb{R}^K} \frac{1}{2} (v - \beta)^2 + \frac{1}{2} \| u - W \|^2 + \lambda \| \beta \|_1$$

  subject to $\| W \|_\infty \leq M \cdot |\beta|$ 

- The **Hier-Prox** operator provides the **global** solution of this **nonconvex** minimization problem

- Integrates seamlessly with **deep learning frameworks**
The **Hier-Prox** operator

- At its core, LassoNet solves $d$ problems of the form

  \[ \begin{align*}
  &\text{minimize}_{\beta \in \mathbb{R}, W \in \mathbb{R}^K} \quad \frac{1}{2} (v - \beta)^2 + \frac{1}{2} \| u - W \|^2 + \lambda \| \beta \|_1 \\
  &\text{subject to} \quad \| W \|_{\infty} \leq M \cdot |\beta| 
  \end{align*} \]

- The **Hier-Prox** operator provides the **global** solution of this **nonconvex** minimization problem

- Integrates seamlessly with **deep learning frameworks** 🌟 PyTorch

- The algorithm has complexity $O(dK \cdot \log(dK))$, where $d$ is the number of features and $K$ the size of the input layer

- **Negligible overhead** compared to gradient computations
Experimental evaluation

- Most other feature selection methods are not *embedded*
- Plug the selected features into external downstream learners:
  - A feedforward neural network
  - A tree-based classifier
- Systematic evaluation on 6 datasets
Results on the ISOLET dataset

- Letter speech data
- Benchmark data set for feature selection
- \( n = 7797, d = 617 \)

Classification accuracies for feature selection methods

**Left:** using a one-hidden-layer feedforward neural network. **Right:** using an extremely randomized tree classifier.
Systematic evaluation

Compare the classification accuracies for a fixed number of features, $k = 50$:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$(n, d)$</th>
<th># Classes</th>
<th>Fisher</th>
<th>HSIC-Lasso</th>
<th>PFA</th>
<th>LassoNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>(10000, 784)</td>
<td>10</td>
<td>0.813</td>
<td>0.870</td>
<td><strong>0.873</strong></td>
<td>0.873</td>
</tr>
<tr>
<td>MNIST-Fashion</td>
<td>(10000, 784)</td>
<td>10</td>
<td>0.671</td>
<td>0.785</td>
<td>0.793</td>
<td><strong>0.800</strong></td>
</tr>
<tr>
<td>ISOLET</td>
<td>(7797, 617)</td>
<td>26</td>
<td>0.793</td>
<td>0.877</td>
<td>0.863</td>
<td><strong>0.885</strong></td>
</tr>
<tr>
<td>COIL-20</td>
<td>(1440, 400)</td>
<td>20</td>
<td>0.986</td>
<td>0.972</td>
<td>0.975</td>
<td><strong>0.991</strong></td>
</tr>
<tr>
<td>Activity</td>
<td>(5744, 561)</td>
<td>6</td>
<td>0.769</td>
<td>0.829</td>
<td>0.779</td>
<td><strong>0.849</strong></td>
</tr>
<tr>
<td>Mice Protein</td>
<td>(1080, 77)</td>
<td>8</td>
<td>0.944</td>
<td><strong>0.958</strong></td>
<td>0.939</td>
<td><strong>0.958</strong></td>
</tr>
</tbody>
</table>

Classification accuracies on a hold-out test set, using a one-hidden-layer feedforward neural network.
Summary

The Neural Network Resurrection

Feature Selection
  Benefits
  Desiderata

LassoNet
  The hierarchy principle
  Formulation

Optimization
  Pruning a dense model
  The hierarchical optimizer

Experimental evaluation
Extensions and applications

- Unsupervised learning
  - Reconstruction loss as the objective
  - Related work: Concrete auto-encoder (Abid et al., *ICML* 2019)

Reconstructing single digit classes of MNIST
Extensions and applications

- Unsupervised Learning
  - Reconstruction loss as the objective
  - Related work: Concrete auto-encoder (Abid et al., ICML 2019)

- Sparse PCA Net
  - enforce an interpretable bottleneck layer
Extensions and applications

- Unsupervised Learning
  - Reconstruction loss as the objective
  - Related work: Concrete auto-encoder (Abid et al., ICML 2019)

- Sparse PCA Net
  - enforce an interpretable bottleneck layer

- Cox Proportional Hazards Model

DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network

Jared L. Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang & Yuvil Kluger

BMC Medical Research Methodology 18, Article number: 24 (2018) | Cite this article

Abstract

Background

Medical practitioners use survival models to explore and understand the relationships between patients' covariates (e.g., clinical and genetic features) and the effectiveness of various treatment options. Standard survival models like the linear Cox proportional hazards model require extensive feature engineering or prior medical knowledge to model treatment interaction at an individual level. While nonlinear survival methods, such as neural networks and survival forests, can inherently model these high-level interaction terms, they have yet to be shown as effective treatment recommender systems.
Resources

- Talk Materials at: https://tinyurl.com/lassonet
- Code at: https://github.com/ilemhadri/lassonet
- Thanks:
  - Rob Tibshirani
  - Feng Ruan
  - PyTorch help: Louis Abraham

Thank you. Be well!
The Hier-Prox algorithm

Algorithm 2 Hierarchical Proximal Algorithm

1: procedure HIER-PROX(\(\theta, W^{(0)}; \lambda, M\))
2:     for \(j \in \{1, \ldots, d\} \) do
3:         Sort the coordinates of \(W_j^{(0)}\) into \(|W_{(j,1)}^{(0)}| \geq \cdots \geq |W_{(j,K)}^{(0)}|\)
4:     for \(m \in \{0, \ldots, K\} \) do
5:         Compute \(w_m \equiv \frac{M}{1+M^2} \cdot S_{\lambda}(|\theta_j| + M \cdot \sum_{i=1}^{m} |W_{(j,i)}^{(0)}|)\)
6:         Find the first \(m\) such that \(|W_{(j,m+1)}^{(0)}| \leq w_m \leq |W_{(j,m)}^{(0)}|\)
7:     end for
8:     \(\tilde{\theta}_j \leftarrow \frac{1}{M} \cdot \text{sign}(\theta_j) \cdot w_m\)
9:     \(\tilde{W}_j^{(0)} \leftarrow \text{sign}(W_j^{(0)}) \cdot \min(w_m, W_j^{(0)})\)
10:    end for
11:    return (\(\tilde{\theta}, \tilde{W}^{(0)}\))
12: end procedure

13: Conventions: Ln. 6, \(W_{(j,K+1)}^{(0)} = 0\), \(W_{(j,0)}^{(0)} = +\infty\); Ln. 9, minimum is applied coordinate-wise.
Systematic evaluation

Compare the classification accuracies for a fixed number of features, $k = 50$:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$(n, d)$</th>
<th># Classes</th>
<th>Fisher</th>
<th>HSIC-Lasso</th>
<th>PFA</th>
<th>LassoNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>(10000, 784)</td>
<td>10</td>
<td>0.813</td>
<td>0.870</td>
<td><strong>0.873</strong></td>
<td><strong>0.873</strong></td>
</tr>
<tr>
<td>MNIST-Fashion</td>
<td>(10000, 784)</td>
<td>10</td>
<td>0.671</td>
<td>0.785</td>
<td>0.793</td>
<td><strong>0.800</strong></td>
</tr>
<tr>
<td>ISOLET</td>
<td>(7797, 617)</td>
<td>26</td>
<td>0.793</td>
<td>0.877</td>
<td>0.863</td>
<td><strong>0.885</strong></td>
</tr>
<tr>
<td>COIL-20</td>
<td>(1440, 400)</td>
<td>20</td>
<td>0.986</td>
<td>0.972</td>
<td>0.975</td>
<td><strong>0.991</strong></td>
</tr>
<tr>
<td>Activity</td>
<td>(5744, 561)</td>
<td>6</td>
<td>0.769</td>
<td>0.829</td>
<td>0.779</td>
<td><strong>0.849</strong></td>
</tr>
<tr>
<td>Mice Protein</td>
<td>(1080, 77)</td>
<td>8</td>
<td>0.944</td>
<td><strong>0.958</strong></td>
<td>0.939</td>
<td><strong>0.958</strong></td>
</tr>
</tbody>
</table>

**Classification accuracies** on a hold-out test set, using Extremely Randomized Tree Classifiers (a variant of random forests).