Discovering and Exploiting Additive Structure for Bayesian Optimization

Jacob R. Gardner  
Department of Computer Science  
Cornell University  
jrg365@cornell.edu

Chuan Guo  
Department of Computer Science  
Cornell University  
cq563@cornell.edu

Kilian Q. Weinberger  
Department of Computer Science  
Cornell University  
kqw4@cornell.edu

Roman Garnett  
Computer Science and Engineering  
Washington University in St. Louis  
garnett@wustl.edu

Roger Grosse  
Department of Computer Science  
University of Toronto  
rgrosse@cs.toronto.edu
Hyperparameter Search

- Most methods in machine learning require hyperparameters
  - Regularization parameters for linear regression, neural network layers, neighbors in kNN, maximum tree depth, etc.

- Performance can crucially depend on their values – think unregularized linear regression with 100,000 predictors or kNN with $k = n$

- Hyperparameters need to be set properly for optimal or even acceptable performance

Difficulties with hyperparameter optimization

- Objective function unknown, no gradients, really expensive to evaluate

Typical solutions

- Grid search, random search, Bayesian optimization
Bayesian Optimization
Bayesian Optimization

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smarter decisions lead to faster convergence</td>
<td>Implementation is not easy</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dependent on own hyperparameters</td>
</tr>
<tr>
<td>Used in practice</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Can’t really be used in high dimension (exponential complexity)</td>
</tr>
</tbody>
</table>

KEY ISSUE

What to do?
EXPLOIT ADDITIVE STRUCTURE
## Objective Function Structure Types

<table>
<thead>
<tr>
<th>Structure</th>
<th>Example</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Dependent</td>
<td>$f(x) = x_1 x_2 x_3 x_4 x_5$</td>
<td>Exponential</td>
</tr>
<tr>
<td>Fully Independent</td>
<td>$f(x) = x_1 + x_2 + x_3 + x_4 + x_5$</td>
<td>Linear</td>
</tr>
<tr>
<td>Mixed</td>
<td>$f(x) = x_1 x_2 x_3 + x_4 + x_5$</td>
<td>Subexponential</td>
</tr>
</tbody>
</table>

Knowing additive structure gives exponential reduction in complexity  
(Kandasamy et. al 2015)
Bayesian Optimization Flow

1. Get initial sample from objective function
2. Update posterior (refit kernel)
3. Optimize acquisition function
4. Sample objective function at point $x^*$
5. Repeat until satisfied
Bayesian Optimization Flow, Structure Discovery

1. Get initial sample from objective function
2. Discover objective function structure
   2.1 Sample model (partition)
      \[ M_k = [1,2,3][4][5] \]
   2.2 Fit additive kernel
      \[ K_k = K(x_{123}, x_{123}, x_{123}) + K(x_4) + K(x_5) \]
   2.3 Optimize acquisition function for \( x^*_k \)
   2.4 Repeat k times (50 in the paper)
3. Set \( x^* \) to be the point from \( (x_1, \ldots, x_k) \) that maximizes marginalized acquisition function
   \[ p(f(x^*) \mid D, x^*) \approx \frac{1}{k} \sum_{j=1}^{k} p(f(x^*) \mid D, x^*, M_j) \]
4. Sample objective function at point \( x^* \)
5. Repeat until satisfied
Metropolis-Hastings Model Sampling

1. Sample from proposal distribution

\[ \mathcal{M} = [1,3] [2] [4] \]

split or merge?

- split
- merge

pick 1 sub-partition:

- perform split

\[ \mathcal{M}' = [1][2][3][4] \]

pick 2 sub-partitions:

- perform merge

\[ \mathcal{M}' = [1,3,2] [1,3,4] [1,3] [2,4] \]

2. Accept sample with probability

\[ \Lambda(\mathcal{M}' | \mathcal{M}_j) = \min \left( 1, \frac{p(y_i | X_i, \mathcal{M}') g(\mathcal{M}_j | \mathcal{M}')}{p(y_i | X_i, \mathcal{M}_j) g(\mathcal{M}' | \mathcal{M}_j)} \right) \]
Results, Simulation

\[
\text{Stybtang}(x) = \frac{1}{2} \sum_{i=1}^{d} x_i^4 - 16 x_i^2 + 5x_i
\]

\[
\text{Michalewicz}(x) = -\sum_{i=1}^{d} \sin(x_i) \sin^{2m}\left(\frac{i x_i}{\pi}\right)
\]
Results, Simulation

![Graph showing minimum function value against number of iterations for different methods: Fully dependent, Bag of Models, BayesOpt+Model MCMC, and Oracle. The graph indicates that the Oracle method reaches the minimum function value at 1.82x the number of iterations compared to the other methods.](image-url)
Results, Real Data

Cosmological Constants Experiment

- Baseline BayesOpt
- Bag of Models
- BayesOpt+Model MCMC

Negative Log Likelihood

Iteration Number

1.38x
Results, Real Data
Conclusion

- Bayesian optimization can select optimal hyperparameter settings with fewer iterations

- ...but is very slow in high dimensions (over 100 hyperparameters)

- One possible solution – exploit additive structure

- Works very well when additive structure is present, not much worse when it isn’t

- Can be a powerful extension to Auto ML applications

- Not free - if the objective function is not too expensive this can be slower
  - Need to evaluate k extra models but each model simpler

- Doesn’t solve all the problems – high dimensionality still a problem, but now less so