BayesOpt was designed for black-box optimization

Goal: Solve $\min_x f(x)$ where $f(x)$ is a black box time-consuming-to-evaluate function
Here's BayesOpt at a glance

Evaluate $f(x)$ for several initial $x$

Build a Bayesian statistical model on $f$ (usually a GP)

while (budget is not exhausted) {
    Find $x$ that maximizes $\text{Acquisition}(x, \text{posterior})$
    Sample at $x$
    Update the posterior distribution
}

This is EGO, a **classical** Bayesian optimization method, optimizing a 1-dimensional objective.

**Background: Expected Improvement**

\[
EI(x) = E[(F(x) - F^*)^+] 
\]
This is EGO, a **classical** Bayesian optimization method, optimizing a 1-dimensional objective.

$$EI(x) = E[(F(x) - F^*)^+]$$
This is EGO, a **classical** Bayesian optimization method, optimizing a 1-dimensional objective

\[ EI(x) = \mathbb{E}[(F(x) - F^*)^+] \]
This is EGO, a **classical** Bayesian optimization method, optimizing a 1-dimensional objective

$$EI(x) = E[(F(x) - F^*)^+]$$
BayesOpt is useful for hyperparameter optimization

Goal: Find a hyperparameter vector $x$ with good cross-validation error

[Snoek, Larochelle, Adams 2012]
BayesOpt is useful for hyperparameter optimization

Google
SIGOPT
yelp
Uber
amazon
Facebook
We can do better by looking **inside** the box

- Hyperparameter vector $x$
- Validation performance
- Other information
We can do better by looking inside the box

- Early Stopping / Freezing & Thawing
- Cheap-to-evaluate proxies / multi-task learning / multi-information source optimization
- Warm starts
- Gradients
- Composite Functions
We can do better by looking inside the box

- Early Stopping / Freezing & Thawing
- Cheap-to-evaluate proxies / multi-task learning / multi-information source optimization
- Warm starts
- Gradients
- Composite Functions
We get a whole learning curve from training

Stopping poor evaluations early is super important

Li, Jamieson, DeSalvo, Rostamizadeh and Talwalkar
"Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization" JMLR 2018
Stopping poor evaluations early is super important

Li, Jamieson, DeSalvo, Rostamizadeh and Talwalkar
"Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization" JMLR 2018
Methods combining BO w/ early stopping has been an active area of research

  "Freeze-thaw Bayesian optimization" 2014

- Domhan, Springenberg, Hutter, IJCAI 2015,
  "Speeding Up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves"

- Klein, Falkner, Bartels, Hennig, Hutter, AISTATS 2017
  "Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets."

- Falkner, Klein, Hutter, ICML 2018
  "BOHB: Robust and Efficient Hyperparameter Optimization at Scale"

- Dai, Yu, Low, Jaillet, ICML 2019
  "Bayesian Optimization Meets Bayesian Optimal Stopping"

- Wu, Toscano-Palmerin, F., Wilson, UAI 2019
  "Practical Multi-fidelity Bayesian Optimization for Tuning Iterative ML Algorithms"
There are also other more general purpose multi-fidelity & multi-information-source methods that can use cheap eval proxies

- Swersky, Snoek & Adams, NIPS 2013
  "Multi-task Bayesian Optimization"

- Kandasamy, Dasarathy, Oliva, Schneider, Poczos, NIPS 2016
  "Gaussian Process Bandit Optimisation with Multi-fidelity Evaluations"

- Poloczek, Wang & F., NIPS 2017
  "Multi-Information Source Optimization"
Here's roughly how these methods work

Evaluate $f(x, t)$ for several initial $x$ and $t$

Build a Bayesian statistical model on $f$ (e.g., GP, TPE)

while (budget is not exhausted) {

  Find $x, t$ that maximizes $\frac{\text{Acquisition}(x, t, \text{posterior})}{\text{Cost}(x, t)}$

  Evaluate validation error at $x$ up to $t$ training iter.

  Update the posterior distribution

}
We can do inference over learning curves

We can do inference over learning curves

\[ f(x,t) \sim \text{GP}(\mu, \Sigma) \]
EI often needs tweaks for use in grey-box optimization

• **Problem:**
  When we observe $f(x,t)$ for $t < T$, we don't observe the objective function
  $\Rightarrow$ there's no "improvement"
  $\Rightarrow$ $EI(x) = E[(f^* - f(x))_+]$ doesn't really make sense

• Some **Partial Solutions:**
  • Choose $x$ via $E[(f^* - f(x,T))_+]$, then choose $t$ in another way
    [choice of $x$ doesn't reflect the cost, or the best $t$]
  • Choose $x$ and $t$ by maximizing $E[(f^* - f(x,t))_+] / \text{Cost}(x,t)$
    [overweights small $t$]
EI often needs tweaks for use in grey-box optimization

- **Problem:**
  When we observe $f(x,t)$ for $t < T$, we don't observe the objective function
  $\Rightarrow$ there's no "improvement"
  $\Rightarrow$ $EI(x) = E[(f^* - f(x))_+]$ doesn't really make sense

- Some **Better Solutions:**
  - Predictive Entropy Search
  - Knowledge Gradient
The Knowledge Gradient doesn't need to be tweaked

- Loss if we stop now:
  \[ \mu^*_n = \min_x \mu_n(x) \]

- Loss if we stop after sampling \( f(x) \):
  \[ \mu^*_{n+1} = \min_x \mu_{n+1}(x) \]

- Reduction in loss due to sampling:
  \[ KG(x) = E_n[\mu^*_n - \mu^*_{n+1} | \text{query } x] \]

For details on computation and optimization, see F., 2018 "A Tutorial on Bayesian Optimization" 1807.02811.pdf
The Knowledge Gradient doesn't need to be tweaked

- Loss if we stop now: 
  \[ \mu^*_n = \min_x \mu_n(x, T) \]

- Loss if we stop after sampling \( f(x, t) \): 
  \[ \mu^*_{n+1} = \min_x \mu_{n+1}(x, T) \]

- Reduction in loss due to sampling: 
  \[ KG(x, t) = E_n[\mu^*_n - \mu^*_{n+1} | \text{query } x, t] \]
You can simultaneously control training data size

Evaluate $f(x, t, d)$ for several initial $x$, $t$ and $d$

Build a Bayesian statistical model on $f$ (e.g., GP, TPE)

while (budget is not exhausted) {
    Find $x, t, d$ that maximizes $\frac{\text{Acquisition}(x, t, d, \text{posterior})}{\text{Cost}(x, t, d)}$
    Evaluate validation error at $x$ up to $t$ training iter. using a training set of size $d$
    Update the posterior distribution
}
We also support...

- Gradient observations
- Batch observations (synchronous or asynchronous)
Feedforward neural network on MNIST:
- EI (q=1)
- EI (q=4)
- vanilla KG (q=1)
- vanilla KG (q=4)
- FABOLAS
- Hyperband
- KG (q=1, 2 pts)
- KG (q=1, 3 pts)
- KG (q=4, 2 pts)
- KG (q=4, 3 pts)

Convolutional neural network on SVHN:
- vanilla KG (q=1)
- FABOLAS
- Hyperband
- KG (q=1, 2 pts)
- KG (q=1, 3 pts)

Convolutional neural network on CIFAR-10:
- vanilla KG (q=1)
- FABOLAS
- Hyperband
- KG (q=1, 2 pts)
- KG (q=1, 3 pts)
- KG (q=4, 2 pts)
- KG (q=4, 3 pts)

Kernel Learning:
- vanilla d-KG (q=1)
- vanilla d-KG (q=4)
- d-KG (q=1, 2 pts)
- d-KG (q=4, 2 pts)

w/derivatives
We can also do better by looking inside the box in other BayesOpt problems

- **Composite Functions**  
  [Astudillo, F., ICML 2019]

- **Gradients**  
  [Wu, Poloczek, Wilson, F., NIPS 2017]

- **Integrals**  
  [Toscano-Palmerin, F., arxiv 1803.08661]
Bayesian optimization of composite functions

Goal: Solve \( \min g(h(x)) \), where

- \( h(x) \) is black-box vector-valued and expensive
- \( g \) is cheap to evaluate
Composite functions arise in inverse RL

- $h(x)$ is black-box vector-valued and expensive
- $x$ describes an agent's utility function
- $h(x)$ is a trajectory from an RL solver, optimizing for utility $x$
- $y$ is a trajectory observed from a real agent
- Inverse RL wants to find $x$ to minimize $\| h(x) - y \| = g(h(x))$
Composite functions arise in autoML

- $h_j(x)$ is the validation error for examples with true class $j$
- Validation error is $g(h(x)) = \Sigma_j h_j(x)$
Our Approach

• Model h using a multi-output GP instead of f directly

• This implies a (non-Gaussian) posterior on \( f(x) = g(h(x)) \)

• To decide where to sample next: Compute and optimize* the EI under this new posterior, \( E[(f^* - g(h(x)))_+] \)

* This is made challenging because \( g(h(x)) \) is non-Gaussian, but we use the reparameterization trick + infinitesimal perturbation analysis + multistart SGA to do it efficiently
\[ f(x) = h(x)^2 \]
Asymptotic Consistency

Theorem.
Under suitable regularity conditions, EI-CF is asymptotically consistent, i.e., it finds the true global optimum as the number of evaluations goes to infinity.
Conclusion

• Looking into the box can dramatically improve performance in Bayesian optimization

• What structure in new applications can you find and leverage?
• For black-box KG methods (allowing batch and/or gradient observations), see the Cornell Metrics Optimization Engine (Cornell MOE) https://github.com/wujian16/Cornell-MOE

• For BayesOpt for Composite Functions, see https://github.com/RaulAstudillo06/BOCF

• Code for KG with freeze/thaw in Wu et al. UAI 2019 will be released with the camera ready paper