natural language processing blog

10 October 2014

Hyperparameter search, Bayesian optimization and related topics

In terms of (importance divided-by glamour), hyperparameter (HP) search is probably pretty close to the top. We all hate finding hyperparameters. Default settings are usually good, but you're always left wondering: could I have done better? I like averaged perceptron for this reason (I believe Yoav Goldberg has also expressed this sentiment): no pesky hyperparameters.
But I want to take a much broader perspective on hyperparameters. We typically think of HPs as { regularization constant, learning rate, architecture } (where "architecture" can mean something like neural network structure, choice of kernel, etc. But I think it's a lot broader and can include things like feature engineering, or at least representation modifications. For instance, vw now supports a number of helpful NLP feature templates: suffix and prefix features (via --affix), spelling features (via --spelling), ngram features (--ngrams), quadratic and cubic features, etc. Picking the right incarnation of these thing is essentially a hyperparameter search process, very akin (IMO) to things like representation learning.

Once you're willing to accept all these things as HPs (and I think you should), something like "grid search", which works for tuning C and eta in your SVM, just doesn't seem to cut it anymore.

Enter the world of automatic HP tuning. Lots of this work, not surprisingly, comes out of the neural networks community, because HP search is a big deal there. Most of my information here comes via Hugo Larochelle, who has done a lot of great work. Some places to start looking:

- Spearmint toolkit by Snoek, Larochelle, Swersky, Zemel and Adams (and a JMLR paper)
- SMAC by Hutter, Hoos, Leyton-Brown, Murphy and Ramage (and a paper)

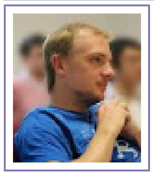
Most of this work falls under the framework of "Bayesian Optimization." The idea comes from the space of derivative-free optimization, where a common strategy is to fit a response surface. Basically you have a bunch of hyperparameters to tune. For any setting of hyperparameters, you can observe some response. In ML land this is usually something like held-out accuracy. Now, fit a regression function that can map from hyperparameters to response. Do something that looks like active learning to explore this space, with a bias toward finding places with high response (high accuracy). In these examples, the function being fit is a Gaussian Process, which is super useful because it can provide realistic estimates of variance, which are useful in the active learning/experimental design.

I first learned about this stuff, not in the context of HP optimization, from Ilya Ryzhov, a faculty member in our business school who works on these topics -- I learned that in his world, exploration/exploitation is also called "learning versus earning" which I think it awesome :).

I would love if hyperparameter optimization were a black box, and Spearmint and SMAC are both great steps in this direction. There are a couple of things that I *haven't* seen (though like I said, I'm not hugely well read in this space) that I would love to see.

my biased thoughts on the fields of natural language processing (NLP), computational linguistics (CL) and related topics (machine learning, math, funding, etc.)

About Me



hal

View my complete profile

Labels

acl (3)
ACS (2)
advising (1)
algorithms (2)
bayesian (10)
chunking (1)
classification (1)
clustering (3)
community (26)
conferences (45)
coreference (1)
data (2)
discourse (3)
domain adaptation (5)
evaluation (9)
finite state methods (1)
graphical models (1)
hiring (7)
information retrieval (1)
journals (3)
language modeling (1)
linguistics (7)
loss functions (1)
machine learning (45)
machine translation (6)
mcmc (1)
news (4)
online learning (2)
papers (17)
parsing (2)
PL (1)
poll (1)
problems (12)

- Learning across multiple datasets, akin to meta-learning. I imagine that if you give me a problem to do HP optimization on, and also give the same problem to an undergraduate, I would be much better. Presumably I've learned from past experience how to do this well.
- More importantly, taking advantage of some of the structure of learning algorithms (rather than oblivious black box optimization) seems like it could be a big win. For instance, most learning algorithms have some notion of early stopping (# of passes over the data, tolerance for convergence, etc.). You can also of course run on subsets of the data (which is equivalent in many cases). If you assume heldout accuracy doesn't get worse over time (e.g., because you do early stopping) then you can think of this as a type of right-censoring. I.e., you run an experiment part of the way through and you know "ok if I kept running it might get better, but I know it's at least 85% accurate now." The SMAC folks had a nice paper on Bayesian optimization with censored data, but my (incomplete) understanding is that it doesn't quite capture this (very common, IMO) case. I should be willing to start and stop processes at various points and try to figure out where to invest more computation to get better estimates. I can presumably even estimate: if I ran another pass, how much better do I think it would get?
- I also think the focus on "finding the best hyperparameters" is somewhat the wrong problem. We want to find the best *parameters*, period. Hyperparameters are a nuisance on their way to that end. For instance, related to the above bullet, I could imagine running a few passes with one setting of hyperparameters, and then doing some other work, and then going back and restarting that previous run with a different setting of hyperparameters (assuming the model being learned is such that "warm starting" makes sense, which is almost always the case--except maybe in some neural network settings).

- Parallelization is a big deal. One of the reasons something akin to grid search is so attractive is that it's trivial to submit 20*20*20 jobs to my cluster and just wait for them to finish. Anything that's less friendly than doing this is not worth it. Again, the SMAC folks have worked on this. But I don't think the problem is solved.

Beyond these technical issues there's always the obnoxious issue of *trust.* Somehow I need to believe that I'm *not* better than these algorithms at tuning hyperparameters. I should be happy to just run them, preferably saying "okay, here are 120 cores, you have four hours -- go to town." And I should believe that it's better than I could do with equivalent time/resources by clever grid search. Or perhaps I should be able to encode my strategies in some way so that it can *prove* to me that it's better than me.

Overall, I'd love to see more work on this problem, especially work that *doesn't* focus on neural networks, but still takes advantage of the properties of machine learning algorithms that are not shared by all black-box derivative-free optimization tasks. In the mean time, from what I hear, SMAC and Spearmint are actually quite good. Would love to hear if any NLP people have played around with them!

Posted by hal at 10/10/2014 11:55:00 AM

5 comments:

Piyush Rai said...
On using the idea of meta-learning, there has been some recent work. There was actually a paper from last year's NIPS where they used *multitask* GPs:
Multi-Task Bayesian Optimization:
http://machinelearning.wustl.edu/mlpapers/paper_files/NIPS2013_5086.pdf)

There was also another paper at this year's AISTATS:
Efficient Transfer Learning Method for Automatic Hyperparameter Tuning:

questions (2)
random (1)
research (12)
reviewing (2)
sentiment (1)
software (1)
speech (1)
statistics (3)
structured prediction (5)
summarization (4)
survey (6)
teaching (3)
theory (1)
topic models (1)

My Blog List

in theory
CS294 Lecture 8: Spectral Algorithms Wrap-up
5 minutes ago

Wadler's Blog
Steven Pinker's The Sense of Style
15 hours ago

Talking Brains
Post-Doc with Tecumseh Fitch in Vienna Department of Cognitive Biology
16 hours ago

Statistical Modeling, Causal Inference, and Social Science
You'll never guess what David Cox wrote about the garden of forking paths!
16 hours ago

Nuit Blanche
Paris Machine Learning Meetup #8 Season 3: Fair and Ethical Algorithms
1 day ago

Daniel Lemire's blog
Cultivating weirdness
2 days ago

The Geomblog
Cartograms only exist in years divisible by 4...
3 days ago

Computational Complexity
∑{p&leq; n} 1/p = ln(ln(n)) + o(1). read it here because....
3 days ago

Gowers's Weblog
FUNC3 — further strengthenings and variants
4 days ago

http://www.cs.cmu.edu/~dyogatam/papers/yogatama+mann.aistats2014.pdf

.. and another paper that uses a somewhat simpler idea of using the results from previous runs (on other related data sets) for better initialization on a new data set: Using Meta-Learning to Initialize Bayesian Optimization of Hyperparameters: http://ceur-ws.org/Vol-1201/MetaSel2014-complete.pdf#page=8

10 October, 2014 16:21

Anonymous said...
For taking advantage of some of the structure of learning algorithms - for example the iterative nature - look no further than Freeze-Thaw Bayesian Optimization (K Swersky, J Snoek, RP Adams) [1] from June or so. They describe what you suggest here: Looking at the performance early, and only continuing promising candidates.

F.

[1] http://arxiv.org/pdf/1406.3896

11 October, 2014 06:56

RPA said...
Thanks for discussing our work, Hal! You are insightful as always. I thought I might just point out a few things in response to your comments. (Please forgive my gratuitous plugging of my own papers.)

The idea of learning across multiple data sets is something we find to be very useful, and Kevin Swersky, Jasper Snoek, and I had a paper on it at NIPS last year. Andreas Krause also had a 2011 paper that discusses this setting.

The parallelism thing is also something we take seriously and have some good results on in Section 3.3 of our 2012 NIPS paper. Again, Andreas Krause and colleagues have also contributed to this with their 2012 ICML paper.

Regarding the idea of leveraging problem structure to speed things up, you might like our Freeze-Thaw Bayesian Optimization paper on arxiv. The idea is to guess what the final result will be of the inner-loop optimization and build that into the exploration/exploitation strategy without actually finishing jobs about which you are pessimistic.

For what it's worth, you might also worry about complicated constraints in your optimization that you can't identify a priori, e.g., "for some complicated hyperparameter regimes, my code just returns NaNs." For this, you might consider Bayesian Optimization with Unknown Constraints by my student Michael Gelbart, or this paper by John Cunningham and colleagues.

In a final gratuitous plug, the most current versions of Spearmint can be found here, and because Spearmint requires some overhead to set up and run, we've also been building a system for "Bayesian optimization as a web service" that you can sign up for at whetlab.com.

Ryan Adams

11 October, 2014 08:10

Bob Carpenter said...
Why not just fit the hyperparameters and parameters together in a hierarchical model? Basically, meta-analysis and what machine learning people call "adaptation" can both be cast as hierarchical modeling techniques. An example is Hal's "embarassingly simple" approach to adaptation, which Finkel and Manning later formulated as a standard hierarchical model.

What's new
Finite time blowup for an Euler-type equation in vorticity stream form
2 weeks ago

Journal of Statistical Software
Notification : Fri, 29 Jan 2016 22:05:39 +0000
2 weeks ago

Machine Learning (Theory)
New York Machine Learning Deadlines
3 weeks ago

my slice of pizza
We Need to Remember and Dream
4 weeks ago

Geeking with Greg
Quick links
4 weeks ago

tombone's blog
The Future of Real-Time SLAM and "Deep Learning vs SLAM"
5 weeks ago

My Biased Coin
Graduating Bits Announcement
5 weeks ago

Mathematics and Computation
A Brown-Palsberg self-interpreter for Gödel's System T
1 month ago

The Scala Programming Language
2016 Scala Release Schedule update
2 months ago

tcs math - some mathematics of theoretical computer science
Follmer's drift, Ito's lemma, and the log-Sobolev inequality
2 months ago

Oddhead Blog
Algorithmic economics postdoc position at Microsoft Research, NYC
3 months ago

XOR's Hammer
Why is the derivative of a generating function meaningful?
5 months ago

Michael Nielsen
Where will the key ideas shaping the future of scientific publishing

The usual reason not to just fit a hierarchical model is twofold. The first problem is computation. MCMC, which will let you do the exact comptuation you want, namely take a posterior mean point estimate which minimizes expected error, will be too slow for large data sets. Optimization simply fails in hierarchical setting because the density grows without bound as hierarchical variance shrinks to zero (or as David McKay aptly put it in his book, "EM goes boom").

What you can do is approximate with point estimates, also known as empirical Bayes (in a huge misnomer --- it's no more empirical than full Bayes in a hierarchical model). The standard approach is maximum marginal likelihood (MML), as in the lme4 package in R. What you do is marginalize out the low level parameters, optimize the hyperparameters, then fix the hyperparameters to estimate the low-level parameters in a second pass.

We're working on doing this in full generality in Stan using Laplace approximations for the marginalization, but we're not quite all the way there yet. We're also working on black-box EP and VB approaches, which may provide better point estimates and better approximations to uncertainty than a simple Laplace estimate based on curvature at the mode.

There's a whole book by Brad Efron, the inventor of the bootstrap, on these kinds of techniques with applications to large-scale genomics problems: *Large-Scale Inference Empirical Bayes Methods for Estimation, Testing, and Prediction*.

13 October, 2014 15:16

Anonymous said...
See also http://jaberg.github.io/hyperopt/

17 October, 2014 09:40

Post a Comment

links to this post

Create a Link

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

come from?
9 months ago

LingPipe Blog
Becky's and My Annotation Paper in TACL
1 year ago

Bayesian Analysis Journal :: Forthcoming Articles
Generalized Quantile Treatment Effect: A Flexible Bayesian Approach Using Quantile Ratio Smoothing
1 year ago

Andy's Math/CS page
Making academic contacts (some thoughts for new researchers)
1 year ago

The StatMT Blog
Easy parallel corpora from Wikipedia
1 year ago

Earning My Turns
Innocence
1 year ago

Learning in Vision
Dual submissions -- busted!
1 year ago

Inductio Ex Machina
What does the "OSS" in MLOSS mean?
2 years ago

WebDiarios de Motocicleta
Presburger Award
3 years ago

Data Wrangling
Slides & Thoughts from Hadoop World NYC
6 years ago

Quantum Algorithms
Polynomial-time quantum algorithm for the simulation of chemical dynamics
7 years ago

Logicomp
When does Bob deserve to be a co-author?
7 years ago

[Lowerbounds, Upperbounds]
Parasitic SEO is not a Victimless Crime
7 years ago

Mainly Data
Hive is alive!
7 years ago

Mathematics Weblog
A Levels
8 years ago

Structured Learning
Corrections to ACL Anthology URLs
8 years ago

http://groundtruth.info/AstroStat/slog/

mstatbiostat :

Undirected Grad

Ganesh Swami

The AstroStat Slog

yw's machine learning blog

Information Retrieval

Mathematics Weblog

Apperceptual

Information Engineering

Blog Archive