Hyperparameter optimization

MIPT

2023

Model selection: coherent inference

First level: select optimal parameters:

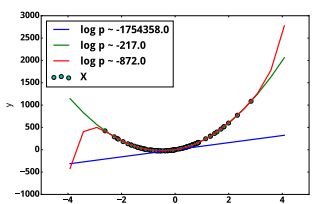
$$\mathrm{w} = \arg\max \frac{p(\mathfrak{D}|\mathrm{w})p(\mathrm{w}|\mathrm{h})}{p(\mathfrak{D}|\mathrm{h})},$$

Second level: select optimal model (hyperparameters).

Evidence:

$$p(\mathfrak{D}|\mathrm{h}) = \int_{\mathrm{w}} p(\mathfrak{D}|\mathrm{w})p(\mathrm{w}|\mathrm{h})d\mathrm{w}.$$

Model selection scheme



Example: polynoms

Hyperparameters

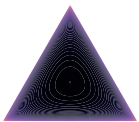
Definition

Prior for parameters w and structure $\boldsymbol{\Gamma}$ of the model f is a distrubution $p(\mathsf{W}, \boldsymbol{\Gamma}|\mathsf{h}) : \mathbb{W} \times \Gamma \times \mathbb{H} \to \mathbb{R}^+$, where $\mathbb{W}$ is a parameter space, Γ is a structure space.

Definition

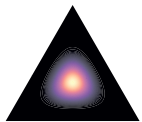
Hyperparameters $\mathsf{h} \in \mathbb{H}$ of the models are the parameters of $p(\mathsf{w}, \boldsymbol{\Gamma}|\mathsf{h})$ (parameters of prior f).

Discrete distribution: relaxation

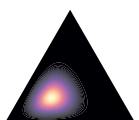


$\bar{\boldsymbol{\alpha}} = [1, 1, 1], t = 0.9$

$t = 1.0$



$t = 10.0$



$\bar{\boldsymbol{\alpha}} = [0.5, 0.25, 0.25], t = 30.0$

$[0.75, 0.125, 0.125]$

$[0.9, 0.05, 0.05]$

Laplace approximation

Nonlinear case with m objects and n features: $\mathrm{y} \sim \mathcal{N}(\mathrm{f}(\mathrm{X}, \mathrm{w}), \lambda^{-1}), \mathrm{w} \sim \mathcal{N}(0, \mathrm{A}^{-1})$.
Write integral:

$$p(\mathfrak{D}|\mathrm{h}) = p(\mathrm{y}|\mathrm{X}, \mathrm{A}, \lambda) = \frac{\sqrt{\lambda \cdot |\mathrm{A}|}}{\sqrt{(2\pi)^{m+n}}} \int_{\mathrm{w}} \exp(-S(\mathrm{w}))d\mathrm{w}.$$

Using Taylor serioes for S:

$$S(\mathrm{w}) \approx S(\hat{\mathrm{w}}) + \frac{1}{2}\Delta\mathrm{w}^{\mathrm{T}}\mathrm{H}\Delta\mathrm{w}$$

Integral reduces to the following expression:

$$\frac{\sqrt{\lambda \cdot |\mathrm{A}|}}{\sqrt{(2\pi)^{m+n}}} S(\hat{\mathrm{w}}) \int_{\mathrm{w}} \exp(-\frac{1}{2}\Delta\mathrm{w}^{\mathrm{T}}\mathrm{H}\Delta\mathrm{w})d\mathrm{w}$$

The expression under integral corresponds to the unnormalized Gaussian PDF.

Graves, 2011

Prior: $p(\mathrm{w}|\sigma) \sim \mathcal{N}(\boldsymbol{\mu}, \sigma\mathrm{I})$.
Variational inference: $q(\mathrm{w}) \sim \mathcal{N}(\boldsymbol{\mu}_q, \sigma_q\mathrm{I})$.
Greedy optimization:

$$\boldsymbol{\mu} = \hat{\mathrm{E}}\mathrm{w}, \quad \sigma = \hat{\mathrm{D}}\mathrm{w}.$$

Prune w_i using relative PDF:

$$\lambda = \frac{q(0)}{q(\boldsymbol{\mu}_{i,q})} = \exp(-\frac{\mu_i^2}{2\sigma_i^2}).$$

Problem statement

Let $\boldsymbol{\theta} \in \mathbb{R}^s$ be the set of all the optimized parameters (including variational parameters if needed).
$L(\boldsymbol{\theta}, \mathrm{h})$ is a differential loss function f.
$Q(\boldsymbol{\theta}, \mathrm{h})$ is a differential validation function.
The problem is to find optimal parameters $\boldsymbol{\theta}^*$ and hyperparameters h^* of the model that minimze

$$\mathrm{h}^* = \arg\max_{\mathrm{h} \in \mathbb{H}} Q(\boldsymbol{\theta}^*(\mathrm{h}), \mathrm{h}),$$

$$\boldsymbol{\theta}(\mathrm{h})^* = \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^s} L(\boldsymbol{\theta}, \mathrm{h}).$$

Bayesian inference

Let $\boldsymbol{\theta} = [\mathsf{w}]^{\mathsf{T}}$.
First level:

$$\boldsymbol{\theta}^* = \arg\max\bigl(-L(\boldsymbol{\theta}, \mathsf{h})\bigr) = p(\mathsf{w}|\mathsf{X}, \mathsf{y}, \mathsf{h}) = \frac{p(\mathsf{y}|\mathsf{X}, \mathsf{w})p(\mathsf{w}|\mathsf{h})}{p(\mathsf{y}|\mathsf{X}, \mathsf{h})}.$$

Second level:

$$p(\mathsf{h}|\mathsf{X}, \mathsf{y}) \propto p(\mathsf{y}|\mathsf{X}, \mathsf{h})p(\mathsf{h}),$$

Considering $p(\mathsf{h})$ improper flat prior we get the following expression:

$$Q(\boldsymbol{\theta}, \mathsf{h}) = p(\mathsf{y}|\mathsf{X}, \mathsf{h}) = \int_{\mathsf{w}\in\mathbb{R}^u} p(\mathsf{y}|\mathsf{X}, \mathsf{w})p(\mathsf{w}|\mathsf{h}) \to \max_{\mathsf{h}\in\mathbb{H}}.$$

Cross-validation

Split the dataset $\mathfrak{D}$ into k equal (maybe stratified) parts:

$$\mathfrak{D} = \mathfrak{D}_1 \sqcup \cdots \sqcup \mathfrak{D}_k.$$

Optimize k modelsfor each data part. Let $\boldsymbol{\theta} = [\mathrm{w}_1, \ldots, \mathrm{w}_k]$, where $\mathrm{w}_1, \ldots, \mathrm{w}_k$ are the model parameters for optimization k.
Let L be a loss function:

$$L(\boldsymbol{\theta}, \mathrm{h}) = -\frac{1}{k} \sum_{q=1}^{k} \Big(\frac{k}{k-1} \log p(\mathrm{y} \setminus \mathrm{y}_q | \mathrm{X} \setminus \mathrm{X}_q, \mathrm{w}_q) + \log p(\mathrm{w}_q | \mathrm{h}) \Big). \tag{1}$$

Let Q be a validation loss:

$$Q(\boldsymbol{\theta}, \mathrm{h}) = \frac{1}{k} \sum_{q=1}^{k} k \log p(\mathrm{y}_q | \mathrm{X}_q, \mathrm{w}_q).$$

ELBO

Let $L = -Q$:

$$\log p(\mathrm{y}|\mathrm{X}, \mathrm{A}) \geq \sum_{\mathrm{x},y} \log p(y|\mathrm{x}, \hat{\mathrm{w}}) - D_{\mathrm{KL}}\big(q(\mathrm{w})||p(\mathrm{w}|\mathrm{A})\big) = -L(\boldsymbol{\theta}, \mathrm{A}^{-1}) = Q(\boldsymbol{\theta}, \mathrm{A}^{-1}),$$

where q is a normal distribution with diagonal covariance matrix:

$$q \sim \mathcal{N}(\boldsymbol{\mu}_q, \mathrm{A}_q^{-1}),$$

$$D_{\mathrm{KL}}\big(q(\mathrm{w})||p(\mathrm{w}|\mathrm{f})\big) = \frac{1}{2}\big(\mathrm{Tr}[\mathrm{A}\mathrm{A}_q^{-1}] + (\boldsymbol{\mu} - \boldsymbol{\mu}_q)^{\mathrm{T}}\mathrm{A}(\boldsymbol{\mu} - \boldsymbol{\mu}_q) - u + \ln\,|\mathrm{A}^{-1}| - \ln\,|\mathrm{A}_q^{-1}|\big).$$

Use variational parameters of q as a vector of optimized parameters $\boldsymbol{\theta}$:

$$\boldsymbol{\theta} = [\alpha_1, \ldots, \alpha_u, \mu_1, \ldots, \mu_u].$$

Evidence vs CV

Evidece estimation:

$$\log p(\mathfrak{D}|\mathsf{f}) = \log p(\mathfrak{D}_1|\mathsf{f}) + \log p(\mathfrak{D}_2|\mathfrak{D}_1, \mathsf{f}) + \cdots + \log p(\mathfrak{D}_n|\mathfrak{D}_1, \ldots, \mathfrak{D}_{n-1}, \mathsf{f}).$$

Leave-one-out estimation:

$$\mathsf{LOU} = \mathsf{E}\log p(\mathfrak{D}_n|\mathfrak{D}_1, \ldots, \mathfrak{D}_{n-1}, \mathsf{f}).$$

Cross-validation uses expected values of the last term $p(\mathfrak{D}_n|\mathfrak{D}_1, \ldots, \mathfrak{D}_{n-1}, \mathsf{f})$ as a complexity estimation.
Evidence considers **full** complexity.

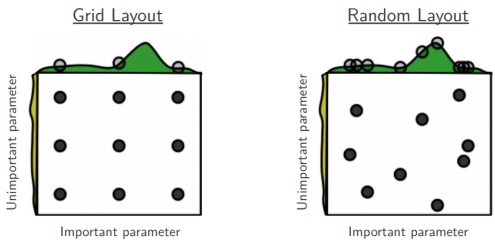
Basic methods of hyperparameter optimization

Variants:

- Grid search;
- random search.

Both methods suffer from curse of dimensionality.

The random search can be more effective if the hyperparameter space is degenerate.



Bergstra et al., 2012

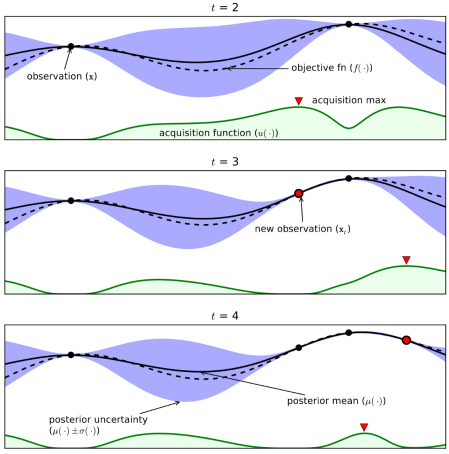
Gaussian process

Idea: Model $Q(\boldsymbol{\theta}(\mathrm{h})^*, \mathrm{h})$ using Gaussian process depending on h.

Pros:

- Flexibility.
- Probabilistic model, cheaper than exhaustive search.

Cons: cubic complexity, $O(|\mathbb{H}|^3)$.



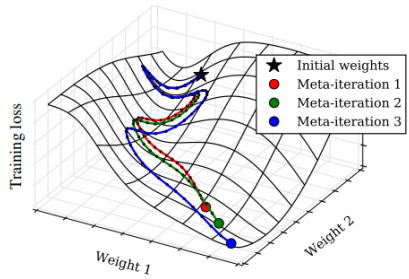
Shahriari et. al, 2016. GP example.

Gradient methods

Idea: Optimize hyperparameters using the full parameter optimization trajectory.
Pros:

- Hyperparameter optimization will consider the features of the parameter optimization.
- The complexity is linear on the number of hyperparameters.

Cons: the computational cost is very expensive.



Maclaurin et. al, 2015. Example.

Problem statement: gradient-based optimization

Definition

A SGD operator T is an operator providing η steps of optimization:

$$\hat{\boldsymbol{\theta}} = T \circ T \circ \cdots \circ T(\boldsymbol{\theta}_0, \mathsf{h}) = T^{\eta}(\boldsymbol{\theta}_0, \mathsf{h}), \tag{2}$$

where

$$T(\boldsymbol{\theta}, \mathsf{h}) = \boldsymbol{\theta} - \lambda \nabla L(\boldsymbol{\theta}, \mathsf{h})|_{\hat{\mathfrak{D}}},$$

λ is a learning rate, $\boldsymbol{\theta}_0$ is an initial value of $\boldsymbol{\theta}$, $\hat{\mathfrak{D}}$ is a random subset of $\mathfrak{D}$.

Rewrite the optimization problem:

$$\mathsf{h}^* = \arg\max_{\mathsf{h} \in \mathbb{H}} Q(T^{\eta}(\boldsymbol{\theta}_0, \mathsf{h})).$$

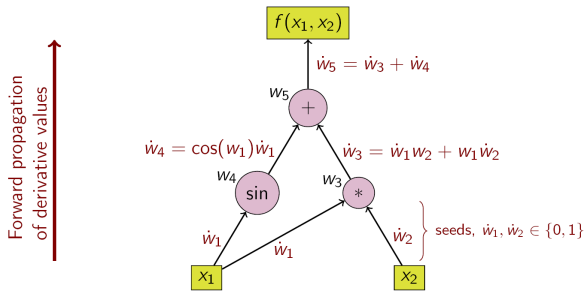
Forward-mode differentiation

Main idea:

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial w_{n-1}} \frac{\partial w_{n-1}}{\partial x} = \frac{\partial y}{\partial w_{n-1}} \left(\frac{\partial w_{n-1}}{\partial w_{n-2}} \frac{\partial w_{n-2}}{x} \partial x \right) = \dots .$$

Example (wiki):

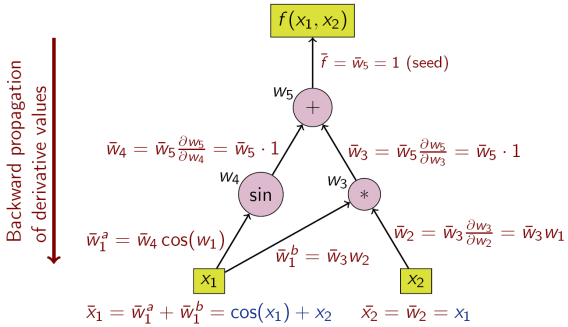
$$x_1 x_2 + \sin(x_1)$$



Reverse-mode differentiation

Idea:

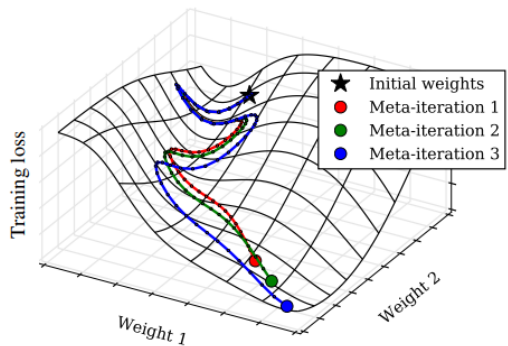
$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial w_1}\frac{\partial w_1}{\partial x} = \left(\frac{\partial y}{\partial w_2}\frac{\partial w_2}{\partial w_1}\right)\frac{\partial w_1}{\partial x} = \dots.$$



RMAD, Maclaurin et. al, 2015

1. Run η steps of memontum optimization γ: $\boldsymbol{\theta} = T(\boldsymbol{\theta}_0, \mathrm{h})$.
2. Let $\hat{\nabla}\mathrm{h} = \nabla_{\mathrm{h}} Q(\boldsymbol{\theta}, \mathrm{h})$.
3. Let $d\mathrm{v} = 0$.
4. For $\tau = \eta \dots 1$ repeat:
5. calucalate $\boldsymbol{\theta}^{\tau-1}$.
6. Calculate gradient on the step $\tau - 1$, using RMD.

RMAD algorithm is based on the Reverse-mode differentiation.



DrMAD

DrMad is a simplified RMAD. Idea: the parameter optimization trajectory is linear.

1. Run η steps of memontum optimization γ: $\boldsymbol{\theta} = T(\boldsymbol{\theta}_0, \mathrm{h})$.
2. Let $\hat{\nabla}\mathrm{h} = \nabla_{\mathrm{h}} Q(\boldsymbol{\theta}, \mathrm{h})$.
3. Let $d\mathrm{v} = 0$.
4. For $\tau = \eta \dots 1$ repeat:
5. Calculate $\boldsymbol{\theta}^{\tau-1}$.
6. Calculate gradient on the step $\tau - 1$, using RMD.

1. Run η steps of memontum optimization γ: $\boldsymbol{\theta} = T(\boldsymbol{\theta}_0, \mathrm{h})$.
2. Let $\hat{\nabla}\mathrm{h} = \nabla_{\mathrm{h}} Q(\boldsymbol{\theta}, \mathrm{h})$.
3. Let $d\mathrm{v} = 0$.
4. For $\tau = \eta \dots 1$ repeat:
5. $\boldsymbol{\theta}^{\tau-1} = \boldsymbol{\theta}_0 + \frac{\tau-1}{\eta}\boldsymbol{\theta}^{\eta}$.
6. Calculate gradient on the step $\tau - 1$, using RMD.

Optimization closed form

Statement (Pedregosa, 2016)

Let L be a differential function such that all the stationary points of L are global minima. Let the Hessian H of L be invertible in each stationary point.
Then

$$\nabla_{\mathsf{h}} Q(T(\boldsymbol{\theta}_0, \mathsf{h}), \mathsf{h}) = \nabla_{\mathsf{h}} Q(\boldsymbol{\theta}^\eta, \mathsf{h}) - \nabla_{\mathsf{h}} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^\eta, \mathsf{h})^\mathsf{T} \mathsf{H}^{-1} \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}^\eta, \mathsf{h}).$$

Greedy optimization of hyperparameters

On each step of $\boldsymbol{\theta}$:

$$\mathsf{h}' = \mathsf{h} - \lambda_{\mathsf{h}} \nabla_{\mathsf{h}} Q\big(T(\boldsymbol{\theta}, \mathsf{h}), \mathsf{h}\big) = \mathsf{h} - \lambda_{\mathsf{h}} \nabla_{\mathsf{h}} Q\big(\boldsymbol{\theta} - \lambda \nabla L(\boldsymbol{\theta}, \mathsf{h}), \mathsf{h}\big)\big),$$

where λ_{h} is a learning rate of hyperparameter optimization.

- The method can be considered as a simplified RMAD that uses only one step of parameter optimization.
- The method can be considered as an approximation of the closed form expression for $\mathsf{H}^{-1} \sim \mathsf{I}$.
- Complexity: $O(|\boldsymbol{\theta}| \cdot |\mathsf{h}|)$
- Can be simplified using finite differences, see DARTS, $O(|\boldsymbol{\theta}| + |\mathsf{h}|)$

HOAG

Approximation of closed-form formula:

$$\nabla_{\mathsf{h}} Q(\boldsymbol{\theta}^{\eta}, \mathsf{h}) - \nabla_{\mathsf{h}} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^{\eta}, \mathsf{h})^{\mathsf{T}} \mathsf{H}^{-1} \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}^{\eta}, \mathsf{h}).$$

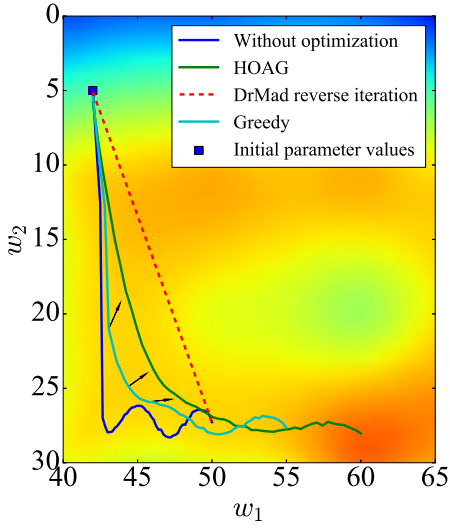
1. run η optimization steps: $\boldsymbol{\theta} = T(\boldsymbol{\theta}_0, \mathsf{h})$.
2. Solve linear system for $\boldsymbol{\lambda}$: $\mathsf{H}(\boldsymbol{\theta})\boldsymbol{\lambda} = \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \mathsf{h})$.
3. Calculate approximate value of hyperparameter gradients $\hat{\nabla}_{\mathsf{h}} Q = \nabla_{\mathsf{h}} Q(\boldsymbol{\theta}, \mathsf{h}) - \nabla_{\boldsymbol{\theta}, \mathsf{h}} L(\boldsymbol{\theta}, \mathsf{h})^{T} \boldsymbol{\lambda}$.

Update formula:

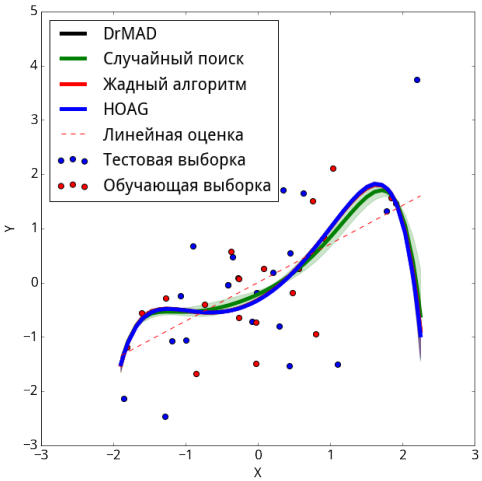
$$\mathsf{h}' = \mathsf{h} - \gamma_{\mathsf{h}} \hat{\nabla}_{\mathsf{h}} Q.$$

Algorithm comparison

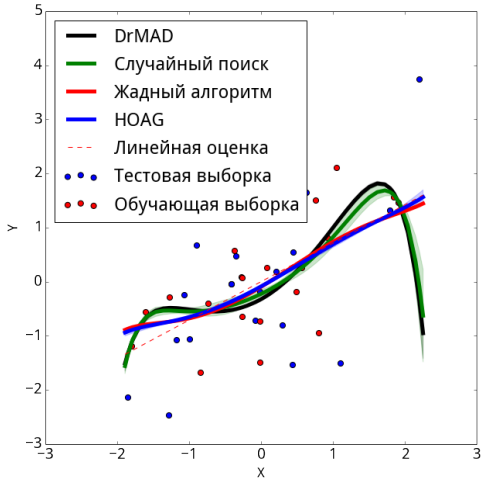
Algorithm	+	-
Random search	Easy to implement	Curse of dimmensionality
Greedy optimization	Easy to implement, can be run inside common gradient optimization loop	Non-optimality.
HOAG	Fast convergence.	Results quality depends on the the linear system solution precision $\mathrm{H}(\boldsymbol{\theta})\boldsymbol{\lambda} = \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \mathrm{h})$.
DrMAD	Considers optimization features. Can be used for metaparameter optimization.	Not very tobust. Strong assumptions about linearity of the optimization trajectory.


Without optimization
HOAG
DrMad reverse iteration
Greedy
Initial parameter values
0
5
10
15
20
25
30
40
45
50
55
60
65
w_1
w_2

Experiment: polynoms

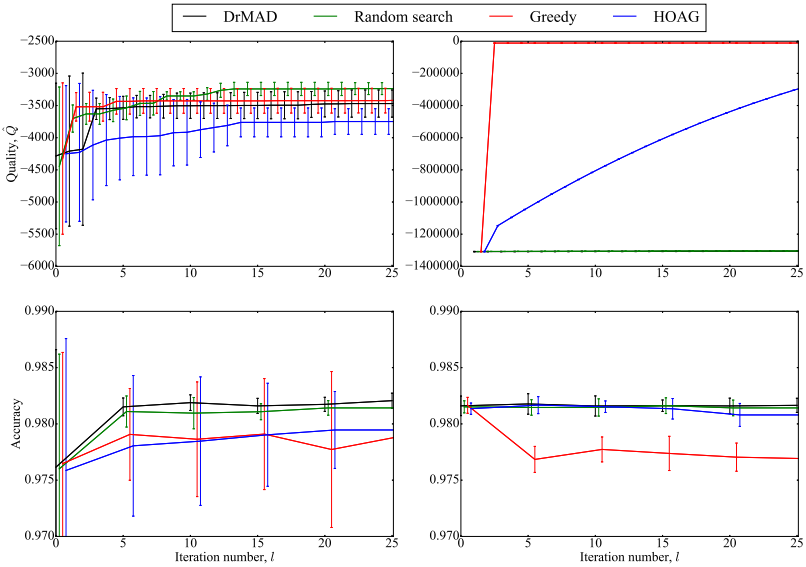


CV



Evidence

Experiment: MNIST



Experiment: MNIST

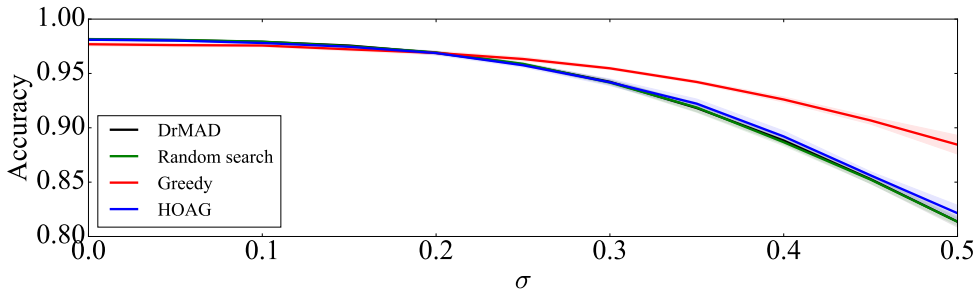
Adding Gaussian noise $\mathcal{N}(0, \sigma^2 \mathrm{I})$:

Без шума

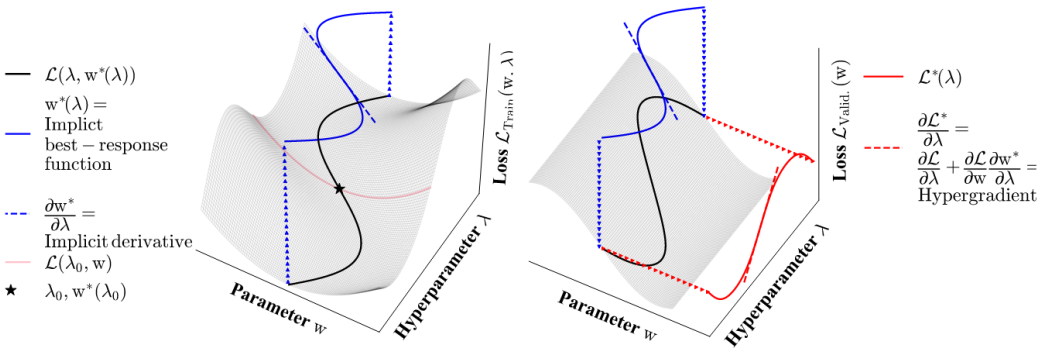
$\sigma = 0.1$

$\sigma = 0.25$

$\sigma = 0.5$



Optimizing Millions of Hyperparameters by Implicit Differentiation



Optimizing Millions of Hyperparameters by Implicit Differentiation

Theorem 1 (Cauchy, Implicit Function Theorem). *If for some* $(\boldsymbol{\lambda}', \mathbf{w}')$, $\frac{\partial \mathcal{L}_{\mathrm{T}}}{\partial \mathbf{w}}|_{\boldsymbol{\lambda}', \mathbf{w}'} = 0$ *and regularity conditions are satisfied, then surrounding* $(\boldsymbol{\lambda}', \mathbf{w}')$ *there is a function* $\mathbf{w}^*(\boldsymbol{\lambda})$ *s.t.* $\frac{\partial \mathcal{L}_{\mathrm{T}}}{\partial \mathbf{w}}|_{\boldsymbol{\lambda}, \mathbf{w}^*(\boldsymbol{\lambda})} = 0$ *and we have:*

$$\frac{\partial \mathbf{w}^*}{\partial \boldsymbol{\lambda}}\Big|_{\boldsymbol{\lambda}'} = -\Big[\underbrace{\frac{\partial^2 \mathcal{L}_{\mathrm{T}}}{\partial \mathbf{w} \partial \mathbf{w}}}_{\text{training Hessian}}\Big]^{-1} \times \underbrace{\frac{\partial^2 \mathcal{L}_{\mathrm{T}}}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}}_{\text{training mixed partials}}\Big|_{\boldsymbol{\lambda}', \mathbf{w}^*(\boldsymbol{\lambda}')} \quad \text{(IFT)}$$

$$\left[\frac{\partial^2 \mathcal{L}_{\mathrm{T}}}{\partial \mathbf{w} \partial \mathbf{w}}\right]^{-1} = \lim_{i \to \infty} \sum_{j=0}^{i} \left[I - \frac{\partial^2 \mathcal{L}_{\mathrm{T}}}{\partial \mathbf{w} \partial \mathbf{w}}\right]^j$$

Optimizing Millions of Hyperparameters by Implicit Differentiation

Jonathan Lorraine, Paul Vicol, David Duvenaud

Method	Steps	Eval.	Hypergradient Approximation
Exact IFT	∞	$\mathbf{w}^*(\boldsymbol{\lambda})$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \left[\frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \mathbf{w}}\right]^{-1} \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\mathbf{w}^*(\boldsymbol{\lambda})}$
Unrolled Diff. [Maclaurin et al., 2015]	i	$\mathbf{w}_0$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \sum_{j \leq i} \left[\prod_{k<j} I - \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \mathbf{w}}\Big\|_{\mathbf{w}_{i-k}}\right] \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\mathbf{w}_{i-j}}$
L-Step Truncated Unrolled Diff.	i	$\mathbf{w}_L$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \sum_{L \leq j \leq i} \left[\prod_{k<j} I - \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \mathbf{w}}\Big\|_{\mathbf{w}_{i-k}}\right] \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\mathbf{w}_{i-j}}$
Larsen et al. [1996]	∞	$\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \left[\frac{\partial \mathcal{L}_T}{\partial \mathbf{w}} \frac{\partial \mathcal{L}_T}{\partial \mathbf{w}}^T\right]^{-1} \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})}$
Bengio [2000]	∞	$\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \left[\frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \mathbf{w}}\right]^{-1} \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})}$
$T1-T2$ [27]	1	$\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times [I]^{-1} \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})}$
Ours	i	$\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \left(\sum_{j \leq i} \left[I - \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \mathbf{w}}\right]^j\right) \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})}$
Conjugate Gradient (CG) $\approx$	-	$\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})$	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} - \left(\arg\min_{\mathbf{x}} \|\mathbf{x} \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \mathbf{w}} - \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}}\|\right) \frac{\partial^2 \mathcal{L}_T}{\partial \mathbf{w} \partial \boldsymbol{\lambda}}\Big\|_{\widehat{\mathbf{w}^*}(\boldsymbol{\lambda})}$
Hypernetwork	-	-	$\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} + \frac{\partial \mathcal{L}_V}{\partial \mathbf{w}} \times \frac{\partial \mathbf{w}^*_\phi}{\partial \boldsymbol{\lambda}}$ where $\mathbf{w}^*_\phi(\boldsymbol{\lambda}) = \arg\min_\phi \mathcal{L}_T(\boldsymbol{\lambda}, \mathbf{w}_\phi(\boldsymbol{\lambda}))$
Bayesian Optimization	-	-	$\frac{\partial \mathbb{E}[\mathcal{L}^*_V]}{\partial \boldsymbol{\lambda}}$ where $\mathcal{L}^*_V \sim$ Gaussian-Process$(\{\boldsymbol{\lambda}_i, \mathcal{L}_V(\boldsymbol{\lambda}_i, \mathbf{w}^*(\boldsymbol{\lambda}_i))\})$

References

- Bishop C. M. Pattern recognition //Machine learning. – 2006. – T. 128. – №. 9.
- Bakhteev O. Y., Strijov V. V. Comprehensive analysis of gradient-based hyperparameter optimization algorithms //Annals of Operations Research. – 2020. – T. 289. – №. 1. – C. 51-65.
- Graves A. Practical variational inference for neural networks //Advances in neural information processing systems. – 2011. – T. 24.
- Bergstra et al., Random Search for Hyper-Parameter Optimization, 2012
- Dougal Maclaurin et. al, Gradient-based Hyperparameter Optimization through Reversible Learning, 2015
- Jelena Luketina et. al, Scalable Gradient-Based Tuning of Continuous Regularization Hyperparameters, 2016
- Jie Fu et. al, DrMAD: Distilling Reverse-Mode Automatic Differentiation for Optimizing Hyperparameters of Deep Neural Networks, 2016
- Fabian Pedregosa, Hyperparameter optimization with approximate gradient, 2016
- Bobak Shahriari et. al, Taking the Human Out of the Loop: A Review of Bayesian Optimization, 2016
- Liu H., Simonyan K., Yang Y. Darts: Differentiable architecture search //arXiv preprint arXiv:1806.09055. – 2018.
- Lorraine, J., Vicol, P., & Duvenaud, D. (2020, June). Optimizing millions of hyperparameters by implicit differentiation. In International Conference on Artificial Intelligence and Statistics (pp. 1540-1552). PMLR. (see also Victor Pankratov's slides, BMM-2021)