Improving Optimizer Evaluation in Deep Learning

ICLR 2022 Workshop - ML Evaluation Standards

Frank Schneider

April 29, 2022

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS

erc

Federal Ministry of Education and Research

imprs-is

The State of Deep Learning Optimization

A crowded field of methods and hyperparameters

A huge number of optimization methods

AcceleGrad	AMSBound	K-BFGS/K-BFGS(L)	RMSProp
ACClip	AMSGrad	KF-QN-CNN	RMSterov
AdaAlter	AngularGrad	KFAC	S-SGD
AdaBatch	ArmijoLS	KFLR/KFRA	SAdam
AdaBayes	ARSG	L4Adam/L4Momentum	Sadam/SAMSGrad
AdaBelief	ASAM	LAMB	SALR
AdaBlock	AutoLRS	LaProp	SAM
AdaBound	AvaGrad	LARS	SC-Adagrad/-RMSProp
AdaComp	BAdam	LHOPT	SDProp
Adadelta	BGAdam	LookAhead	SGD
Adafactor	BPGrad	M-SVAG	SGD-BB
AdaFix	BRMSProp	MADGRAD	SGD-G2
AdaFom	BSGD	MAS	SGDEM
AdaFTRL	C-ADAM	MEKA	SGDHess
Adagrad	CADA	MTAdam	SGDM
ADAHESSIAN	Cool Momentum	MVRC-1/MVRC-2	SGDR
Adai	CProp	Nadam	SHAdagrad
AdaLoss	Curveball	NAMSB/NAMSG	Shampoo
Adam	Dadam	ND-Adam	SignAdam++
Adam^{+}	DeepMemory	Nero	SignSGD
AdamAL	DGNOpt	Nesterov	SKQN/S4QN
AdaMax	DiffGrad	Noisy Adam/Noisy K-FAC	SM3
AdamBS	EAdam	NosAdam	SMG
AdamNC	EKFAC	Novograd	SNGM
AdaMod	Eve	NT-SGD	SoftAdam
AdamP/SGDP	Expectigrad	Padam	SRSGD
AdamT	FastAdaBelief	PAGE	Step-Tuned SGD
AdamW	FRSGD	PAL	SWATS
AdamX	G-AdaGrad	PolyAdam	SWNTS
ADAS	GADAM	Polyak	TAdam
AdaS	Gadam	PowerSGD/PowerSGDM	TEKFAC
AdaScale	GOALS	Probabilistic Polyak	VAdam
AdaSGD	GOLS-I	ProbLS	VR-SGD
AdaShift	Grad-Avg	PStorm	vSGD-b/vSGD-g/vSGD-l
AdaSqrt	GRAPES	QHAdam/QHM	vSGD-fd
Adathm	Gravilon	RAdam	WNGrad
AdaX/AdaX-W	Gravity	Ranger	YellowFin
AEGD	HAdam	RangerLars	Yogi
ALI-G	HyperAdam		

The State of Deep Learning Optimization

A crowded field of methods and hyperparameters

A huge number of optimization methods

AcceleGrad	AMSBound	K-BFGS/K-BFGS(L)	RMSProp
ACClip	AMSGrad	KF-QN-CNN	RMSterov
AdaAlter	AngularGrad	KFAC	S-SGD
AdaBatch	ArmijoLS	KFLR/KFRA	SAdam
AdaBayes	ARSG	L4Adam/L4Momentum	Sadam/SAMSGrad
AdaBelief	ASAM	LAMB	SALR
AdaBlock	AutoLRS	LaProp	SAM
AdaBound	AvaGrad	LARS	SC-Adagrad/-RMSProp
AdaComp	BAdam	LHOPT	SDProp
Adadelta	BGAdam	LookAhead	SGD
Adafactor	BPGrad	M-SVAG	SGD-BB
AdaFix	BRMSProp	MADGRAD	SGD-G2
AdaFom	BSGD	MAS	SGDEM
AdaFTRL	C-ADAM	MEKA	SGDHess
Adagrad	CADA	MTAdam	SGDM
ADAHESSIAN	Cool Momentum	MVRC-1/MVRC-2	SGDR
Adai	CProp	Nadam	SHAdagrad
AdaLoss	Curveball	NAMSB/NAMSG	Shampoo
Adam	Dadam	ND-Adam	SignAdam++
Adam^+	DeepMemory	Nero	SignSGD
AdamAL	DGNOpt	Nesterov	SKQN/S4QN
AdaMax	DiffGrad	Noisy Adam/Noisy K-FAC	SM3
AdamBS	EAdam	NosAdam	SMG
AdamNC	EKFAC	Novograd	SNGM
AdaMod	Eve	NT-SGD	SoftAdam
AdamP/SGDP	Expectigrad	Padam	SRSGD
AdamT	FastAdaBelief	PAGE	Step-Tuned SGD
AdamW	FRSGD	PAL	SWATS
AdamX	G-AdaGrad	PolyAdam	SWNTS
ADAS	GADAM	Polyak	TAdam
AdaS	Gadam	PowerSGD/PowerSGDM	TEKFAC
AdaScale	GOALS	Probabilistic Polyak	VAdam
AdaSGD	GOLS-I	ProbLS	VR-SGD
AdaShift	Grad-Avg	PStorm	vSGD-b/vSGD-g/vSGD-l
AdaSqrt	GRAPES	QHAdam/QHM	vSGD-fd
Adathm	Gravilon	RAdam	WNGrad
AdaX/AdaX-W	Gravity	Ranger	YellowFin
AEGD	HAdam	RangerLars	Yogi
ALI-G	HyperAdam		

Tuning hyperparameters

SGD:

$$\theta = \theta - \boxed{\eta}\, \boldsymbol{g}$$

The State of Deep Learning Optimization

A crowded field of methods and hyperparameters

A huge number of optimization methods

AcceleGrad	AMSBound	K-BFGS/K-BFGS(L)	RMSProp
ACClip	AMSGrad	KF-QN-CNN	RMSterov
AdaAlter	AngularGrad	KFAC	S-SGD
AdaBatch	ArmijoLS	KFLR/KFRA	SAdam
AdaBayes	ARSG	L4Adam/L4Momentum	Sadam/SAMSGrad
AdaBelief	ASAM	LAMB	SALR
AdaBlock	AutoLRS	LaProp	SAM
AdaBound	AvaGrad	LARS	SC-Adagrad/-RMSProp
AdaComp	BAdam	LHOPT	SDProp
Adadelta	BGAdam	LookAhead	SGD
Adafactor	BPGrad	M-SVAG	SGD-BB
AdaFix	BRMSProp	MADGRAD	SGD-G2
AdaFom	BSGD	MAS	SGDEM
AdaFTRL	C-ADAM	MEKA	SGDHess
Adagrad	CADA	MTAdam	SGDM
ADAHESSIAN	Cool Momentum	MVRC-1/MVRC-2	SGDR
Adai	CProp	Nadam	SHAdagrad
AdaLoss	Curveball	NAMSB/NAMSG	Shampoo
Adam	Dadam	ND-Adam	SignAdam++
Adam$^+$	DeepMemory	Nero	SignSGD
AdamAL	DGNOpt	Nesterov	SKQN/S4QN
AdaMax	DiffGrad	Noisy Adam/Noisy K-FAC	SM3
AdamBS	EAdam	NosAdam	SMG
AdamNC	EKFAC	Novograd	SNGM
AdaMod	Eve	NT-SGD	SoftAdam
AdamP/SGDP	Expectigrad	Padam	SRSGD
AdamT	FastAdaBelief	PAGE	Step-Tuned SGD
AdamW	FRSGD	PAL	SWATS
AdamX	G-AdaGrad	PolyAdam	SWNTS
ADAS	GADAM	Polyak	TAdam
AdaS	Gadam	PowerSGD/PowerSGDM	TEKFAC
AdaScale	GOALS	Probabilistic Polyak	VAdam
AdaSGD	GOLS-I	ProbLS	VR-SGD
AdaShift	Grad-Avg	PStorm	vSGD-b/vSGD-g/vSGD-l
AdaSqrt	GRAPES	QHAdam/QHM	vSGD-fd
Adathm	Gravilon	RAdam	WNGrad
AdaX/AdaX-W	Gravity	Ranger	YellowFin
AEGD	HAdam	RangerLars	Yogi
ALI-G	HyperAdam		

Tuning hyperparameters

SGD:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \boxed{\eta}\, \boldsymbol{g}$$

ADAM:

$$\boldsymbol{m} = \boxed{\beta_1}\, \boldsymbol{m} + (1 - \beta_1)\boldsymbol{g}$$

$$\boldsymbol{v} = \boxed{\beta_2}\, \boldsymbol{v} + (1 - \beta_2)\boldsymbol{g} \odot \boldsymbol{g}$$

$$\hat{\boldsymbol{m}} = \frac{\boldsymbol{m}}{1 - \beta_1^t} \qquad \hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{1 - \beta_2^t}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \frac{\boxed{\eta}}{\sqrt{\hat{\boldsymbol{v}}} + \boxed{\varepsilon}} \odot \hat{\boldsymbol{m}}$$

The State of Deep Learning Optimization

A crowded field of methods and hyperparameters

A huge number of optimization methods

AcceleGrad
ACClip
AdaAlter
AdaBatch
AdaBayes
AdaBelief
AdaBlock
AdaBound
AdaComp
Adadelta
Adafactor
AdaFix
AdaFom
AdaFTRL
Adagrad
ADAHESSIAN
Adai
AdaLoss
Adam
Adam$^+$
AdamAL
AdaMax
AdamBS
AdamNC
AdaMod
AdamP/SGDP
AdamT
AdamW
AdamX
ADAS
AdaS
AdaScale
AdaSGD
AdaShift
AdaSqrt
Adathm
AdaX/AdaX-W
AEGD
ALI-G
AMSBound
AMSGrad
AngularGrad
ArmijoLS
ARSG
ASAM
AutoLRS
AvaGrad
BAdam
BGAdam
BPGrad
BRMSProp
BSGD
C-ADAM
CADA
Cool Momentum
CProp
Curveball
Dadam
DeepMemory
DGNOpt
DiffGrad
EAdam
EKFAC
Eve
Expectigrad
FastAdaBelief
FRSGD
G-AdaGrad
GADAM
Gadam
GOALS
GOLS-I
Grad-Avg
GRAPES
Gravilon
Gravity
HAdam
HyperAdam
K-BFGS/K-BFGS(L)
KF-QN-CNN
KFAC
KFLR/KFRA
L4Adam/L4Momentum
LAMB
LaProp
LARS
LHOPT
LookAhead
M-SVAG
MADGRAD
MAS
MEKA
MTAdam
MVRC-1/MVRC-2
Nadam
NAMSB/NAMSG
ND-Adam
Nero
Nesterov
Noisy Adam/Noisy K-FAC
NosAdam
Novograd
NT-SGD
Padam
PAGE
PAL
PolyAdam
Polyak
PowerSGD/PowerSGDM
Probabilistic Polyak
ProbLS
PStorm
QHAdam/QHM
RAdam
Ranger
RangerLars
RMSProp
RMSterov
S-SGD
SAdam
Sadam/SAMSGrad
SALR
SAM
SC-Adagrad/-RMSProp
SDProp
SGD
SGD-BB
SGD-G2
SGDEM
SGDHess
SGDM
SGDR
SHAdagrad
Shampoo
SignAdam++
SignSGD
SKQN/S4QN
SM3
SMG
SNGM
SoftAdam
SRSGD
Step-Tuned SGD
SWATS
SWNTS
TAdam
TEKFAC
VAdam
VR-SGD
vSGD-b/vSGD-g/vSGD-l
vSGD-fd
WNGrad
YellowFin
Yogi

Benchmarks

Tuning hyperparameters

SGD:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta\, \boldsymbol{g}$$

Adam:

$$\boldsymbol{m} = \beta_1 \boldsymbol{m} + (1 - \beta_1)\boldsymbol{g}$$

$$\boldsymbol{v} = \beta_2 \boldsymbol{v} + (1 - \beta_2)\boldsymbol{g} \odot \boldsymbol{g}$$

$$\hat{\boldsymbol{m}} = \frac{\boldsymbol{m}}{1 - \beta_1^t} \qquad \hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{1 - \beta_2^t}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \frac{\eta}{\sqrt{\hat{\boldsymbol{v}}} + \varepsilon} \odot \hat{\boldsymbol{m}}$$

The State of Deep Learning Optimization

A crowded field of methods and hyperparameters

A huge number of optimization methods

AcceleGrad
ACClip
AdaAlter
AdaBatch
AdaBayes
AdaBelief
AdaBlock
AdaBound
AdaComp
Adadelta
Adafactor
AdaFix
AdaFom
AdaFTRL
Adagrad
ADAHESSIAN
Adai
AdaLoss
Adam
Adam^{+}
AdamAL
AdaMax
AdamBS
AdamNC
AdaMod
AdamP/SGDP
AdamT
AdamW
AdamX
ADAS
AdaS
AdaScale
AdaSGD
AdaShift
AdaSqrt
Adathm
AdaX/AdaX-W
AEGD
ALI-G
AMSBound
AMSGrad
AngularGrad
ArmijoLS
ARSG
ASAM
AutoLRS
AvaGrad
BAdam
BGAdam
BPGrad
BRMSProp
BSGD
C-ADAM
CADA
Cool Momentum
CProp
Curveball
Dadam
DeepMemory
DGNOpt
DiffGrad
EAdam
EKFAC
Eve
Expectigrad
FastAdaBelief
FRSGD
G-AdaGrad
GADAM
Gadam
GOALS
GOLS-I
Grad-Avg
GRAPES
Gravilon
Gravity
HAdam
HyperAdam
K-BFGS/K-BFGS(L)
KF-QN-CNN
KFAC
KFLR/KFRA
L4Adam/L4Momentum
LAMB
LaProp
LARS
LHOPT
LookAhead
M-SVAG
MADGRAD
MAS
MEKA
MTAdam
MVRC-1/MVRC-2
Nadam
NAMSB/NAMSG
ND-Adam
Nero
Nesterov
Noisy Adam/Noisy K-FAC
NosAdam
Novograd
NT-SGD
Padam
PAGE
PAL
PolyAdam
Polyak
PowerSGD/PowerSGDM
Probabilistic Polyak
ProbLS
PStorm
QHAdam/QHM
RAdam
Ranger
RangerLars
RMSProp
RMSterov
S-SGD
SAdam
Sadam/SAMSGrad
SALR
SAM
SC-Adagrad/-RMSProp
SDProp
SGD
SGD-BB
SGD-G2
SGDEM
SGDHess
SGDM
SGDR
SHAdagrad
Shampoo
SignAdam++
SignSGD
SKQN/S4QN
SM3
SMG
SNGM
SoftAdam
SRSGD
Step-Tuned SGD
SWATS
SWNTS
TAdam
TEKFAC
VAdam
VR-SGD
vSGD-b/vSGD-g/vSGD-l
vSGD-fd
WNGrad
YellowFin
Yogi

Tuning hyperparameters

SGD:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \boxed{\eta}\, \boldsymbol{g}$$

ADAM:

$$\boldsymbol{m} = \boxed{\beta_1}\, \boldsymbol{m} + (1 - \beta_1)\boldsymbol{g}$$

$$\boldsymbol{v} = \boxed{\beta_2}\, \boldsymbol{v} + (1 - \beta_2)\boldsymbol{g} \odot \boldsymbol{g}$$

$$\hat{\boldsymbol{m}} = \frac{\boldsymbol{m}}{1 - \beta_1^t} \qquad \hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{1 - \beta_2^t}$$

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \frac{\boxed{\eta}}{\sqrt{\hat{\boldsymbol{v}}} + \boxed{\varepsilon}} \odot \hat{\boldsymbol{m}}$$

Benchmarks

Debugging Tools

The How, What & Why of Deep Learning Optimization

Making neural network training more user-friendly in three steps

How can we fairly compare deep learning optimizers? DEEPOBS is a Python package for benchmarking optimizers.

ICLR 2019

What are the best deep learning optimization methods? An extensive empirical comparison of fifteen popular deep learning optimizers.

ICML International Conference On Machine Learning 2021

Why do/don't deep learning optimizers work? COCKPIT is a visual and statistical debugger specifically designed for deep learning.

NEURAL INFORMATION PROCESSING SYSTEMS 2021

The How, What & Why of Deep Learning Optimization

Making neural network training more user-friendly in three steps

How can we fairly compare deep learning optimizers? DeepOBS is a Python package for benchmarking optimizers.

ICLR 2019

What are the best deep learning optimization methods? An extensive empirical comparison of fifteen popular deep learning optimizers.

ICML 2021

Why do/don't deep learning optimizers work? Cockpit is a visual and statistical debugger specifically designed for deep learning.

Neural Information Processing Systems 2021

The Status Quo of Benchmarking Deep Learning Optimizers

Why we need an independent & standardized optimizer benchmark protocol

DEEPOBS (Schneider, Balles, et al. 2019)

Deep learning benchmarks require time

- Use of rather small test problems
- Use only a few test problems
- Repeated work & incomparable results

Misaligned incentives

- Own optimizer gets more attention than the competition
- Potential for cherry-picking

The Status Quo of Benchmarking Deep Learning Optimizers

Why we need an independent & standardized optimizer benchmark protocol

DEEPOBS (Schneider, Balles, et al. 2019)

Deep learning benchmarks require time

- Use of rather small test problems
- Use only a few test problems
- Repeated work & incomparable results

Misaligned incentives

- Own optimizer gets more attention than the competition
- Potential for cherry-picking

Solution: Independent & standardized benchmark protocol

DEEPOBS: The Deep Learning Optimizer Benchmark Suite

Fairer, faster & more comparable evaluations with less work

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

DEEPOBS (Schneider, Balles, et al. 2019)

1. Run the optimizer automatically on multiple meaningful test problems.
2. Compare to the provided state of the art without extra costs.
3. Plot the results & the standardized summary evaluation.

The How, What & Why of Deep Learning Optimization

Making neural network training more user-friendly in three steps

How can we fairly compare deep learning optimizers? DEEPOBS is a Python package for benchmarking optimizers.

ICLR 2019

What are the best deep learning optimization methods? An extensive empirical comparison of fifteen popular deep learning optimizers.

ICML International Conference On Machine Learning 2021

Why do/don't deep learning optimizers work? COCKPIT is a visual and statistical debugger specifically designed for deep learning.

NEURAL INFORMATION PROCESSING SYSTEMS 2021

The Benchmarking Process

Many dimensions to explore at once

Benchmark (Schmidt et al. 2021)

$$\underset{\textbf{Optimizer}}{\begin{Bmatrix} \text{Adam} \\ \text{NAG} \\ \cdots \\ \text{SGD} \end{Bmatrix}_{15}} \times \underset{\textbf{Problem}}{\begin{Bmatrix} \text{P1} \\ \text{P2} \\ \cdots \\ \text{P8} \end{Bmatrix}_{8}} \times \underset{\textbf{Tuning}}{\begin{Bmatrix} \text{one-shot} \\ \text{small} \\ \text{medium} \\ \text{large} \end{Bmatrix}_{4}} \times \underset{\textbf{Schedule}}{\begin{Bmatrix} \text{constant} \\ \text{cosine} \\ \text{cosine wr} \\ \text{trapez.} \end{Bmatrix}_{4}}$$

The Benchmarking Process

Many dimensions to explore at once

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Benchmark (Schmidt et al. 2021)

$$\underbrace{\begin{Bmatrix} \text{Adam} \\ \text{NAG} \\ \ldots \\ \text{SGD} \end{Bmatrix}_{15}}_{\textbf{Optimizer}} \times \underbrace{\begin{Bmatrix} \text{P1} \\ \text{P2} \\ \ldots \\ \text{P8} \end{Bmatrix}_{8}}_{\textbf{Problem}} \times \underbrace{\begin{Bmatrix} \text{one-shot} \\ \text{small} \\ \text{medium} \\ \text{large} \end{Bmatrix}_{4}}_{\textbf{Tuning}} \times \underbrace{\begin{Bmatrix} \text{constant} \\ \text{cosine} \\ \text{cosine wr} \\ \text{trapez.} \end{Bmatrix}_{4}}_{\textbf{Schedule}}$$

- AMSBound
- AMSGrad
- AdaBelief
- AdaBound
- AdaDelta
- AdaGrad
- Adam
- LA(Mom.)
- LA(RAdam)
- Momentum
- NAG
- Nadam
- RAdam
- RMSProp
- SGD

The Benchmarking Process

Many dimensions to explore at once

Optimizer		Problem		Tuning		Schedule
$\left\{ \begin{matrix} \text{ADAM} \\ \text{NAG} \\ \dots \\ \text{SGD} \end{matrix} \right\}_{15}$	$\times$	$\left\{ \begin{matrix} \text{P1} \\ \text{P2} \\ \dots \\ \text{P8} \end{matrix} \right\}_{8}$	$\times$	$\left\{ \begin{matrix} \text{one-shot} \\ \text{small} \\ \text{medium} \\ \text{large} \end{matrix} \right\}_{4}$	$\times$	$\left\{ \begin{matrix} \text{constant} \\ \text{cosine} \\ \text{cosine wr} \\ \text{trapez.} \end{matrix} \right\}_{4}$

	Data set	Model	Task	Metric	Batch size	Budget *in epochs*	Approx. run time
P1	Artificial	**Noisy quadratic**	Minimization	Loss	128	100	$<$ 1 min
P2	MNIST	**VAE**	Generative	Loss	64	50	10 min
P3	Fashion-MNIST	**Simple CNN: *2c2d***	Classification	Accuracy	128	100	20 min
P4	CIFAR-10	**Simple CNN: *3c3d***	Classification	Accuracy	128	100	35 min
P5	Fashion-MNIST	**VAE**	Generative	Loss	64	100	20 min
P6	CIFAR-100	***All-CNN-C***	Classification	Accuracy	256	350	4 h 00 min
P7	SVHN	***Wide ResNet 16-4***	Classification	Accuracy	128	160	3 h 30 min
P8	War and Peace	**RNN**	Character Prediction	Accuracy	50	200	5 h 30 min

$$\begin{array}{cccc} \textbf{Optimizer} & \textbf{Problem} & \textbf{Tuning} & \textbf{Schedule} \\ \left\{\begin{array}{l}\text{ADAM}\\ \text{NAG}\\ \dots\\ \text{SGD}\end{array}\right\}_{15} \times & \left\{\begin{array}{l}\text{P1}\\ \text{P2}\\ \dots\\ \text{P8}\end{array}\right\}_{8} \times & \left\{\begin{array}{l}\text{one-shot}\\ \text{small}\\ \text{medium}\\ \text{large}\end{array}\right\}_{4} \times & \left\{\begin{array}{l}\text{constant}\\ \text{cosine}\\ \text{cosine wr}\\ \text{trapez.}\end{array}\right\}_{4} \end{array}$$

- **One-Shot** - 1 Run
  No tuning, uses default hyperparameters
- **Small** - 25 Runs
  Tuned via random search
- **Medium** - 50 Runs
  Tuned via random search, superset of *small budget*
- **Large** - 75 Runs
  Tuned via random search, refined search spaces

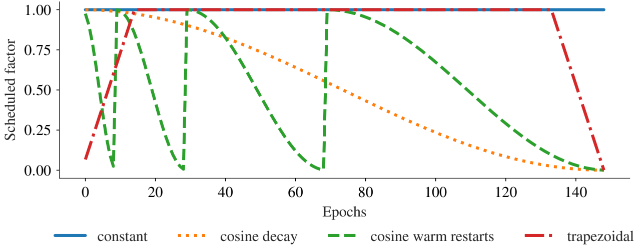
The Benchmarking Process

Many dimensions to explore at once

Benchmark (Schmidt et al. 2021)

Optimizer		Problem		Tuning		**Schedule**

$$\begin{Bmatrix} \text{ADAM} \\ \text{NAG} \\ \cdots \\ \text{SGD} \end{Bmatrix}_{15} \times \begin{Bmatrix} \text{P1} \\ \text{P2} \\ \cdots \\ \text{P8} \end{Bmatrix}_{8} \times \begin{Bmatrix} \text{one-shot} \\ \text{small} \\ \text{medium} \\ \text{large} \end{Bmatrix}_{4} \times \begin{Bmatrix} \text{constant} \\ \text{cosine} \\ \text{cosine wr} \\ \text{trapez.} \end{Bmatrix}_{4}$$



The Benchmarking Process

Many dimensions to explore at once

$$\underset{\textbf{Optimizer}}{\left\{\begin{matrix}\textsc{Adam}\\ \text{NAG}\\ \ldots\\ \text{SGD}\end{matrix}\right\}_{15}} \times \underset{\textbf{Problem}}{\left\{\begin{matrix}\text{P1}\\ \text{P2}\\ \ldots\\ \text{P8}\end{matrix}\right\}_{8}} \times \underset{\textbf{Tuning}}{\left\{\begin{matrix}\text{one-shot}\\ \text{small}\\ \text{medium}\\ \text{large}\end{matrix}\right\}_{4}} \times \underset{\textbf{Schedule}}{\left\{\begin{matrix}\text{constant}\\ \text{cosine}\\ \text{cosine wr}\\ \text{trapez.}\end{matrix}\right\}_{4}}$$

1,920 **Configurations**
fifteen optimizers, eight problems, four budgets, & four schedules

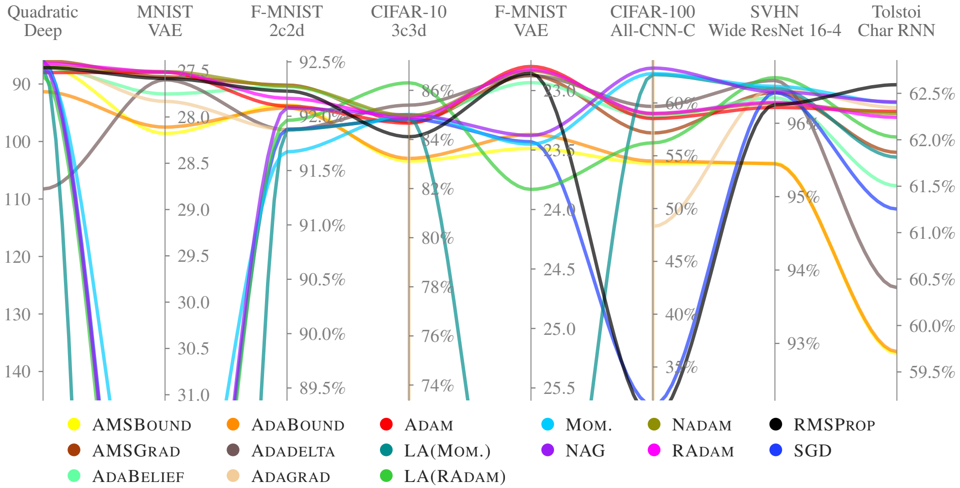
>50,000 **Individual Runs**
includes tuning runs & runs stochastic fidelity

128 **Settings** each optimizer is tested in
eight problems, four budgets, & four schedules

The Benchmarking Results

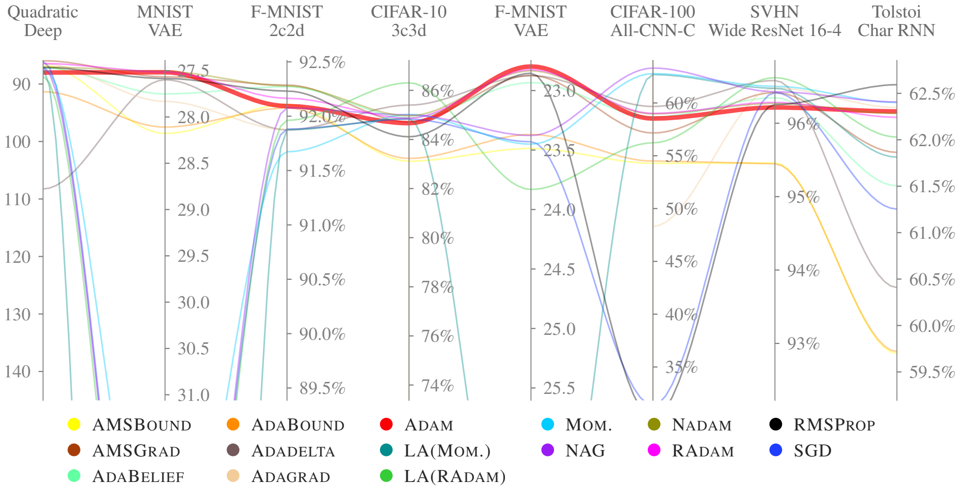
Why ADAM is still a good choice | *large budget with a trapezoidal schedule*

Benchmark (Schmidt et al. 2021)



The Benchmarking Results

Why ADAM is still a good choice | *large budget with a trapezoidal schedule*

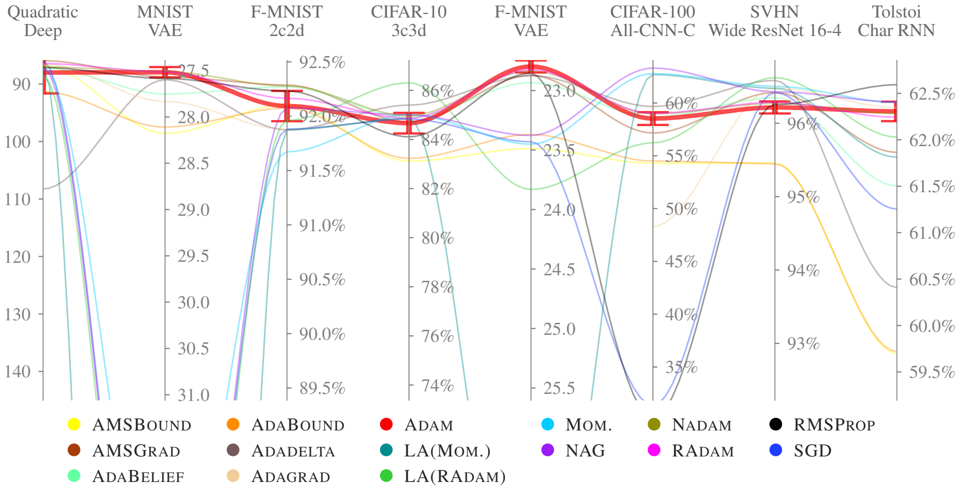


The Benchmarking Results

Why ADAM is still a good choice | *large budget with a trapezoidal schedule*

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Benchmark (Schmidt et al. 2021)

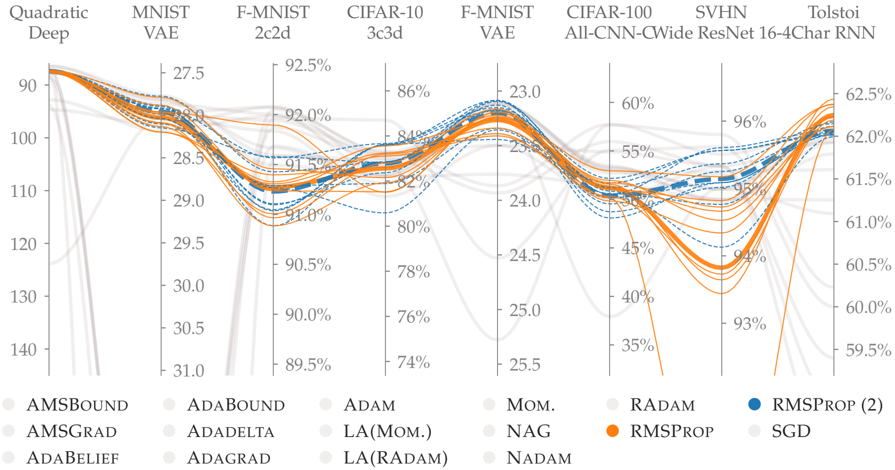


The Benchmarking Results

Individual results can change after re-tuning but overall trends remain

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Benchmark (Schmidt et al. 2021)



MLCommons

A natural extension of our benchmarking efforts

ML Commons

Algorithms Working Group

A competition to measure neural network training speedups due to algorithmic changes.

Chairs:

George Dahl Google

Frank Schneider University of Tübingen

MLCommons

Benchmarking more than just optimizers

Training Algorithm Track

- **Tests training algorithms** (update rules, data selection, hyperparameter spaces, etc.)
- **Scored by “time-to-result”** on multiple tasks
- **No manual workload-specific adaptation** is allowed

Model Track

- **Tests models** (architecture, initialization, data augmentation, etc.)
- **Scored by “time-to-result”** on a single task
- **Uses standard optimizer** and tuning procedures

MLCommons

Three ways to interact and contribute!

- **Join the Working Group!**
  Consider contributing to the working group to improve the benchmark.
- **Submit to the Competition!**
  Consider submitting your novel algorithm to our competition.
- **Check the Results!**
  Consider reading our (future) summary of the competition.

The How, What & Why of Deep Learning Optimization

Making neural network training more user-friendly in three steps

How can we fairly compare deep learning optimizers? DeepOBS is a Python package for benchmarking optimizers.

ICLR 2019

What are the best deep learning optimization methods? An extensive empirical comparison of fifteen popular deep learning optimizers.

ICML 2021

Why do/don't deep learning optimizers work? Cockpit is a visual and statistical debugger specifically designed for deep learning.

NEURAL INFORMATION PROCESSING SYSTEMS 2021

Deep Learning Requires New Debuggers

As coding is replaced by learning, we need to replace our debuggers

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

COCKPIT (Schneider, Dangel, et al. 2021)

Classic Programming

Functions

Level of Abstraction

Deep Learning Requires New Debuggers

As coding is replaced by learning, we need to replace our debuggers

Cockpit (Schneider, Dangel, et al. 2021)

Classic Programming

Zeros and Ones

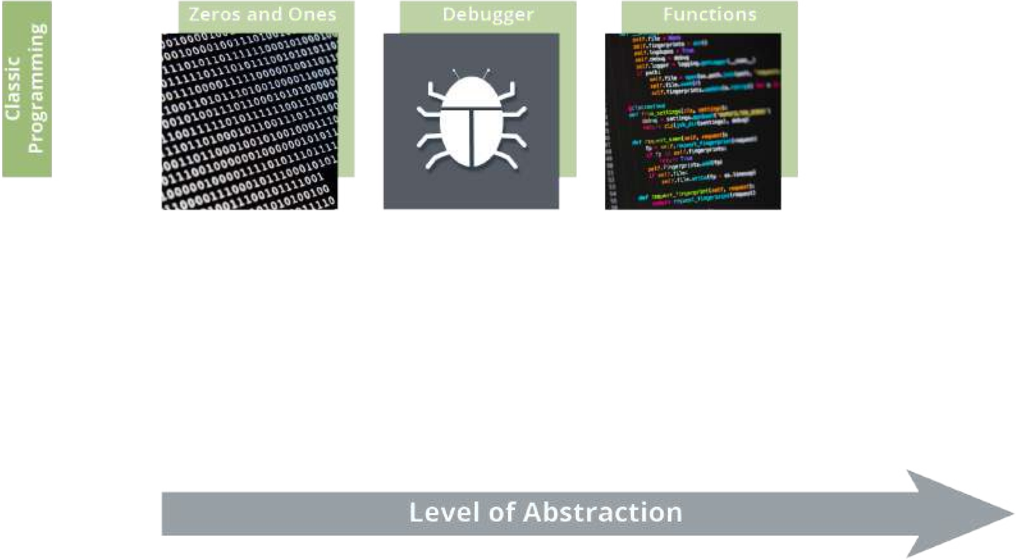
Functions

Level of Abstraction

Deep Learning Requires New Debuggers

As coding is replaced by learning, we need to replace our debuggers

COCKPIT (Schneider, Dangel, et al. 2021)



Deep Learning Requires New Debuggers

As coding is replaced by learning, we need to replace our debuggers

Cockpit (Schneider, Dangel, et al. 2021)

Classic Programming

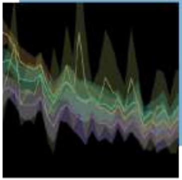
Zeros and Ones

Debugger

Functions

Deep Learning

Model Training



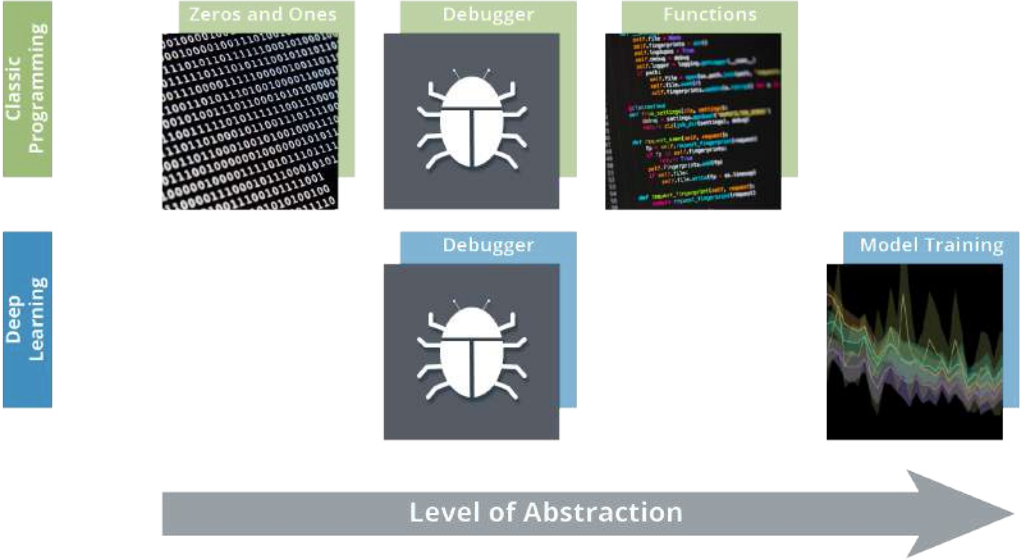
Level of Abstraction

Deep Learning Requires New Debuggers

As coding is replaced by learning, we need to replace our debuggers

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

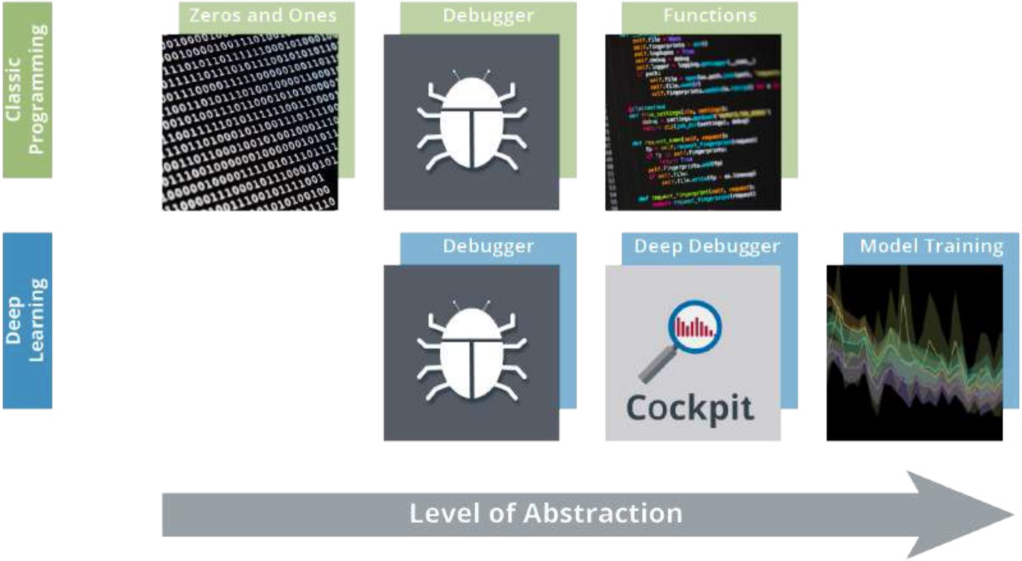
COCKPIT (Schneider, Dangel, et al. 2021)



Deep Learning Requires New Debuggers

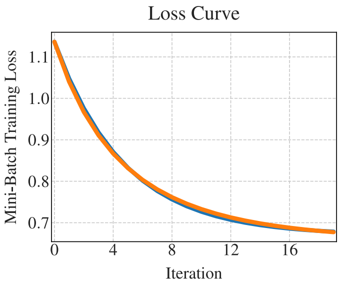
As coding is replaced by learning, we need to replace our debuggers

COCKPIT (Schneider, Dangel, et al. 2021)



Loss Curves Do Not Tell the Full Story

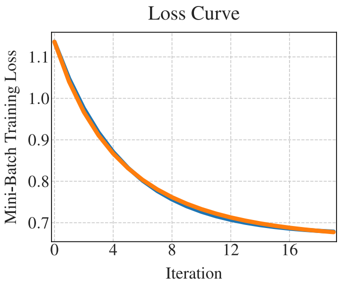
Why we need better observables in neural network training

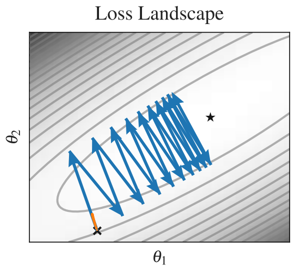


Loss Curves Do Not Tell the Full Story

Why we need better observables in neural network training

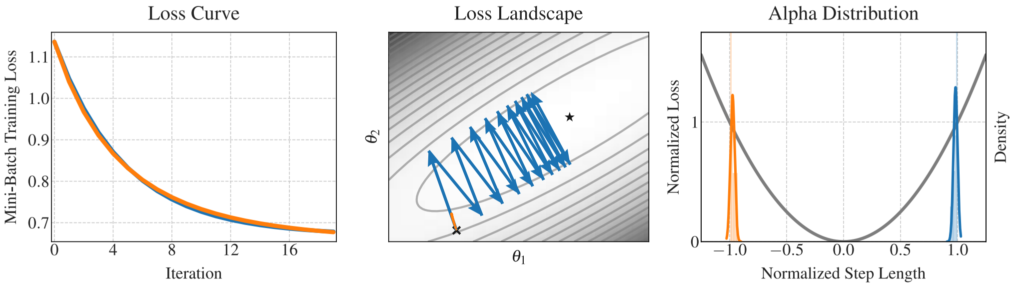
COCKPIT (Schneider, Dangel, et al. 2021)





Loss Curves Do Not Tell the Full Story

Why we need better observables in neural network training

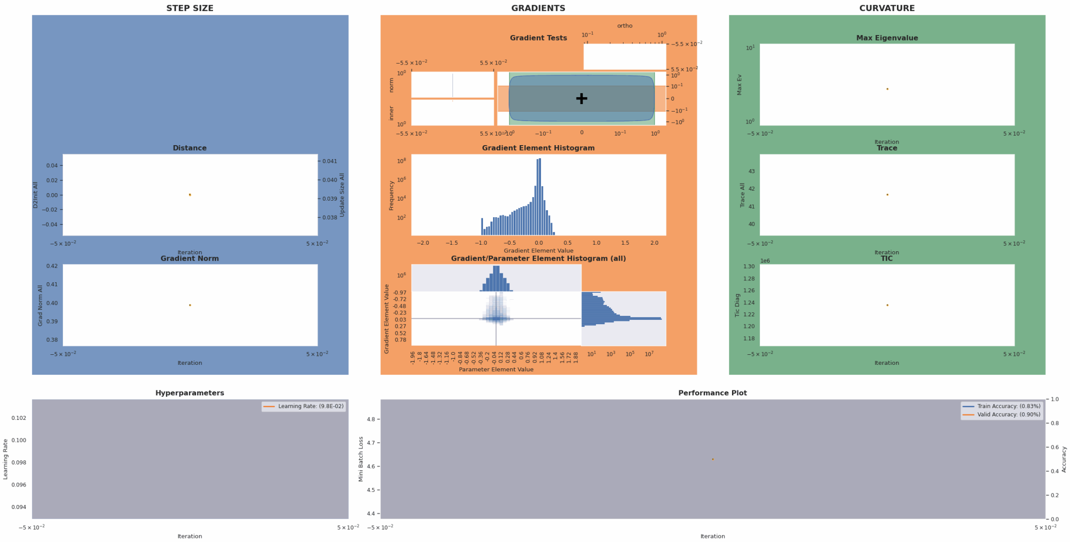


Cockpit in Action!

Training the All-CNN-C network through the lens of Cockpit

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Cockpit (Schneider, Dangel, et al. 2021)



Summary

Improving Optimizer Evaluation in Deep Learning

- **DeepOBS**, a benchmarking suite for deep learning optimizers.
- An empirical comparison of fifteen optimization methods for deep learning.
- **Cockpit**, a visual debugging tool for deep learning.

DeepOBS: `pip install deepobs`

Benchmark: github.com/SirRob1997/Crowded-Valley---Results

Cockpit: `pip install cockpit-for-pytorch`

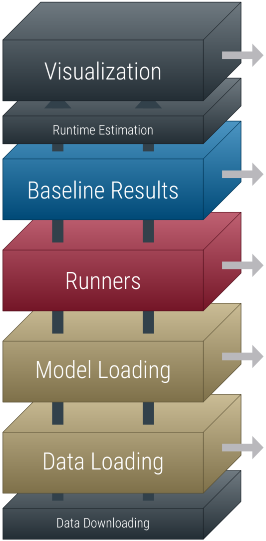
DeepOBS

The DeepOBS Stack

A modular environment for every step of evaluating deep learning optimizers

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

DeepOBS (Schneider, Balles, et al. 2019)



`.tex` files of learning curves for new optimizer and the baselines.

Performance results of the most popular optimizers.

Optimizer performance on a specific test problem.

Losses and accuracy of a deep learning model.

Pre-processed and batched data.

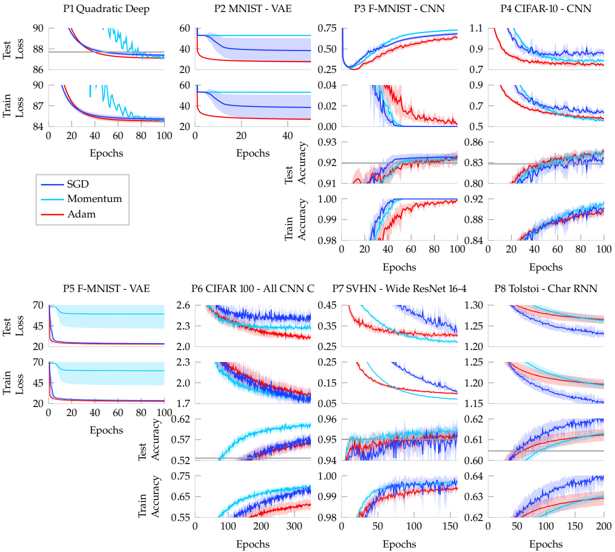
The DeepOBS Test Problems

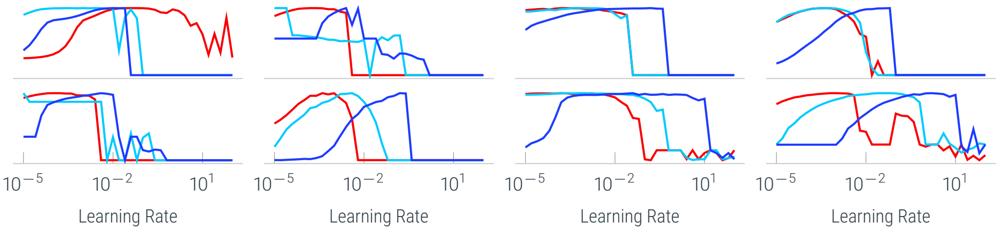
A wide selection of varied problems with room to grow.

Data set	Model	Description	*Conv*	*RNN*	*Drop*	*BN*	L^2
2D	Noisy Beale	*Noisy version of the Beale function* (Beale 1958)					
	Noisy Branin	*Noisy version of the Branin function* (Branin 1972)					
	Noisy Rosenbrock	*Noisy version of the Rosenbrock function* (Rosenbrock 1960)					
Quadratic	Deep	*100-dimensional ill-conditioned noisy quadratic* (Chaudhari et al. 2017)					
MNIST (LeCun et al. 1998)	Log. Regr.	*Logistic regression*					
	MLP	*Four layer fully connected network*					
	2c2d	*Two conv. and two fully connected layers*	✓				
	VAE	*Variational Autoencoder*	✓		✓		
Fashion MNIST (Xiao et al. 2017)	Log. Regr.	*Logistic regression*					
	MLP	*Four layer fully connected network*					
	2c2d	*Two conv. and two fully connected layers*	✓				
	VAE	*Variational Autoencoder*	✓		✓		
CIFAR-10 (Krizhevsky et al. 2009)	3c3d	*Three conv. and three fully connected layers*	✓				✓
	VGG16	*Adapted version of VGG16 (Simonyan et al. 2015)*	✓		✓		✓
	VGG19	*Adapted version of VGG19*	✓		✓		✓
CIFAR-100 (Krizhevsky et al. 2009)	3c3d	*Three conv. and three fully connected layers*	✓				✓
	VGG16	*Adapted version of VGG16*	✓		✓		✓
	VGG19	*Adapted version of VGG19*	✓		✓		✓
	All-CNN-C	*The all convolutional net from Springenberg et al. (2015)*	✓		✓		✓
	Wide ResNet-40-4	*Wide Residual Network (Zagoruyko et al. 2016)*	✓			✓	✓
SVHN (Netzer et al. 2011)	3c3d	*Three conv. and three fully connected layers*	✓				✓
	Wide ResNet-16-4	*Wide Residual Network*	✓			✓	✓
ImageNet (Deng et al. 2009)	VGG16	*VGG16*	✓		✓		✓
	VGG19	*VGG19*	✓		✓		✓
	Inception-v3	*Inception-v3 network as described by Szegedy et al. (2016)*	✓		✓	✓	✓
Tolstoi	CharRNN	*Recurrent Neural Network for character-level language modeling*		✓	✓		

Results Output of DEEPOBS

A showcase of DEEPOBS' capabilities





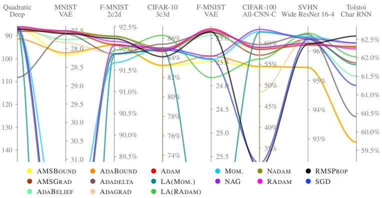
10^{-5}
10^{-2}
10^{1}
Learning Rate
10^{-5}
10^{-2}
10^{1}
Learning Rate
10^{-5}
10^{-2}
10^{1}
Learning Rate
10^{-5}
10^{-2}
10^{1}
Learning Rate

Results Output of DeepOBS

A showcase of DeepOBS' capabilities

Test Problem		SGD	Momentum	Adam
P1 **Quadratic Deep**	Performance	87.40	87.05	87.11
	Speed	51.1	70.5	39.9
	Tuneability	η: 1.58e-02	η: 2.51e-03 μ: 0.99	η: 3.98e-02 β_1: 0.9 β_2: 0.999 ε: 1e-08
P2 **MNIST VAE**	Performance	38.46	52.93	27.83
	Speed	1.0	1.0	1.0
	Tuneability	η: 3.98e-03	η: 2.51e-05 μ: 0.99	η: 1.58e-04 β_1: 0.9 β_2: 0.999 ε: 1e-08
P3 **F-MNIST CNN**	Performance	92.27 %	92.14 %	92.34 %
	Speed	40.6	59.1	40.1
	Tuneability	η: 1.58e-01	η: 2.51e-03 μ: 0.99	η: 2.51e-04 β_1: 0.9 β_2: 0.999 ε: 1e-08
P4 **CIFAR-10 CNN**	Performance	83.71 %	84.41 %	84.75 %
	Speed	42.5	40.7	36.0
	Tuneability	η: 6.31e-02	η: 3.98e-04 μ: 0.99	η: 3.98e-04 β_1: 0.9 β_2: 0.999 ε: 1e-08

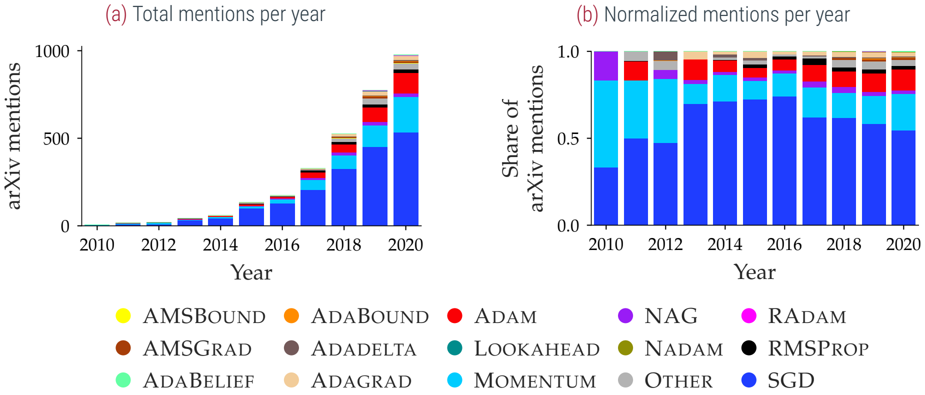
Test Problem		SGD	Momentum	Adam
P5 **F-MNIST VAE**	Performance	23.80	59.23	23.07
	Speed	1.0	1.0	1.0
	Tuneability	η: 3.98e-03	η: 1.00e-05 μ: 0.99	η: 1.58e-04 β_1: 0.9 β_2: 0.999 ε: 1e-08
P6 **CIFAR-100 All CNN C**	Performance	57.06 %	60.33 %	56.15 %
	Speed	128.7	72.8	152.6
	Tuneability	η: 1.58e-01	η: 3.98e-03 μ: 0.99	η: 1.00e-03 β_1: 0.9 β_2: 0.999 ε: 1e-08
P7 **SVHN Wide ResNet**	Performance	95.37 %	95.53 %	95.25 %
	Speed	28.3	10.8	12.1
	Tuneability	η: 2.51e-02	η: 6.31e-04 μ: 0.99	η: 1.58e-04 β_1: 0.9 β_2: 0.999 ε: 1e-08
P8 **Tolstoi Char RNN**	Performance	62.07 %	61.30 %	61.23 %
	Speed	47.7	88.0	62.8
	Tuneability	η: 1.58e+00	η: 3.98e-02 μ: 0.99	η: 2.51e-03 β_1: 0.9 β_2: 0.999 ε: 1e-08


Quadratic Deep
MNIST VAE
F-MNIST 2c2d
CIFAR-10 3c3d
F-MNIST VAE
CIFAR-100 All-CNN-C
SVHN Wide ResNet 16-4
Tolstoi Char RNN
AMSBound
AdaBound
Adam
Mom.
NAdam
RMSProp
AMSGrad
AdaDelta
LA(Mom.)
NAG
RAdam
SGD
AdaBelief
AdaGrad
LA(RAdam)

Benchmark

arXiv Mentions of Optimizers per Year

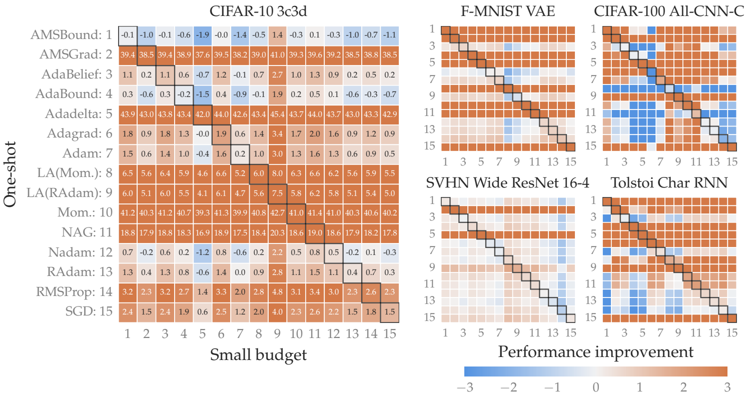
Our selected optimizers cover the most popular choices of the increasing number of methods



Results: Out-of-the-box Performance

Why trying out multiple optimizers can be better than tuning them

Benchmark (Schmidt et al. 2021)



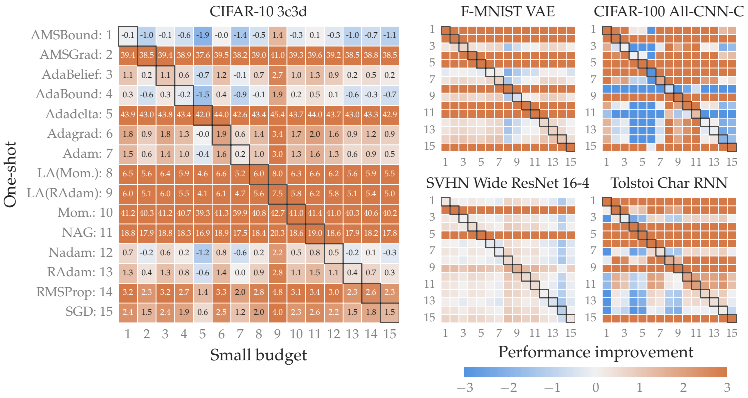
Results: Out-of-the-box Performance

Why trying out multiple optimizers can be better than tuning them

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Benchmark (Schmidt et al. 2021)

- **Orange rows**
  bad default hyperparameters
  SGD, NAG, MOMENTUM, AMSGRAD, ADADELTA

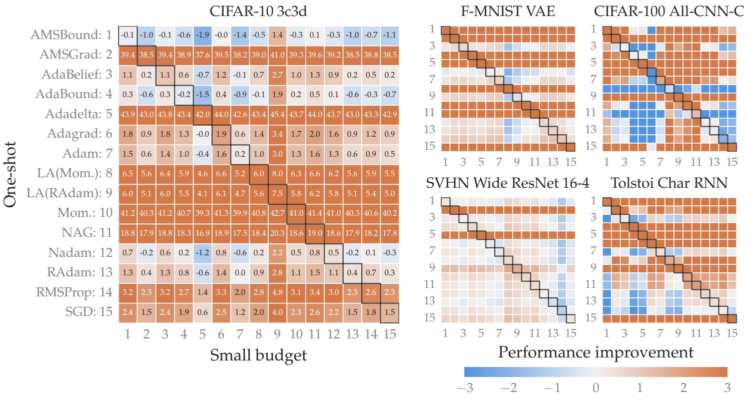


Results: Out-of-the-box Performance

Why trying out multiple optimizers can be better than tuning them

Benchmark (Schmidt et al. 2021)

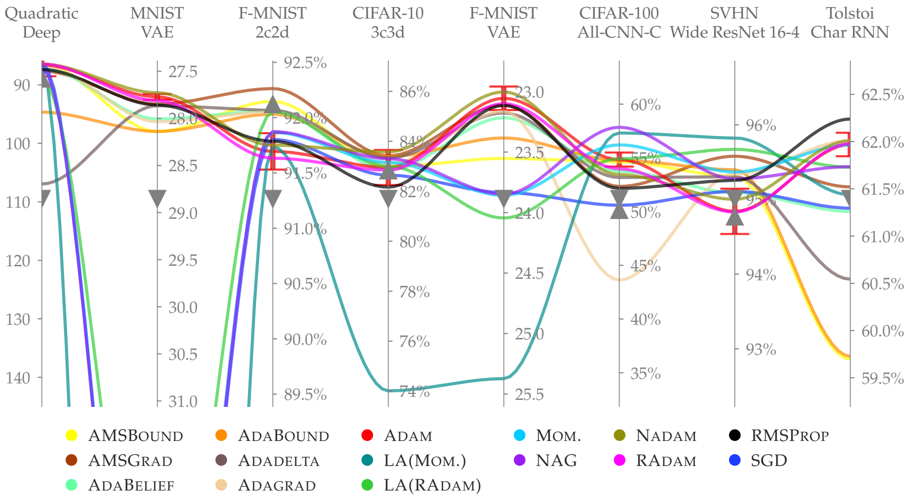
- **Orange rows**
  bad default hyperparameters
  SGD, NAG, MOMENTUM, AMSGRAD, ADADELTA
- **White & blue rows**
  good default hyperparameters
  ADAM, NADAM, RADAM, AMSBOUND, ADABOUND



Results: Which Optimizer Should You Pick?

ADAM is still a reasonable choice | *large budget without a learning rate schedule*

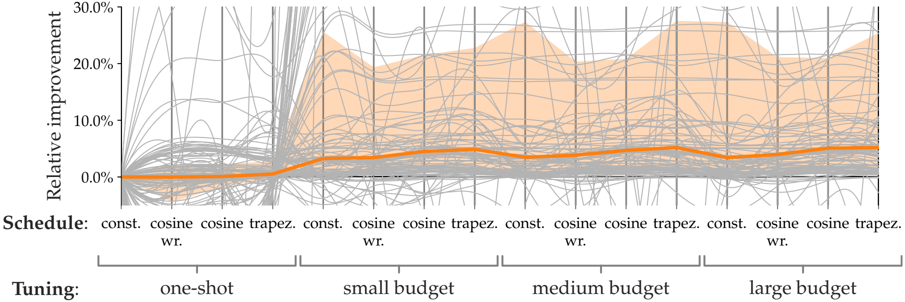
Benchmark (Schmidt et al. 2021)



Results: Effect of Tuning and Schedules

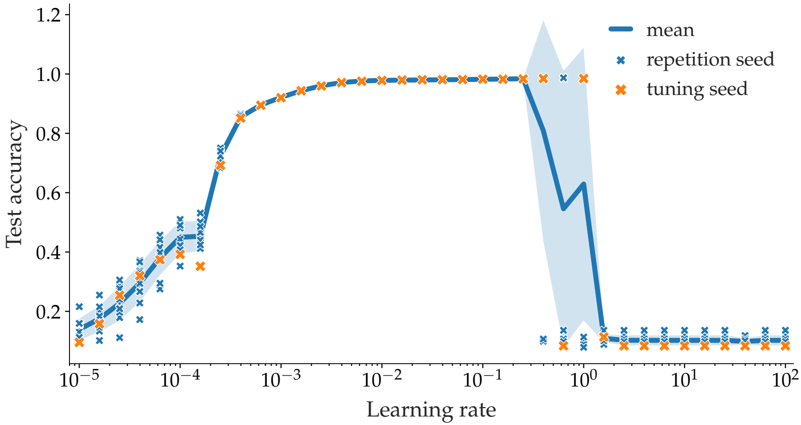
Increasing the budget improves the performance on average with diminishing returns

Benchmark (Schmidt et al. 2021)



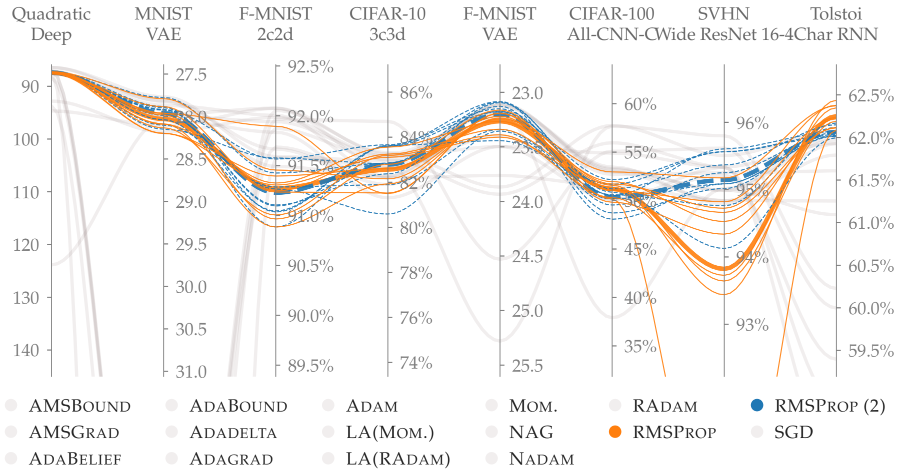
Results: Robustness to Random Seeds

Single seed tuning can result in hyperparameters in an unstable “danger zone” (<0.5% of cases)



Results: Re-tuning Experiment

Individual results can change after re-tuning but overall trends remain

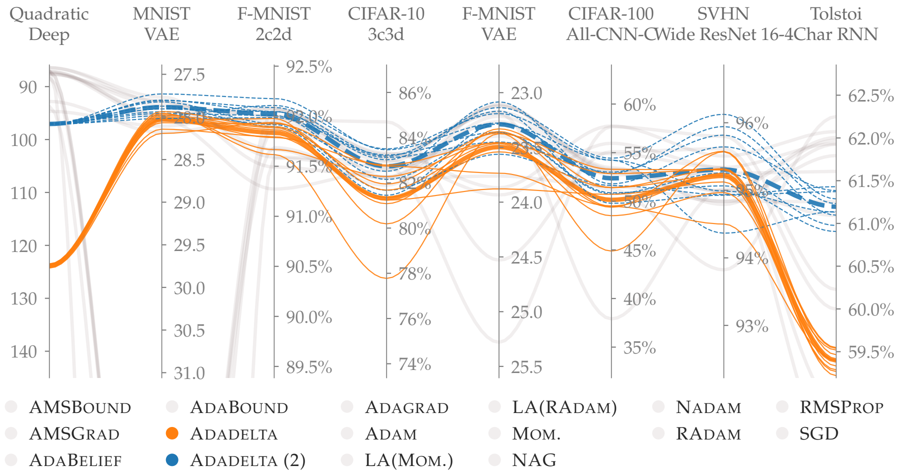


Results: Re-tuning Experiment

Individual results can change after re-tuning but overall trends remain

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Benchmark (Schmidt et al. 2021)



Cockpit

List of Quantities in Cockpit

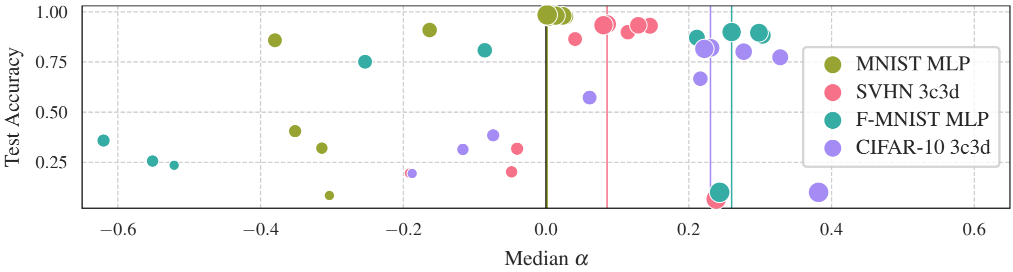
A wide selection to choose from and easy to extend

Name	Description	Configuration
`Loss`	Mini-batch training loss at current iteration, $L_{\mathbb{B}}$	*economy*
`Distance`	L^2 distance from initialization $\|\|\boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^{(0)}\|\|_2$	*economy*
`UpdateSize`	Update size of the current iteration $\|\|\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}\|\|_2$	*economy*
`GradNorm`	Mini-batch gradient norm $\|\|\boldsymbol{g}_{\mathbb{B}}(\boldsymbol{\theta})\|\|_2$	*economy*
`Alpha`	Normalized step on a noise-informed quadratic interpolation between $\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)}$	*economy*
`GradHist1d`	Histogram of individual gradient elements, $\{\boldsymbol{g}_{\mathbb{B}}^{(i)}(\theta_j)\}_{i\in\mathbb{B}}^{j=1,\dots,D}$	*economy*
`NormTest`	Normalized fluctuations of the residual norms $\|\|\boldsymbol{g}_{\mathbb{B}} - \boldsymbol{g}_{\mathbb{B}}^{(i)}\|\|$, proposed in Byrd et al. 2012	*economy*
`InnerTest`	Normalized fluctuations of $\boldsymbol{g}_{\mathbb{B}}^{(i)}$'s parallel components along $\boldsymbol{g}_{\mathbb{B}}$, proposed in Bollapragada et al. 2017	*economy*
`OrthoTest`	Normalized fluctuations of $\boldsymbol{g}_{\mathbb{B}}^{(i)}$'s orthogonal components along $\boldsymbol{g}_{\mathbb{B}}$, proposed in Bollapragada et al. 2017	*economy*
`HessTrace`	Exact or approximate Hessian trace, $\mathrm{Tr}(\boldsymbol{H}_{\mathcal{B}}(\boldsymbol{\theta}))$, inspired by Yao et al. 2020	*business*
`TICDiag`	Relation between (diagonal) curvature and gradient noise, inspired by Thomas et al. 2020	*business*
`HessMaxEV`	Maximum Hessian eigenvalue, $\lambda_{\max}(\boldsymbol{H}_{\mathcal{B}}(\boldsymbol{\theta}))$, inspired by Yao et al. 2020	*full*
`GradHist2d`	Histogram of weights and individual gradient elements, $\{(\theta_j, \boldsymbol{g}_{\mathbb{B}}^{(i)}(\theta_j))\}_{i\in\mathbb{B}}^{j=1,\dots,D}$	*full*
`Parameters`	Parameter values $\boldsymbol{\theta}^{(t)}$ at the current iteration	-
`Time`	Time of the current iteration	-
`CABS`	Adaptive batch size for SGD, optimizes expected objective gain per cost, adapted from Balles et al. 2017	-
`EarlyStopping`	Evidence-based early stopping criterion for SGD, proposed in Mahsereci et al. 2017	-
`TICTrace`	Relation between curvature and gradient noise trace, inspired by Thomas et al. 2020	-
`MeanGSNR`	Average gradient signal-to-noise-ratio (GSNR), inspired by Liu et al. 2020	-

Cockpit as a Tool for Inspiring Research

Neural network training requires systematically overstepping the local minima

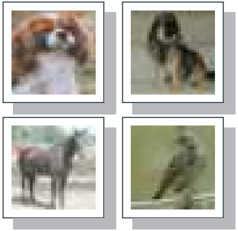
Cockpit (Schneider, Dangel, et al. 2021)



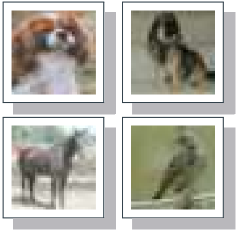
CockpIT as a Tool for Finding Data Bugs

Normalized and raw data can *look* the same but *behave* differently

CockpIT (Schneider, Dangel, et al. 2021)



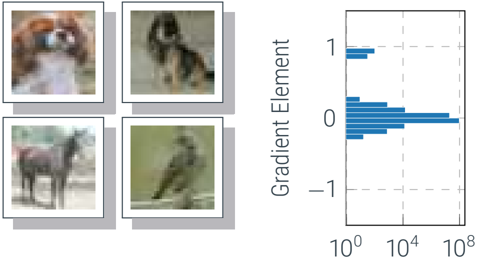
(a) Normalized Data



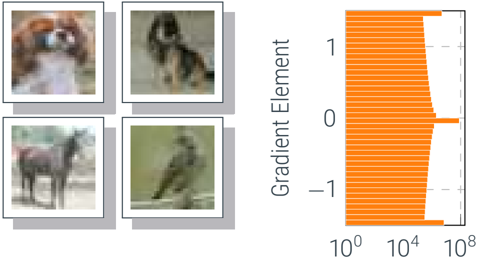
(b) Raw Data

Cᴏᴄᴋᴘɪᴛ as a Tool for Finding Data Bugs

Normalized and raw data can *look* the same but *behave* differently



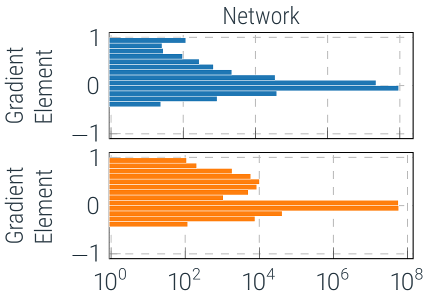
(a) Normalized Data



(b) Raw Data

Cockpit as a Tool for Finding Model Bugs

Looking at the gradients layerwise can reveal model inefficiencies

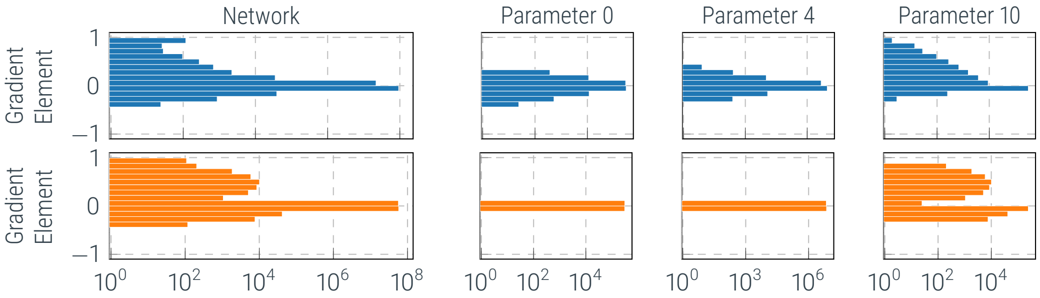


(a) Network Histogram

COCKPIT as a Tool for Finding Model Bugs

Looking at the gradients layerwise can reveal model inefficiencies

COCKPIT (Schneider, Dangel, et al. 2021)



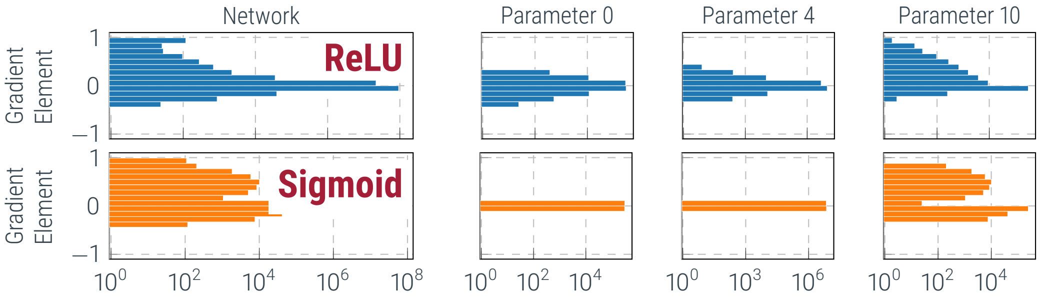
(a) Network Histogram

(b) Layer-wise Histograms

Cockpit as a Tool for Finding Model Bugs

Looking at the gradients layerwise can reveal model inefficiencies

Cockpit (Schneider, Dangel, et al. 2021)



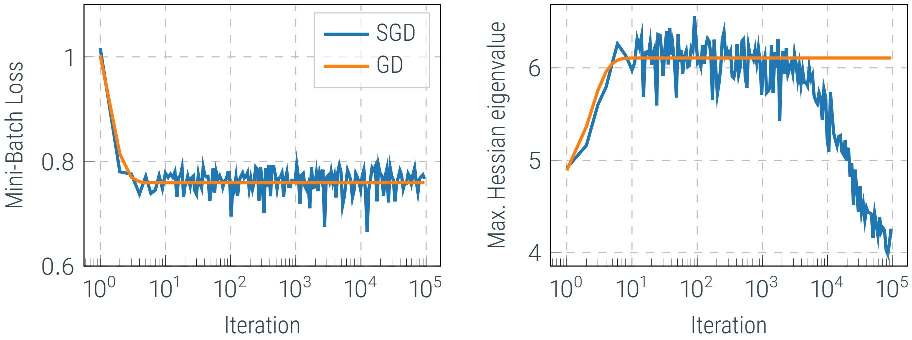
(a) Network Histogram

(b) Layer-wise Histograms

Observing Implicit Regularization with COCKPIT

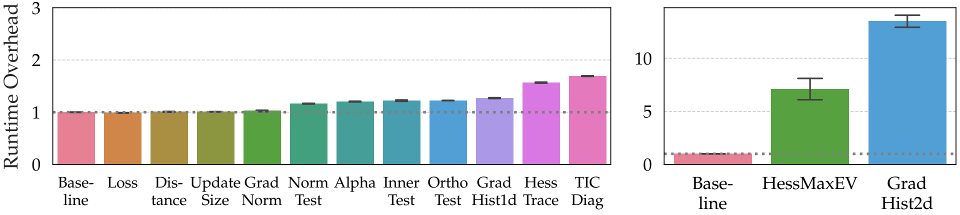
An effect that cannot be seen by monitoring the only the loss

COCKPIT (Schneider, Dangel, et al. 2021)



Efficient Implementation for Practical Overhead

COCKPIT's quantities are affordable in practice: CIFAR-10 and 3C3D

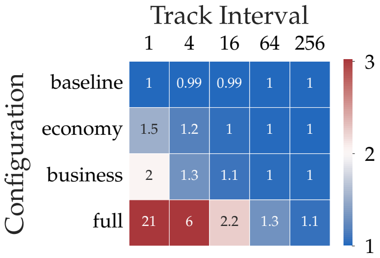


Efficient Implementation for Practical Overhead

Combining quantities and reducing the tracking interval reduces the overhead

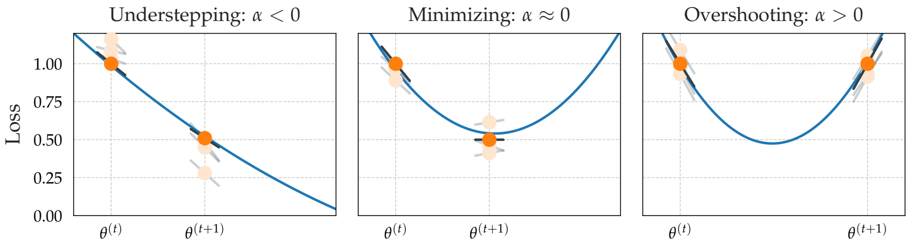
EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Cockpit (Schneider, Dangel, et al. 2021)



Illustrating the Alpha Quantity

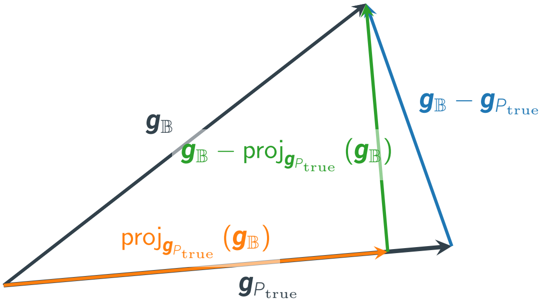
Are we understepping or overshooting?



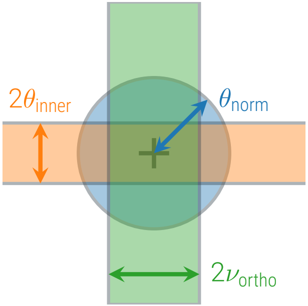
Illustrating the Gradient Tests

Defining geometric constraints between the mini-batch and the true gradient

(a) Gradient test vectors



(b) Visualization in COCKPIT



Bibliography I

Standing on the shoulders of giants

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

- Balles, Lukas, Javier Romero, and Philipp Hennig (2017). “Coupling Adaptive Batch Sizes with Learning Rates”. In: *Uncertainty in Artificial Intelligence - Proceedings of the 33rd Conference, UAI*.
- Beale, Evelyn M. L. (1958). *On an iterative method for finding a local minimum of a function of more than one variable*. Statistical Techniques Research Group, Section of Mathematical Statistics, Department of Mathematics, Princeton University.
- Bollapragada, Raghu, Richard Byrd, and Jorge Nocedal (2017). “Adaptive Sampling Strategies for Stochastic Optimization”. In: *SIAM Journal on Optimization*.
- Branin, Franklin H. (1972). “Widely convergent method for finding multiple solutions of simultaneous nonlinear equations”. In: *IBM Journal of Research and Development*.
- Byrd, Richard H. et al. (2012). “Sample Size Selection in Optimization Methods for Machine Learning”. In: *Math. Program.*
- Chaudhari, Pratik et al. (2017). “Entropy-SGD: Biasing gradient descent into wide valleys”. In: *5th International Conference on Learning Representations, ICLR*.
- Deng, Jia et al. (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. In: *The IEEE Conference on Computer Vision and Pattern Recognition, CVPR*.

Bibliography II

Standing on the shoulders of giants

- Krizhevsky, Alex and Geoffrey Hinton (2009). *Learning multiple layers of features from tiny images*. Technical Report.
- LeCun, Yann et al. (1998). “Gradient-Based Learning Applied to Document Recognition”. In: *Proceedings of the IEEE*.
- Liu, Jinlong et al. (2020). “Understanding Why Neural Networks Generalize Well Through GSNR of Parameters”. In: *8th International Conference on Learning Representations, ICLR*.
- Mahsereci, Maren et al. (2017). *Early Stopping without a Validation Set*. arXiv: `1703.09580`.
- Netzer, Yuval et al. (2011). “Reading digits in natural images with unsupervised feature learning”. In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.
- Rosenbrock, Howard H. (1960). “An automatic method for finding the greatest or least value of a function”. In: *The Computer Journal*.
- Schmidt, Robin M., Frank Schneider, and Philipp Hennig (2021). “Descending through a Crowded Valley - Benchmarking Deep Learning Optimizers”. In: *38th International Conference on Machine Learning, ICML*.
- Schneider, Frank, Lukas Balles, and Philipp Hennig (2019). “DeepOBS: A Deep Learning Optimizer Benchmark Suite”. In: *7th International Conference on Learning Representations, ICLR*.

Bibliography III

Standing on the shoulders of giants

- Schneider, Frank, Felix Dangel, and Philipp Hennig (2021). "Cockpit: A Practical Debugging Tool for the Training of Deep Neural Networks". In: *Advances in Neural Information Processing Systems 34, NeurIPS*.
- Simonyan, Karen and Andrew Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations, ICLR*.
- Springenberg, Jost T. et al. (2015). "Striving for simplicity: The all convolutional net". In: *3rd International Conference on Learning Representations, ICLR (workshop track)*.
- Szegedy, Christian et al. (2016). "Rethinking the Inception Architecture for Computer Vision". In: *The IEEE Conference on Computer Vision and Pattern Recognition, CVPR*.
- Thomas, Valentin et al. (2020). "On the interplay between noise and curvature and its effect on optimization and generalization". In: *23rd International Conference on Artificial Intelligence and Statistics, AISTATS*.
- Xiao, Han, Kashif Rasul, and Roland Vollgraf (2017). *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv: `1708.07747`.
- Yao, Zhewei et al. (2020). *ADAHESSIAN: An Adaptive Second Order Optimizer for Machine Learning*. arXiv: `2006.00719`.

Bibliography IV

Standing on the shoulders of giants

- Zagoruyko, Sergey and Nikos Komodakis (2016). “Wide Residual Networks”. In: *Procedings of the British Machine Vision Conference*.