Neural Optimizer Search with Reinforcement Learning

Irwan Bello[1] Barret Zoph[1] Vijay Vasudevan[1] Quoc V. Le[1]

[1]Google Brain

ICLR, 2017/ Presenter: Anant Kharkar

Outline

Outline

Motivation

Classical optimizers:

- SGD
- SGD w/Momentum
- Adam
- RMSProp

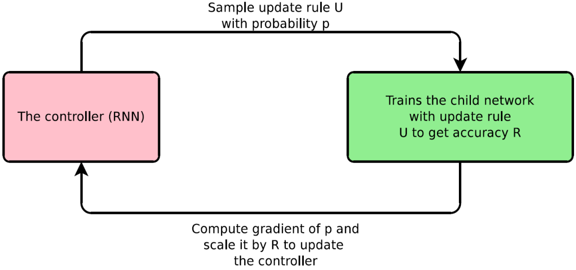
Combination of stochastic methods and heuristic approximations

Want to automate process of generating update rules
Produce equation, not just numerical updates

Outline

Approach

- RNN controller produces update rule string
- Controller updated based on performance of optimizer
- RL approach to training



- How to generate update rules? First define space of update rules

Previous Work

- LSTM for numerical updates (Andrychowicz et al., 2016)
  - Equations are more transferrable
- Genetic programming for update equations (Orchard & Wang, 2016)
  - Slow and needs heuristics
- Neural Architecture Search (Zoph & Le, 2017) - seen earlier
  - RNN produces network architecture

Outline

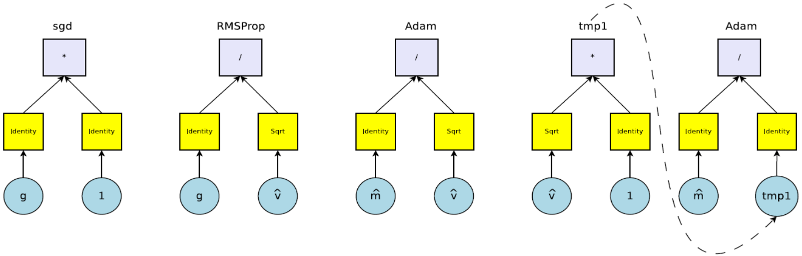
Domain-Specific Language

- Each optimizer has computational graph - binary expression tree

Components:

- 2 operands
- Unary function for each operand
- Binary function to combine

$$\Delta w = \lambda * b(u_1(op_1), u_2(op_2))$$

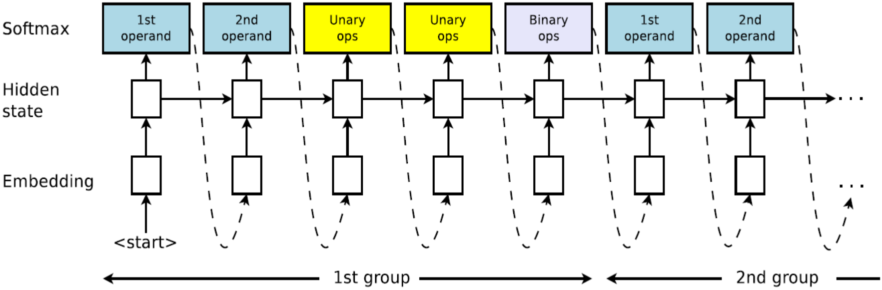


Outline

1. Introduction
   - Motivation
   - Approach
2. Methods
   - Domain-Specific Language
   - Controller RNN
3. Experiments
   - Optimizer Discovery
   - Transfer Experiment
4. Summary

Controller RNN

Trained with Adam



Objective function: $J(\theta) = \mathbb{E}_{\Delta \sim p_\theta(.)}[R(\Delta)]$
Optimize reward (accuracy of target model)

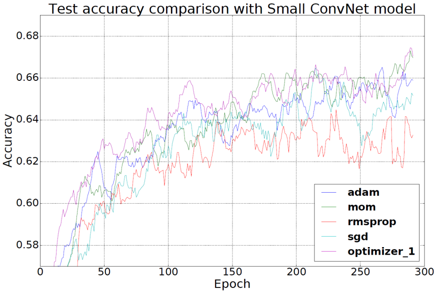
- Operands
  - Gradients: $g, g^2, g^3, sign(g)$
  - Moving averages: $\hat{m}, \hat{v}, \hat{y}, sign(\hat{m})$
  - Weights: $10^{-4}w, 10^{-3}w, 10^{-2}w, 10^{-1}w$
  - ADAM, RMSProp, 1, small noise
- Unary Functions
  - $x, -x, e^x, log|x|, clip, drop, sign$
- Binary Functions
  - $x + y, x - y, x * y, \frac{x}{y+\varepsilon}$

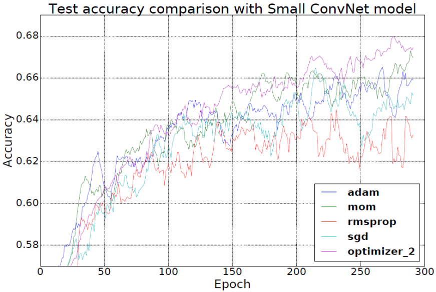
- Optimizers tested on 3x3 ConvNet (32 filters) for 5 epochs
- Favors early progress

Outline

1. Introduction
   - Motivation
   - Approach
2. Methods
   - Domain-Specific Language
   - Controller RNN
3. Experiments
   - Optimizer Discovery
   - Transfer Experiment
4. Summary

Optimizer Discovery





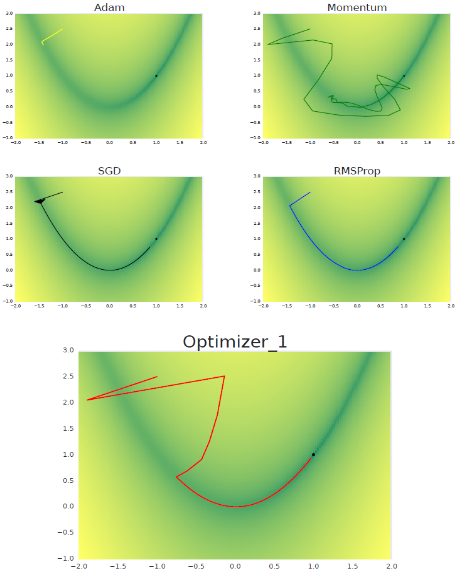
Recurring element:

$$e^{sign(g)*sign(m)} * g$$

- If $sign(g)$ agrees with running average, scale e - g keeps decreasing
- Else scale $\frac{1}{e}$ - gradient direction changed

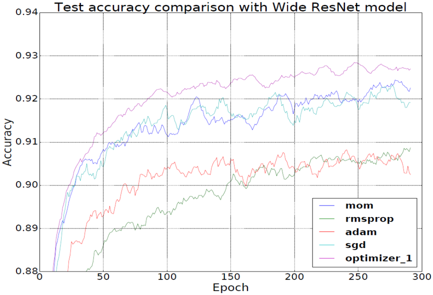
Outline

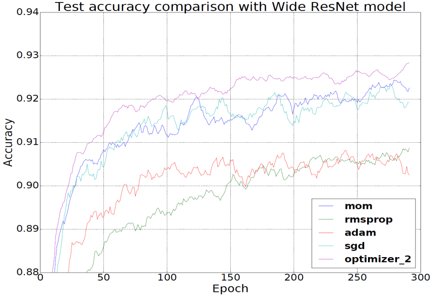
Rosenbrock Function



CIFAR-10

Wide ResNet (Zagoruyko & Komodakis, 2016) - 300 epochs





CIFAR-10

Optimizer	Final Val	Final Test	Best Val	Best Test
SGD	92.0	91.8	92.9	91.9
Momentum	92.7	92.1	93.1	92.3
ADAM	90.4	90.1	91.8	90.7
RMSProp	90.7	90.3	91.4	90.3
$[e^{\mathrm{sign}(g)*\mathrm{sign}(m)} + \mathrm{clip}(g, 10^{-4})] * g$	92.5	92.4	93.8	93.1
$\mathrm{clip}(\hat{m}, 10^{-4}) * e^{\hat{v}}$	93.5	92.5	93.8	92.7
$\hat{m} * e^{\hat{v}}$	93.1	92.4	93.8	92.6
$g * e^{\mathrm{sign}(g)*\mathrm{sign}(m)}$	93.1	92.8	93.8	92.8
$\mathrm{drop}(g, 0.3) * e^{\mathrm{sign}(g)*\mathrm{sign}(m)}$	92.7	92.2	93.6	92.7
$\hat{m} * e^{g^2}$	93.1	92.5	93.6	92.4
$\mathrm{drop}(\hat{m}, 0.1)/(e^{g^2} + \epsilon)$	92.6	92.4	93.5	93.0
$\mathrm{drop}(g, 0.1) * e^{\mathrm{sign}(g)*\mathrm{sign}(m)}$	92.8	92.4	93.5	92.2
$\mathrm{clip}(\mathrm{RMSProp}, 10^{-5}) + \mathrm{drop}(\hat{m}, 0.3)$	90.8	90.8	91.4	90.9
$\mathrm{ADAM} * e^{\mathrm{sign}(g)*\mathrm{sign}(m)}$	92.6	92.0	93.4	92.0
$\mathrm{ADAM} * e^{\hat{m}}$	92.9	92.8	93.3	92.7
$g + \mathrm{drop}(\hat{m}, 0.3)$	93.4	92.9	93.7	92.9
$\mathrm{drop}(\hat{m}, 0.1) * e^{g^3}$	92.8	92.7	93.7	92.8
$g - \mathrm{clip}(g^2, 10^{-4})$	93.4	92.8	93.7	92.8
$e^g - e^{\hat{m}}$	93.2	92.5	93.5	93.1
$\mathrm{drop}(\hat{m}, 0.3) * e^{10^{-3} w}$	93.2	93.0	93.5	93.2

Neural Machine Translation

Completely different model & task: WMT 2014 English → German task
GNMT model - 8 LSTM layers

Optimizer	Train perplexity	Test BLEU
Adam	1.49	24.5
$g * e^{\mathrm{sign}(g) * \mathrm{sign}(m)}$	1.39	25.0

Summary

- RNN generates optimizer equations
- Train RNN via RL setup
- Optimizers tested on small ConvNet
- New optimizers on par with state of the art