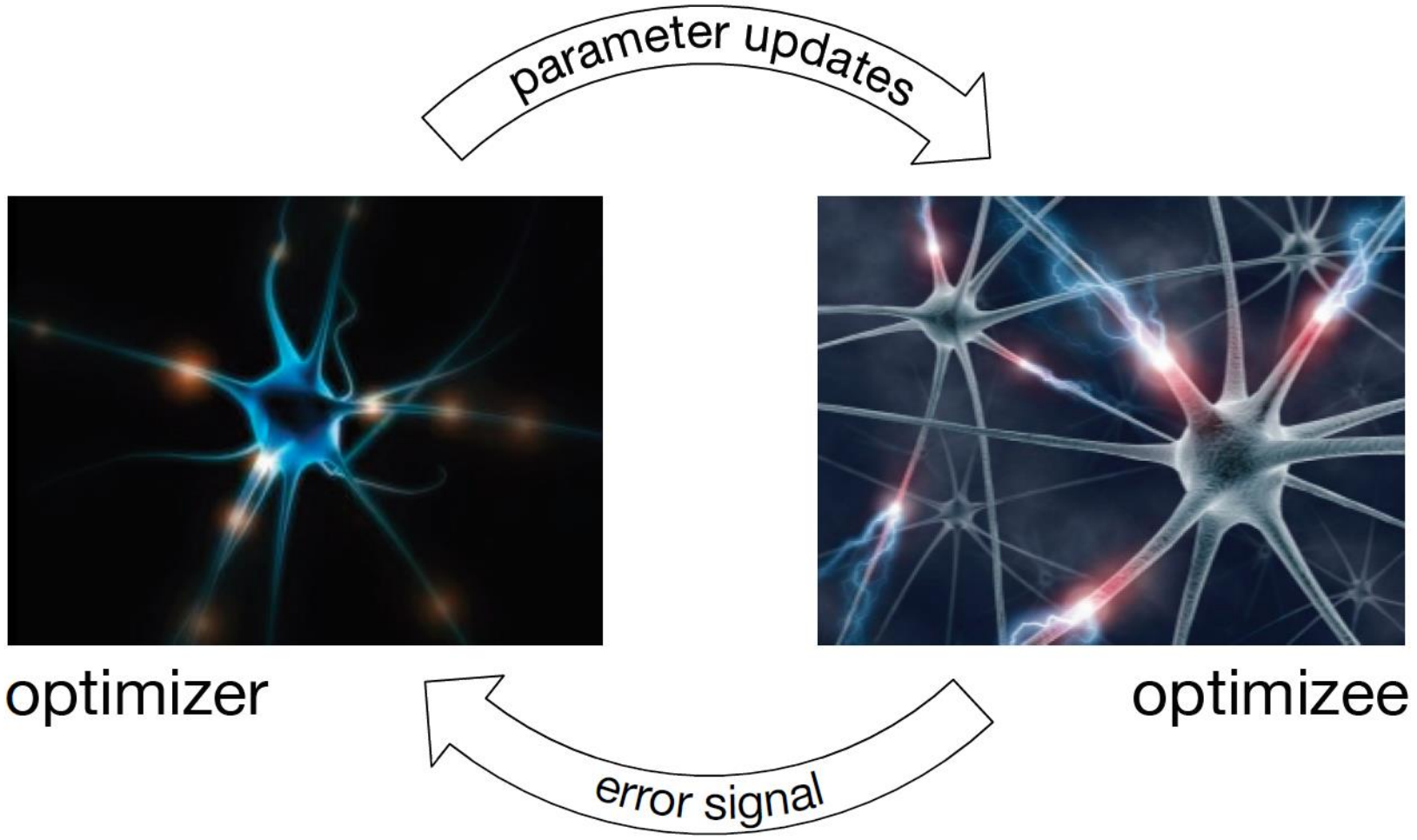
Meta Learning, part 2

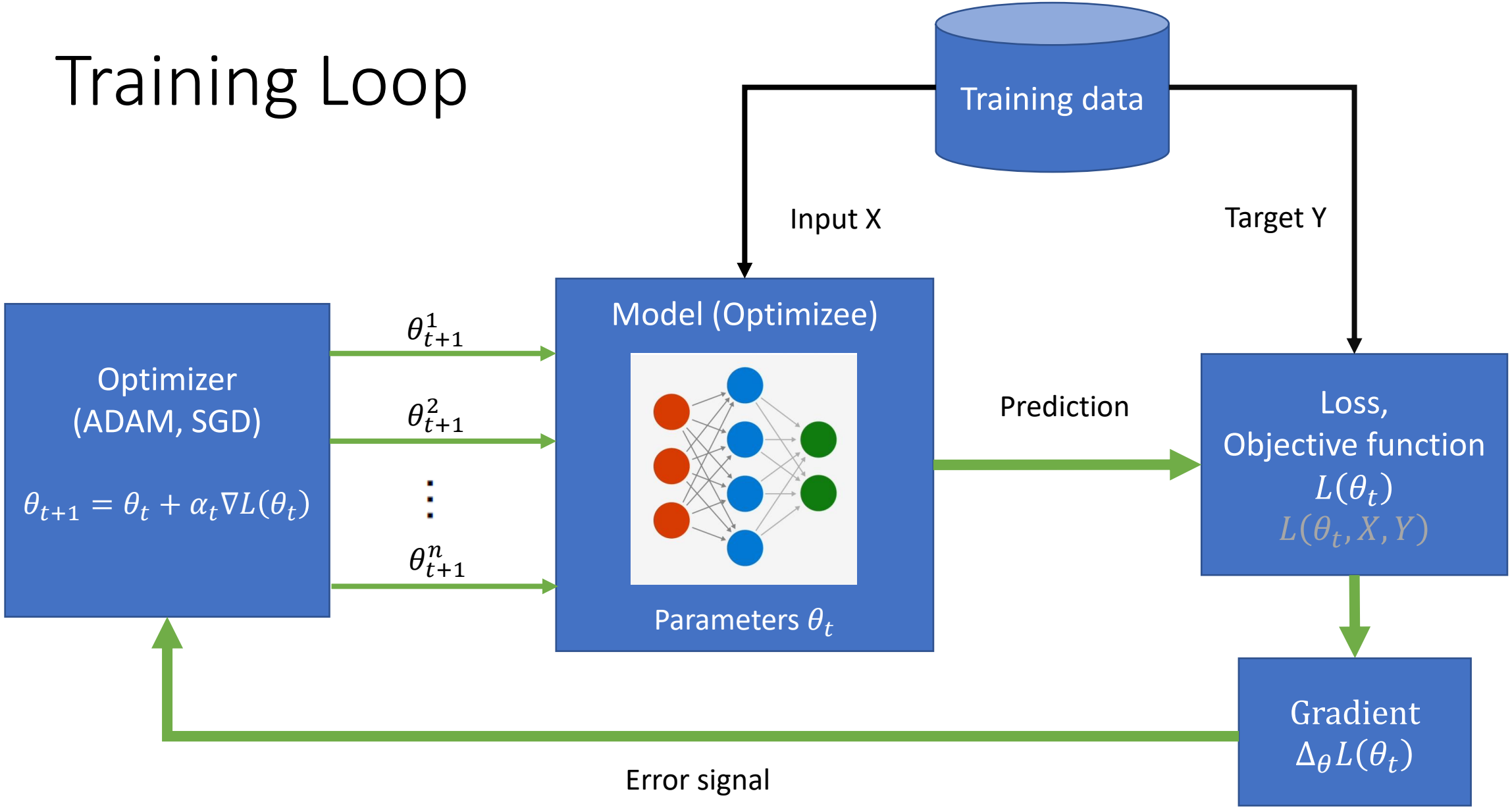
Seminar in Deep Neural Networks - 2021

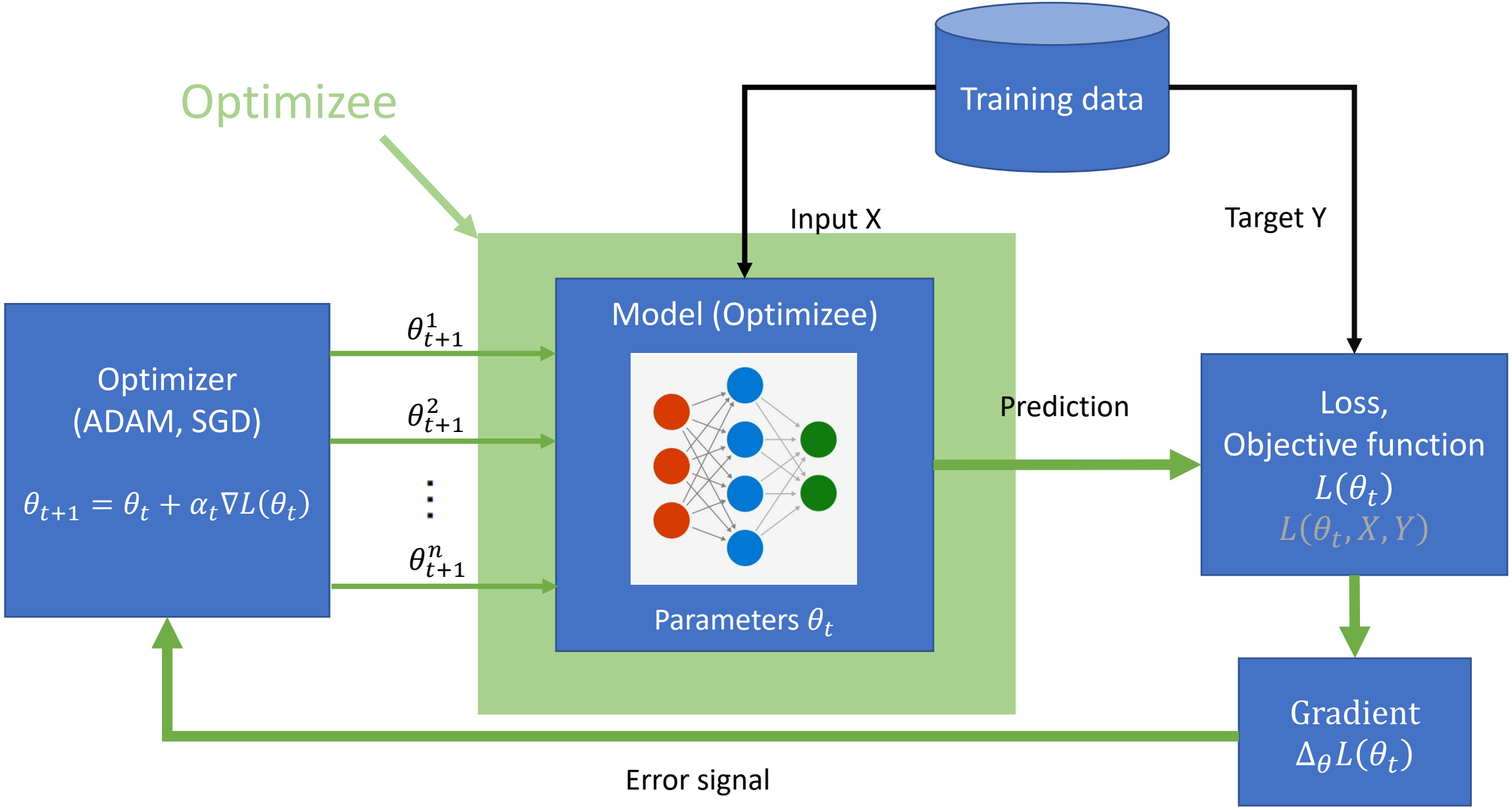
Davide Plozza

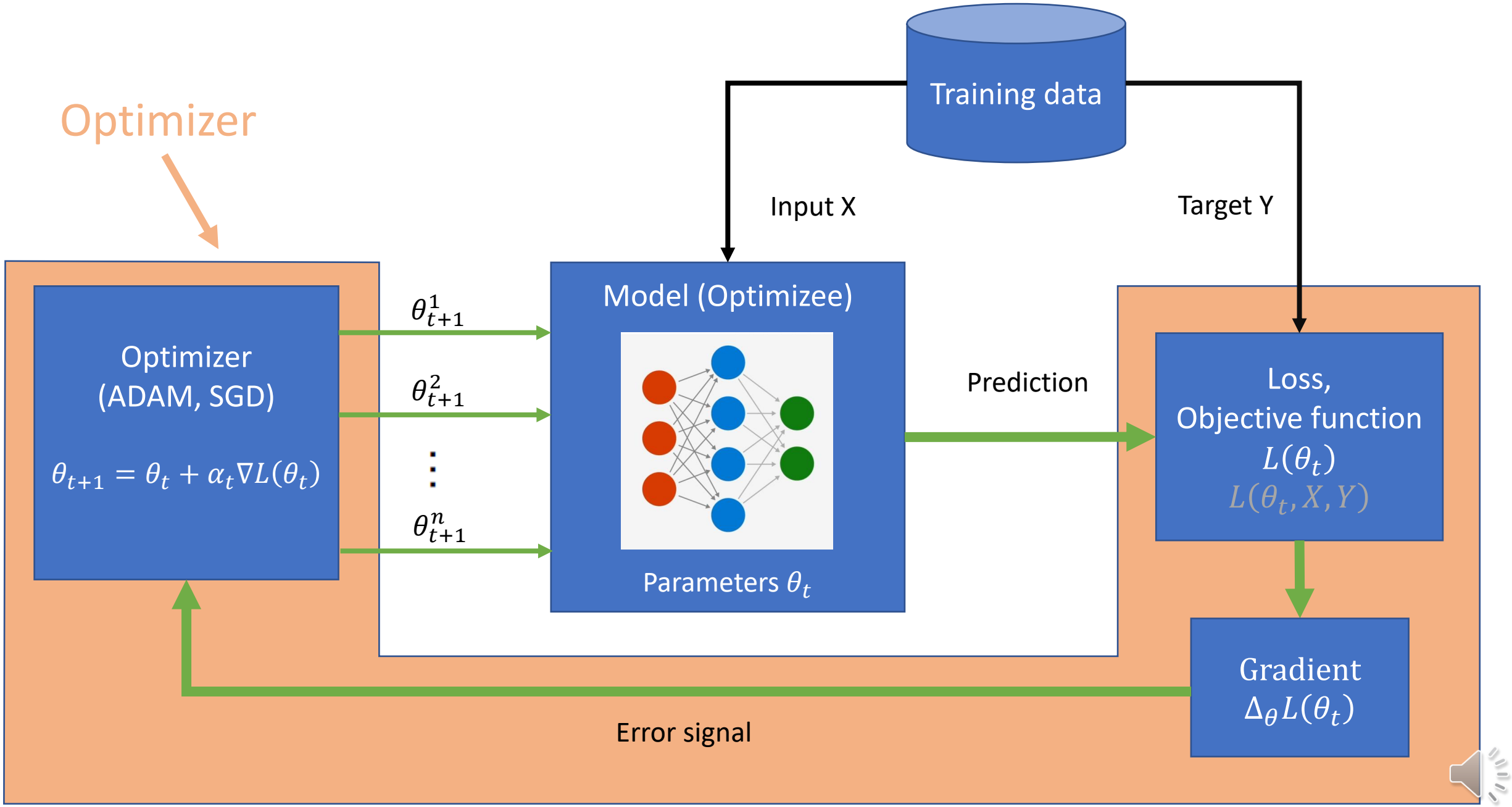
Meta learning = learning to learn



Training Loop




Optimizee
Training data
Input X
Target Y
Model (Optimizee)
Optimizer
(ADAM, SGD)
$\theta_{t+1} = \theta_t + \alpha_t \nabla L(\theta_t)$
θ_{t+1}^1
θ_{t+1}^2
θ_{t+1}^n
Parameters θ_t
Prediction
Loss,
Objective function
$L(\theta_t)$
$L(\theta_t, X, Y)$
Gradient
$\Delta_\theta L(\theta_t)$
Error signal


Optimizer
Training data
Input X
Target Y
Optimizer
(ADAM, SGD)
$\theta_{t+1} = \theta_t + \alpha_t \nabla L(\theta_t)$
θ_{t+1}^1
θ_{t+1}^2
θ_{t+1}^n
Model (Optimizee)
Parameters θ_t
Prediction
Loss,
Objective function
$L(\theta_t)$
$L(\theta_t, X, Y)$
Gradient
$\Delta_\theta L(\theta_t)$
Error signal

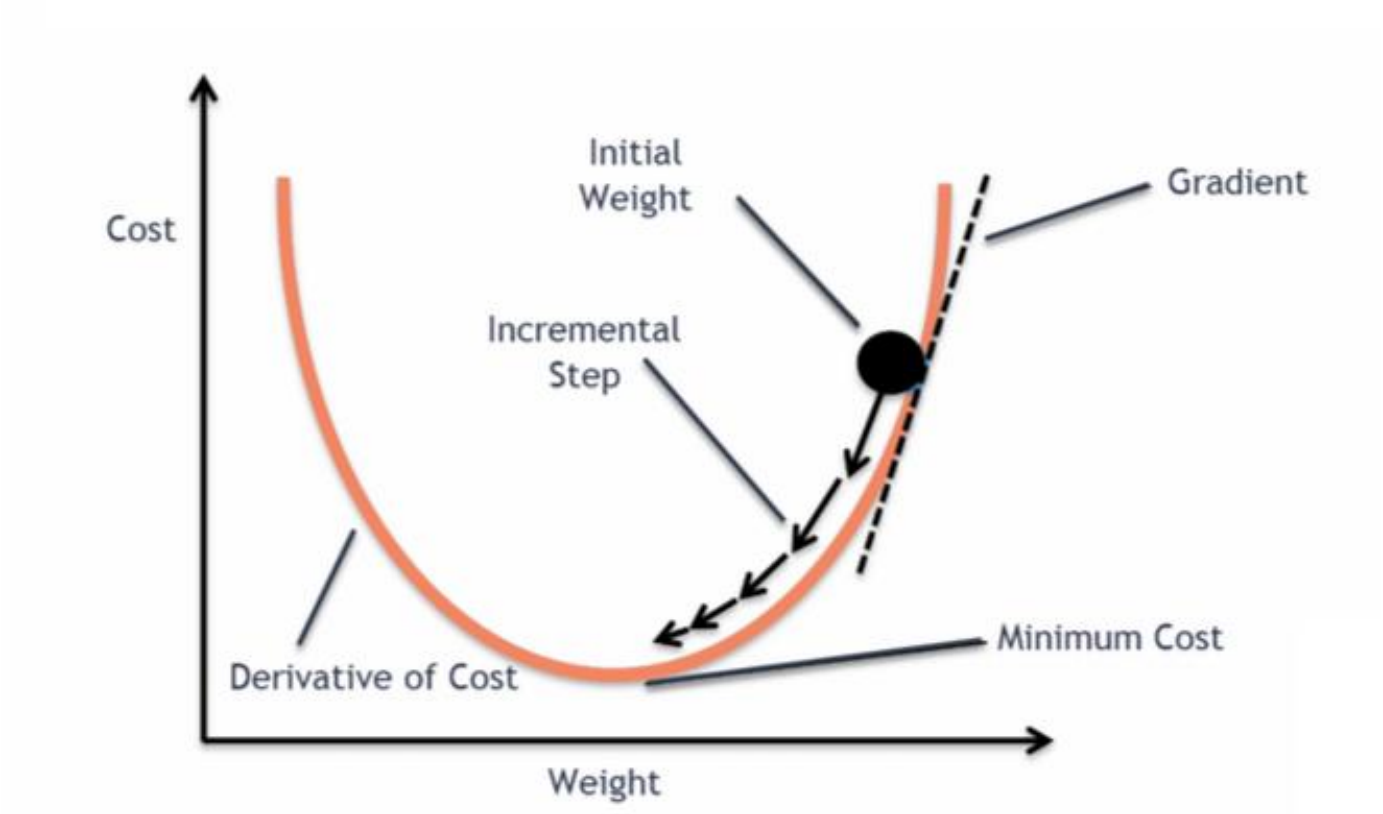
Learning to learn by gradient descent by gradient descent

Marcin Andrychowicz[1], **Misha Denil**[1], **Sergio Gómez Colmenarejo**[1], **Matthew W. Hoffman**[1], **David Pfau**[1], **Tom Schaul**[1], **Brendan Shillingford**[1,2], **Nando de Freitas**[1,2,3]

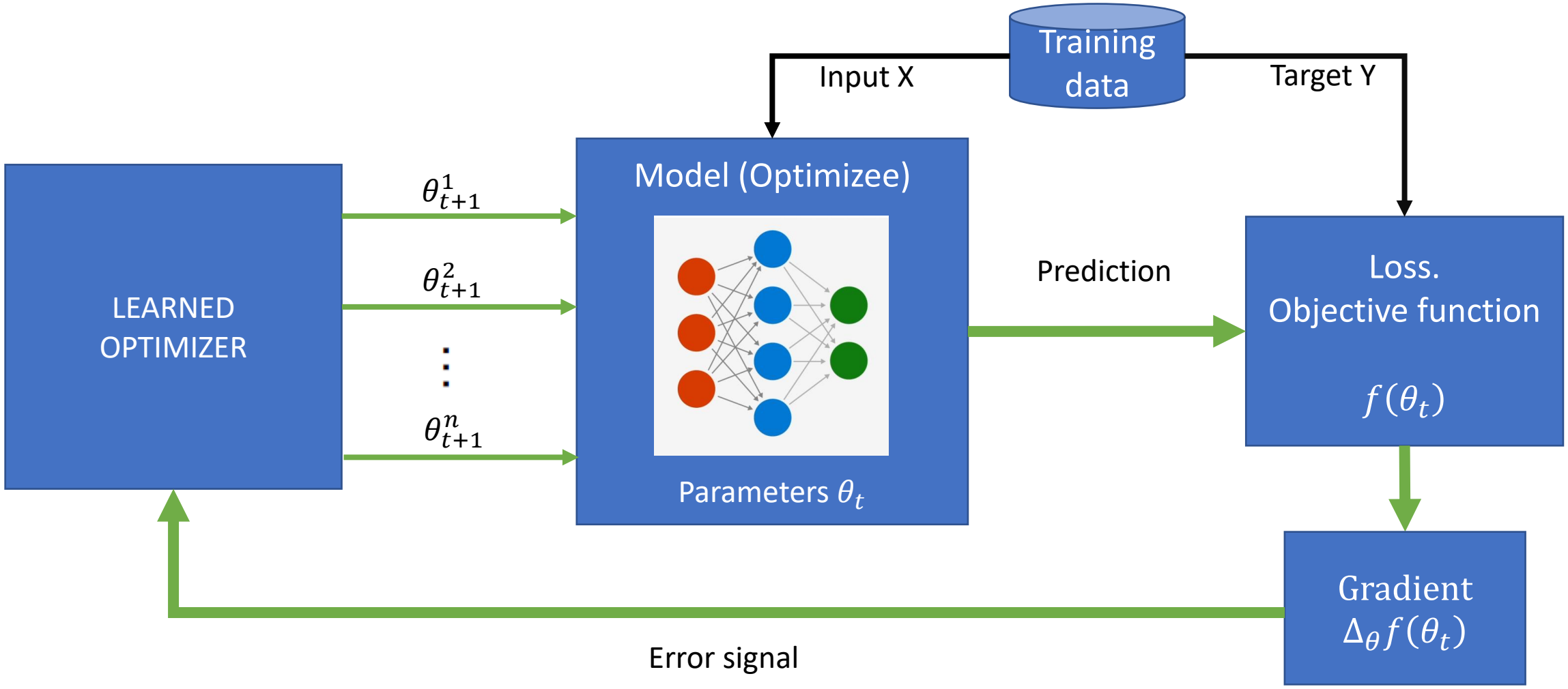
[1]Google DeepMind [2]University of Oxford [3]Canadian Institute for Advanced Research

Revisiting gradient descent

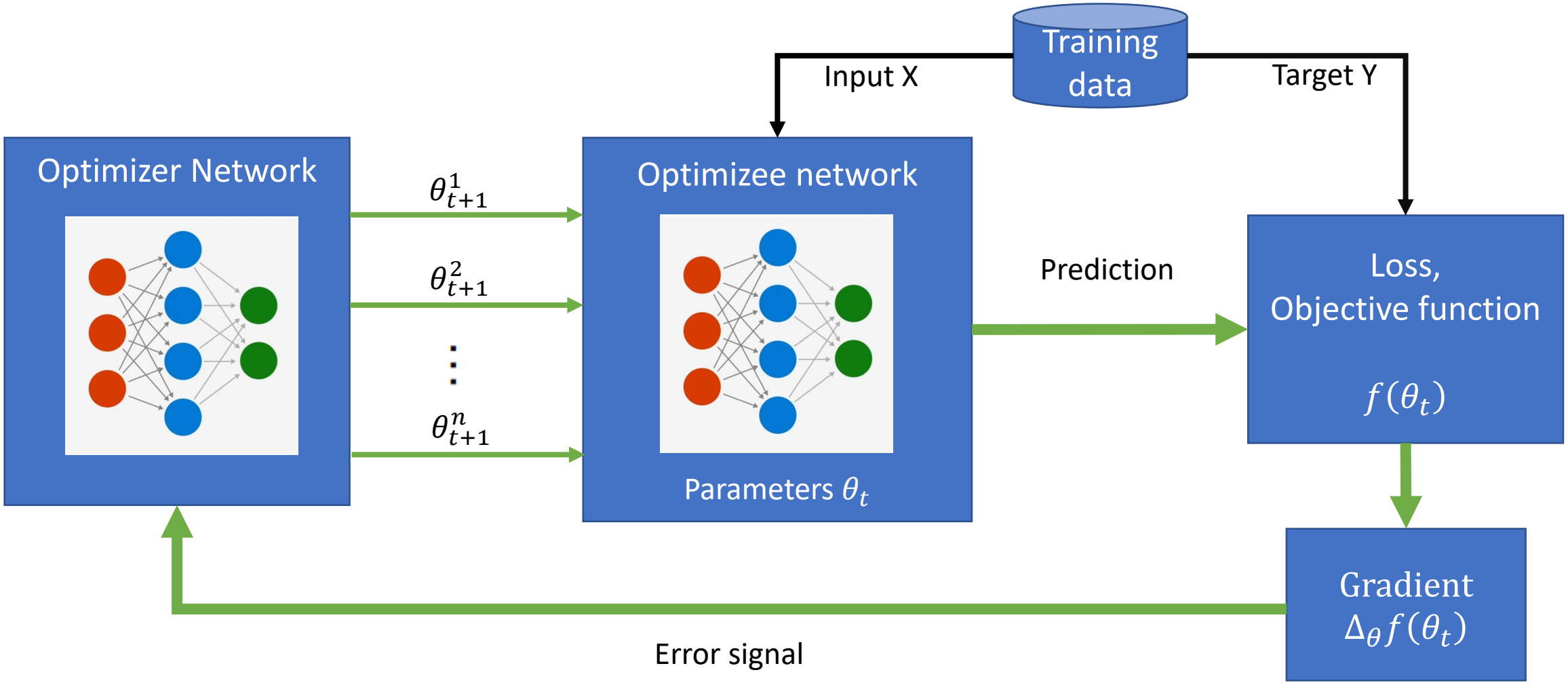
$$\theta_{t+1} = \theta_t - \alpha_t \nabla f(\theta_t)$$



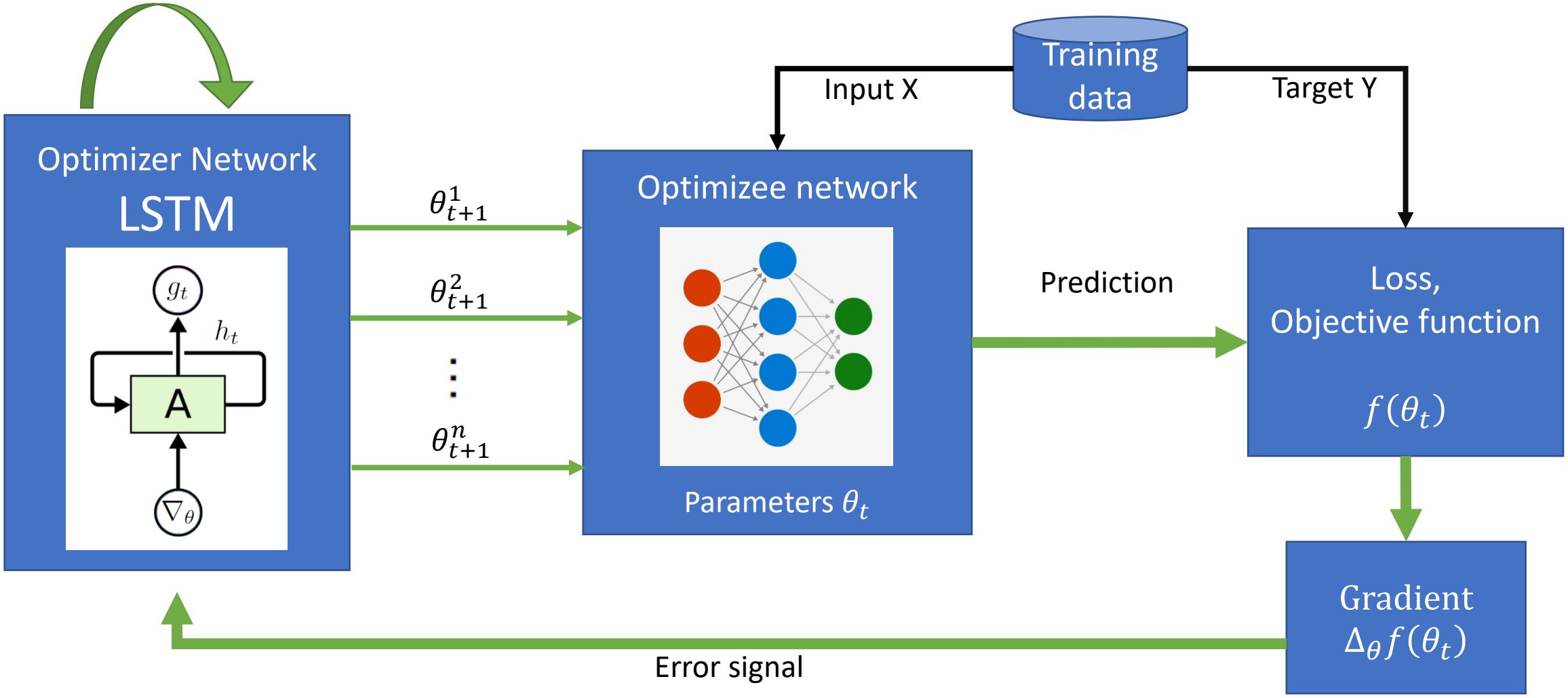
Can we learn gradient descent?



Two network architecture



Recurrent NN (LSTM) as optimizer

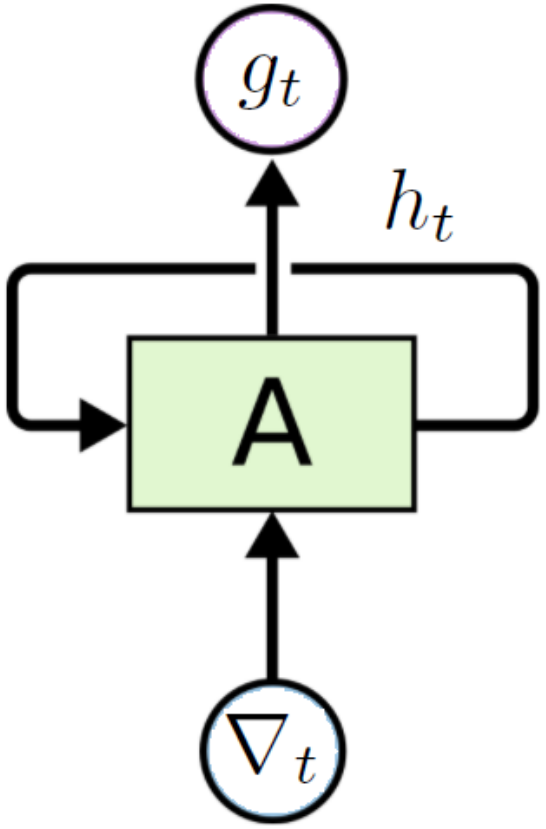


Recurrent NN (LSTM) as optimizer

$$\theta_{t+1} = \theta_t - \alpha_t \nabla f(\theta_t)$$

$$\theta_{t+1} = \theta_t + g_t(\nabla f(\theta_t), \phi)$$

Recurrent NN (LSTM) as optimizer

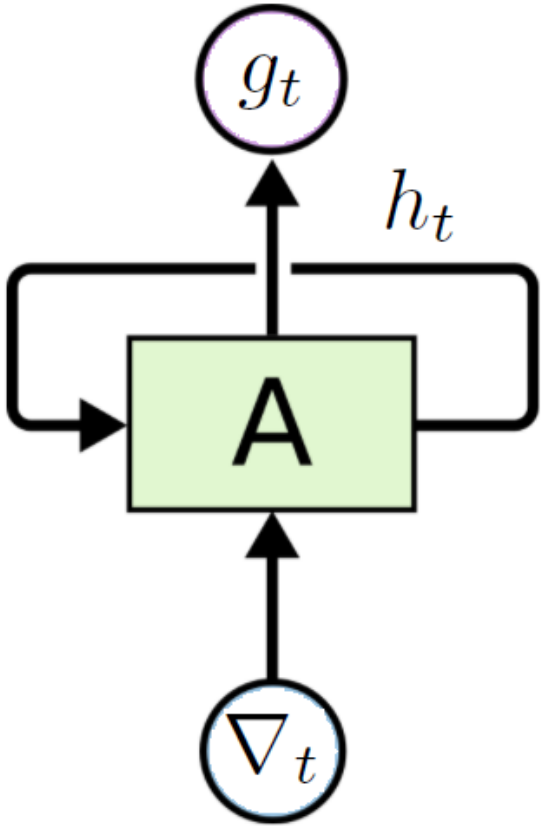


$$\theta_{t+1} = \theta_t + g_t \ ,$$

$$\begin{bmatrix} g_t \\ h_{t+1} \end{bmatrix} = m(\nabla_t, h_t, \phi)$$

$$\nabla_t = \nabla_\theta f(\theta_t)$$

Recurrent NN (LSTM) as optimizer



$$\theta_{t+1} = \theta_t + g_t \ ,$$

$$\begin{bmatrix} g_t \\ h_{t+1} \end{bmatrix} = m(\nabla_t, h_t, \phi)$$

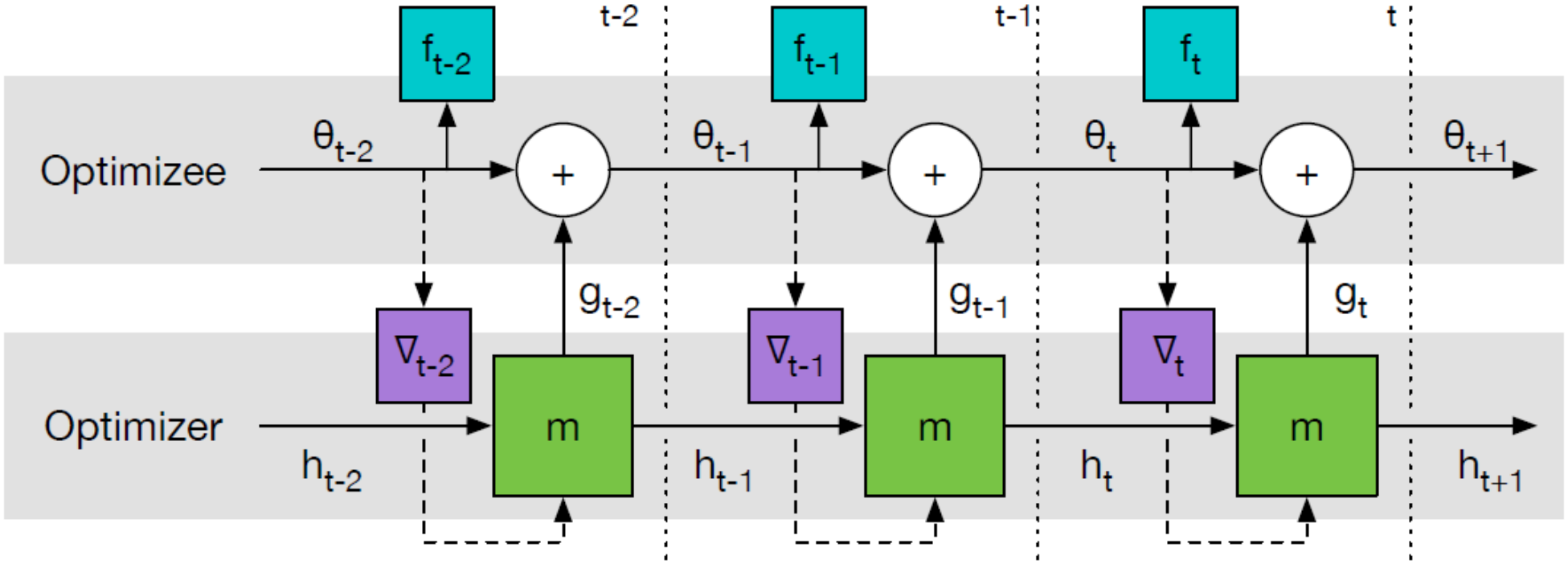
$$\nabla_t = \nabla_\theta f(\theta_t)$$

Outer optimization loss

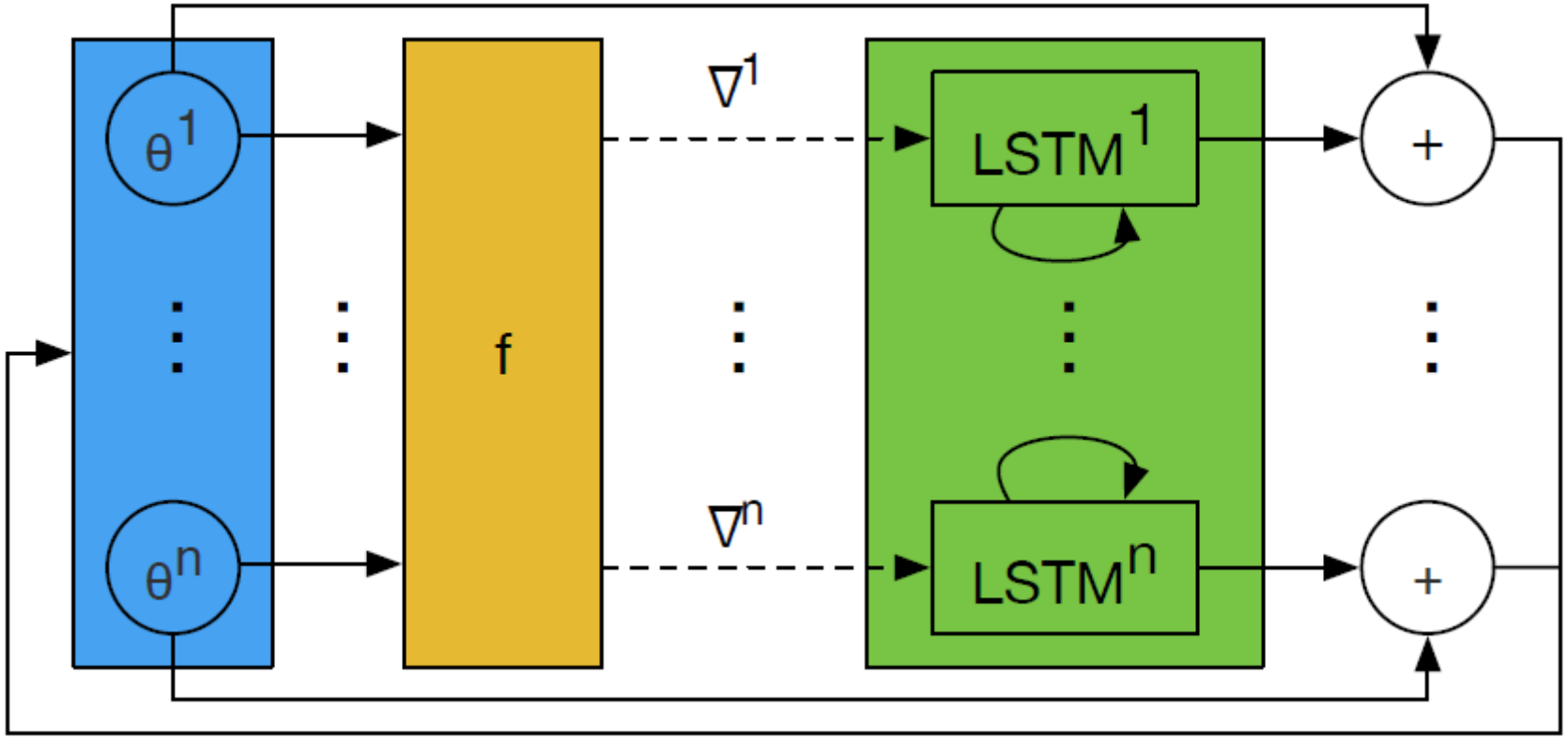
$$\mathcal{L}(\phi) = \mathbb{E}_f \left[\sum_{t=1}^{T} w_t f(\theta_t) \right]$$

$$\text{where} \quad \theta_{t+1} = \theta_t + g_t$$

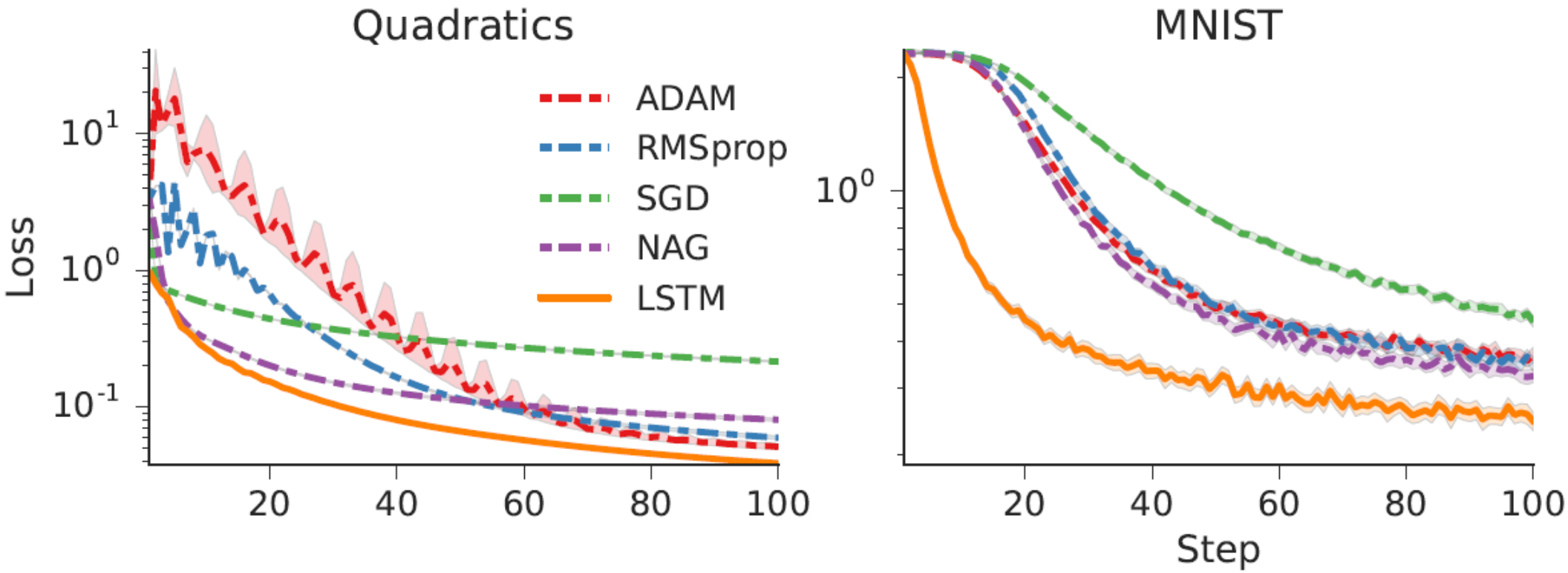
Outer optimization computational graph



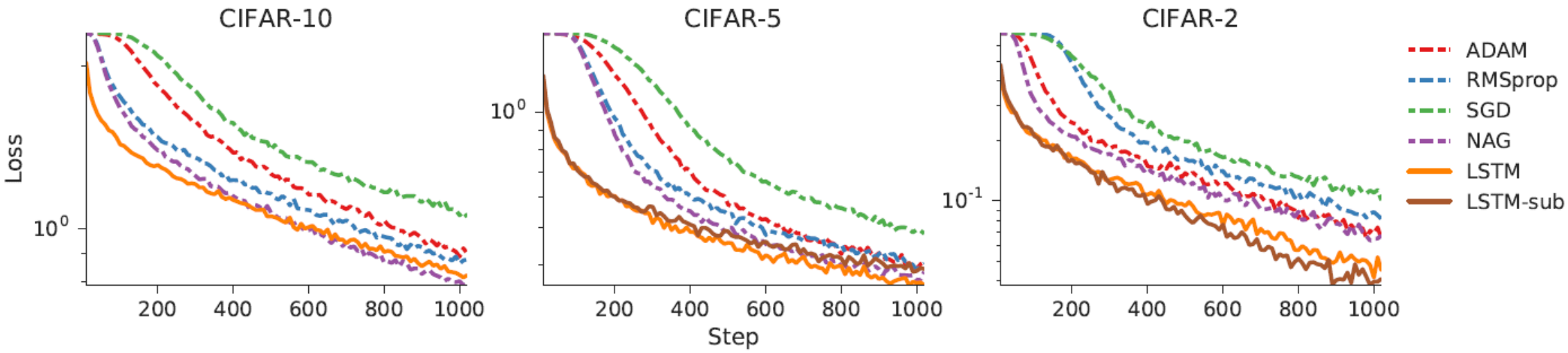
Coordinatewise LSTM optimizer



Results


Quadratics
MNIST
ADAM
RMSprop
SGD
NAG
LSTM
Loss
10^1
10^0
10^{-1}
20
40
60
80
100
Step

Results


CIFAR-10
CIFAR-5
CIFAR-2
Loss
10^0
10^0
10^{-1}
200
400
600
800
1000
Step
ADAM
RMSprop
SGD
NAG
LSTM
LSTM-sub

Conclusion

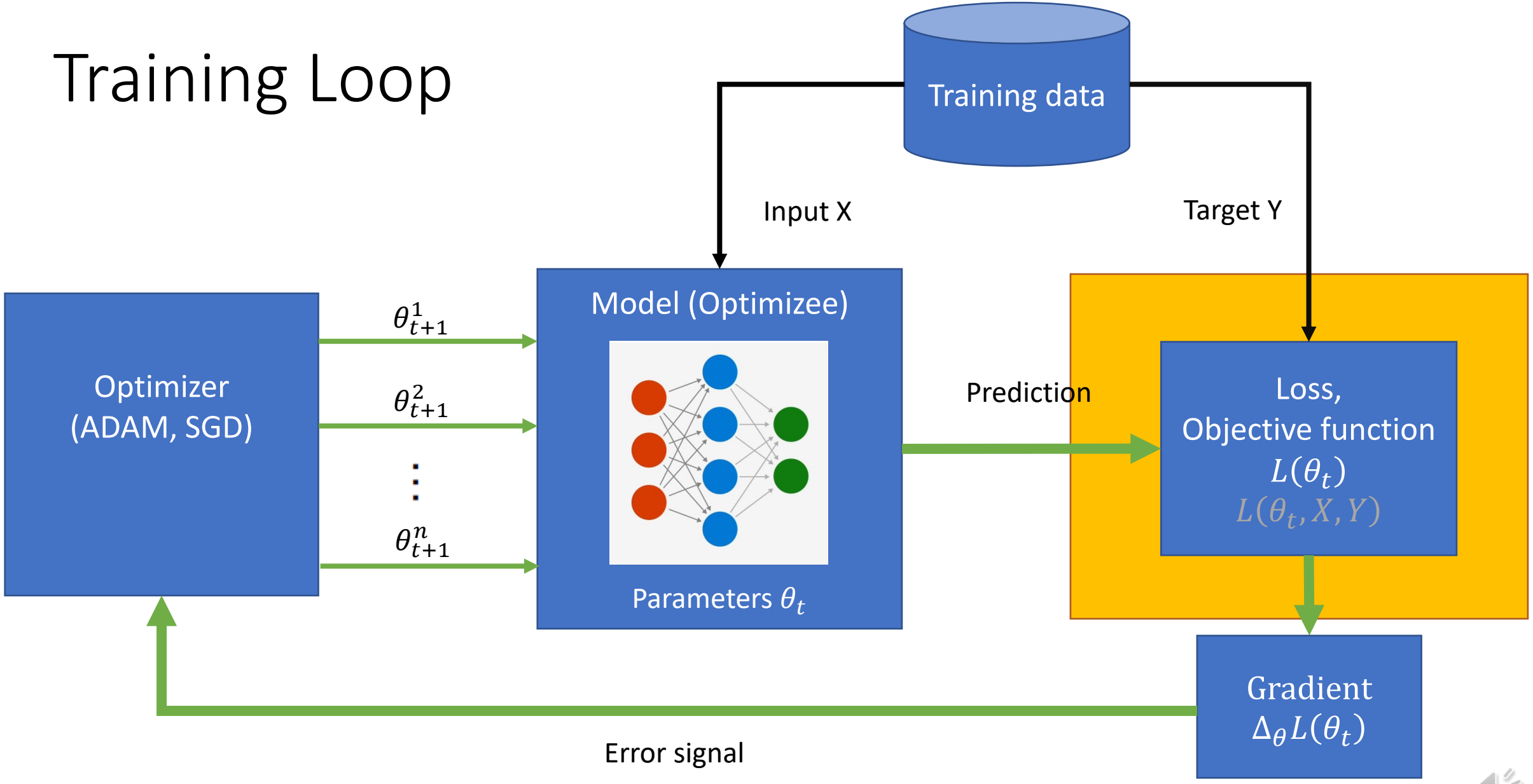
- Better performance than state-of-the-art optimizers
- High degree of transfer between different tasks and different architectures

- LSTM is costlier to run than SGD
- Need meta training

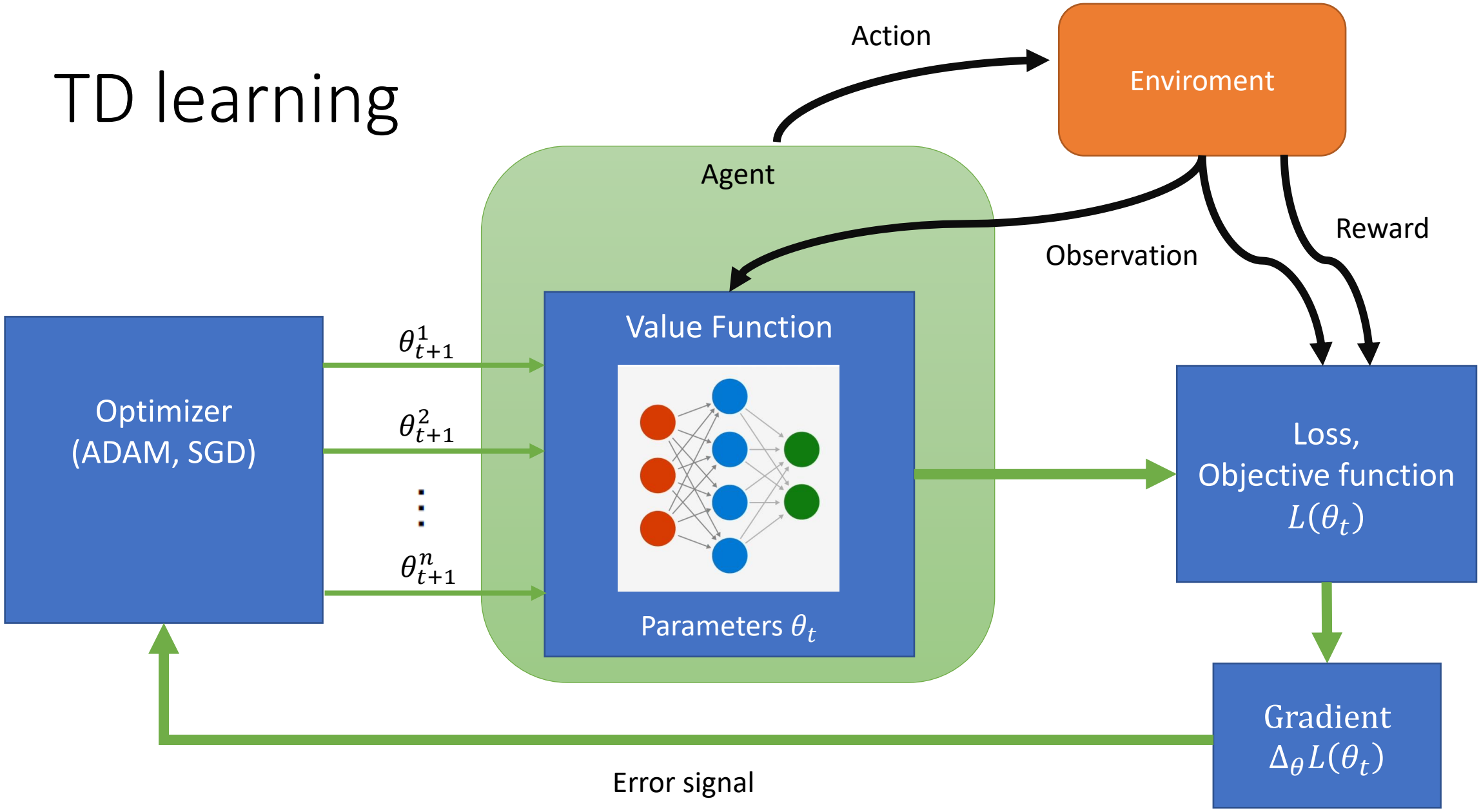
Meta-Gradient Reinforcement Learning with an Objective Discovered Online

Zhongwen Xu, Hado van Hasselt, Matteo Hessel
Junhyuk Oh, Satinder Singh, David Silver
DeepMind

Training Loop



TD learning


Action
Enviroment
Agent
Observation
Reward
Value Function
Optimizer
(ADAM, SGD)
θ^1_{t+1}
θ^2_{t+1}
θ^n_{t+1}
Parameters θ_t
Loss,
Objective function
$L(\theta_t)$
Gradient
$\Delta_\theta L(\theta_t)$
Error signal

Previous work

- RL^2: solves the problem by applying a RL algorithm to learn a RNN which represents the RL algorithm

- MAML: searches for a good initialisation of gradient based models.

Proposed solution

- Black box method
- Parameterizes target (loss) of RL algorithm
- Meta-learned online
- No «meta training»

Update Targets in RL algorithms

$$\tau_t = \{S_t, A_t, R_{t+1}, \dots\}$$

$$G_t = R_{t+1} + \gamma v_\theta(S_{t+1})$$

$$\theta \leftarrow \theta + \alpha(G_t - v_\theta(S_t))\nabla_\theta v_\theta(S_t)$$

Update Targets in RL algorithms

$$\tau_t = \{S_t, A_t, R_{t+1}, \dots\}$$

$$G_t = R_{t+1} + \gamma \max_a q_\theta(S_{t+1}, a)$$

$$\theta \leftarrow \theta + \alpha(G_t - q_\theta(S_t, A_t))\nabla_\theta q_\theta(S_t, A_t)$$

Idea: parameterize update target

$$G_t = g_\eta(\tau_t)$$

$$g_\eta : \tau_t \rightarrow \mathbb{R}$$

Meta gradients

$$\Delta\theta_i \propto \nabla_{\theta_i} L_{\eta}^{\text{inner}}(\tau_i, \theta_i) \qquad \theta_{i+1} = \theta_i + \Delta\theta_i$$

$$\theta_i \xrightarrow{\eta} \theta_{i+1} \xrightarrow{\eta} \ldots \xrightarrow{\eta} \theta_{i+M-1} \xrightarrow{\eta} \theta_{i+M}$$

$$\Delta\eta \propto \nabla_{\eta} L^{\text{outer}}(\tau_{i+M+1}, \theta_{i+M}) \qquad \eta \leftarrow \eta + \Delta\eta$$

Value based control

$$\Delta\theta \propto (g_\eta(\tau) - v_\theta(S))\nabla_\theta v_\theta(S).$$

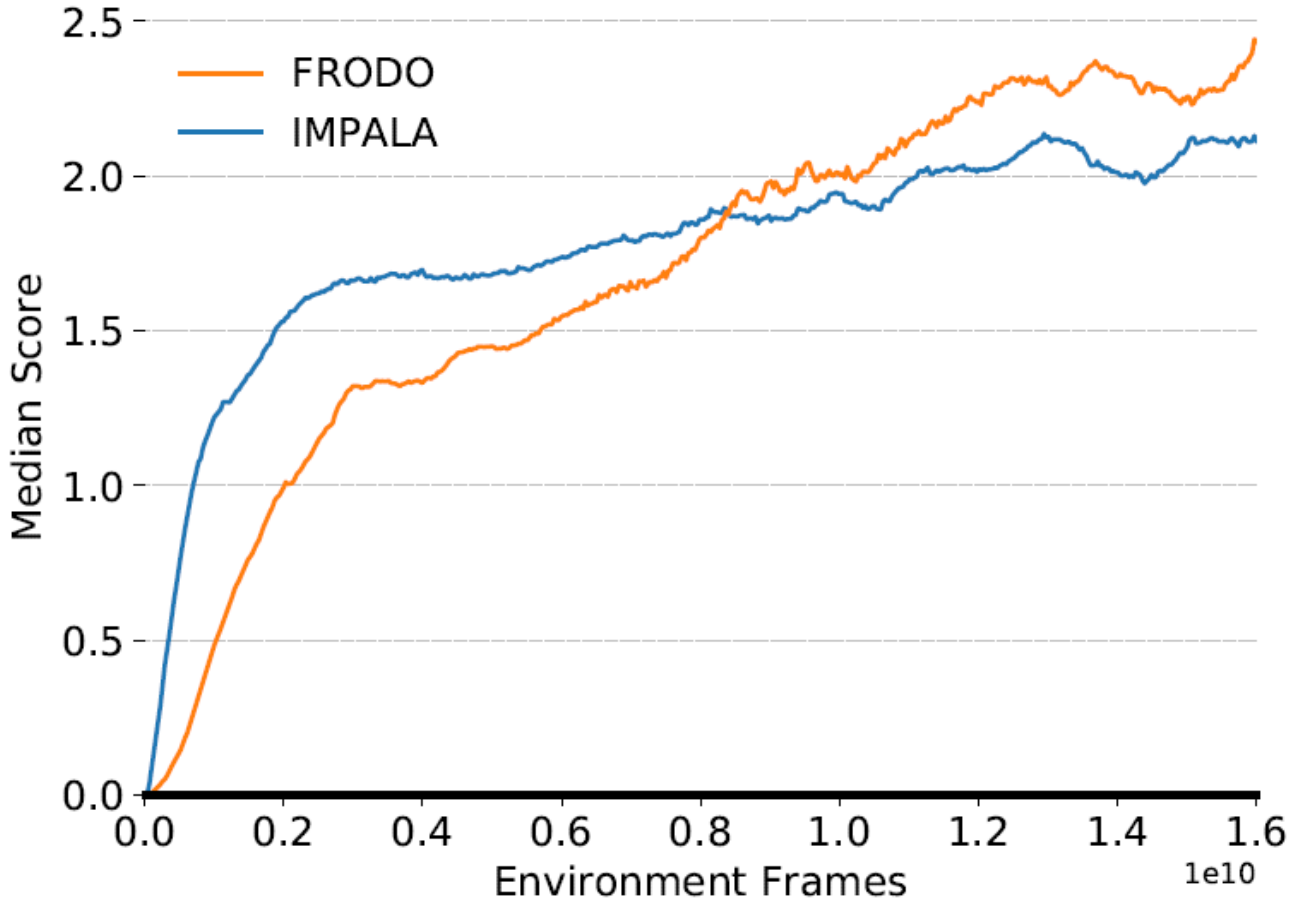
$$\nabla_{\theta'} L^{\text{outer}} = (G(\tau') - v_{\theta'}(S'))\nabla_{\theta'} v_{\theta'}(S')$$

Consistent update targets heuristic

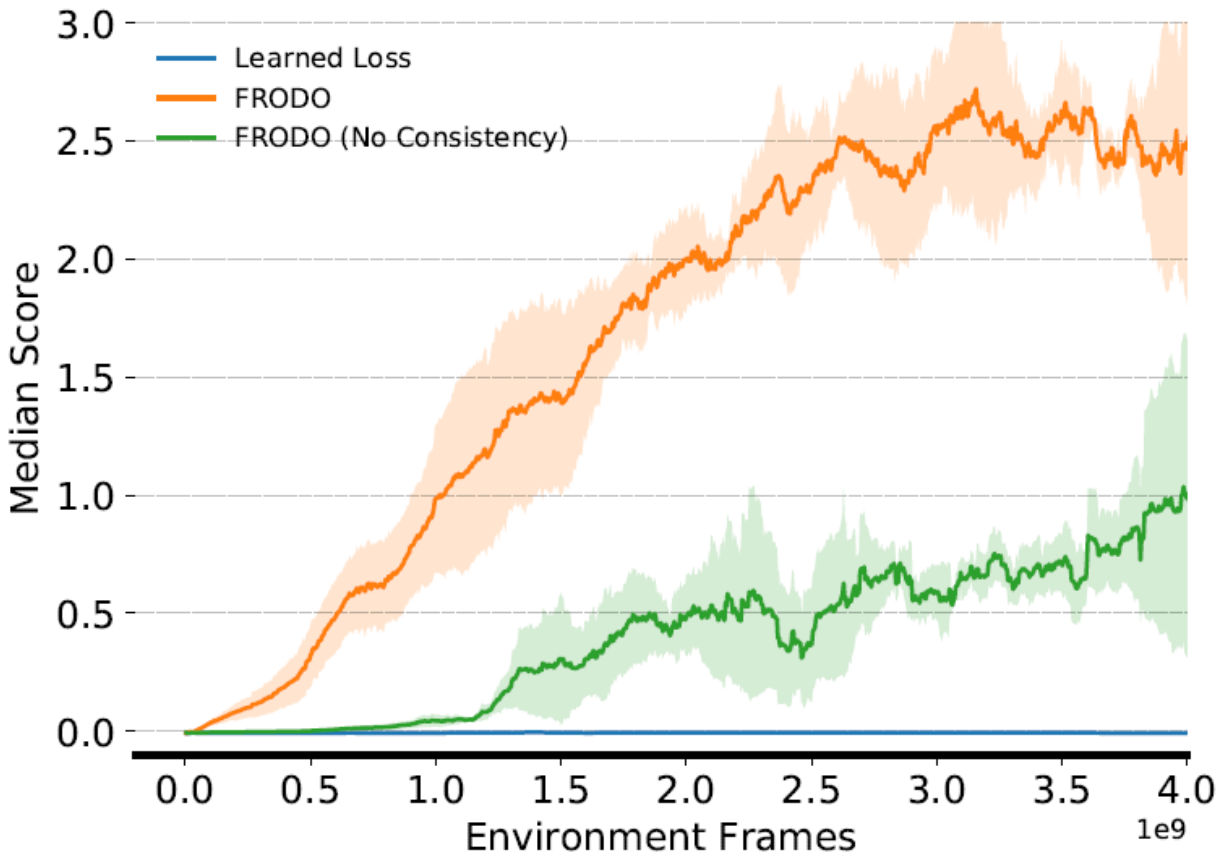
$$G_t^\eta = R_{t+1} + \gamma G_{t+1}^\eta$$

$$L^{\text{outer}} \leftarrow L^{\text{outer}} + c||\perp(R_{t+1} + \gamma G_{t+1}^\eta) - G_t^\eta||^2$$

Results


FRODO
IMPALA
Median Score
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
Environment Frames
1e10

Results


Learned Loss
FRODO
FRODO (No Consistency)
Median Score
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Environment Frames
1e9

Conclusion

- FRODO can outperform a strong actor critic baseline
- Solves boostrapping
- Solves non-stationarity
- Learns online, during single agent lifetime
- Can adapt to changes

- Needs heuristic
- Still needs hand tuning

Summary

- First paper proposes learned optimizer in form of recurrent NN, for supervised learning taks

- FRODO proposes to learn the update target of RL algorithms

Benchmarks: DeepMind Alchemy

Alchemy: A structured task distribution for meta-reinforcement learning

Jane X. Wang[*,†,1], **Michael King**[*,1], **Nicolas Porcel**[1], **Zeb Kurth-Nelson**[1,2], **Tina Zhu**[1], **Charlie Deck**[1], **Peter Choy**[1], **Mary Cassin**[1], **Malcolm Reynolds**[1], **Francis Song**[1], **Gavin Buttimore**[1], **David P. Reichert**[1], **Neil Rabinowitz**[1], **Loic Matthey**[1], **Demis Hassabis**[1], **Alexander Lerchner**[1], **Matthew Botvinick**[‡,2]

[1]DeepMind, London, UK

[2]University College London, London, UK

February 8, 2021

References

- Marcin Andrychowicz et al. (2016). **Learning to learn by gradient descent by gradient descent**. [cs.AI]
- Zhongwen Xu et al. (2020). **Meta-Gradient Reinforcement Learning with an Objective Discovered Online**. [cs.LG]
- Jane X. Wang et al. (2021). **Alchemy: A structured task distribution for meta-reinforcement learning**.[cs.LG]

Additional references

- https://sites.google.com/view/icml19metalearning
- https://lilianweng.github.io/lil-log/2018/11/30/meta-learning.html
- https://medium.com/dataseries/learning-to-learn-gradient-descent-by-gradient-descent-a-paper-review-44292f2fb1ff
- https://becominghuman.ai/paper-repro-learning-to-learn-by-gradient-descent-by-gradient-descent-6e504cc1c0de