Learning to Optimize

Marc Katzenmaier
Recent Trends in Automated Machine-Learning
Technical University of Munich
Thursday, 11th July, 2019
Motivation

- Improvement with automated machine learning
  - data augmentation
  - architecture
  - activation functions
- learn an optimizer
  - faster convergence
  - lower final loss value
  - more stable training
  - no hyperparameter optimisation
Optimizer

- can be rewritten in general form
- update rule can be seen as a policy

known handcrafted examples:
- gradient descent: \[ \pi(f, \{x^{(0)}, \ldots, x^{(i-1)}\}) = -\gamma \nabla f(x^{(i-1)}) \]
- gradient descent with momentum: \[ \pi(f, \{x^{(0)}, \ldots, x^{(i-1)}\}) = -\gamma \left( \sum_{j=0}^{i-1} \alpha^{i-1-j} \nabla f(x^{(j)}) \right) \]

**Algorithm 1** General structure of optimization algorithms

**Require:** Objective function \( f \)
\[ x^{(0)} \leftarrow \text{random point in the domain of } f \]
for \( i = 1, 2, \ldots \) do
\[ \Delta x \leftarrow \pi(f, \{x^{(0)}, \ldots, x^{(i-1)}\}) \]
if stopping condition is met then
\[ \text{return } x^{(i-1)} \]
end if
\[ x^{(i)} \leftarrow x^{(i-1)} + \Delta x \]
end for
Reinforcement Learning

- negative loss of the child network as reward
  - encourage fast convergence and low final loss value
  - undiscounted reward
- learn the policy in continuous state and action space
- using Guided Policy Search

\[ R = \sum_{t=1}^{T} -l_t(x) \]
Guided Policy Search

- Trajectory Optimization (learn dynamics)
- Supervised Learning (optimize policy)
Guided Policy Search - Trajectory Optimization

- dynamics calculates the response of the system when changing the variables
- trajectory is the path of optimization steps during training
- initial trajectory is chosen to behave like SGD with momentum
- trajectory distributions are more stable, especially for discontinuous dynamics
- need to approximate dynamics
  - sample distribution and linearize at each time step
  - number of samples can be reduced with knowledge of previous samples
- trajectories mustn’t deviate too much for good linear approximation

\[
\min_{p(\tau) \in \mathcal{N}(\tau)} E_p[\ell(\tau)] \quad \text{s.t.} \quad D_{\text{KL}}(p(\tau) \| \hat{p}(\tau)) \leq \epsilon
\]

- solve with Lagrangian with dual gradient descent
Guided Policy Search - Supervised Learning

- take the samples of the trajectory distribution
- learn policy supervised by minimizing

\[
\sum_{t=1}^{T} \lambda_t D_{KL}(p(x_t, u_t | x_t) \parallel p(x_t, u_t))
\]

  - minimize the difference between the optimized trajectory and the policy
  - converges to a policy which produces the trajectory
Network for the Policy

last 25 loss values

last 25 gradients

50 hidden neurons

update value which is added to the current weights
Experiment Logistic Regression

- convex optimization surface
  \[
  \min_{w,b} \frac{1}{n} \sum_{i=1}^{n} y_i \log(\sigma(w^T x_i + b)) + (1 - y_i) \log(1 - \sigma(w^T x_i + b)) + \frac{\lambda}{2} \|w\|^2
  \]

- artificial data
  - single set created based on two multivariate gaussians, 50 samples each
  - 90 of these sets for training
  - 100 of these sets for testing

- only L-BFGS converges faster
  - known for fast convergence with convex problems
Experiment Robust Linear Regression

- nonconvex problem
- artificial data
  - datapoints: 100 samples from 4 multivariate gaussians per trainings set
  - labels: datapoints of each gaussian projected on a different random vector, a random bias is added perturbed with i.i.d. gaussian noise
  - 120 sets for training
  - 100 sets for testing
- outperforms other policies after 30 epochs
Experiment Neural Net Classifier

- complex optimization surface with multiple local optima
  \[
  \min_{W,U,b,c} \frac{1}{n} \sum_{i=1}^{n} \log \left( \frac{\exp((U \max(W x_i + b, 0) + c)y_i)}{\sum_j \exp((U \max(W x_i + b, 0) + c)_j)} \right) + \frac{\lambda}{2} ||W||_F^2 + \frac{\lambda}{2} ||U||_F^2
  \]

- Fully Conected NN with 2 input, 2 hidden and 2 output neurons and regularization

- artificial data
  - datapoints are sampled from 4 different gaussians
  - labels randomly 0 or 1 assigned per gaussian
    - at least 1 gaussian of each label
  - 120 sets for training
  - 100 sets for testing

- outperforms all other policies
  - the first epochs similar to SGD with momentum
Conclusion

- **Strength**
  - simple idea to learn the policy
  - outperformed other optimizer
  - no hyperparameter tuning

- **Weaknesses**
  - only toy problems, no real data or application
  - scalability problem, need to cache 25 gradients and 25 loss values per weight
  - guided policy search not straight forward to train