ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. ICLR

Contents

1. Introduction

- 여러 parameter로 구성된 미분가능한 objective function의 maximization / minimization은 1st order partial derivatives 를 활용한 gradient ascent/decent를 통해 효과적으로 optimization이 가능함
- Objective function의 loss는 매 time step (=mini-batch)마다 다른 loss를 갖는 stochastic function의loss의 합으로 치환될 수 있고, 이 step마다 다른 gradient를 활용해 parameter를 개선하는 stochastic gradient ascent/descent는 ML/DL 분야에 성공적으로 적용해왔음
- 논문의 저자들은 Objective function에 dropout과 같은 noise가 추가되면 high-dimensional parameter space를 갖는 stochastic optimization을 할 때는 high-order optimization보다는 Momentum, Adagrad와 같은 1st order 방법이 좋다고 주장하며, 1st order method인 Adam을 제시함
- Adam은 parameter의 1st moment와 2nd moment를 활용해 parameter마다 learning rate를 조절하며, gradient rescaling이 필요없고, stepsize가 설정한 learning rate hyperparameter로 bound되며, non-stationary objective에도 잘 작동하며, sparse gradient에도 잘 작동한다는 장점이 있음

2. Related Work

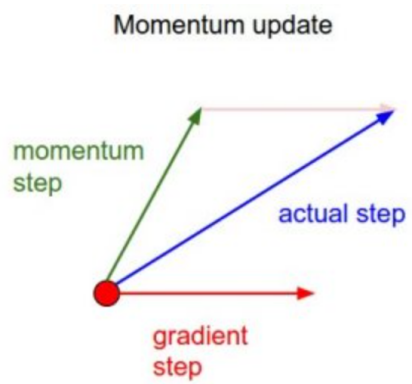
- Adam는 Momentum + RMSprop
  - Momentum의 이전 update에 활용한 gradient의 관성 방향 활용
  - Adagrad/RMSProp의 parameter마다 이전 update들에서 이동한 gradient 크기를 활용해 parameter마다 learning rate를 adaptive하게 적용하는 것 활용

- θ_t : $parameter\ at\ time\ step\ t$
- $g_t\ (= \nabla_t J(\theta_t)\ )$: $gradient\ at\ time\ step\ t$
- α : $learning\ rate$

2. Related Work

- Momentum : 과거의 방향성을 참고해서 이동
  - $v_t = \gamma \cdot v_{t-1} + \alpha \cdot g_t \quad (\gamma : momentum\ term)$
  - $v_t = \alpha \cdot g_t + \alpha \cdot \gamma \cdot g_{t-1} + \alpha \cdot \gamma \cdot \gamma \cdot g_{t-2} + ..$
  - $\theta_t = \theta_t - v_t$

  - Gradient의 지수 이동 평균을 이용해 gradient를 update함
  - Local Minimum 에 빠졌을 시에 ($g_t = 0$), 이전 gradient의 관성을 활용해 빠져나올 수 있음
  - Oscilation시에 SGD보다 빠르게 수렴이 가능함



2. Related Work

- Adagrad : 과거의 update를 고려해 parameter마다 다른 stepsize를 설정
  - $G_t = G_{t-1} + (g_t)^2$
  - $\theta_t = \theta_{t-1} - \alpha \cdot \frac{1}{\sqrt{G_t+\epsilon}} \cdot g_t$
  - Gradient의 제곱 값을 저장해 과거 update 정보를 저장
  - G_t 값을 이용해 과거에 많이 update한 gradient(G_t ↑)는 상대적으로 작은 stepsize를 주고 적게 update (G_t ↓) 한 gradient (sparse gradient)는 상대적으로 큰 stepsize를 주어 parameter마다 다른 stepsize를 줌
  - 학습(t)이 증가됨에 따라 $\alpha \cdot \frac{1}{\sqrt{G_t+\epsilon}}$가 0으로 수렴해 학습이 진행되지 않음

2. Related Work

- RMSProp : gradient 제곱의 단순합 대신 gradient 제곱의 지수 평균을 저장하는 방법
  - $G_t = \gamma \cdot G_{t-1} + (1-\gamma) \cdot (g_t)^2$
  - $\theta_t = \theta_{t-1} - \alpha \cdot \frac{1}{\sqrt{G_t + \epsilon}} \cdot g_t$

  - Gradient의 제곱의 지수평균을 저장해 과거 update 정보를 저장
  - 학습(t)이 증가됨에 따라 G_t가 무한대로 커지는 것을 방지함
  - 전체 timestep의 gradient 크기를 동일하게 고려하는 것이 아닌 최근의 gradient를 상대적으로 더 반영해 parameter마다의 stepsize를 적용 (최근에 많이 update한 gradient는 조금 덜 이동, 덜 update한 gradient는 조금 더 이동)

3. Algorithm

- Adam은 β_1과 β_2로 gradient의 이전 방향성(momentum)과 크기(RMSprop)를 동시에 활용해 update하는 방식
- $\alpha = 0.001,\ \ \beta_1 - 0.9,\ \ \beta_2 = 0.999, \epsilon = 1e-7$ 이 default로 추천됨

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

3. Algorithm

- Adam의 1st momentum과 2nd momentum이 아래와 같을 때,
  - $m_t = \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$
  - $v_t = \beta_2 \cdot G_{t-1} + (1-\beta_2) \cdot (g_t)^2$
  - $\theta_t = \theta_{t-1} - \alpha \cdot \frac{m_t}{\sqrt{v_t}+\epsilon}$
  - 의 식을 사용하면 initial 값에 bias가 있음

  - $\widehat{m_t} = m_t/(1-\beta_1^t)$
  - $\widehat{v_t} = v_t/(1-\beta_2^t)$
  - $\theta_t = \theta_{t-1} - \alpha \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\epsilon}$
  - 로 수정해 initial 값에 bias 되는 것을 보정함

3. Algorithm

- Adam Update Rule
  - $\epsilon = 0$ 이라 가정하면, $stepsize\ at\ t:\ \Delta_t = \alpha \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}}$
  - 일반적인 경우, $\left|\frac{E[g]}{\sqrt{E[g^2]}}\right| \leq 1$이기 때문에 $\frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}} \approx \pm 1$이 되고,
  - $|\Delta_t| \leq \alpha \cdot \frac{(1-\beta_1)}{\sqrt{(1-\beta_2)}}$ $if\ (1-\beta_1) > \sqrt{(1-\beta_2)}$: sparse gradient
  - $|\Delta_t| \leq \alpha$ otherwise
  - 따라서 저자들은 gradient의 stepsize는 $|\Delta_t| \lesssim \alpha$ 로 bound되고, 이는 곧 gradient가 항상 정해진 step으로 탐색해 특정 횟수 내에서 전역/국소 최적값으로 수렴이 가능하다고 주장함
  - 또한 저자들은 $\frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}}$을 signal-to-noise ratio로 정의하며, signal-to-noise ratio가 작을수록 stepsize가 작아진다고 함
    - signal-to-noise ratio가 작다는 것은 $\widehat{m_t}$의 방향과 실제 gradient 방향에 대한 불확실성이 증가했다는 것을 의미해 stepsize를 작게 가져가고, 이는 바람직한 속성임
    - 특히 optimal에 가까워질 수록 gradient 방향성에 대한 불확실성이 증가하는데, stepsize를 줄여 자동적으로 annealing이 가능하다고 주장

4. Initial bias correction

- 논문의 저자들은
  - $E[g]= E[m_t]$, $E[g^2]= E[v_t]$로 추정하고 싶어했음
  - 1st moment (m_t) 로 설명하면,
    - $m_t = \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$ 에서 $m_0 = 0$으로 initialize, $\beta_1 = 0.9$라 가정하면
      - $m_1 = 0.9 \cdot m_0 + 0.1 \cdot g_1 = 0.1 \cdot g_1$
      - $m_2 = 0.9 \cdot m_1 + 0.1 \cdot g_2 = 0.9 \cdot 0.1 \cdot g_1 + 0.1 \cdot g_2$
      - $m_3 = 0.9 \cdot m_2 + 0.1 \cdot g_3 = 0.9 \cdot \ 0.9 \cdot 0.1 \cdot g_1 + 0.9 \cdot 0.1 \cdot g_2 + 0.1 \cdot g_3$

      m_t가 Initial 값 $m_0 = 0$에 영향을 많이 받아
      이후 m_1~m_{t-1}정보를 충분히 반영 X
      (m_1은 첫 step인데도 첫 gradient인 g_1의 0.1에 반영)
    - 일반화해서 다시 쓰면
    - $m_t = (1-\beta_1)\sum_{i=1}^{t} \beta_{i=1}^{t-i} \cdot g_i$
      - $\mathrm{E}[m_t] = E[(1-\beta_1)\sum_{i=1}^{t} \beta_{i=1}^{t-i} \cdot g_i]$
      - $\mathrm{E}[m_t] = E[g_t] \cdot (1-\beta_1)\sum_{i=1}^{t} \beta_{i=1}^{t-i} + \xi$
      - $\mathrm{E}[m_t] = E[g_t] \cdot (1-\beta_1^t) + \xi$
      - ξ는 β_1이 weight decay로 먼 과거의 gradient를 반영하지 않게 해 매우 작은 값을 주어서 무시가 가능함
    - $\mathrm{E}[m_t] = E[g_t] \cdot (1-\beta_1^t)$
      - $(1-\beta_1^t)$는 $m_0 = 0$으로 설정해서 생긴 값으로,
      - 저자들은 이를 양변에 나눠 줌으로 bias 보정

4. Initial bias correction

- 이어서
  - 1st moment로 설명하면,
    - $m_t = \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$ 에서 $m_0 = 0$으로 initialize, $\beta_1 = 0.9$로 가정하고
    - $\widehat{m_t} = m_t/(1-\beta_1^t)$ 를 하면,
      - $m_1 = 0.9 \cdot m_0 + 0.1 \cdot g_1 = 0.1 \cdot g_1$
      - $\widehat{m_1} = \frac{m_1}{1-0.9} = 0.1 \cdot \frac{g_1}{0.1} = g_1$
      - $m_2 = 0.9 \cdot m_1 + 0.1 \cdot g_2 = 0.9 \cdot g_1 + 0.1 \cdot g_2$
      - $\widehat{m_2} = \frac{m_1}{1-0.9^2} = \frac{0.9}{1-0.9^2} \cdot g_1 + \frac{0.1}{1-0.9^2} \cdot g_2$

m_t가 Initial 값 $m_0 = 0$에 영향을 받지않고
이후 $m_1 \sim m_{t-1}$정보를 충분히 반영하면서 update

- 2nd moment도 마찬가지로 적용이 가능함
- 이에 따라 식을 수정해 Adam 수식을 수정
  - $m_t = \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t \quad > \widehat{m_t} = m_t/(1-\beta_1^t)$
  - $v_t = \beta_2 \cdot G_{t-1} + (1-\beta_2) \cdot (g_t)^2 \; > \widehat{v_t} = v_t/(1-\beta_2^t)$
  - $\theta_t = \theta_{t-1} - \alpha \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\epsilon}$ 로 식을 수정

5. Convergence Analysis

- 임의의 알려지지 않은 일련의 convex cost function이 주어질 때 ($f_1(\theta), f_2(\theta), \ldots, f_T(\theta)$), θ_{t-1}와 f_t를 가지고 θ_t을 예측하는 task에서
- Online prediction 인 $f_t(\theta_t)$와 이전 모든 step의 가능한 feasible set χ의 best fixed point parameter인 $f_t(\theta_t^*)$의 timestep마다의 차이를 전부 합한 score인 Regret을 아래와 같이 정의할 때,

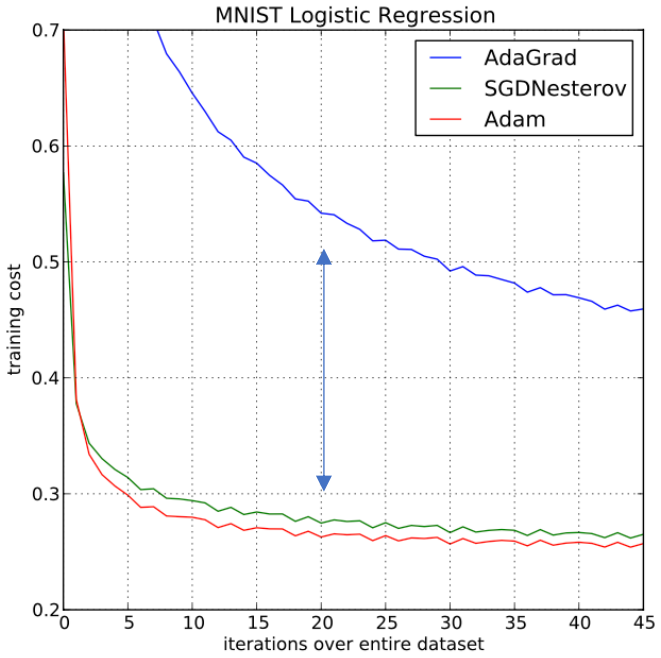
$$R(T) = \sum_{t=1}^{T} [f_t(\theta_t) - f_t(\theta^*)]$$

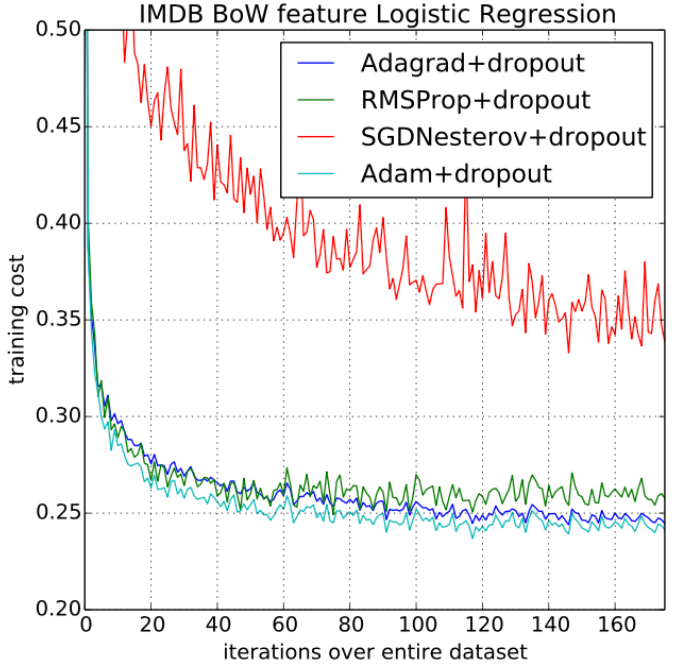
- Adam의 Regret bound는 $O(\sqrt{T})$로 당시 convex online problem의 bound 중 최고를 달성
- Adam과 Adagrad와 같이 parameter별로 다른 stepsize를 주는 방법이 그렇지 않은 방법들에 비해 bound가 더 작음
- 또한 훈련이 지날수록 momentum coefficient β_1^t이 작아지는 것은 훈련 막바지에 수렴을 향상시킴

6. Experiment

- L2 Regularized multi-class Logistic Regression
  - Logistic Regression은 local minimum을 신경쓰지 않고 서로 다른 optimizer의 performance를 비교하기에 좋은 task
  - Learning rate α를 time step t마다 $1/\sqrt{t}$로 decay 시킴
  - Sparse gradient에 대한 성능도 알아보기 위해 IMDB movie review dataset을 고빈도 단어 10,000로 BoW로 embedding 후 50% dropout을 적용해 실험

*Momentum term*이 있는 경우 수렴속도가 빠름

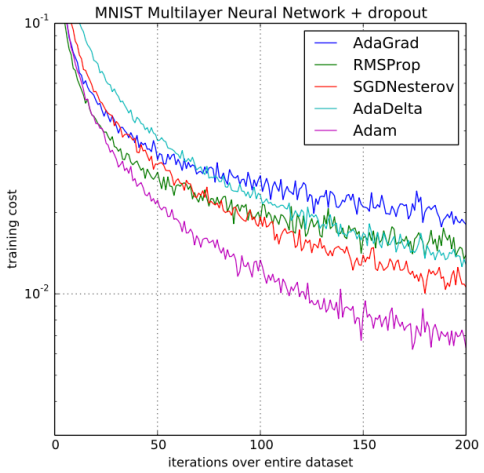


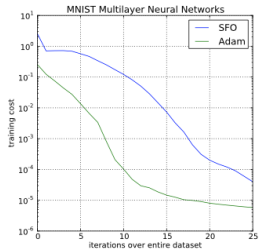


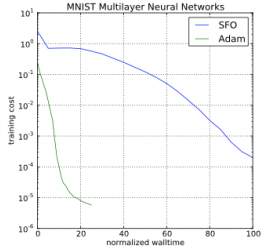
*parameter*별로 *learning rate* 조절하는 Adagrad, RMSprop, Adam이 sparse gradient에서 수렴이 빠름

6. Experiment

- Multi-layer neural network
  - Multi-layer neural network은 non-convex objective function으로 5절에서 보인 convergence analysis를 적용할 수는 없지만 다른 optimizer와 성능 비교를 통해 adam의 우수성을 보임
  - 2개의 Layer에 hidden unit을 1,000개 두고 ReLU를 activation으로 활용
  - Dropout을 추가해 stochastic regularization 상황에서의 converge performance 측정
  - 이전에 Multi-layer neural network의 optimization에서 좋은 성과를 보인 SFO(sum of functions)는 mini-batch마다 메모리를 선형적으로 추가로 요구하고 stochastic regularization 에서는 수렴을 못하지만 adam은 더 빠른 속도로 안정적으로 수렴

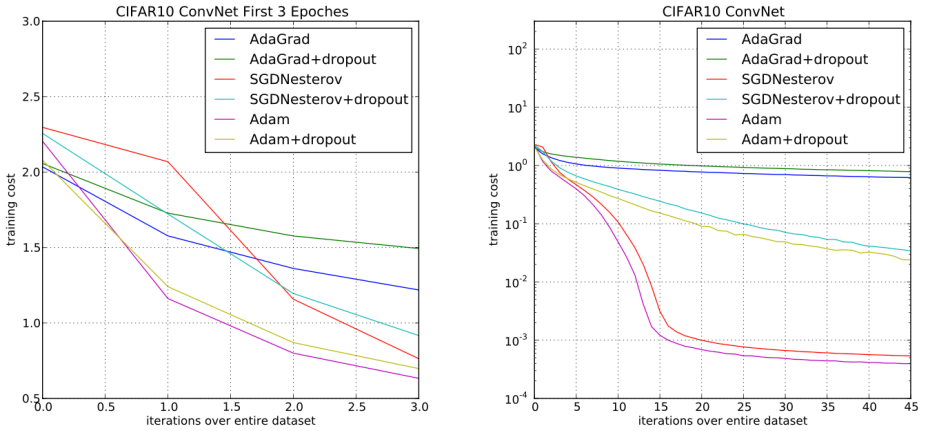






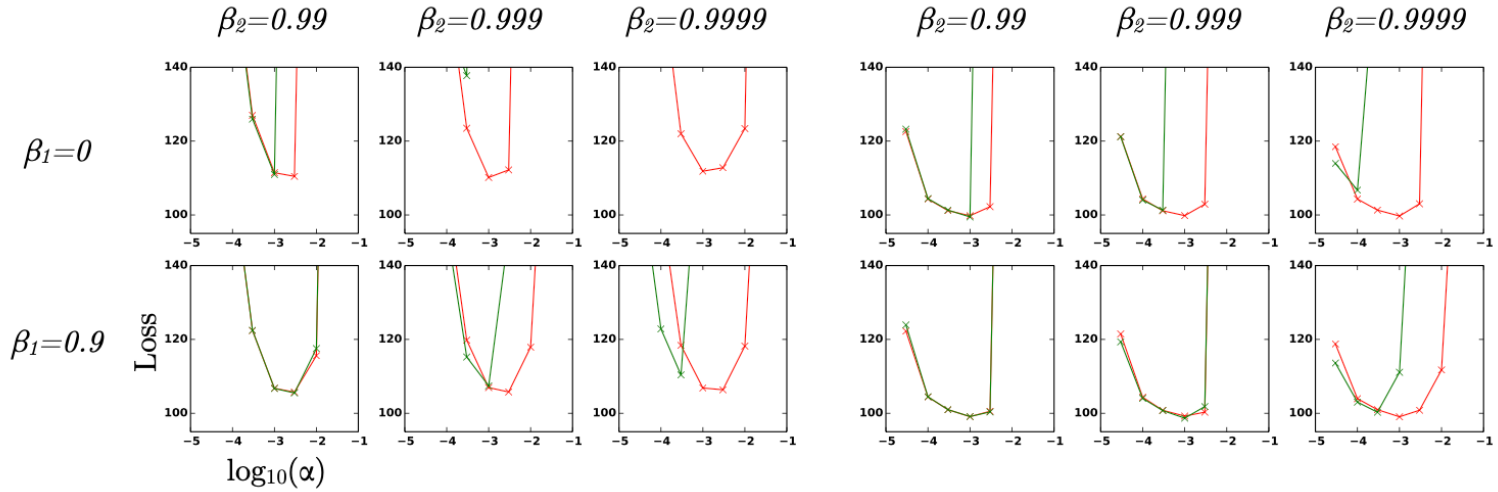
6. Experiment

- Convolutional neural network
  - CNN의 weight sharing은 Multi-layer neural network보다 서로 다른 layer에 영향을 미치는게 커 상대적으로 낮은 learning rate를 사용
  - 5 x 5 convolution layer, 3 x 3 maxpooling layer, 1,000 ReLU unit으로 된 2 layer FFN으로 실험
  - CNN 학습 시에는 parameter마다 learning rate (step size)를 조정하는 $\hat{v}_t$가 거의 0이 되어 ϵ에 dominated 되고, 1st moment을 조절하는 SGDNesterov와 Adam 그렇지 않은 Adagrad보다 수렴이 빠름
  - 저자들은 $\hat{v}_t$가 거의 0이 되어도 parameter마다 learning rate를 조절한 Adam이 수동으로 조절해야하는 SGDNesterov보다 성능이 조금 더 좋음을 보인다고 설명



6. Experiment

- Bias correction term
  - Bias correction term의 효과를 보기 위해 bias correction이 없지만 gradient의 관성 방향과 parameter별 stepsize를 동시에 계산하도록 하는 optimizer인 RMSprop + Momentum과 Adam을 VAE(Variational Auto Encoder)로 훈련해 비교실험을 진행
  - β_2가 1에 가까울수록 sparse gradient에는 robust해지지만 그만큼 initialization bias가 커짐
  - 또한 훈련 epoch이 길어질수록, 세부적인 pattern을 찾기 위해서 gradient의 update는 sparse해지는데 이 때 bias correction term이 있는 Adam이 RMSprop + Momentum보다 Loss를 더 줄임을 보임



(a) after 10 epochs (b) after 100 epochs

7. Extension and Conclusion

- Extension
  - Adam은 현재와 과거 gradient의 L-2 Norm에 반비례하게 parameter별 learning rate를 조절함
  - L-2 Norm을 L-p Norm으로 확장하면 unstable하지만, $p \to \infty$면 안정적인 알고리즘이 도출됨
  - 이를 AdaMax라 하며, AdaMax는 v_t 제약식이 max로 바뀌면서 initial bias correction term이 생략되고,

$\alpha = 0.002,\ \beta_1 - 0.9,\ \beta_2 = 0.999$가 default로 추천됨

Algorithm 2: *AdaMax*, a variant of Adam based on the infinity norm. See section 7.1 for details. Good default settings for the tested machine learning problems are $\alpha = 0.002$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. With β_1^t we denote β_1 to the power t. Here, $(\alpha/(1-\beta_1^t))$ is the learning rate with the bias-correction term for the first moment. All operations on vectors are element-wise.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1st moment vector)
$u_0 \leftarrow 0$ (Initialize the exponentially weighted infinity norm)
$t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$u_t \leftarrow \max(\beta_2 \cdot u_{t-1}, |g_t|)$ (Update the exponentially weighted infinity norm)
$\theta_t \leftarrow \theta_{t-1} - (\alpha/(1-\beta_1^t)) \cdot m_t/u_t$ (Update parameters)
end while
return θ_t (Resulting parameters)

7. Extension and Conclusion

- Conclusion
  - Stochastic objective function의 gradient-based optimization에 효과적인 알고리즘을 제시
  - Sparse gradient에 우수한 Adagrad와 non-stationary에 우수한 RMSprop의 장점을 합친 알고리즘인 Adam을 제시
  - Adam은 convex problem 뿐만 아니라 실험적으로 non-convex optimization에서도 robust함을 보임

- ✓https://stats.stackexchange.com/questions/232741/why-is-it-important-to-include-a-bias-correction-term-for-the-adam-optimizer-for
- ✓https://ruder.io/optimizing-gradient-descent/
- ✓http://shuuki4.github.io/deep%20learning/2016/05/20/Gradient-Descent-Algorithm-Overview.html
- ✓https://dalpo0814.tistory.com/29

감사합니다.