"확률적 목표함수의 1차 기울기 기반 최적화 기법 "

ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION (확률적 최적화 기법)

ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION

Diederik P. Kingma[*]
University of Amsterdam, OpenAI
dpkingma@openai.com

Jimmy Lei Ba[*]
University of Toronto
jimmy@psi.utoronto.ca

ABSTRACT

We introduce *Adam*, an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. The method is straightforward to implement, is computationally efficient, has little memory requirements, is invariant to diagonal rescaling of the gradients, and is well suited for problems that are large in terms of data and/or parameters. The method is also appropriate for non-stationary objectives and problems with very noisy and/or sparse gradients. The hyper-parameters have intuitive interpretations and typically require little tuning. Some connections to related algorithms, on which *Adam* was inspired, are discussed. We also analyze the theoretical convergence properties of the algorithm and provide a regret bound on the convergence rate that is comparable to the best known results under the online convex optimization framework. Empirical results demonstrate that Adam works well in practice and compares favorably to other stochastic optimization methods. Finally, we discuss *AdaMax*, a variant of *Adam* based on the infinity norm.

ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION

- 2015. ICLR(International Conference on Learning Representations) 발표

논문 기여도

- 기존의 확률적 기울기 하강법(SGD)은 고정된 학습률로 인해 학습 과정이 불안정하고 희소한 기울기 문제에 취약함.
- Adam 알고리즘은 적응형 학습률을 도입하여 이러한 한계를 극복하며, 딥러닝 모델 학습의 효율성과 안정성을 크게 향상시켰음

* ICML: 국제 학습표현 회의, 딥러닝 및 표현 학습 분야에서 매우 권위 있는 학술대회, 혁신적인 알고리즘이 주로 발표

주요 개념들

1. 확률적 최적화(Stochastic Optimization)와 Adam 알고리즘
2. 적응형 학습률(Adaptive Learning Rate)
3. 바이어스 보정(Bias Correction)
4. AdaMax
5. 파라미터 업데이트 식

1. 확률적 최적화(Stochastic Optimization)

1. 개념 : 대규모 데이터셋에서 *일부 샘플(미니배치)만을 사용해 매번 조금씩 다른 데이터로, 반복적으로 모델을 학습시키는 최적화 방법
2. 사용하는 이유: 모든 데이터를 한 번에 사용하지 않고 일부만 사용하기 때문에, 컴퓨터가 더 빨리 배우고 계산하는 데 시간이 덜 걸리며, 배울 때 생기는 *작은 실수들에도 잘 적응할 수 있음.
3. 단점: 샘플링으로 인한 변동성 큼

*작은 실수나 노이즈는 여러 번의 반복을 통해 자연스럽게 평균화됨

2. 본 논문에서 확률적 최적화 기법(ADAM)이란?

- 무작위로 선택된 미니배치(일부 샘플)를 통해, 각 파라미터를 적응적 학습률로 조정하여 최적화를 수행하는 기법
- 1차 및 2차 모멘트를 활용해 학습률을 동적으로 조정하여, 학습 과정의 효율성과 안정성을 높이는 데 중점을 둠

3. 고정된 학습률 & 적응형 학습률

- 학습률(learning rate): 한번에 얼마만큼씩 배울지를 결정하는 숫자

- **고정된 학습률**: 일정한 학습 속도를 유지해 구현이 간단하지만, 최적점에 빠르게 도달하거나 안정적으로 수렴하지 못할 수 있다.
- **적응형 학습률**: 학습 과정에 따라 속도가 조정되어 더 효율적이고 안정적이지만, 구현이 복잡하고 계산 비용이 더 많이 들 수 있다.

4. 바이어스 보정(Bias Correction)

1. Adam 알고리즘에서 초기 단계에서 발생하는 편향(bias)을 수정하기 위해 사용되는 중요한 기법

2. 왜 보정하는가?
   - Adam 알고리즘이 초기 학습 단계에서 기울기 평균(1차 모멘트)과 제곱 평균(2차 모멘트)을 계산할 때, 이 값들이 0에서 시작한다. 각 파라미터에 대해 이전 학습 결과가 전혀 없기 때문이다.
   - 이로 인해 학습이 느리게 진행될 수 있는데, 이때 바이어스 보정이 필요함

5. 파라미터 업데이트 관련 식

1. 파라미터 업데이트 식

$$\theta_t = \theta_{t-1} - \alpha \cdot \frac{m_t}{\sqrt{v_t} + \epsilon}$$

2. 첫번째 모멘트 추정식(방향 업데이트)

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

3. 두번째 모멘트 추정식(이동 크기 업데이트)

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

AdaMax(ADAM 알고리즘 변형 중 하나)

1. Adam의 업데이트 규칙을 좀 더 단순하고 안정적으로 변형한 버전
2. 필요성: Adam에서 발생할 수 있는 큰 기울기 변동에 대한 불안정성을 줄임
   - 예) 경사가 급한 봉우리 등산(하산)을 할때, 보폭이 크면 움직임이 불안정 해진다.
3. 해결법: 기울기의 최대값(가장 가파른 경사, *L∞ 노름)를 기준으로 파라미터 업데이트 함
   - 예) 경사가 가장 큰 부분을 기준으로 발걸음 조정하면, 어떤 경사를 만나더라도 안정적으로 등산(하산)

* L∞ 노름: 각 파라미터의 업데이트 크기를 기울기의 최대 절대값을 기준으로 조정

Google Scholar

Diederik P. Kingma

FOLLOW

Other names

Research Scientist, Google Brain
Verified email at google.com - Homepage

Machine Learning Deep Learning Neural Networks
Generative Models Artificial Intelligence

TITLE	CITED BY	YEAR
Adam: A method for stochastic optimization DP Kingma, J Ba arXiv preprint arXiv:1412.6980 <확률적 최적화 기법: ADAM>	193697	2014
Auto-Encoding Variational Bayes <변형 베이즈 자동 인코딩> DP Kingma, M Welling arXiv preprint arXiv:1312.6114	38520	2013
Score-based generative modeling through stochastic differential equations <확률적 미분 방정식을 통한 점수기반 생성 모델링> Y Song, J Sohl-Dickstein, DP Kingma, A Kumar, S Ermon, B Poole arXiv preprint arXiv:2011.13456	4247	2020
Semi-Supervised Learning with Deep Generative Models <심층 생성모델 사용 빈지도 학습> DP Kingma, S Mohamed, DJ Rezende, M Welling Advances in Neural Information Processing Systems, 3581-3589	3593	2014

Jimmy Ba <지미 바>

FOLLOW

University of Toronto
Verified email at cs.toronto.edu - Homepage

Neural Networks Artificial Intelligence Machine Learning
Deep Learning

TITLE	CITED BY	YEAR
Adam: A method for stochastic optimization <확률적 최적화 기법> D Kingma, J Ba International Conference on Learning Representations	193209	2015
Show, attend and tell: Neural image caption generation with visual attention <보여주고, 참석하고, 말하기: 시각적 주의를 기울인 신경 이미지 캡션 생성> K Xu, J Ba, R Kiros, K Cho, A Courville, R Salakhudinov, R Zemel, ... International conference on machine learning, 2048-2057	12624	2015
Layer normalization <레이어 정규화> J Ba, JR Kiros, GE Hinton Advances in NIPS 2016 Deep Learning Symposium, arXiv preprint arXiv:1607.06450	12426	2016
Do deep nets really need to be deep? J Ba, R Caruana Advances in neural information processing systems, 2654-2662	2623	2014

“어떻게 하면 대규모 데이터셋과 복잡한 신경망 모델에서 효율적이고 안정적으로 학습할 수 있는 최적화 알고리즘을 개발할 수 있을까?”

대규모 데이터셋과 복잡한 신경망에서 안정적인 학습을 저해하는 요인들

1. **기울기 폭발 및 소실**: 기울기가 너무 커지거나 작아져 학습이 불안정해짐
2. **고정된 학습률**: 모든 파라미터에 동일한 학습률을 적용해 최적화가 어려움
3. ***노이즈**: 데이터의 노이즈로 인해 모델이 안정적으로 수렴하지 못함.
4. **희소한 기울기**: 일부 파라미터에 기울기가 거의 없어 학습이 느려짐
5. **높은 메모리 및 계산 비용**: 대규모 데이터로 인해 학습 속도와 효율성이 떨어짐

***예) 시끄러운 환경에서는 많은 노이즈가 섞여 있어서, 사람들이 말하는 내용을 정확하게 이해하기 어렵다.**
데이터에 노이즈가 많으면, 중요한 패턴을 학습하는 대신, 쓸모없는 정보(노이즈)에 영향 받아 잘못된 방향으로 학습될 수 있다.

<Abstract>

- Adam 알고리즘을 제안한다.
- Adam은 '확률적 목표 함수에 대한 1차 기울기 기반 최적화 알고리즘'이다.
- 적응형 모멘트 추정을 사용해 학습률을 동적으로 조정함
- 대규모 데이터와 희소하거나 노이즈가 많은 문제에도 적합함
- 안정적인 수렴 속도를 보장하고, 실험결과에서도 다른 최적화 방법보다 뛰어난 성능을 입증함.
- **AdaMax**라는 Adam의 변형 알고리즘도 소개함

1.INTRODUCTION

1. SGD(Stochastic Gradient Descent) 기여

- SGD: 확률적 경사하강법 + 손실함수 최소화 기법
- 효율성: 전체 데이터셋 대신 샘플 사용: 계산량 줄임
- 딥러닝 성공에 핵심적 역할 수행
  - ✓ImageNet 분류(Krizhevsky et al., 2012)
  - ✓음성 인식(Hinton et al., 2012a)
  - ✓심층 신경망(Graves et al., 2013)
- 드롭아웃(Hinton et al., 2012b) 등 다른 기술과 결합하여 일반화 성능 향상

2. SGD(Stochastic Gradient Descent) 한계

- 고차원 문제 비효율성: 매개변수 공간이 고차원일 경우 최적화 속도와 안정성이 저하될 수 있음.
- 정적 학습률: 학습률을 고정하거나 단순히 감소시키는 방식은 최적화 과정에서 효과적이지 않을 수 있음
- 비정상(non-stationary) 환경에서의 불안정성: 데이터 분포나 목적 함수가 변화할 경우 성능 저하
- 초기화 변향 문제: 1차 평균 및 2차 분산 모멘트 추정할 때, 초기값 0으로 설정하는 경우 많음, 그러나 초기 단계 충분한 데이터 누적 되지 안하, 실제 값보다 작아지고, 학습률 과소 또는 과대 조정 초래함

3. Adam(Stochastic Gradient Descent) 제안

- 적응형 학습률(adaptive learning rate)
- 초기화 편향 조정
- 효율적 메모리 사용
- 기존의 장점 통합
  - ✓ AdaGrad : 희소 그래디언트 문제에서 강정
  - ✓ RMSProP: 정상 환경에서의 안정성

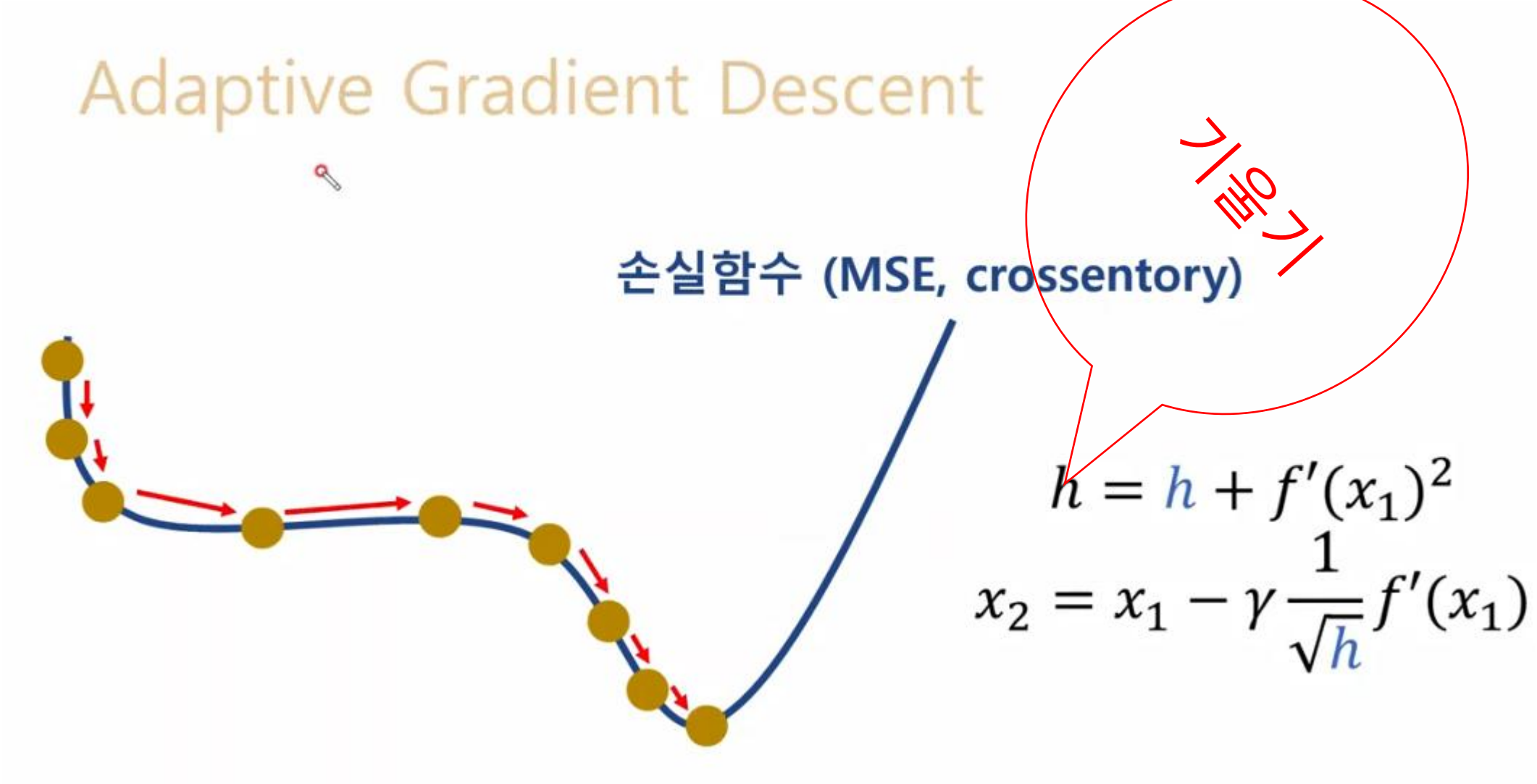
AdaGrad(Adaptive Gradient, 적응적 기울기) 예시

1. 문제: 영화 리뷰 데이터 분석+긍정 또는 부정적인지? 분류
2. Feature(입력데이터의 구성요소) 분석
   - 중요한 단어: "great", "awesome", "terrible“ : 자주등장하므로 학습률 감소
   - 덜 중요한 단어: "the", "and", "is " : 감정을 나타내지 않으므로 업데이트 적음 ….. 학습률 유지 또는 증가

AdaGrad는 각 Feature의 중요도와 학습 패턴에 따라

학습률을 조정하여 모델이 더욱 효율적으로 학습할 수 있도록 돕습니다.

<누적 기울기 고려 방향 결정>



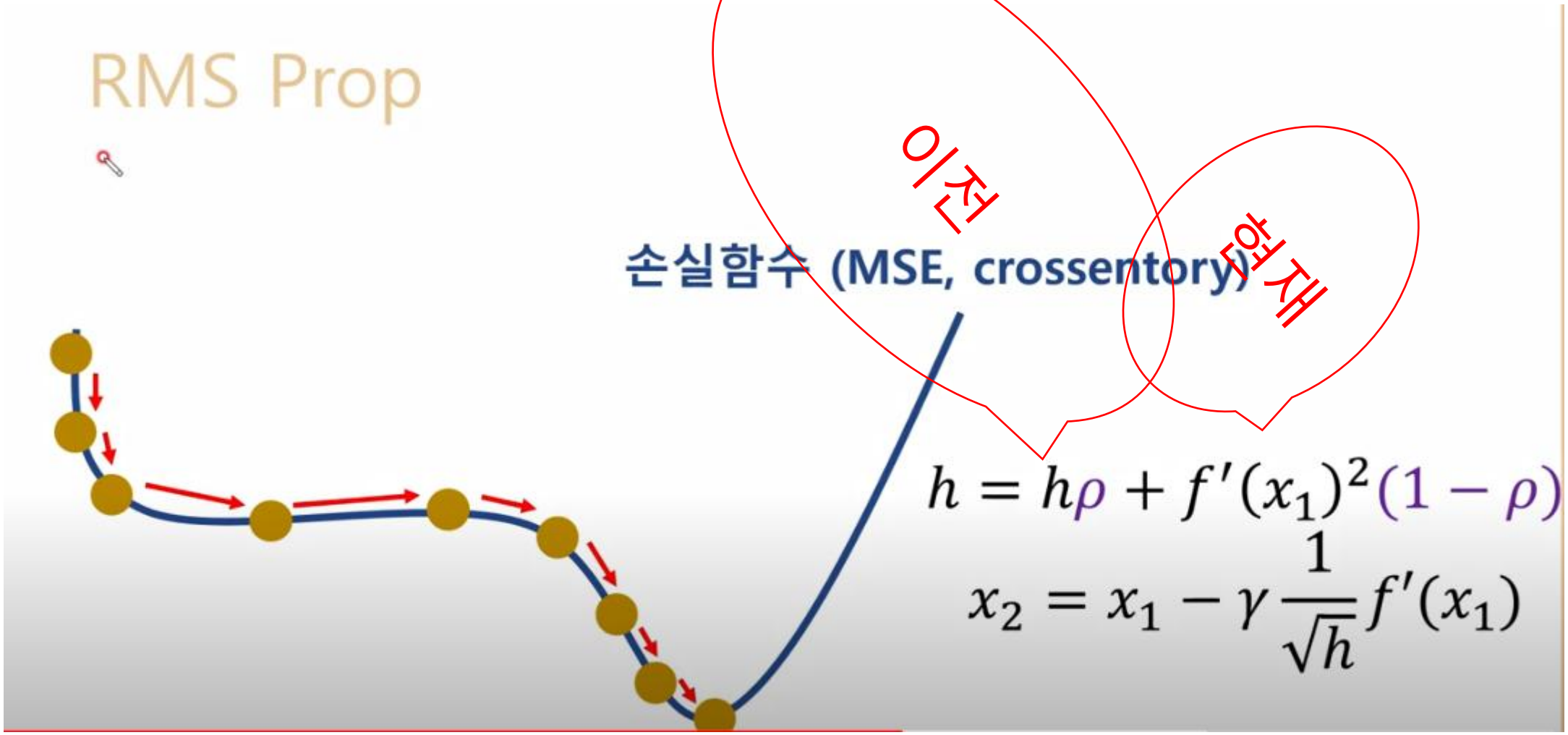
RMSProp(Root Mean Sqaure Propagation)

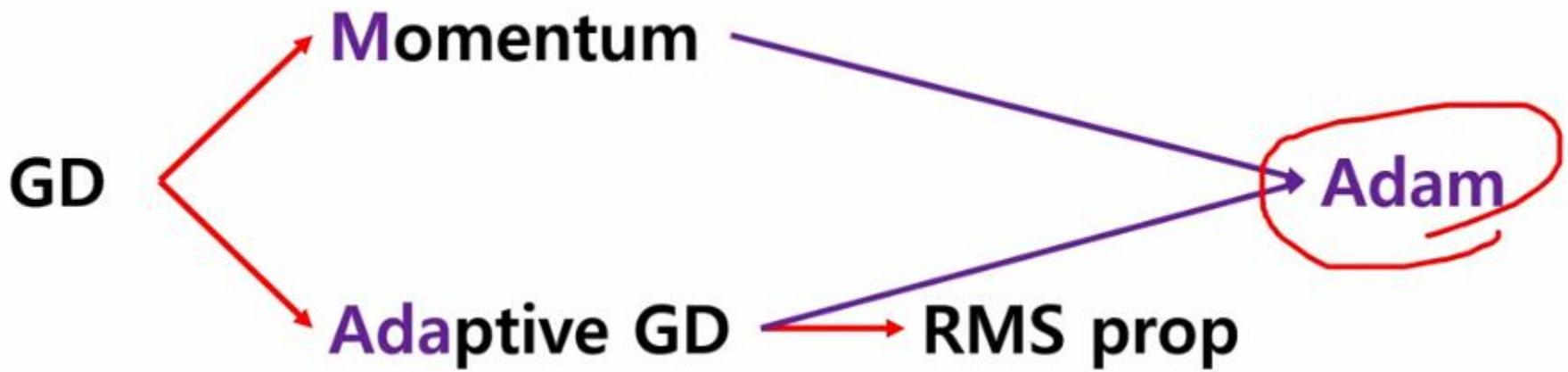
1. AdaGrad 한계: 학습이 진행될 때 학습률이 꾸준히 감소하다 나중에는 0으로 수렴하여 더 이상 학습이 진행되지 않음
2. Featue(변수_별로 학습률을 조정하되 기울기 업데이트 방식에서 차이 있음
3. 지수이동 평균 활용 기울기 업데이트

“가장 최근 time step에서의 기울기는 많이 반영하고

먼 과거의 time step에서의 기울기 조금 반양하는 점이 특징이다.”

<현재 기울기 고려하여 방향 결정>





$$v = \alpha v - \gamma f'(x_1)$$
$$h = h\rho + f'(x_1)^2(1-\rho)$$
$$x_2 = x_1 + \alpha v - \gamma \frac{1}{\sqrt{h}} f'(x_1)$$

2. AlGORITHM

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)

1. 초기 설정

- α: 학습률(Stepsize).
- β_1, β_2: 모멘트 추정을 위한 지수 감쇠율(Exponential decay rates, 일반적으로 $\beta_1 = 0.9$, $\beta_2 = 0.999$).
- ϵ: 수치 안정성을 위한 작은 값(보통 10^{-8}).
- $f(\theta)$: 매개변수 θ를 포함한 확률적 목적 함수.
- **초기화**:
  - $m_0 = 0$: 1차 모멘트 벡터 초기화.
  - $v_0 = 0$: 2차 모멘트 벡터 초기화.
  - $t = 0$: 시간 단계 초기화.

Require: α: Stepsize “필요조건 α 학습률)
Require: β1, β2 ∈ [0, 1): Exponential decay rates for the moment estimates
“β1와 β2: 모멘트 추정을 위한 지수 감쇠율”

- Adam 알고리즘에서 β1와 β2는 각각 **1차 모멘트(그래디언트의 평균)**와 **2차 모멘트(그래디언트 크기의 분산)**를 계산할 때, 과거와 현재 값에 부여하는 **가중치 비율**을 결정함.
- 이 가중치는 지수적으로 감소하며, 이를 "지수 감쇠율 (Exponential Decay Rate)"이라고 부름

2. 반복 수행

- **그래디언트 계산**: 현재 단계 t에서 확률적 목적 함수에 대한 그래디언트 g_t를 계산.
- **모멘트 업데이트**:

1차 모멘트: 어느 방향으로 이동할까?

  - $m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (편향된 1차 모멘트 추정값).
  - $v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (편향된 2차 모멘트 추정값).

2차 모멘트: 얼마 만큼 이동할까?

- **편향 보정**:
  - $\hat{m}_t = m_t / (1 - \beta_1^t)$ (1차 모멘트 보정값). 방향성 보정
  - $\hat{v}_t = v_t / (1 - \beta_2^t)$ (2차 모멘트 보정값). 이동크기 보정
- **매개변수 업데이트**:
  - $\theta_t = \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$.

Adam 알고리즘의 직관적 설명

1. **파라미터 초기화**
   - 모든 매개변수(파라미터) 초기값으로 설정
   - 1차 모멘트 벡터(m0)와 2차 모멘트 벡터(v0)도 0으로 초기화
2. **손실 함수의 그래디언트 계산**
   - 현재 단계 손실함수 그래디언트 계산
   - 방향성과 크기를 결정하는 기본 정보
3. **모멘트 업데이트**
   - 1차 모멘트 업데이트: 그래디언트 평균(방향성) 계산
   - 2차 모멘트 업데이트: 그래디언트 크기의 제곱 평균(변화크기, 이동성) 계산

4. 편향 보정
   - 초기화시 모멘트 값이 0에서 시작됨으로, 초기 단계에서 방향성과 크기가 과소 추정될 수 있음
   - 이를 보정하기 위해, 1모멘트 보정값 계산 $\widehat{m}_t \leftarrow m_t/(1-\beta_1^t)$
   - 2차 모멘트 보정값 계산 $\widehat{v}_t \leftarrow v_t/(1-\beta_2^t)$

5. 매개변수 없데이트
   - 모멘트 보정값을 활용해 매개변수를 어느방향으로, 얼마나 이동할 지 결정

$$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t/(\sqrt{\widehat{v}_t} + \epsilon)$$

2. 1 ADAM'S UPDATE RULE

1. 효과적인 스텝 크기를 신중이 결정한다. 학습률이 너무 크거나 불안정해지는 것을 방지한다.

(예시: 산 정상에 도달할 때, 보폭이 너무 크면 산을 벗어 나거나 더 높은 곳으로 올라가지 못할 위험이 있다. 또한 너누 작게 설정하면 시간이 매우 오래 걸릴 수 있다)

$$\Delta_t = \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

가파른 경사일 수록 이동크기 줄임

- Δ_t: 매개변수의 스텝 크기(얼마나 이동할지).
- α: 학습률(초기 설정 값, 보통 0.001).
- $\hat{m}_t$: 1차 모멘트 보정값 (방향성 정보: 어느 방향으로 이동할지).
- $\hat{v}_t$: 2차 모멘트 보정값 (크기 정보: 이동 크기 조정).
- ϵ: 안정성을 위한 작은 값(0으로 나누는 것을 방지).

기호 해설

기호	한글 독음	의미 및 역할
Δ_t	델타 티	이번 스텝에서 **얼마나 이동할지** (스텝 크기)
α	알파	**학습률** (기본 이동 크기)
$\hat{m}_t$	엠 햇 티 / 모멘트 햇 티	**1차 모멘트 편향 보정값** (이동 방향 정보)
$\hat{v}_t$	브이 햇 티 / 브이 편향 보정값	**2차 모멘트 편향 보정값** (변화 크기 정보)
ϵ	엡실론	**작은 수** (0으로 나누는 오류 방지용)

2. 신뢰영역(Trust region) 설정: 매개변수 공간에서 효과적인 탐색이 가능하도록 설계

- 신뢰 영역은 "현재 위치에서 합리적으로 이동할 수 있는 거리"를 제한
- 경사가 가파르면 신뢰 영역을 좁게 설정하여 작은 걸음으로 이동
- 경사가 완만하면 신뢰 영역을 넓게 설정하여 더 큰 걸음으로 이동

"신뢰 영역 자동설정 "

효과적인 스텝 크기:

$$\Delta_t = \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

- **1차 모멘트 ($\hat{m}_t$)**: 이동할 방향을 결정.
- **2차 모멘트 ($\hat{v}_t$)**: 그래디언트 크기에 따라 스텝 크기를 제한.
- α: 신뢰 영역의 상한선(스텝 크기 상한을 결정하는 학습률).

예시) 신뢰 영역 설정

상황

- 현재 위치: $x = 5$.
- 정상(최적값): $x = 0$.
- 현재 경사: $g_t = 2$ (오른쪽으로 이동해야 함).
- 학습률 $\alpha = 0.1$.

Case 1: 가파른 경사 (크기 큼)

- 그래디언트가 크다면 $\sqrt{\hat{v}_t}$ 값이 커져서 스텝 크기 Δ_t가 작아집니다.
- 신뢰 영역이 좁아지므로, **작은 걸음**으로 신중히 이동:
  - $\Delta_t = 0.05 \rightarrow x = 5 - 0.05 = 4.95.$

Case 2: 완만한 경사 (크기 작음)

- 그래디언트가 작다면 $\sqrt{\hat{v}_t}$ 값이 작아져서 스텝 크기 Δ_t가 커집니다.
- 신뢰 영역이 넓어지므로, **큰 걸음**으로 빠르게 이동:
  - $\Delta_t = 0.2 \rightarrow x = 5 - 0.2 = 4.8.$

효과적인 스텝 크기:

$$\Delta_t = \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

- **1차 모멘트 ($\hat{m}_t$)**: 이동할 방향을 결정.
- **2차 모멘트 ($\hat{v}_t$)**: 그래디언트 크기에 따라 스텝 크기를 제한.
- α: 신뢰 영역의 상한선(스텝 크기 상한을 결정하는 학습률).

경사가 가파르면 ($\hat{v}_t$가 크면):

- $\sqrt{\hat{v}_t}$ 값이 커지므로 Δ_t가 작아집니다.
- 결과적으로 작은 걸음으로 이동 → 안정적 학습.

경사가 완만하면 ($\hat{v}_t$가 작으면):

- $\sqrt{\hat{v}_t}$ 값이 작아지므로 Δ_t가 커집니다.
- 결과적으로 더 큰 걸음으로 이동 → 빠른 학습.

SNR(Signal-to-Noise Ratio, 신호 대 잡음비)의 의미

3. SNR은 최적화 과정에서 불확실성을 반영하고, 최적값에 가까워질수록 자동으로 스텝크기를 줄이는데 기여한다.

- SNR은 최적화 과정에서 유용한 정보(신호)와 불확실성(잡음)의 비율

직관적 예: 산 오르기

1. **목표**: 산 정상에 도달하기 위해 경사를 따라 이동.
2. **SNR의 역할**:
   - **신호(평균, $\hat{m}_t$)**: "이쪽 방향이 산 정상으로 가는 길이야!"라고 알려주는 정보.
   - **잡음(분산, $\sqrt{\hat{v}_t}$)**: "그런데 이 방향이 얼마나 신뢰할 수 있는지 확실하지 않아..."라는 불확실성.
   - **SNR이 크면**: 명확한 방향성을 가지고 정상으로 빠르게 이동.
   - **SNR이 작으면**: 불확실성이 커서 조심스럽게 작은 걸음으로 이동.

$$\mathrm{SNR} = \frac{\hat{m}_t}{\sqrt{\hat{v}_t}}$$

(그래디언트 방향성(평균): 어느방향으로 이동해야 하는가?

그래디언트 크기(분산): 현재 방향의 신뢰도는 얼마인가?

- **SNR이 크다**: 신호가 잡음보다 강함 → 방향성이 명확함 → 더 큰 스텝 크기를 사용

- **SNR이 작다**: 신호가 잡음에 가려져 있음 → 방향성이 불확실함 → 더 작은 스텝 크기를 사용

"Adam은 SNR을 기반으로 스텝 크기를 자동으로 줄이는 메커니즘을 제공"

4. 스텝 크기가 그래디언트 크기에 대해 불변한다.

- Adam 알고리즘에서 스텝 크기(Δt)는 그래디언트의 크기에 영향을 받지 않고 **동일한 스케일**을 유지함
- 즉, 그래디언트 값이 커지거나 작아져도 스텝 크기는 자동으로 조정되어 안정적으로 학습할 수 있음

예시 1) 그래디언트 크기가 큰 경우

- $g = 10$으로 그래디언트가 크다고 가정.
- Adam은 분모 $\sqrt{\hat{v}_t}$를 증가시켜 스텝 크기를 제한:

$$\Delta_t = \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

  - 큰 그래디언트 g는 $\hat{v}_t$를 증가시키므로, Δ_t는 안정적으로 유지.
  - 결과: 가파른 경사에서도 지나치게 멀리 이동하지 않음.

예시 2) 그래디언트 크기가 작은 경우

- $g = 0.1$로 그래디언트가 작다고 가정.
- Adam은 분모 $\sqrt{\hat{v}_t}$를 감소시켜 스텝 크기를 적절히 유지:
  - 작은 그래디언트 g는 $\hat{v}_t$를 감소시키므로, Δ_t는 안정적으로 유지.
  - 결과: 완만한 경사에서도 너무 느리게 이동하지 않음.

요약하면,

- Adam은 경사가 가파르든 완만하든 **스텝 크기를 적절히 조정**하여,
- 너무 크게 이동하거나 너무 작게 이동하지 않도록 설계됨
- 이는 사람이 산을 오를 때 경사의 가파름에 상관없이 일정한 걸음 크기로 이동하는 것과 비슷한 방식임
- 이를 통해 학습 과정에서 안정성을 높이고, 효율적인 최적화를 보장함

3. INITIALIZATION BIAS CORRECTION (초기화 편향 보정)

파라미터 업데이트 식

$$\theta_t = \theta_{t-1} - \alpha \cdot \frac{m_t}{\sqrt{v_t} + \epsilon}$$

- θ_t: 업데이트된 파라미터.
- α: 학습률.
- v_t: 두 번째 모멘트 추정값(기울기 제곱의 지수 이동 평균).
- ϵ: 매우 작은 값으로, 분모가 0이 되는 것을 방지.

시작적 이해

- 파라미터 축
- <----------|----------->
- 감소 방향 (m_t > 0) 현재 값 증가 방향 (m_t < 0)

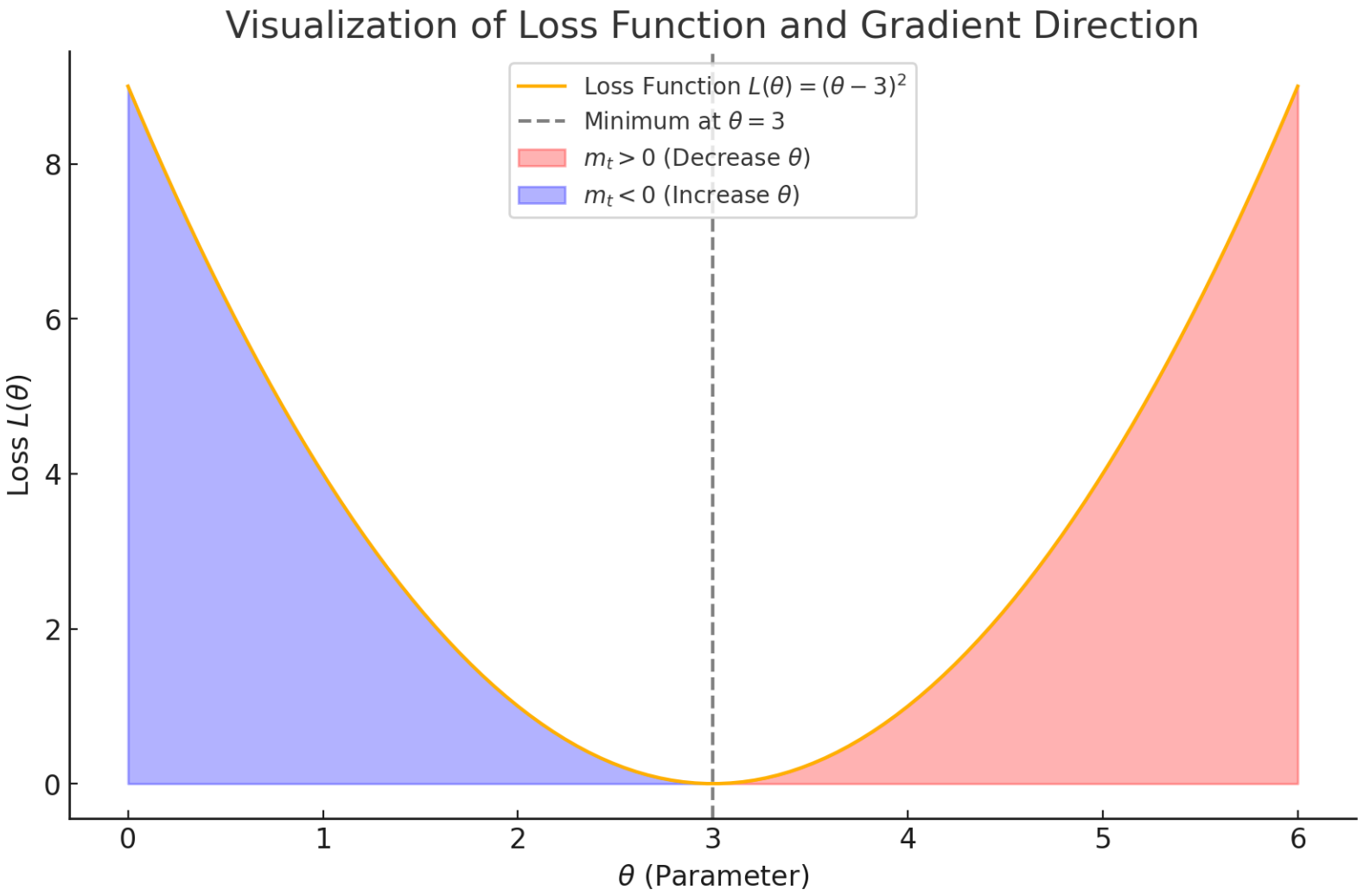
  - $m_t > 0$: 왼쪽(감소 방향)으로 이동.
  - $m_t < 0$: 오른쪽(증가 방향)으로 이동.

Adam 옵티마이저는,

1. 기계학습에서 널리 사용되는 최적화 알고리즘이다.
2. 기울기의 1차 모멘트(기울기 평균)과 2차 모멘트(기울기 분산)을 사용하여 학습률을 조정한다.
3. 첫 번째 모멘트 추정값 m_t 는 기울기의 지수 이동 평균으로 모델 파라미터를 업데이트할 방향을 결정한다.
4. m_t 의 부호에 따른 이동 방향

   - $m_t > 0$ (양수): 파라미터를 **감소**시키는 방향으로 이동
   - $m_t < 0$ (음수): 파라미터를 **증가**시키는 방향으로 이동

(예시) 손실 함수 $L(\theta) = (\theta - 3)^2$와 파라미터 업데이트 방향



1. **노란 곡선**: 손실 함수 $L(\theta)$
   - $\theta = 3$에서 최소값을 가짐.
2. **회색 점선**: 최적 파라미터 ($\theta = 3$)
   - 손실 함수의 최소점.
3. **파란 영역 ($m_t < 0$)**:
   - 조건: $\theta < 3$, 기울기 음수.
   - 파라미터를 **증가**시켜야 손실 감소.
4. **빨간 영역 ($m_t > 0$)**:
   - 조건: $\theta > 3$, 기울기 양수.
   - 파라미터를 **감소**시켜야 손실 감소.

1. 첫번째 모멘트 설명: "이동방향 결정"

수식:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

주요 구성:

- m_t: 현재 시점 t에서의 첫 번째 모멘트 추정값.
- m_{t-1}: 이전 시점에서 계산된 첫 번째 모멘트 추정값.
- g_t: 현재 시점의 기울기.
- β_1: 과거 데이터를 얼마나 신뢰할지를 결정하는 가중치 (보통 0.9로 설정).
- $(1 - \beta_1)$: 현재 기울기 g_t에 부여되는 가중치.

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

 # 과거까지의 평균값에 가중치 β_1 을 곱하고, 지금 새로 들어온 값어 $(1 - \beta_1)$ 을 곱하는 이유?

📌 이유 1: 평균의 스케일 유지

- 만약 두 가중치의 합이 **1이 아니라면**,
  → 결과값이 **원래 값보다 크거나 작아지는 왜곡**이 생깁니다.
- 예를 들어 $\beta = 0.6, 1 - \beta = 0.5$이면
  → 총합이 1.1이므로, 평균값이 **실제보다 커집니다.**

📌 이유 2: 균형 잡힌 반영

- β와 $1 - \beta$는 각각 **얼마나 과거를 신뢰할지**와
  얼마나 새 정보를 반영할지를 결정하는 가중치입니다.
- 합이 1이면
  → 전체 평균이 항상 일정한 기준을 중심으로 계산됩니다.
  → **새 값과 과거 값의 힘겨루기가 공정하게 일어납니다.**

✅ 요약:

- $\beta + (1 - \beta) = 1$이 되게 하는 이유는
  평균값의 왜곡을 막고,
  과거와 현재의 기여도를 공정하게 나누기 위해서입니다.

1번 모멘트 요약하면,

1. 첫번째 모멘트 추정값 m_t는 기울기 지수 이동 평균으로, 이동 방향을 결정한다.
2. 이 값이 양수(+)인지 음수(-)인지에 따라 이동 방향이 달라진다.

파라미터 업데이트 관련 식

1. 파라미터 업데이트 식 $$\theta_t = \theta_{t-1} - \alpha \cdot \frac{m_t}{\sqrt{v_t} + \epsilon}$$

2. 첫번째 모멘트 추정식(방향 업데이트) $$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

3. 두번째 모멘트 추정식(이동 크기 업데이트)

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

2. 두번째 모멘트

"안정적으로 학습하기 위해서는 기울기(gradient) 값의 크기를 어떻게 반영할까?"

<가중치>

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

기울기 크기의 변화를 축적한 값
- 학습률 조정위해 사용됨
- 현시점 t에서의 기울기 제곱+이전 시점까지의 평균

기울기 벡터 각요소 제곱값

- v_t: 현재 시점 t에서의 "평균적인 값" (여기서는 기울기의 제곱의 지수 이동 평균).
- β_2: 과거 값 v_{t-1}에 얼마나 의존할지를 결정하는 **가중치** (보통 0.999처럼 과거에 높은 가중치를 부여함).
- $(1 - \beta_2)$: 새로운 값 g_t^2에 부여되는 가중치.

v_t는 새로운 데이터와 과거 데이터를 점진적으로 반영하며 평균값을 추정

<예시>

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

지수 이동 평균의 날씨 예측 설명

- v_t: 오늘 날씨의 "평균 온도 추정치".
- v_{t-1}: 어제까지의 평균 온도 데이터.
- g_t^2: 오늘 관측된 온도 데이터(제곱된 값).
- β_2: 과거 데이터를 얼마나 신뢰할지 결정하는 가중치 비율.

1. β_2 **값이 크면** (예: 0.999):
   - 과거 데이터를 더 많이 반영.
   - 오늘 데이터를 조금만 반영.
   - **적용 사례:** 과거 패턴이 중요한 경우.
2. β_2 **값이 작으면** (예: 0.9):
   - 새로운 데이터인 g_t^2가 더 큰 영향을 미침.
   - **적용 사례:** 환경이 자주 변하는 경우.

"두번째 모멘트, 즉 기울기의 크기를 어떻게 결정할까?"

지수 이동 평균(Exponential Moving Average, EMA)

- **최근 값**이 더 중요하다고 생각해 평균을 구하는 방법
- <예시> 달리기 기록 "내 달리기 시간은 얼마나 걸릴까?"
  - ✓ 첫째날 : 10초, 둘째날: 12초, 셋째 날: 11초, 넷째 날: 9초
  - ✓ 평균:(10+12+11+9)/4 = 10.5초
  - ✓ 지수이동 평균 >>> 최근기록에 더 많은 가중치를 둔다.

<논문에서>

* 모든 과거 기울기 제곱값을 사용하여 두번째 모멘트 계산

$$v_t = (1 - \beta_2) \sum_{i=1}^{t} \beta_2^{t-i} \cdot g_i^2 \quad \text{<본 논문 이전에 사용한 방법>} \tag{1}$$

We wish to know how $\mathbb{E}[v_t]$, the expected value of the exponential moving average at timestep t, relates to the true second moment $\mathbb{E}[g_t^2]$, so we can correct for the discrepancy between the two. Taking expectations of the left-hand and right-hand sides of eq. (1):

$$\mathbb{E}[v_t] = \mathbb{E}\left[(1 - \beta_2) \sum_{i=1}^{t} \beta_2^{t-i} \cdot g_i^2\right] \quad \text{두번째 모멘트 기대값} \tag{2}$$

* 이론적 기대값 + 편향으로 발생하는 오차항
* 초기 이동평균 = 0으로 시작하기 때문에 발생하는 오차항

$$= \mathbb{E}[g_t^2] \cdot (1 - \beta_2) \sum_{i=1}^{t} \beta_2^{t-i} + \zeta \tag{3}$$

$$= \mathbb{E}[g_t^2] \cdot (1 - \beta_2^t) + \zeta \quad \text{(기대값 간소화 식)} \tag{4}$$

where $\zeta = 0$ if the true second moment $\mathbb{E}[g_i^2]$ is stationary; otherwise ζ can be kept small since the exponential decay rate β_1 can (and should) be chosen such that the exponential moving average assigns small weights to gradients too far in the past. What is left is the term $(1 - \beta_2^t)$ which is caused by initializing the running average with zeros. In algorithm 1 we therefore divide by this term to correct the initialization bias.

In case of sparse gradients, for a reliable estimate of the second moment one needs to average over many gradients by chosing a small value of β_2; however it is exactly this case of small β_2 where a lack of initialisation bias correction would lead to initial steps that are much larger.

1. 왜 기대값의 개념을 도입하여 편향보정을 설명하는가?

- 처음에는 **계산이 부족**해서 v_t 값(기울기의 크기 평균)이 정확하지 않음
- 우리가 알고 싶은 것은 진짜 평균(기대값)이다.
- 기대값을 사용하여, 초기 평균을 0으로 설정한 것을 보정해서 더 정확하게 학습하도록 돕는다.

2. 왜 초기 평균이 0으로 설정될까?

- 첫번째 모멘트와 두번째 모멘트 초기값을 0으로 설정하고 시작하기 때문이다.(초기화 편향 문제 발생)
- 초기에 아직 학습된 정보가 없기 때문이다.

3. 보정하면 어떤 효과가 있는가?

- 초기값이 0으로 시작되어 발생하는 과소평가를 기대값으로 보정하여, 첫 번째와 두 번째 모멘트가 **실제 기울기와 크기를 더 정확히 반영한다.**
- **초기 편향이 줄어들면서, 학습이 더 빠르고 안정적으로 진행된다.**
- **특히, 기울기가 작은 경우(희소한 데이터)에서 안정적인 학습률을 제공한다.**
- **그리고, 장기적으로 학습 최적화가 일어난다.**
- **즉, 시간이 지남에 따라 실제 기울기 평균과 분산을 점점 더 정확히 추적한다.**

4. CONVERGENCE ANALYSIS (수렴 분석)

1. 수렴분석이란?

- 최적화 알고리즘이 학습과정에서 최적의 값으로 수렴하는지? 수학적으로 증명하는 정량적 분석 과정
- 손실함수가 최소화될 수 있는지? 최소값에 수렴하는지? 수학적으로 증명하는 과정
- 수렴 분석의 목적
  - ✓ 손실 함수의 최소값에 도달하는가?
  - ✓ 최소값 근처에서 안정적으로 멈추는가?

2. 수렴 분석의 핵심 질문

① 손실 함수의 값이 점점 줄어드는가?

② 기울기가 0에 가까워지는가?

③ 최소값 근처에서 발산하지 않고 멈추는가?

3. 본 논문에서 사용된 수렴 분석 방법은?

1) Regret Analysis(후회값 분석): 알고리즘을 사용했을 때, 최적의 방법과 비교해서 얼마나 더 손해를 보았는가? 측정하는 개념(손해값 측정도구)
2) Regret Bound (Regret 경계): 후회값 분석의 결과를 바탕으로, 후회값이 시간에 따라 제한된 범위에서 증가하거나, 감소한다는 것을 수학적으로 보장함.(양적 범위, 수학적 상한)
3) Convex Optimization (볼록 최적화): 종모양의 볼록함수의 최소값을 찾는 과정

Regret Analysis(후회값 분석) 알고리즘

- 후회(Regret)? “아, 다른 선택을 했으면 더 좋았을 텐데!” 하는 마음

- “잘못된 선택을 하고 난 다음에 ‘더 좋은 선택을 하면 좋았를 텐데…’ 라고 후회 하듯이 시간이 지날수록 더 좋은 선택을 배우는 알고리즘”

- 후회 경계(Regret Bound)란? “후회의 한계를 정하는 기준선”이다. 즉, 후회가 너무 커지지 않도록 정해놓은 기준선

“후회 기준선을 줄이는 방향이 알고리즘의 학습 방향이다 "

- 처음에는 후회가 클 수도 있지만, 점점 후회 기준선(Regret Bound)이 낮아짐
- 좋은 알고리즘일수록 후회 경계를 빠르게 낮추는 방향으로 학습하며,

- 결국, 더 똑똑해지고, 더 좋은 결정을 내리는 것이다.

- 즉, 좋은 알고리즘일수록 후회를 빨리 줄이고, 더 좋은 선택을 배운다.

‘후회(Regret)’를 어떻게 계산할까?

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

(실제선택-최고선택)

- 가장좋은 선택과 우리가 실제로 선택한 것의 차이만큼 후회를 한다.
- 그렇다면, (가장좋은 선택)-(실제선택)의 차이를 모두더하면, 후회를 계산할 수 있음

그렇다면, 후회의 기준선을 어떻게 낮출까?

"논문에서는 아래와 같이 설명하고 있다 " "에고 복잡해라....."

Theorem 4.1. *Assume that the function f_t has bounded gradients, $\|\nabla f_t(\theta)\|_2 \leq G$, $\|\nabla f_t(\theta)\|_\infty \leq G_\infty$ for all $\theta \in R^d$ and distance between any θ_t generated by Adam is bounded, $\|\theta_n - \theta_m\|_2 \leq D$, $\|\theta_m - \theta_n\|_\infty \leq D_\infty$ for any $m, n \in \{1, ..., T\}$, and β_1, $\beta_2 \in [0, 1)$ satisfy $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$. Let $\alpha_t = \frac{\alpha}{\sqrt{t}}$ and $\beta_{1,t} = \beta_1 \lambda^{t-1}$, $\lambda \in (0, 1)$. Adam achieves the following guarantee, for all $T \geq 1$.*

$$R(T) \leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T\hat{v}_{T,i}} + \frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 + \sum_{i=1}^{d} \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}$$

Our Theorem 4.1 implies when the data features are sparse and bounded gradients, the summation term can be much smaller than its upper bound $\sum_{i=1}^{d} \|g_{1:T,i}\|_2 << dG_\infty\sqrt{T}$ and $\sum_{i=1}^{d} \sqrt{T\hat{v}_{T,i}} << dG_\infty\sqrt{T}$, in particular if the class of function and data features are in the form of section 1.2 in (Duchi et al., 2011). Their results for the expected value $\mathbb{E}[\sum_{i=1}^{d} \|g_{1:T,i}\|_2]$ also apply to Adam. In particular, the adaptive method, such as Adam and Adagrad, can achieve $O(\log d\sqrt{T})$, an improvement over $O(\sqrt{dT})$ for the non-adaptive method. Decaying $\beta_{1,t}$ towards zero is important in our theoretical analysis and also matches previous empirical findings, e.g. (Sutskever et al., 2013) suggests reducing the momentum coefficient in the end of training can improve convergence.

Finally, we can show the average regret of Adam converges,

$$R(T) \leq \textcircled{1} + \textcircled{2} + \textcircled{3}$$

"후회값 R(T)가 특정한 상한선 이하로 제한된다"는 의미

$$R(T) \leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T\widehat{v}_{T,i}} + \frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2} \sum_{i=1}^{d} \|g_{1:T,i}\|_2 + \sum_{i=1}^{d} \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}$$

$$R(T) \leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T\hat{v}_{T,i}} + \frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2} \sum_{i=1}^{d} ||g_{1:T,i}||_2 + \sum_{i=1}^{d} \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}$$

1. 3개의 공식이 더해진 형태로 이루어짐.
2. Adam이 학습할 때, 후회를 어떻게 줄여가는지를 수학적으로 나타낸 것
3. 첫번째 공식: (학습초반)처음에는 실수가 많지만 점점 줄어듦
4. 두번째 공식: (학습중반)학습이 진행될 수록 더 나은 선택을 하게 됨
5. 세번째 공식: (최종)결국, 후회가 거의 0에 가까워짐 "최적화 완료"

(학습초반)

$$R(T) \leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T\hat{v}_{T,i}}$$

실제발생 후회

후회의 최대경계선

- 학습 초반에 후회의 기준선을 정하는 과정
- 실제 발생한 후회는 후회 기준선보다 적어야 한다.
- 즉, 이와 같은 방향으로 모델은 학습해야 한다

우측 항 설명

- 경사하강법: 기울기를 따라 내려가면서 후회를 줄이는 과정
- 기울기가 크면: 속도를 줄이고(학습률 감소)
- 기울기가 작으면: 속도를 올려서(학습률 증가)
- 후회를 줄이는 방향으로 학습하는 것

$$\frac{D^2}{2\alpha(1-\beta_1)}\sum_{i=1}^{d}\sqrt{T\hat{v}_{T,i}}$$

✔ **기울기(Gradient)** ∇f → 모델이 학습하는 방향을 결정하는 값

✔ **경사하강법(Gradient Descent)** → 기울기를 따라 가장 낮은 값(최적해)로 이동하는 방법

✔ $\hat{v}_{T,i}$ **(기울기 제곱 평균)** → 기울기의 크기를 반영하여 학습률을 조절

✔ $\sum_{i=1}^{d}$ **(합산 기호)** → 모든 변수(파라미터)에 대해 학습률을 다르게 적용

👉 **즉, 기울기를 따라 내려가면서 학습하지만, 너무 빠르거나 느리게 가지 않도록 조정하는 과정이 포함됨!**

(학습 중반)

"어떻게 후회를 줄이고, 안정적으로 학습하는가?"

공식 분석

$$\frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2}$$

- α → 학습률 (이 값이 크면 빠르게 학습, 작으면 천천히 학습)
- β_1 → 모멘텀 계수 (이전 기울기의 영향을 반영)
- G_∞ → 기울기의 최대 크기 (큰 기울기일수록 변화가 큼)
- β_2 → 2차 모멘텀 계수 (기울기 변화량을 반영)
- γ → 조정 계수 (학습 안정성을 조절)

📌 공식의 분자와 분모 분석 & 전체 의미 🎯

공식:

$$\frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2}$$

- ✔ **분자(위쪽): 학습 속도를 조절하는 요소들**
- ✔ **분모(아래쪽): 학습의 안정성을 조절하는 요소들**
- ✔ **전체 의미: 학습 속도를 조절하면서도 안정성을 유지하는 균형 잡힌 학습 과정**

"속도는 내되, 너무 흔들리지 않게!"

즉, **빠르게 배우면서도 안정성을 유지하는 균형 잡힌 학습**을 목표로 한다는 뜻

분자/분모 관계

- 분자가 크면? 현재 방향을 더 신뢰하고 빠르게 이동
- 분모가 크면? 방향을 신중하게 조절하며 이동

"Adam은 방향을 유지하면서도, 안정적으로 학습할 수 있도록 분자와 분모를 조절함으로써, 최적의 학습속도를 유지하는 역할을 수행한다 "

분자 설명

📌 정리: 분자의 $(1 + \beta_1)$ 의미

- ✔ **현재 기울기를 1로 기준**
- ✔ **과거 기울기(모멘텀, β_1) 를 추가하여 현재 방향을 보정**
- ✔ **즉, "과거 방향을 일부 유지하면서 현재 기울기를 반영해 더 부드럽게 학습!"**

$$\alpha(1+\beta_1)G_\infty$$

- ✔ α **(학습률)** → 학습 속도를 조절
- ✔ $(1+\beta_1)$ → 현재 기울기(1) + 과거의 기울기(모멘텀 β_1) 반영
- ✔ G_∞ → 기울기의 크기(변화량)

✅ **즉, 현재와 과거 기울기의 영향을 모두 고려하여 학습률을 동적으로 조정하는 역할을 함!** 🚀

분모 설명 '분모가 크면 학습속도 늦음 '

$$(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2$$

- ✔ $(1-\beta_1)$ 증가 → 학습 속도를 줄여서 더 신중하게 조절
- ✔ $\sqrt{1-\beta_2}$ 증가 → 기울기의 변화를 부드럽게 반영하여 학습 속도가 느려짐
- ✔ $(1-\gamma)^2$ 증가 → 안정성을 높여서 학습 속도를 조절

✅ 즉, 분모가 크면 학습이 느려지고 더 안정적으로 진행됨!

$$(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2$$

- (현재기울기-1차 모멘텀(과거기울기): 급격한 속도 변화 방지
- (현재 기울기 변화량-과거기울기 변화량): 기울기의 변화량을 고려하여 학습 속도를 조절함
- (현재의 조정계수-과거의 조정계수): 학습의 안정성을 조절하는 값도, 너무 급격하게 조정이 되지 않도록 함

"급격한 학습 속도를 조절하여 학습의 안정을 도모하기 위해 기울기(Gradient), 기울기의 변화량(2차 모멘텀), 조정 계수(γ)를 고려한다."

즉, **Adam 알고리즘은 학습이 너무 튀거나 불안정하지 않도록, 학습 속도를 부드럽게 조절하는 요소들을 포함하고 있다.**

전체 공식

$$\frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2}\sum_{i=1}^{d}\|g_{1:T,i}\|_2$$

< 전체학습 과정에서 기울기 크기의 합>
즉, 학습 동안 기울기 변화를 추적하여 얼마나 많은 변화가 있었는지 측정하는 요소

" 학습의 과정에서 기울기의 크기변화(오른쪽 항)를 고려하여, 학습속도를 조절하고(왼쪽항), 안정적으로 최적화하는 방법을 설명하는 식 "

(학습 후반)

**"학습 후반부 기울기 변화가 줄어들때,
최적화 과정을 안정적으로 진행 보장 역할"**

$$\sum_{i=1}^{d} \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}$$

- ✔ D_∞^2 → 초기 파라미터 거리(변화 가능 범위)
- ✔ G_∞ → 최대 기울기 크기
- ✔ $\sqrt{1-\beta_2}$ → 2차 모멘텀(기울기 변화량 조절)
- ✔ α → 학습률 (너무 크면 불안정, 작으면 느려짐)
- ✔ $1-\beta_1$ → 1차 모멘텀 (과거 기울기 영향 조절)
- ✔ $(1-\lambda)^2$ → 학습 안정성을 조절하는 계수

✅ **즉, 학습 후반부에서는 기울기 변화량과 학습 속도를 조절하여 안정적으로 최적 값을 찾을 수 있도록 보장하는 역할!** 🚀

📌 1 분자의 의미 (최대 변화량, 학습 진행도) 🚀

$$D_{\infty}^{2} G_{\infty} \sqrt{1-\beta_{2}}$$

- ✔ D_{∞}^{2} → 초기 파라미터 거리(최대 변화 가능 범위)
- ✔ G_{∞} → 최대 기울기 크기 (기울기가 클수록 변화량 증가)
- ✔ $\sqrt{1-\beta_{2}}$ → 2차 모멘텀 보정 (기울기 변화량 반영)

✅ **즉, 분자는 "학습을 통해 얼마나 큰 변화가 가능한지"를 나타냅니다.**

 # 해설) 분자는 초기 파라미터 거리(학습 가능한 최대 변화 범위), 기울기(Gradient, 변화 방향), 그리고 2차 모멘텀 보정(기울기 변화량을 안정적으로 조절)을 반영하여, "학습을 통해 모델이 얼마나 크게 변화할 수 있는지를 결정한다"

📌 2 분모의 의미 (학습 속도 & 안정성 조절) 🔒

$$2\alpha(1-\beta_1)(1-\lambda)^2$$

- ✔ $2\alpha \rightarrow$ 학습률(너무 크면 불안정, 작으면 느림)
- ✔ $(1-\beta_1) \rightarrow$ 1차 모멘텀 (과거 기울기의 영향 반영)
- ✔ $(1-\lambda)^2 \rightarrow$ 추가적인 안정성 조절 요소

✅ **즉, 분모는 학습 속도를 조절하여 너무 급격한 변화를 방지하고, 안정성을 높이는 역할을 합니다!**

 # 해설) 분모는 학습률(α), 1차 모멘텀(과거 기울기의 영향), 그리고 추가적인 안정성 요소를 반영하여 학습 속도를 조절하고, 급격한 변화 없이 안정적으로 최적화가 진행되도록 조정하는 역할을 한다."

$$R(T) \leq \underbrace{\frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{T\hat{v}_{T,i}}}_{\text{(학습 초반)}} + \underbrace{\frac{\alpha(1+\beta_1)G_\infty}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2} \sum_{i=1}^{d} \|g_{1:T,i}\|_2}_{\text{(학습 중반)}} + \underbrace{\sum_{i=1}^{d} \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}}_{\text{(학습 후반)}}$$

예시) 마라톤 페이스 조절

1 초반:

- 에너지가 많아서 빠르게 달릴 수 있음
- 학습률(α)이 비교적 커서 빠르게 최적화

2 중반:

- 적절한 속도로 페이스 유지
- 모멘텀(β1, β2)을 조절하여 균형 유지

3 후반:

- 너무 빨리 달리면 지칠 수 있으므로 속도를 안정적으로 조절
- $(1-\lambda)^2$ **가 속도를 줄여주어 안전한 완주를 돕는 역할!**

“Adam과 유사한 최적화 기법: RMSProp, AdaGrad

5. RELATED WORK (관련 연구)

📌 RMSProp vs. Adam – 핵심 차이점

🚀 알고리즘	🤖 업데이트 방식	🔍 Bias Correction(편향 보정)	📈 모멘텀 사용 방식
RMSProp	기울기의 2차 모멘텀(평균 제곱)만 사용하여 학습률 조정	❌ 없음 → β_2 값이 1에 가까우면 학습이 불안정할 수 있음	🔹 모멘텀을 추가할 수도 있지만 필수는 아님
Adam	1차 모멘텀(기울기 평균)과 2차 모멘텀(기울기 제곱 평균)을 함께 사용하여 학습률 조정	✅ 있음 → 편향을 보정하여 학습 안정성 증가	🔹 항상 모멘텀을 사용하여 부드러운 업데이트 진행

<RMSProp & Adam>

마라톤 코치가 선수를 훈련시키는 방식으로 비교

🟢 RMSProp 코치 🏃

- ✔ 선수의 **최근 페이스(기울기 변화량)만 반영**하여 속도를 조절
- ✔ **Bias Correction(편향 보정)이 없어** 너무 빨리 뛰면 지칠 가능성이 높음
- ✔ 예를 들면?

- 선수: "코치, 요즘 컨디션이 좋아서 더 빠르게 뛸 수 있을 것 같아요!"
- RMSProp 코치: "좋아! 최근 기록만 보고 바로 속도를 조절하자!"
- ⚠️ 하지만 초반에 너무 빨리 뛰면 후반에 지칠 수 있음

🔵 Adam 코치 🏃

- ✔ **과거 페이스(1차 모멘텀)와 현재 페이스(2차 모멘텀) 둘 다 고려하여 속도를 조절**
- ✔ **Bias Correction(편향 보정)을 적용하여 안정적인 훈련 진행**
- ✔ 예를 들면?

- 선수: "코치, 요즘 컨디션이 좋아요!"
- Adam 코치: "좋아! 하지만 과거 기록도 함께 고려해서 천천히 속도를 올리자!"
- ✅ 안정적으로 속도를 조절하여 마지막까지 균형 있게 달릴 수 있음

📌 AdaGrad vs. Adam – 핵심 차이점

🚀 알고리즘	🤖 학습률 조정 방식	🔍 Bias Correction(편향 보정)	📈 모멘텀 사용 방식
AdaGrad	모든 기울기의 제곱합을 누적하여 학습률 조정 → 학습이 진행될수록 학습률 감소	❌ 없음 → 학습률이 너무 빠르게 감소할 수 있음	❌ 없음 (기본적으로 모멘텀을 사용하지 않음)
Adam	1차 모멘텀(기울기 평균)과 2차 모멘텀(기울기 제곱 평균)을 함께 사용하여 학습률 조정	✅ 있음 → 학습이 후반까지 안정적으로 진행됨	🔹 항상 모멘텀을 사용하여 부드러운 업데이트 진행

<AdaGrad & Adam>

마라톤 코치가 선수를 훈련시키는 방식으로 비교

🟢 AdaGrad 공부법 📚

- ✔ 처음에는 **모든 과목을 빠르게 공부**
- ✔ 하지만 **공부한 내용이 쌓일수록 새로운 내용을 학습하는 속도가 느려짐**
- ✔ 예를 들면?

- 학생: "시험이 얼마 남지 않았어! 모든 과목을 다 공부해야 해!"
- 📖 **초반에는 속도가 빠르지만, 시간이 지나면서 학습 속도가 줄어듦**
- ⚠️ 결국 후반부에는 새로운 내용을 거의 학습하지 못하고 복습만 하게 됨

🔵 Adam 공부법 📚

- ✔ **새로운 내용을 배울 때, 기존의 학습한 내용을 고려하면서 적절한 속도로 조정**
- ✔ **Bias Correction(편향 보정)을 적용하여 학습 속도를 균형 있게 유지**
- ✔ 예를 들면?

- 학생: "시험 준비를 잘해야겠어! 적절한 페이스로 공부하자!"
- 📖 **새로운 내용을 배울 때 기존 내용을 복습하면서도 적절한 학습 속도를 유지**
- ✅ **마지막까지 균형 있는 학습이 가능!**

📌 결론: AdaGrad vs. Adam, 언제 사용하면 좋을까?

✔ **AdaGrad → 희소한 기울기(Sparse Gradient) 문제를 해결할 때 유리** (예: 자연어 처리, 추천 시스템)

✔ **Adam → 전반적으로 안정적이고 빠른 학습이 필요할 때 유리** (예: 이미지 인식, 강화 학습)

✅ **즉, AdaGrad는 초반 학습이 빠르지만 후반에 속도가 줄어드는 반면, Adam은 초반부터 후반까지 균형 잡힌 학습이 가능하도록 만든 알고리즘!** 🚀🎯

"희소한 기울기 문제란?
: 모델이 학습해야 할 정보(Gradient)가 부족하여 학습이 원할하지 않은 상태 "

📌 학습할 정보(Gradient)가 부족할 때, 왜 AdaGrad가 좋은가? 🎯

✅ AdaGrad는 희소한 기울기(Sparse Gradient) 문제를 해결하기 위해 "자주 등장하는 특징은 학습률을 낮추고, 희귀한(거의 등장하지 않는) 특징은 학습률을 높이는 기법"을 사용합니다.

✅ 즉, "자주 등장하는 정보에 대한 학습을 줄이고, 부족한 정보(학습이 거의 안 되는 부분)에는 학습을 집중하도록 조절하는 방식!"

“Adam 알고리즘이 다른 최적화 알고리즘(RMSProp, AdaGrad 등)과 비교하여 얼마나 효과적인지 실험을 통해 평가”

6. EXPERIMENTS (실험 결과)

실험 결과

1. Adam은 초반 학습 속도가 빠르고 안정적
   - 초기 학습: **Adam이 RMSProp, AdaGrad보다 빠르게 손실을 줄임**
   - 특히 **기울기가 희소(Sparse Gradient)한 데이터에서도 우수한 성능**
2. 학습 후반부에도 안정적인 수렴
   - AdaGrad는 학습률이 너무 작아져 학습이 느려짐
   - RMSProp은 Bias Correction이 없어 일부 상황에서 발산하는 문제 발생
   - Adam은 끝까지 안정적으로 학습이 진행됨
3. 일반화 성능(테스트 성능)이 더 우수
   - Adam은 **훈련 데이터 + 테스트 데이터에서도 좋은 성능**을 보임
   - 과적합(Overfitting) 위험이 낮고, 더 일반적인 최적화 기법으로 적합

* 과적합: 모델이 훈련 데이터에는 **너무 잘 맞지만**, 새로운 데이터(테스트 데이터)에서는 **성능이 나빠지는 현상**

<참고>

"공부 못하는 예) 시험에 안나오는 불필요한 것 학습+1단원만 학습+
전지훈련 없이 우리학교 운동장에서만 학습

📌 과적합이 발생하는 이유 🚨

✔ **훈련 데이터에 너무 맞춰진 모델 구조**

- 너무 복잡한 모델(파라미터가 많음) → 불필요한 패턴까지 학습

✔ **훈련 데이터가 부족하거나 편향됨**

- 특정 유형의 데이터만 반복 학습 → 새로운 데이터에 대한 일반화 능력 부족

✔ **학습을 너무 오래 시킴 (Epoch 수가 너무 많음)**

- 모델이 훈련 데이터에 완벽히 맞추려다 보니 새로운 데이터에서는 성능 저하

📌 과적합 해결 방법 🛠️

✅ 1️⃣ 정규화(Regularization) 사용

- L1/L2 정규화 → 가중치를 너무 크게 학습하지 않도록 제어

✅ 2️⃣ 데이터 양 늘리기

- 훈련 데이터를 다양하게 만들어 과적합 방지 (데이터 증강, Augmentation)

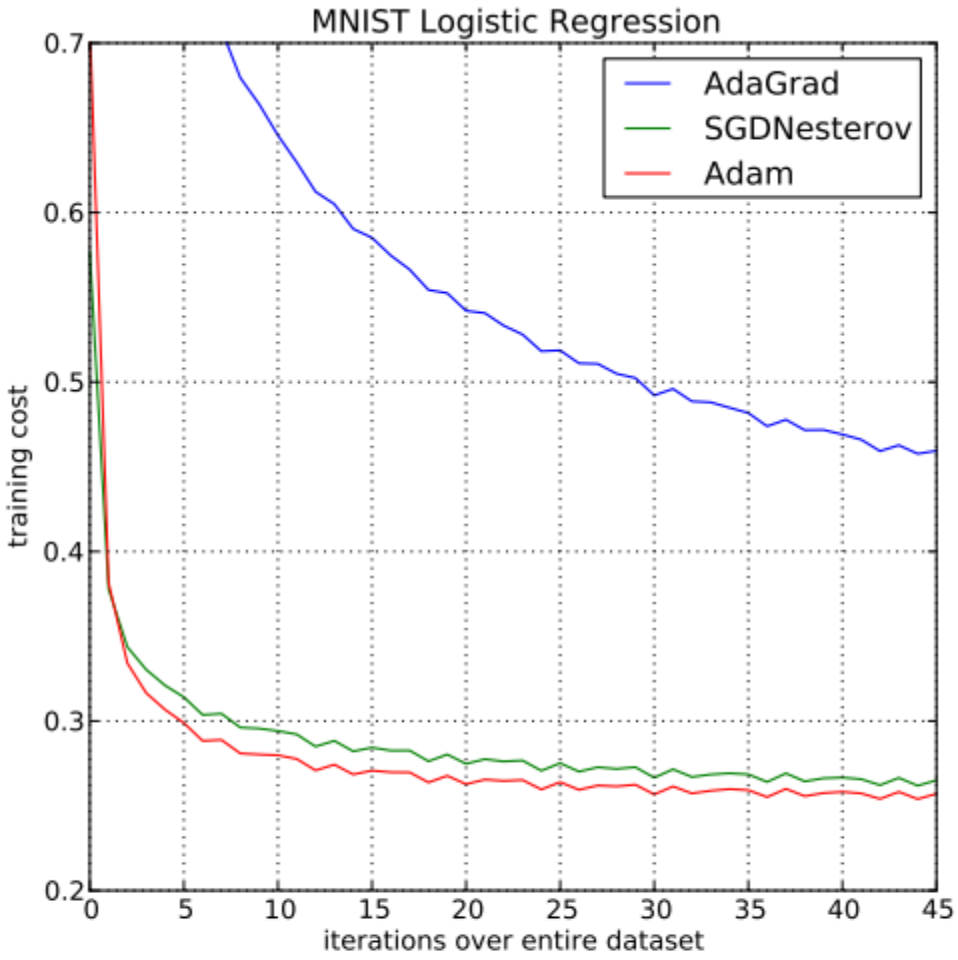
✅ 3️⃣ 조기 종료(Early Stopping)

- **훈련 데이터 성능은 올라가지만, 테스트 데이터 성능이 떨어지기 시작하면 학습 중단**

✅ 4️⃣ Dropout 사용

- 신경망에서 일부 뉴런을 랜덤하게 비활성화하여 특정 패턴에 과하게 의존하지 않도록 함

논문에서



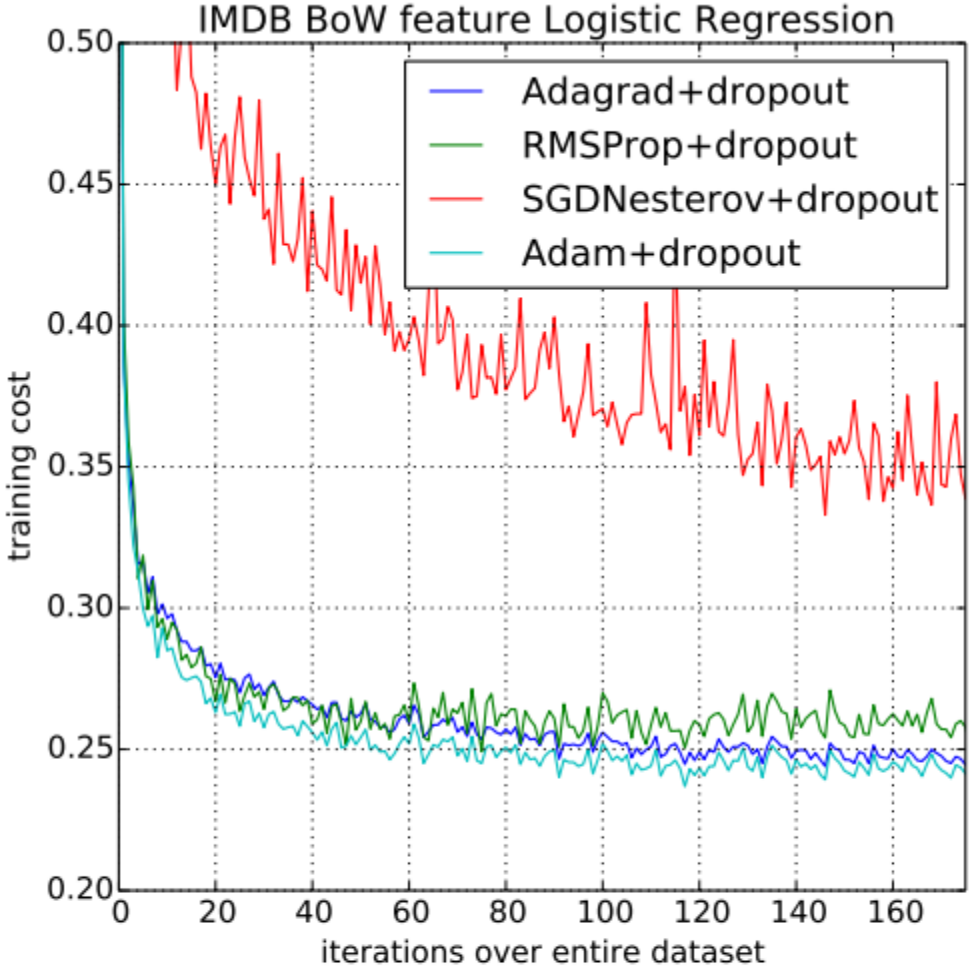
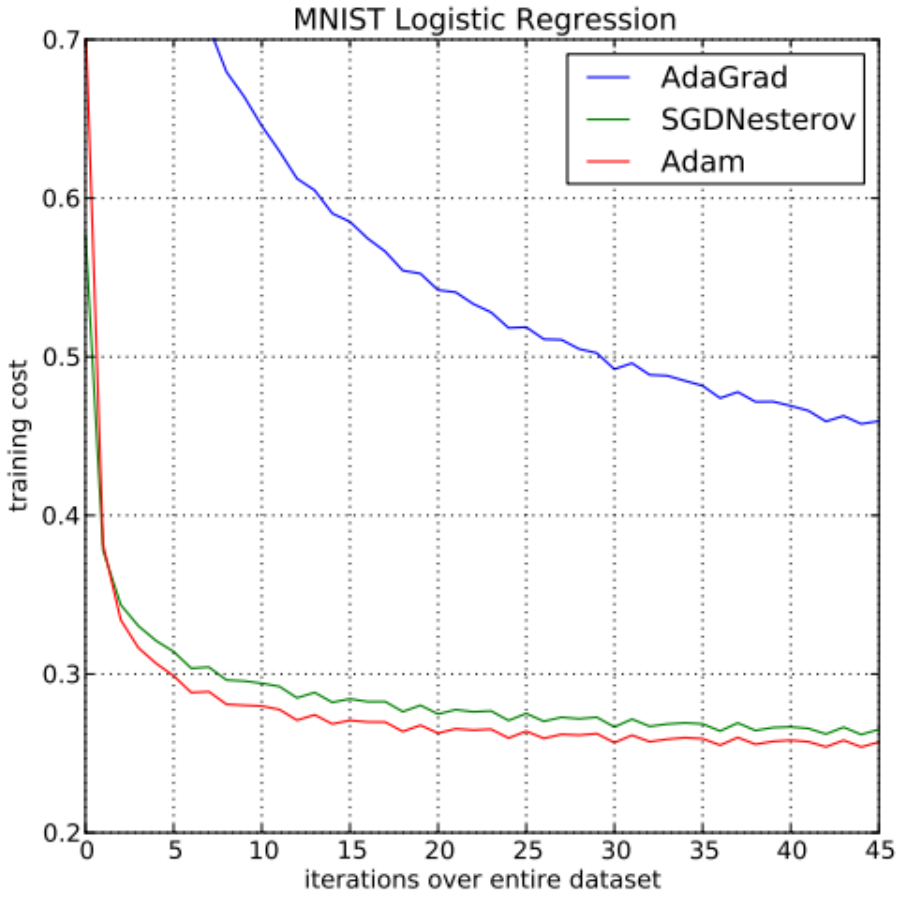


Figure 1: Logistic regression training negative log likelihood on MNIST images and IMDB movie reviews with 10,000 bag-of-words (BoW) feature vectors.

MNIST 데이터셋(손글씨 이미지 분류)

"로지스틱 회귀(Logistic Regression) 모델: 분류모델"

"Adam이 비용이 가장 저렴하게 든다 "



- **X축:** 전체 데이터셋에 대한 반복 횟수 (iterations)
- **Y축:** 훈련 비용(Training Cost)
- **파란색 (AdaGrad)** → 학습 속도가 상대적으로 느리고, 비용 감소가 더디게 진행됨
- **녹색 (SGDNesterov)** → 비용이 빠르게 감소하지만 Adam보다는 느림
- **빨간색 (Adam)** → 가장 빠르게 비용이 감소하며 안정적인 학습을 보여줌

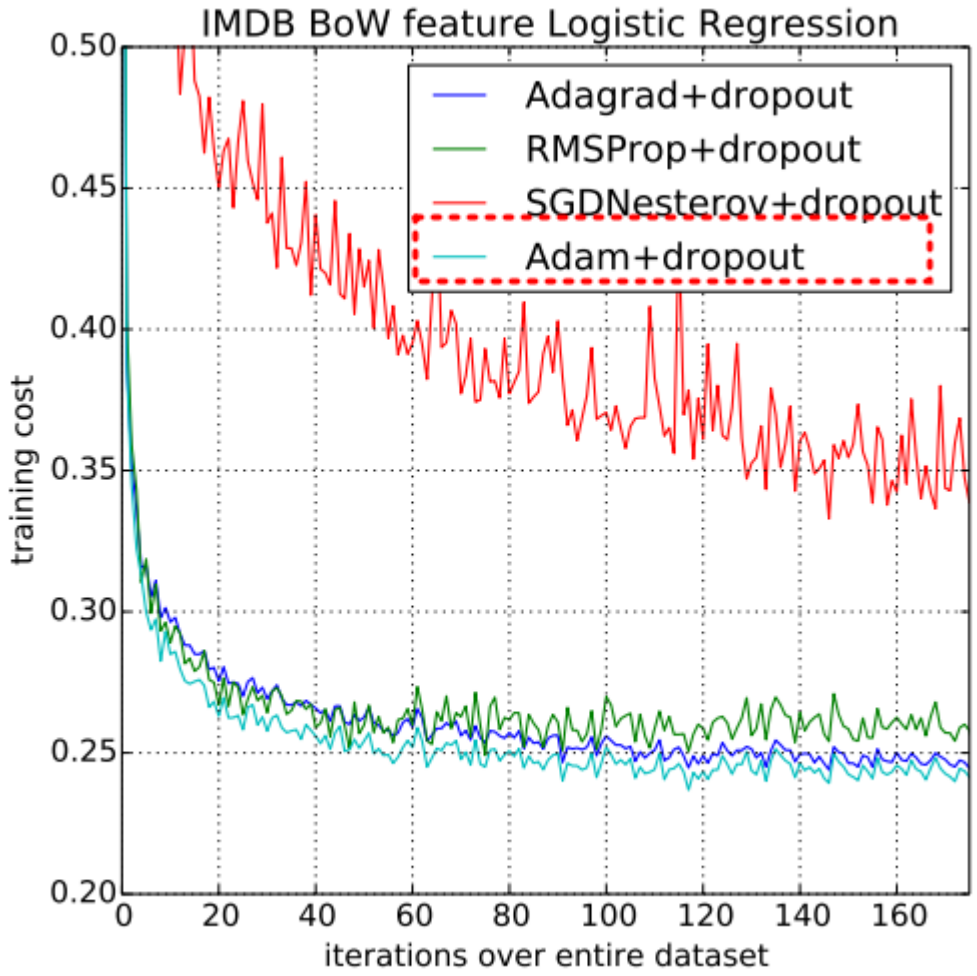
IMDB BoW(Bag-of-Words) 특징을 사용한 영화 리뷰 감성 분석

📌 IMDB BoW (Bag-of-Words)란? 🎯

✅ IMDB BoW는 영화 리뷰 데이터(IMDB)에서 단어의 출현 빈도를 기반으로 문장을 수치화하여 감성 분석(Sentiment Analysis) 등을 수행하는 방법입니다.

✅ BoW(Bag-of-Words, 단어 주머니)는 문장의 단어 개수를 세어 벡터로 변환하는 방식!

IMDB BoW(Bag-of-Words) 특징을 사용한 영화 리뷰 감성 분석



- **X축:** 전체 데이터셋에 대한 반복 횟수 (iterations)
- **Y축:** 훈련 비용(Training Cost

7. CONCLUSION (결론)

- Adam은 확률적 경사하강법(Gradient-Based Optimization) 기반의 최적화 알고리즘으로, 단순하면서도 계산 효율성이 높은 방법이다.
- 대규모 데이터셋과 고차원 파라미터 공간을 가진 머신러닝 문제에 효과적이다.