OptimABS: 절댓값 기반 적응적 옵티마이저의 연산 효율성 개선

이건수, 김정훈, 김지홍, 정재훈*

ksl1165@kau.kr, ang0811@kau.kr, ghd8876@kau.kr, jhjung@kau.kr

OptimABS: Computational Efficiency Improvement of Absolute Value-based Adaptive Optimizer

Lee Keon Soo, Kim Jeong Hun, Kim Ji Hong, Jay Hoon Jung*

Korea Aerospace University

요 약

딥러닝 모델 학습 과정에서 사용되는 현대 옵티마이저에서의 제곱근 연산은 파라미터별 학습률 조정에 필수적이지만 연산 복잡도를 증가시킨다. 본 논문은 이러한 연산 과정에서의 제곱-제곱근 구조를 절댓값으로 대체하는 OptimABS 방법론을 제안한다. MNIST 데이터셋과 Multi-Layer Perceptron(MLP) 모델을 사용한 실험을 통해 학습 성능을 유지하며 연산 복잡도의 감소를 확인했다. 이러한 결과는 절댓값 기반 옵티마이저가 효율적인 딥러닝 학습을 위한 실용적인 대안이 될 수 있음을 보여준다.

Ⅰ. 서론

기존의 적응적 학습률 옵티마이저들은 딥러닝 모델 학습에 있어서 중요한 역할을 해왔다. Adagrad[1]는 누적된 기울기 제곱을 활용하여 빈번하게 갱신되는 파라미터의 학습률을 줄이고 드물게 갱신되는 파라미터의 학습률은 유지하는 방식으로 효율적인 학습을 가능케 하였다. Adagrad 의 학습률 감소 문제를 해결하기 위해 등장한 RMSProp 은 지수 이동 평균을 적용하여 과거 기울기 정보의 영향을 점진적으로 줄였으며 Adam 의 경우[2]는 1 차, 2 차 모멘트 추정치를 결합하여 파라미터별로 더욱 세밀한 적응적 학습률을 조정했다.

이러한 적응적 학습률 옵티마이저들이 널리 활용되는 주요 원인은 파라미터별 학습률 조정, 희소 데이터 학습 효율 보장, 양·음수 기울기의 절대 크기 누적에 있다. 그러나 이들 방법은 기울기의 크기 정보를 얻기 위해 제곱 및 제곱근 연산을 필수적으로 수행해야 한다. 이러한 부수적인 연산 과정은 모델 학습의 연산 효율성 측면에서 재검토될 필요가 있다고 판단하였다.

본 연구는 이러한 기존 방식의 한계점에서 출발한다. 우리는 연산상에서 기울기의 크기 정보만을 필요로 한다면 굳이 제곱 후 제곱근을 취하는 복잡한 과정 대신 절댓값 연산을 사용하는 것이 연산 효율성 측면에서 훨씬 유리하다는 가설을 세웠다. 이에 따라 본 논문은 Adagrad 의 핵심 로직을 재검토하고 제곱 연산 대신 절댓값을 활용하는 새로운 적응적 학습률 옵티마이저 OptimABS 를 제안한다. 또한 이 절댓값 기반 접근 방식이 RMSProp 과 Adam 과 같은 주요 옵티마이저에도 성공적으로 적용될 수 있음을 논의한다.

Ⅱ. 본론

본 논문은 OptimABS 와 기존 옵티마이저 간의 핵심 차별점을 두 가지로 정리한다. 첫째, 해당 OptimABS 기법은 적응적 학습률 계산 시 기존 기울기의 제곱 $g_t{}^2$를 사용하는 수식 (1)과 다르게 수식 (2)와 같이 절댓값 $|g_t|$을 사용한다.

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

$$v_t = \beta v_{\{t-1\}} + (1-\beta)|g_t| \quad (2)$$

여기서 β는 지수 감쇠율, v_t는 t 번째 timestep 에서의 모멘트 추정치이다.

둘째, 이로 인해 파라미터 업데이트 과정에서 기존 수식 (3)의 제곱근 연산이 제거되어 수식 (4)의 식이 된다.

$$\theta_{\{t+1\}} = \theta_t - \frac{\alpha g_t}{\sqrt{v_t} + \epsilon} \quad (3)$$

$$\theta_{\{t+1\}} = \theta_t - \frac{\alpha g_t}{v_t + \epsilon} \quad (4)$$

여기서 θ_t는 t 번째 timestep 에서의 모델 파라미터, α는 학습률, ε은 수치적 안정성을 위한 작은 상수이다.

연산 종류	기존 방식	제안 방식
파라미터 업데이트	$O(\log n)$	$O(1)$
전체 복잡도	$O(p \log n)$	$O(p)$

표 1: 연산 복잡도 비교

표 1 을 볼 때 기존 방식에서 파라미터 업데이트에는 $O(\log n)$의 시간이 걸렸으나 절댓값을 적용한 방식에서는 $O(1)$의 시간이 소요된다. 따라서 전체 복잡도는 $O(p \log n)$에서 $O(p)$로 감소할 수 있게 된다.

Ⅱ-1. 실험 설정

MNIST 데이터셋을 Multi-Layer Perceptron 을 사용하여 배치 크기는 64, 128, 256 으로 설정하고 학습 에포크 200 회, 학습률 0.001, 가중치 감쇠 1e-4, β_1=0.9, β_2=0.999, ε=1e-8 파라미터를 적용하여 연산 복잡도 차이를 분석했다.

Ⅱ-2. 실험 결과

Optimizer \ Batch	64	128	256	Average
RMSProp	936.1	565.3	560.2	687.2
RMSPropABS(ours)	**882.5**	**589.4**	**528.2**	**666.7**
Adam	1112.1	651.7	540.0	767.9
AdamW	1124.2	669.1	542.6	778.6
AdamABS(ours)	**1071.0**	**632.6**	**514.6**	**739.4**

표 2: 배치 크기별 Epoch 당 학습 시간(초)

Optimizer \ Batch	64	128	256	Average
RMSProp	98.54	98.96	99.11	98.87
RMSPropABS(ours)	97.86	98.40	98.66	98.30
Adam	98.64	98.92	99.06	98.87
AdamW	98.89	99.18	99.10	99.05
AdamABS(ours)	97.86	98.32	98.81	98.33

표 3: 배치 크기별 테스트 정확도(%)

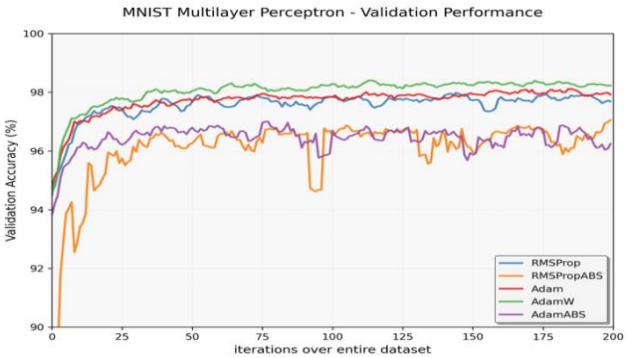

그림 1: Epoch 당 검증 성능

해당 실험 결과로 알 수 있는 본 방법의 이점은 작은 배치 사이즈일수록 학습 당 전체 연산량의 증가 대비 시간 효율성을 극대화할 수 있다는 점이다. 표 **2** 를 보면 배치 크기 64 에서 RMSPropABS 는 RMSProp 대비 5.7%(53.6 초), AdamABS 는 Adam 대비 3.7%(41.1 초)의 시간 단축을 보였으며 배치 크기가 작을수록 이러한 시간 효율성 개선이 더욱 두드러졌다. 정확도 측면에서는 표 **3** 에 나타난 바와 같이 Adam 대비 AdamABS 는 평균 98.87%에서 98.33%로 0.54%p, RMSProp 대비 RMSPropABS 는 98.87%에서 98.30%로 0.57%p 의 소폭 하락하였으나 여전히 98% 이상의 높은 정확도를 유지하며 안정적인 수렴 양상을 보였다.

이는 MLP 와 같은 완전연결 신경망에서 ABS 를 적용한 옵티마이저가 효과적으로 동작함을 보여준다. 반면 해당 방법의 한계점은 **그림 1** 에 나타난 바와 같이 Epoch 당 정확도의 변동성이 크다는 점이다. 정확도 또한 여전히 ABS 적용 옵티마이저들이 더 낮다. 따라서 해당 방법은 여전히 개선의 여지가 있다.

Ⅲ. 결론

본 논문에서는 Adagrad 기반 적응적 옵티마이저들의 제곱근 연산을 절댓값으로 대체하는 OptimABS 방법론을 제안했다. MNIST 데이터셋과 MLP 모델을 사용한 실험에서 기존 RMSProp, Adam 대비 연산 효율성을 개선하면서도 유사한 분류 성능을 달성하였다. 특히 작은 배치 크기에서 시간 효율성이 극대화됨을 확인했다. 향후 연구에서는 다양한 데이터셋과 모델 아키텍처에서의 성능 검증, 이론적 수렴 보장, 학습 안정성에 대한 추가적인 연구가 필요하다.

ACKNOWLEDGEMENT

본 과제(결과물)은 교육부와 한국연구재단의 재원으로 지원을 받아 수행된 첨단분야 혁신융합대학사업(차세대통신)의 연구 결과입니다.

참 고 문 헌

[1] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," Journal of Machine Learning Research, vol. 12, pp. 2121-2159, Jul. 2011.

[2] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. 3rd Int. Conf. Learning Representations (ICLR), San Diego, CA, USA, May 2015.