퓨샷 러닝 및 경량 딥러닝 응용을 위한 최적의 학습률 및 최적화 기법 탐색 알고리즘

윤수용, 조혜정, 문지원, 이상진, 정찬호*

한밭대학교

20181435@edu.hanbat.ac.kr, 20201566@edu.hanbat.ac.kr, 20221094@edu.hanbat.ac.kr, 20151709@edu.hanbat.ac.kr, peterjung@hanbat.ac.kr*

An Algorithm to Search Optimal Learning Rate and Optimization Technique for Few-Shot Learning and Lightweight Deep Learning Applications

Suyoung Yun, Hyejeong Cho, Jiwon Moon, Sangjin Lee, Chanho Jung*

Hanbat National University

요 약

본 논문에서는 퓨샷 러닝 및 경량 딥러닝 응용을 위한 최적의 학습률 및 최적화 기법 탐색 알고리즘을 제안한다. 제안하는 방법에서는 매 반복(iteration)마다 손실을 최소로 하는 학습률 및 최적화 방법을 선택한다. 또한 선택적으로 모멘텀 최적화 기법을 이용한다. 제안하는 방법의 효율성을 검증하기 위하여 MNIST 데이터셋을 이용하였다. 본 논문에서는 퓨샷 러닝을 위해 클래스 당 학습데이터 개수를 제한하였다. 또한 경량 딥러닝 응용을 위해 매우 간단한 구조를 가지는 네트워크 아키텍처를 이용하였다. 실험결과는 제안하는 방법이 기존 방법에 비해 1) 배치사이즈가 클수록, 2) 학습률이 작을수록 비교적 높은 성능을 제공함을 보여준다. 뿐만 아니라 클래스 당 학습데이터 개수가 비교적 적을 경우(예를 들어 1, 3) 히든 레이어 내 뉴런 개수가 적을수록 비교적 우수한 성능을 제공함을 볼 수 있었다.

Ⅰ. 서 론

딥러닝 모델을 학습시키는 과정에서 대부분의 경우 학습률과 같은 하이퍼파라미터를 적절하게 설정하는 것과 딥러닝 모델의 예측 성능은 밀접한 관련이 있는 것으로 알려져 왔다. 이러한 이유로 다양한 학습률 스케줄링 기법이 제안 및 이용되어왔다.

본 논문에서는 퓨샷 러닝 및 경량 딥러닝 실현 가능성을 알아보기 위해 최적의 학습률 탐색 알고리즘을 제안한다. 적절한 학습률을 탐색하는 동시에 적절한 최적화 기법(경사 하강법 또는 경사 상승법)을 탐색한다. 구체적으로 매 반복(iteration)마다 손실을 최소로 하는 학습률 및 최적화 방법을 선택한다. 또한 선택적으로 모멘텀 최적화 기법을 이용한다. 이는 다음과 같은 가정을 기반으로 한다: “각 반복에서 손실함수의 값이 global minimum에 가까운 값이 아닌 local minima에 가까운 값일 가능성이 있다.”. 제안하는 방법의 효율성을 검증하기 위하여 MNIST 데이터셋을 이용하였다. 본 논문에서는 퓨샷 러닝 가능성을 알아보기 위해 클래스 당 학습데이터 개수를 제한하였다. 또한 경량 딥러닝 응용을 위해 매우 간단한 구조를 가지는 네트워크 아키텍처(two-layer network)를 이용하였을 뿐만 아니라 히든 레이어의 사이즈가 입력 레이어의 사이즈 대비 매우 작도록 설정하였다. MNIST 데이터셋에 대한 실험결과는 제안하는 방법이 기존 방법에 비해 1) 배치사이즈가 클수록, 2) 학습률이 작을수록 비교적 높은 성능을 제공함을 보여주었다.

Ⅱ. 제안하는 방법

본 논문에서는 매 반복마다 손실함수를 최소로 하는 학습률 및 최적화 방법을 선택한다. 이를 위해 학습률에 대한 프로포절 집합 $H=\{\eta_1,\eta_2,\ldots,\eta_n\}$ 를 정의한다. 또한 최적화 기법 후보군을 경사 하강법 및 경사 상승법으로 구성한다. 따라서 학습률 및 최적화 방법을 각각 선택할 수 있는 경우의 수는 총 $2n$가지다.

제안하는 방법에서는 매 반복에서 총 $2n$개의 경우의 수 중 손실함수를 최소로 하는 학습률 및 최적화 방법을 각각 선택한다. 단, 그림 1과 같은 동작 순서를 따라 선택해야 한다.

Constraint: 현재 반복에서의 손실함수 값 L_i가 이전 반복에서의 손실함수 값 L_{i-1}보다 작아야 한다.

Step 1: Constraint를 만족시키는 최적화 방법이 1) 경사 하강법 및 경사 상승법인 경우, 2) 오직 경사 하강법인 경우, 3) 없는 경우 선택할 수 있는 최적화 방법을 경사 하강법으로 제한한다.

Step 2: Constraint를 만족시키는 최적화 방법이 오직 경사 상승법인 경우 선택할 수 있는 최적화 방법을 경사 상승법으로 제한한다.

그림 1. 제안하는 알고리즘의 동작 순서

선택된 학습률 및 최적화 기법을 기반으로 현재 반복에서의 손실함수 값을 결정한다. 다음 반복으로 넘어가기 전에 다음 조건을 만족하는 경우 모멘텀 최적화 기법[1]을 적용한다.

$$|L_i - L_{i-1}| < \theta. \quad (1)$$

식 (1)에서 θ는 사전에 정의된 threshold를 나타낸다.

Ⅲ. 실험결과

본 논문에서는 제안하는 방법의 성능을 평가하기 위하여 two-layer network를 이용하였다. 히든 레이어 사이즈의 영향을 평가하기 위하여 히든 레이어의 사이즈를 각각 20, 50으로 설정하였다. 제안하는 방법의 구현에서 식 (1)의 θ는 0.37로 설정하였다. 또한 학습률 프로포절 집합은 $H=\{0.2, 0.3, 1, 3, 5\}$로 설정하였다. 베이스라인에 대한 실험결과를 기반으로 손실함수 값이 0.3 이하인 경우 기존 방법으로 학습을 진행한다.

표 1. 배치사이즈, 학습률, 클래스 당 학습데이터 개수, 히든 레이어 내 뉴런 개수 변화에 따른 베이스라인 및 제안하는 방법 간의 성능 비교 (1행 : 베이스라인(히든 레이어 내 뉴런 개수 : 20), 2행 : 제안하는 방법(히든 레이어 내 뉴런 개수 ; 20), 3행 : 베이스라인(히든 레이어 내 뉴런 개수 : 50), 4행 : 제안하는 방법(히든 레이어 내 뉴런 개수 ; 50))

배치사이즈	학습률	클래스 당 학습데이터 개수				
		1	3	5	7	9
1	0.1	**0.5094**	**0.5559**	**0.6203**	**0.6368**	**0.6840**
		0.1176	0.1182	0.1195	0.1135	0.1135
		0.5062	**0.6072**	**0.6434**	**0.6386**	**0.6688**
		0.1177	0.1135	0.1204	0.1859	0.1136
	0.01	**0.5324**	**0.6092**	**0.6474**	**0.6657**	**0.6849**
		0.5160	0.5830	0.6405	0.6547	0.6836
		0.5152	**0.6064**	0.6492	**0.6787**	0.6974
		0.5146	0.5994	**0.6692**	0.6730	**0.7002**
2	0.1	**0.5271**	**0.6145**	**0.6523**	**0.6721**	**0.6808**
		0.1611	0.1221	0.1294	0.1139	0.1374
		0.5129	**0.6110**	**0.6581**	**0.6839**	**0.6914**
		0.4193	0.1881	0.2515	0.1651	0.1459
	0.01	0.4343	**0.6013**	**0.6447**	**0.6663**	**0.6852**
		0.5212	0.5934	0.6126	0.6215	0.6658
		0.4835	**0.6080**	**0.6489**	**0.6781**	**0.6946**
		0.5180	0.6042	0.6394	0.6765	0.6847
4	0.1	**0.5325**	**0.6163**	**0.6467**	**0.6682**	**0.6795**
		0.4395	0.2411	0.2945	0.3961	0.2174
		0.5160	**0.6110**	**0.6531**	**0.6815**	**0.6969**
		0.4411	0.5834	0.6149	0.6545	0.4226
	0.01	0.4019	0.4684	0.6377	**0.6536**	**0.6822**
		0.5287	**0.5951**	**0.6499**	0.6423	0.6752
		0.4266	0.5768	0.6457	0.6728	0.6934
		0.5183	**0.6025**	**0.6657**	**0.6793**	**0.6973**
8	0.1	**0.5250**	**0.6076**	**0.6497**	**0.6664**	**0.6881**
		0.4173	0.5895	0.5582	0.6414	0.6447
		0.5134	**0.6091**	**0.6527**	**0.6787**	**0.6984**
		0.4805	0.5963	0.6403	0.6644	0.6822
	0.01	0.2482	0.2189	0.3353	0.5094	0.6461
		0.5305	**0.6019**	**0.6372**	**0.6607**	**0.6824**
		0.3667	0.3669	0.5507	0.6259	0.6791
		0.5135	**0.6072**	**0.6459**	**0.6779**	**0.7051**

MNIST 학습데이터셋으로부터 추출한 데이터를 two-layer network 학습을 위한 학습 데이터셋으로 이용하였다. (퓨샷 러닝을 위한) 학습데이터 개수에 따른 성능 변화 정도를 알아보기 위하여 데이터를 추출하는 과정에서 클래스 당 학습데이터 개수가 각각 1, 3, 5, 7, 9가 되도록 하였다. 성능 평가를 위해 10,000개의 샘플로 구성된 MNIST 테스트데이터셋을 이용하였다. 배치사이즈에 따른 성능 변화를 평가하기 위하여 배치사이즈를 각각 1, 2, 4, 8이 되도록 설정하였다. 즉, 히든 레이어의 사이즈 및 클래스 당 학습데이터 개수를 일정하게 유지시키면서 배치사이즈를 각각 1, 2, 4, 8로 하여 미니배치학습을 수행하였다. 표 1은 배치사이즈, 학습률, 클래스 당 학습데이터 개수, 히든 레이어 내 뉴런 개수 변화에 따른 베이스라인 및 제안하는 방법 간의 성능 비교 결과를 보여준다. 모든 epoch에 걸쳐서 테스트데이터셋에 대한 분류 정확도를 측정하고, 그 중 최대값을 각 방법의 성능으로 판단하였다. 표 1에서 보는 바와 같이 제안하는 방법이 기존 방법에 비해 1) 배치사이즈가 클수록, 2) 학습률이 작을수록 비교적 높은 성능을 제공함을 알 수 있다. 또한 클래스 당 학습데이터 개수가 비교적 적을 경우(예를 들어 1, 3) 히든 레이어 내 뉴런 개수가 적을수록 비교적 우수한 성능을 제공함을 볼 수 있었다.

Ⅳ. 결론

본 논문에서는 퓨샷 러닝 및 경량 딥러닝 응용을 위한 최적의 학습률 및 최적화 기법 탐색 알고리즘을 제안하였다.

ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2021R1I1A3041191).

참 고 문 헌

[1] S. Ruder, “An overview of gradient descent optimization algorithms,” arXiv preprint arXiv:1609.04747 (2016).