ResNet 논문에서의 다양한 매개변수 변경 및 데이터 증강을 통한 성능 개선 실험

김동현*, 이나현*

Performance Improvement Experiments with Varied Parameters Modification and Data Augmentation in the ResNet Paper

DongHyeon Kim*, and NaHyun Lee*

요 약

본 보고서에서는 CNN 신경망 구조를 사용한 ResNet18 모델을 구현하고, 학습률과 활성화 함수 등 다양한 인자 값을 변경하고 데이터 증강 기법을 적용하여 실험을 진행한다. 이를 통해 실험 결과를 분석하여 인자 값들의 조정과 데이터 증강 기법 적용이 ResNet18 모델의 성능에 어떤 영향을 미치는지 평가한다. 또한 ResNet18 모델에서 최적의 성능을 달성하기 위한 적절한 인자 값과 데이터 증강 기법을 확인한다.

Abstract

In this report, we implement the ResNet18 model using a CNN neural network architecture and conduct experiments by varying different parameters such as learning rate and activation functions and applying data augmentation techniques. By analyzing the experimental results, we analyze and evaluate how adjusting the various parameters and applying data augmentation techniques affect the performance of the ResNet18 model. We also identify the appropriate parameters and data augmentation techniques that can achieve the best performance to achieve optimal performance in the ResNet18 model.

Key words
deep learning,, convolutional neural network, ResNet

Ⅰ. 서 론

CNN(Convolution Neural Network) 모델은 이미지 인식, 이미지 분류 등 컴퓨터 비전 관련 작업에서 사용되는 신경망 구조이다. CNN은 컨볼루션, 풀링 등을 통해 이미지의 다양한 특징을 추출하고 학습하는 방식이다.

VGGNet은 3x3 크기의 컨볼루션 필터를 사용하여 깊은 네트워크를 구성하여 이미지에서 특징을 추출하는 모델이며, GoogleNet은 Inception 모듈이라는 새로운 구조를 도입하여 깊은 네트워크를 구성

하는 모델이다. 하지만 이러한 모델들은 네트워크의 깊이가 깊어질수록 정확도가 낮아지는 degradation 문제를 가지고 있다. 이는 CNN의 특성으로 레이어가 깊어지면 역전파 과정에서 활성화 함수 미분값이 소실되어 그레이디언트가 희미해지는 현상인 그레이디언트 소실 문제가 발생하기 때문이다. 하지만 CNN은 깊은 레이어를 가질수록 더 정확한 결과를 얻을 수 있기 때문에, 그레이디언트 소실 문제를 억제하여 더 깊은 레이어를 구성해야 한다.

ResNet 모델은 입력 값이 일정 층들을 건너뛰어 출력에 더해지는 잔차 연결(Residual Connection)을 이용해 그레이디언트 소실 문제를 해결한 모델이다. 입력과 출력간의 차이(residual)을 계산하므로 모델이 깊어져도 그레이디언트 소실 문제를 극복할 수 있다. 따라서 ResNet은 이전 모델들보다 더 깊은 레이어를 가지면서도 빠른 속도와 개선된 성능을 제공한다.

본 보고서에서는 직접 ResNet18 모델을 직접 구현하고 다양한 인자 값들을 변경하며 여러 실험을 수행한다. 또한 원본 데이터로부터 새로운 데이터를 만들어내는 데이터 증강 기법을 적용하여 실험을 진행한다.

Ⅱ. 실험 환경

1. 데이터셋

본 실험에서 사용한 데이터셋은 CIFAR 10이다. 해당 데이터셋은 32x32 사이즈의 이미지 60,000개로 구성되어있다. 기존에는 MNIST를 사용하려 했으나, MNIST는 특징이 잘 뽑히도록 구성되어있는 데이터셋이라 학습을 몇 번만 돌려도 정확도가 99%를 넘었기에 비교 분석 실험에는 적합하지 않다고 판단했다.

2. 레이어 개수

본 실험에서는 18개의 레이어를 사용하는 ResNet18 모델을 사용했다. 여러 방법론을 적용해서 결과를 비교분석 할 것이기에 빠르게 학습이 가능한 것이 중요했다. 그리하여 모델의 레이어 개수는 18개로 정했다.

Ⅲ. 실험 결과

본 보고서는 기존 ResNet 논문에 기반하여 다양한 방법론을 적용하였다. 본 실험의 목표는 다양한 방법론을 적용해보고, 이에 따라 결과가 어떻게 변하는지 등을 비교 분석해보는 것이다. 적용해본 방법론은 다음과 같다.

1. 학습률(Learning rate) 비율 조절 2. 활성화 함수(Activation Function) 변경 3. 데이터 증강 기법 적용

1. 학습률 비율 조절

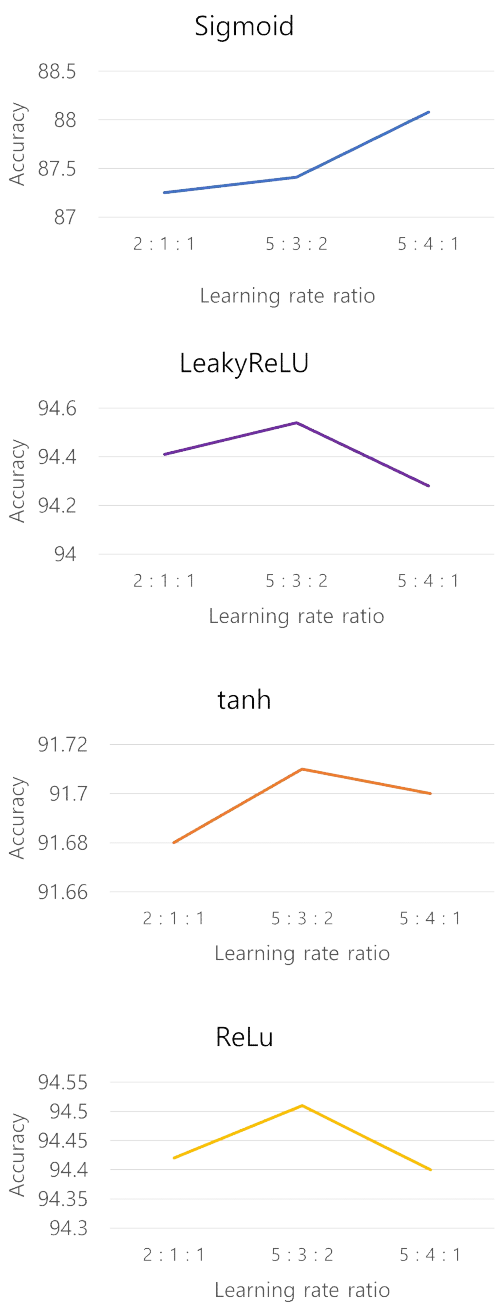
모델이 파라미터를 학습하는 과정에서 '학습률'이라는 파라미터가 사용된다. 이는 가중치에 오차를 얼마나 반영할 것인지에 대한 정도를 나타낸다. 보통의 경우 처음에는 큰 값을 사용해서 빠르게 최적화한 후에, 조금씩 값을 줄이면서 최적의 값에 근사하도록 한다.

기존 모델에서는 이를 2:1:1의 비율으로, 0.1, 0.01, 0.001을 적용했다. 학습의 0~50%에서 0.1, 50~75%에서 0.01, 75~100%에서 0.001을 사용한 것이다.

본 실험에서는 이 비율을 5:3:2(이하 [1-1])와 5:4:1(이하[1-2])로 총 2가지 적용해보았다. 중간 학습률의 비율을 높이는 것이 본 실험의 의도였다. 이는 최적값에 접근하는 속도를 올리는 정도라고 해석할 수 있다. [1-1]은 조금만 올린 것이고, [1-2]는 더 많이 올린 것이다.

실험 결과, [1-1]은 성능이 소폭 상승하였으나, [1-2]는 성능이 오히려 낮아졌다. 이는 가파른 비율로 가중치를 업데이트하면 최적값에 근사하지 못하고 오히려 멀어지는 현상을 보여준다.

그림 1 . 학습률 비율에 따른 모델의 성능



결론적으로 적절한 학습률 값을 선택하여 최적값에 효율적으로 근사할 수 있도록 하는 것이 중요하다는 것을 알 수 있다.

2. 활성화 함수 변경

기존 논문에서는 활성화 함수로 ReLU를 사용하였다. 이에 본 실험에서는 sigmoid, tanh, LeakyReLU를 적용하여 실험해보았다.

Sigmoid 활성화 함수는 가장 기본적인 형태의 활성화 함수이다. 하지만 해당 함수의 미분 값 최대치가 0.5 이기 때문에 레이어를 깊게 쌓는 경우 역전파 과정에서 loss가 제대로 전달되지 않는 그레이디언트 소실 현상이 발생하게 된다. 이러한 현상을 방지하기 위해 기존 논문에서 사용된 ReLU라는 함수가 등장하게 되었다.

실험을 통해 Sigmoid 함수와 ReLU 함수가 어느 정도 차이가 있는지 확인했다. 실험 결과, 활성화 함수로 Sigmoid 함수를 사용한 경우 기존 논문과 비교하여 약 5%정도 성능이 하락하는 것을 확인할 수 있었으며, 이는 역전파 과정에서 발생하는 그레이디언트 소실로 인한 영향으로 해석할 수 있다.

tanh는 Sigmoid 함수를 -1~+1 범위로 확장한 형태이다. 이렇게 함수의 범위를 확장한 덕분에 tanh 함수의 미분 값의 최대치는 1이 되어, Sigmoid 함수의 0.5보다 더 큰 값을 가지게 된다. 따라서 Sigmoid보다 역전파 과정에서 발생하는 그레이디언트 소실의 영향을 덜 받을 것으로 보인다.

실험 결과, Sigmoid보다 성능이 향상된 것을 확인할 수 있다. 하지만 기존 모델인 ReLU보다는 좋지 못했는데, 이는 0 이상의 값은 그대로 전달하는 ReLU에 비해, tanh는 일부만 전달하기 때문으로 해석할 수 있다.

ReLU에서 0 이하의 값은 모두 무시되는 “dying ReLU” 현상이 발생할 수 있다. 이러한 단점을 보완하기 위해 LeakyReLU 함수가 등장하였으며, 이 함수는 0 이하의 값을 일정한 비율로 전달함으로써 “dying ReLU” 현상을 방지한다.

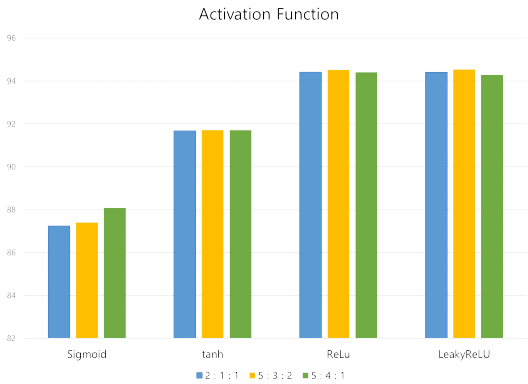
본 실험에서 LeakyReLU를 적용한 경우 기존 ReLU보다 더 좋은 성능이 나올 것으로 예상했으나, 실제로는 대동소이한 결과를 보였다.

이는 신경망에서 유의미한 0 이하의 값이 거의

나오지 않는다는 것을 의미한다. 즉 CIFAR10 데이터셋과 ResNet18 모델 구조에서는 ReLU 함수로도 충분한 성능을 발휘할 수 있어 LeakyReLU의 장점이 그다지 부각되지 않았던 것으로 해석할 수 있다.

결론적으로 활성화 함수마다 각자의 특성이 있고, 이러한 함수의 특성을 잘 고려하여 학습하고자 하는 데이터와 적용하고자 하는 신경망의 특성에 맞는 활성화 함수를 선택해야 한다는 것을 알 수 있다.

그림 2 . 활성화 함수에 따른 모델의 성능



3. 데이터 증강 기법 적용

데이터 증강은 훈련 데이터를 다양한 방법으로 변형 및 증강하여 학습에 활용하는 기법이다. 해당 실험에서는 두 가지의 데이터 증강 기법을 적용했다.

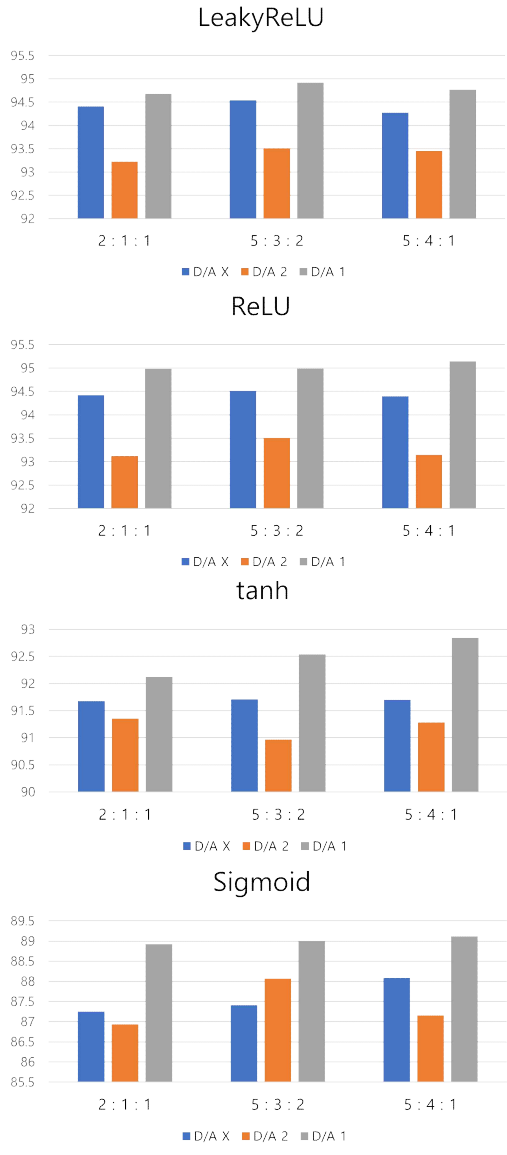
첫 번째는 이미지를 무작위로 자르고 뒤집는 기법(Random crop and flip)을 적용하였고 (이하 [3-1]), 두 번째는 [3-1]에 추가로 블러와 색 변환을 적용한 기법이다 (이하 [3-2]). 결론적으로 [3-1]은 모델의 성능을 향상시켰지만, [3-2] 방법은 오히려 모델의 성능을 저하시켰다.

이론적으로 데이터 증강은 학습 데이터의 다양성을 높여 전반적인 모델의 성능을 향상시킨다고 알려져 있다. 따라서 [3-2]도 당연히 [3-1]처럼 성능이 올라가야 했고, 심지어 더 많은 증강 기법을 적용했기 때문에 [3-1]보다 성능이 잘 나와야 했지만 그러지 않았다.

그 이유는 [3-2]에서 블러 방법을 적용했기 때문이다. 이번 실험에서 사용한 CIFAR10 데이터셋은 이미지 사이즈가 32x32로 매우 작다. 보통의 이미지에는 블러를 적용해도 학습에 필요한 중요한 특징은 날아가지 않고 노이즈만 적당히 날아가지만, 이런 작은 이미지에 블러를 적용하면 학습해야하는 데이터의 특징이 사라지게 된다. 그로 인해 오히려 블러를 적용한 [3-2]에서 성능이 낮아진 것이다.

결론적으로 학습하고자 하는 데이터의 특성에 맞게 증강 기법을 적용해야한다는 것을 알 수 있다.

그림 3 . D/A X: 증강 적용 안함 D/A 2: 랜덤 크롭 + 뒤집기 + 블러 + 색 변환 D/A 1: 랜덤 크롭 + 뒤집기



		2 : 1 : 1	5 : 3 : 2	5 : 4 : 1
Data Augmentation X	Sigmoid	87.25	87.41	88.08
	tanh	91.68	91.71	91.70
	ReLu	94.42	94.51	94.40
	LeakyReLU	94.41	94.54	94.28
Data Augmentation (crop&flip + color&blur)	Sigmoid	86.94	88.07	87.16
	tanh	91.35	90.97	91.28
	ReLu	93.12	93.51	93.15
	LeakyReLU	93.22	93.51	93.45
Data Augmentation (crop + flip)	Sigmoid	88.92	89.00	89.12
	tanh	92.12	92.54	92.84
	ReLu	94.98	94.99	95.14
	LeakyReLU	94.68	94.92	94.77

그림 4 . 최종 결과 표 (노란색: 기존 모델, 분홍색: 최고 향상 모델)

Ⅲ. 결 론

본 보고서에서는 CNN 신경망 구조를 활용한 ResNet18 모델을 직접 구현하고, CIFAR10 데이터셋을 사용한 모델에 학습률 변경, 활성화 함수 변경 및 데이터 증강 기법을 적용한 실험을 진행했다.

실험 결과 학습률 비율은 5:4:1로 조정하고 ReLU 활성화 함수를 적용하고 자르기와 뒤집기 데이터 증강 기법을 적용한 실험의 성능이 95.14%로 가장 우수했다.

Ⅳ. 부 록

GitHub 주소

https://github.com/NaDongHyeon/ComputerVision__2023

CIFAR10 데이터셋 URL

https://www.cs.toronto.edu/~kriz/cifar.html

참 고 문 헌

[1] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[3] 핸즈온 머신러닝, 오렐리앙 제롱, 한빛미디어, 2020

[4] 혼자 공부하는 딥러닝, 박해선, 한빛미디어 2020

[5] https://towardsdatascience.com/covolutional-neural-network-cb0883dd6529

[6] https://manu1992.medium.com/what-are-deep-residual-networks-or-why-resnets-are-important-40a94b562d81

[7] https://velog.io/@uonmf97/%EB%AA%A8%EB%91%90%EB%A5%BC-%EC%9C%84%ED%95%9C-%EB%94%A5%EB%9F%AC%EB%8B%9D-%EC%8B%9C%EC%A6%8C-2-ResNet-CIFAR10

[8] https://github.com/ndb796/Deep-Learning-Paper-Review-and-Practice/blob/master/code_practices/ResNet18_CIFAR10_Train.ipynb

[9] https://www.youtube.com/watch?v=671BsKl8d0E