Journal of Digital Convergence
Vol. 19. No. 10, pp. 253-263, 2021

ISSN 2713-6434 / eISSN 2713-6442
https://doi.org/10.14400/JDC.2021.19.10.253

비용함수와 파라미터를 이용한 효과적인 디지털 데이터 기계학습 방법론

지상민[1], 박지은[2*]
[1]충남대학교 수학과 박사과정 학생, [2]대구대학교 성산교양대학 교수

An efficient machine learning for digital data using a cost function and parameters

Sangmin Ji[1], Jieun Park[2*]
[1]Student, Department of Mathematics, Chungnam National University
[2]Professor, Seongsan Liberal Arts College, Daegu University

요 약 기계학습은 학습에 이용되는 학습 데이터와 데이터를 예측할 인공신경망을 이용하여 비용함수를 만들고, 비용함수를 최소화시키는 파라미터들을 찾는 과정이다. 파라미터들은 비용함수의 그래디언트 기반 방법들을 이용하여 변화하게 된다. 디지털 신호가 복잡할수록, 학습하고자 하는 문제가 복잡할수록, 인공신경망의 구조는 더욱 복잡해지고 깊어진다. 복잡하고, 깊어지는 인공신경망 구조는 과적합(Over-fitting) 문제를 발생시킨다. 과적합 문제를 해결하기 위하여 파라미터의 가중치 감소 정규화 방법이 사용되고 있다. 우리는 이러한 방법에서 추가로 비용함수의 값을 이용한다. 이러한 방법으로 기계학습의 정확도가 향상되는 결과를 얻었으며 이는 수치 실험을 통하여 우수성이 확인된다. 이러한 결과는 기계학습을 통한 인공지능의 폭넓은 데이터에 대한 정확한 값을 도출한다.

주제어 : 최적화, 디지털 신호, 기계학습, 분류, 정규화

Abstract Machine learning is the process of constructing a cost function using learning data used for learning and an artificial neural network to predict the data, and finding parameters that minimize the cost function. Parameters are changed by using the gradient-based method of the cost function. The more complex the digital signal and the more complex the problem to be learned, the more complex and deeper the structure of the artificial neural network. Such a complex and deep neural network structure can cause over-fitting problems. In order to avoid over-fitting, a weight decay regularization method of parameters is used. We additionally use the value of the cost function in this method. In this way, the accuracy of machine learning is improved, and the superiority is confirmed through numerical experiments. These results derive accurate values for a wide range of artificial intelligence data through machine learning.

Key Words : Optimization, Digital signal, Machine learning, Classification, Regularization

*This work was supported by the National Research Foundation of Korea (NRF-2017R1E1A1A03070311)
*Corresponding Author : Jieun Park(writer2yah@daegu.ac.kr)
Received July 8, 2021 Revised September 9, 2021
Accepted October 20, 2021 Published October 28, 2021

1. 서론

알파고가 2016년 인간과의 대결에서 4:1의 승리한 것은 기계가 인간을 뛰어넘는 하나의 큰 사건이었다. 인간을 모방하는 것을 넘어 인간을 뛰어넘는 일이 현실화한 것이다. 인간의 능력을 뛰어넘는 일은 인공지능에서 특히 Deep Learning의 발전으로 가능하고 최근 무인 자동차, 번역기, 기사를 쓰는 일 등과 같은 다양한 분야에 사용하고 있다. 다양한 분야로의 활용은 어떻게 가능한지 기계학습의 작동 원리부터 살펴보자.

기계학습은 데이터 집합 $D=\{D_1, D_2, \cdots, D_n\}$, $D_i=(X_i, Y_i)$, $X_i \in X$, $Y_i \in Y$. 를 이용하여 학습시키는 방법으로 입력 집합 X와 목표 집합(Target) Y가 필요하다. 예를 들어 개와 고양이 사진을 분류한다고 했을 때, 입력 집합 X는 디지털화 된 사진(개, 고양이 사진), 목표 집합 Y는 $Y_1=0$과 $Y_2=1$로 설정할 수 있다. 0의 값은 개라고 생각하고, 1은 고양이라고 생각하면 된다. 집합 X의 원소 X_i를 인공신경망을 통하여 나온 예측값 $H(X_i, Y_i, w)$이 목표값 Y_i와 일치되게 하고 싶다. 즉 예측값과 목표값이 같아지도록 파라미터 w의 값을 변화시키는 것이다. 예측값과 목표값의 일치가 잘 이루어지도록 인공신경망의 구조를 깊게 하고, 많은 수의 파라미터 w를 사용하는 것이 Deep learning이다.

예측값이 목표값에 접근하고자 보통은 둘 사이의 거리를 최소로 줄이는 방법을 사용하며, 둘사이의 거리값을 우리는 비용함수($C(w)$)라고 한다. 따라서 비용함수의 값이 0이 되면 아주 이상적이나, 최소가 되는 상황일수록 학습은 잘 이루어진 것으로 본다. 따라서 인공지능 학습이 잘 이루어진 것인지를 확인하는 방법은 이 비용함수의 값으로 가능하다. 비용함수의 최솟값을 찾는 방법으로 가장 널리 쓰이는 방법은 비용함수의 그래디언트(Gradient)를 기반으로 하고 있다. 그래디언트를 기반으로 한 알고리즘으로는 GD, Momentum, RMSprop, Adadelta, Adam 등이 있다. 비용함수의 그래디언트와 이를 이용한 변화된 방법으로 파라미터 w를 변화시켜 비용함수의 최솟값을 찾아가는 방법이다.

많은 데이터와 복잡한 문제의 해결을 위하여, Deep Learning은 복잡한 신경망을 필요로 하고, 많은 수의 파라미터를 요구한다. 이것은 특정 입력 데이터에는 아주 잘 작동하지만, 입력 집합의 원소에서 학습에 사용된 데이터와 조금만 벗어난 다른 데이터에 대해서는 오류 값을 출력하는 문제를 발생시킨다. 이러한 문제를 과적합 문제(Over-Fitting)라고 이야기한다. 다시 말해서 우리는 이 세상의 모든 개, 고양이 사진을 가지고 올 수 없다. 따라서 우리가 학습에 다룰 수 있는 입력 집합은 전제 집합의 부분 집합이다. 이러한 부분 집합을 통하여 학습을 하더라도 전체 집합에 있는 데이터(학습 상황 이외의 데이터)에 대해서도 목표값(목표 집합 원소)을 잘 나타내기를 바란다. 학습에 사용되지 않은 개의 사진에 대해서도 잘 판단하기를 바라는 것이다. 과적합이 발생한 Deep Learning에서는 이러한 문제에서 오판의 여지가 있다.

과적합 문제를 해결하기 위하여 파라미터의 크기를 작게 해주는 가중치 감소 정규화(Weight Decay Regularization)가 널리 이용되고 있다[1-4]. 가중치 감소 정규화 방법은 파라미터의 크기를 감소하는 방향으로 기계학습이 이루어지는 방법이다.

가중치 감소 정규화 방법은 과적합 문제를 어느 정도 잘 수행한다. 그러나 보편적으로 예측값과 목표값의 불일치 정도가 커진다. 즉 가중치 감소 정규화 방법을 사용하지 않은 Deep Learning에 비하여 가중치 감소 정규화 방법을 사용한 Deep Learning에서 비용함수의 값은 증가하였다. 물론 가중치 감소 정규화 방법을 사용하지 않은 Deep Learning에서도 비용함수는 0이 될 수는 없다. 어느 정도의 불일치는 발생한다. 그러나 비용함수의 최솟값의 증가는 기계학습이 이루어지고 있는지를 판단하는데 어려움을 준다. 우리는 기계학습이 학습 데이터뿐만 아니라, 학습에 이용되지 않는 데이터에서도, 모든 부분에서 학습이 잘 이루어지기를 바란다. 모든 데이터에서 비용함수의 최솟값을 구하는 것이 목표이다. 학습의 정도를 비용함수를 이용하여 판단할 수 있기 때문이다.

가중치 감소 정규화 방법을 사용하여 과적합 문제를 해결하고, 동시에 비용함수 또한 작은 값을 얻고자 한다. 이러한 이유에서 파라미터 w의 변화는 비용함수 값 $C(w)$의 값에 따라 변화가 계속 이루어져야 한다. 우리는 그래디언트 기반으로한 최적화 알고리즘에 과적합 문제를 해결하기 위한 가중치 감소 정규화 방법을 사용하고, 최솟값을 구하기 위해 비용함수의 함수값을 더하는 새로운 최적화 알고리즘을 제시한다.

학습의 정도를 파악하기 위하여 비용함수는 계산하고 있으므로 추가적인 계산량은 더 요구되지 않는다. 또한 비용함수와 가중치 감소 정규화 방법에 가중치를 부여함으로 두 방법론에 적당한 비율로 알고리즘에 작용하게 한다. 이를 통해서 비용함수 방법을 알고리즘에 사용하지

않은 즉, 그래디언트 기반에 가중치 감소 정규화 방법만을 사용한 기존의 알고리즘에 비해 더 높은 정확도를 보여준다. 학습 데이터뿐만 아니라 학습에 사용되지 않은 데이터에서도 기존의 방법론 보다 높은 정확도를 보였다.

이 논문의 2장에서는 기존의 알고리즘을 설명하고 3장에서는 우리의 알고리즘을 설명한다. 4장에서는 각 최적화 알고리즘의 기계학습 수치 실험 결과를 제시한다. 기계학습 수치 실험에는 Python과 Tensorflow2를 이용하였으며 실험의 내용은 각 장(Chapter)에서 추가적인 설명을 한다.

2. 기존의 알고리즘

이 장에서는 기존의 기계학습 최적화 알고리즘들을 소개한다.

지도학습에서 학습 데이터 집합 D는 입력 데이터 X와 목표 데이터 Y로 구성되어 있다. 즉,

$$D = \{D_1, D_2, \cdots, D_n\},$$
$$D_i = (X_i, Y_i),$$
$$X_i \in X,\ Y_i \in Y.$$

(여기서 n은 데이터의 개수이다.)

기계학습에 이용할 인공 신경망 N의 레이어(layer)의 개수를 m이라고 하자. 그러면 인공 신경망 N 의 예측값 $H(D, w)$는 다음과 같이 정의된다.

$$H(D, w) = \frac{1}{n}\sum_{i=1}^{n} H(D_i, w),$$
$$H(D_i, w) = w^m \sigma^m (H^{m-1}(D_i, w)),$$
$$H^j(D_i, w) = w^{j-1} H^{j-1}(D_i, w),$$
$$H^0(D_i, w) = X_i.$$

여기서 $1 \le j \le m-1$이고, 파라미터들의 집합w는 $w = \{w^1, \cdots, w^m\}$이며, σ^j는 j번째 레이어에서의 활성화 함수이다.

기계학습에서 말하는 학습이란 i번째 데이터의 예측값 $H(D_i, w)$가 Y_i와 같아지도록 파라미터 w값을 바꾸는 것을 의미한다. 비용함수를 $C(w)$라고 하면 $H(D_i, w)$가 Y_i와 값이 가까워질수록 비용함수 $C(w)$의 값은 작아지게 된다. 따라서 우리의 목표는 비용함수 $C(w)$의 함수값을 작게 만드는 파라미터 w를 찾는 것이라고 할 수 있고, 이것을 의미하는 수식은 $argmin_w\{C(w)\}$이다.

과적합을 방지하기 위하여 이용되는 가중치 감소 정규화는 비용함수의 함수값을 줄이는 파라미터 w중에서도 크기가 작은 파라미터를 선택하는 방법이며 이것은 수식으로 표현하면 $argmin_w\{C(w) + \lambda \| w \|_2^2\}$이다.

최적화 알고리즘들은 반복법(iteration method)으로 같은 수식을 반복하면서 학습을 진행한다. 기존의 최적화 알고리즘들은 그래디언트가 감소하는 방향으로 파라미터를 변화시켜가는 GD(Gradient decent)을 기반으로 하고 있다[5-10]. GD 알고리즘은 파라미터 w의 변화에 비용함수 C의 그래디언트 $\frac{\partial C}{\partial w}$를 빼주는 방식의 알고리즘으로 식으로 쓰면 다음과 같다.

$$w_{i+1} = w_i - \eta \frac{\partial C(w_i)}{\partial w}. \quad (1)$$

여기서 η는 학습률(learning rate or step size)라고 불리는 상수로 파라미터값을 변화시킬 그래디언트의 양을 조절해주는 역할을 한다. 학습률 η의 값이 크면 학습이 잘 이루어지지 않고 η의 값이 너무 작으면 학습하는데 너무 많은 시간이 걸리게 된다. 학습률 η는 경험적으로 0.1에서 0.001정도의 숫자를 이용하며 상황에 따라서는 학습 정도에 따라서 값을 바꾼다[11-14].

여기에 가중치 감소 정규화를 적용하면 다음과 같은 방법으로 파라미터 w가 변화하게 된다.

$$w_{i+1} = w_i - \eta\left(\frac{\partial C(w_i)}{\partial w} + \lambda w_i\right). \quad (2)$$

여기서 λ는 decay rate로 주로 10^{-3}정도의 수를 이용한다.

만일 가중치 감소 정규화가 적용되지 않은 GD 알고리즘은 비용함수의 그래디언트에 의존하여 학습이 진행된다. 하지만 그래디언트는 부분 최솟값이나 부분 최댓값에서 0이 되기 때문에 학습이 더 이상 진행되지 않게 된다. 하지만 가중치 감소 정규화가 적용된 GD 알고리즘은

$$w_{i+1} = w_i - \eta\left(\frac{\partial C(w_i)}{\partial w} + \lambda w_i\right)$$
$$= (1-\eta\lambda) w_i - \eta \frac{\partial C(w_i)}{\partial w} \quad (3)$$

이므로 만일 w_i의 함수값이 부분 최솟값(또는 부분 최댓값)을 갖는다고 해도 η와 λ가 양수이기 때문에

$$w_{i+1} = (1-\eta\lambda) w_i - \eta 0, \quad (4)$$
$$w_{i+2} = (1-\eta\lambda) w_{i+1} - \eta \frac{\partial C(w_{i+1})}{\partial w} \quad (5)$$

인 상황이 되어 학습이 진행되지 않는 상황을 방지할 수

있다. 또한 $1-\eta\lambda<1$이기 때문에 파라미터 w는 크기가 작은 값을 선택하게 된다.

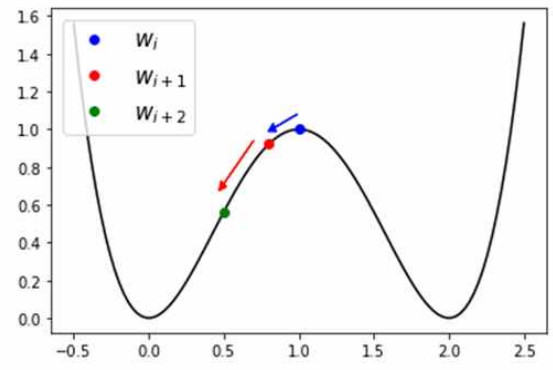


Fig. 1. Mechanism of weight decay regularization

2.1 Momentum

GD 알고리즘은 부분 최솟값에서 그래디언트 $\frac{\partial C}{\partial w}$가 0이 되어 학습이 되지 않는 문제점을 가지고 있는데, 이러한 문제점을 해결하기 위하여 지난 학습들에서의 그래디언트 값들을 비율로 더한 값 m_i를 학습에 이용하는 방법이 Momentum 알고리즘이다. 즉, $w_{i+1}=w_i-m_i$, $m_i=\beta m_{i-1}+\eta\frac{\partial C(w_i)}{\partial w}$식으로 계산된다. 여기서 $m_0=0$, η는 학습률이고 β는 momentum 상수로 주로 0.9를 이용한다. 이렇게 계산된 m_i에 의해 그래디언트 $\frac{\partial C}{\partial w}$의 값이 계속 누적되어 w값의 변화에 영향을 미치는 방법이다[15].

여기에 가중치 감소 정규화를 적용하면 다음과 같은 방법으로 파라미터 w가 변화하게 된다.

$$w_{i+1}=w_i-m_i, \tag{6}$$

$$m_i=\beta m_{i-1}+\eta\left(\frac{\partial C(w_i)}{\partial w}+\lambda w_i\right). \tag{7}$$

여기서 λ는 decay rate로 주로 10^{-3}정도의 수를 이용한다.

2.2 RMSprop

Momentum 알고리즘은 파라미터 w값의 변화를 줄 때 m_i를 이용하지만 학습이 어느 정도 진행된 후에는 m_i값이 상대적으로 커서 w값이 수렴하지 못하는 문제가 있다. 이러한 문제점 때문에 학습이 진행됨에 따라서 학습률(learning rate)이 줄어드는 기능을 추가한 알고리즘이 RMSprop (Root Mean Square Propagation)이다. RMSprop 알고리즘은 다음과 같다.

$$w_{i+1}=w_i-\frac{\eta}{\sqrt{G_i+\varepsilon}}\frac{\partial C(w_i)}{\partial w}, \tag{8}$$

$$G_i=\gamma G_{i-1}+(1-\gamma)\left(\frac{\partial C(w_i)}{\partial w}\right)^2 \tag{9}$$

여기서 $G_0=0$이고 ε은 분모가 0이 되는 것을 막아주기 위한 상수로 10^{-7}을 쓰며 β_1,β_2는 이동평균(Moving Average) 상수로 주로 $\beta_1=0.9,\beta_2=0.999$를 이용한다[16].

여기에 가중치 감소 정규화를 적용하면 다음과 같은 방법으로 파라미터 w가 변화하게 된다.

$$w_{i+1}=w_i-\frac{\eta}{\sqrt{G_i+\varepsilon}}\left(\frac{\partial C(w_i)}{\partial w}+\lambda w_i\right), \tag{10}$$

$$G_i=\gamma G_{i-1}+(1-\gamma)\left(\frac{\partial C(w_i)}{\partial w}+\lambda w_i\right)^2 \tag{11}$$

여기서 λ는 decay rate로 주로 10^{-3}정도의 수를 이용한다.

2.3 Adadelta

RMSprop은 G_i를 이용하여 학습이 진행됨에 따라서 학습률을 줄여나가지만 학습을 많이 진행하여야 하는 상황에서도 학습률을 줄이는 문제가 있다. 이러한 문제를 해결하기 위하여 Adadelta 알고리즘이 도입되었다. Adadelta 알고리즘은 다음과 같다.

$$w_{i+1}=w_i+\triangle w_i, \tag{12}$$

$$\triangle w_i=-\frac{RMS[\triangle w]_{i-1}}{RMS[g]_i+\varepsilon}\frac{\partial C(w_i)}{\partial w}, \tag{13}$$

$$E[\triangle w_i^{2]}=\rho E[\triangle w^2]_{i-1}+\ (1-\rho)\triangle w_i^2, \tag{14}$$

$$E[g_i^{2]}=\rho E[g^2]_{i-1}+\ (1-\rho)\left(\frac{\partial C(w_i)}{\partial w}\right)^2. \tag{15}$$

여기서 $E[g^2]_0=E[\triangle w^2]_0=0$, ε은 분모가 0이 되는 것을 막아주기 위한 상수로 10^{-7}을 쓰며 RMS는 root mean square의 약자로 제곱 평균의 제곱근을 의미하

고 ρ는 이동평균(Moving Average) 상수로 주로 0.9를 이용한다[17].

여기에 가중치 감소 정규화를 적용하면 다음과 같은 방법으로 파라미터 w가 변화하게 된다.

$$w_{i+1} = w_i + \triangle w_i,$$

$$\triangle w_i = -\frac{RMS[\triangle w]_{i-1}}{RMS[g]_i + \varepsilon}\left(\frac{\partial C(w_i)}{\partial w} + \lambda w_i\right), \quad (16)$$

$$E[\triangle w_i^{2]} = \rho E[\triangle w^2]_{i-1} + (1-\rho)\triangle w_i^2,$$

$$E[g^2]_i = \rho E[g^2]_{i-1} + (1-\rho)\left(\frac{\partial C(w_i)}{\partial w} + \lambda w_i\right)^2. \quad (17)$$

여기서 λ는 decay rate로 주로 10^{-3}정도의 수를 이용한다.

2.4 Adam

RMSprop 알고리즘은 상수를 그래디언트의 제곱의 이동평균의 제곱근으로 나눠서 학습률을 결정하고 Adadelta 알고리즘은 $\triangle w_i^2$의 이동평균 rms와 그래디언트의 제곱의 이동평균 rms의 비율의 제곱근으로 학습률을 결정한다. 하지만 RMSprop과 Adadelta는 변화량을 누적하여 파라미터를 변화시키는 Momentum의 성질을 가지고 있지 않기 때문에 Momentum의 성질에 학습률을 변화시키는 성질을 추가한 알고리즘이 Adam이다. Adam 알고리즘은 다음과 같다.

$$w_{i+1} = w_i - \eta\frac{\widehat{M}_i}{\sqrt{\widehat{V}_i + \varepsilon}}, \quad (18)$$

$$\widehat{M}_i = M_i/(1-\beta_1^i),\ \widehat{V}_i = V_i/(1-\beta_2^i), \quad (19)$$

$$M_i = \beta_1 M_{i-1} + (1-\beta_1)\frac{\partial C(w_i)}{\partial w} \quad (20)$$

$$V_i = \beta_2 V_{i-1} + (1-\beta_2)\left(\frac{\partial C(w_i)}{\partial w}\right)^2 \quad (21)$$

여기서 $M_0 = V_0 = 0$이고 ε은 분모가 0이 되는 것을 막아주기 위한 상수로 10^{-7}을 쓰며 β_1, β_2는 이동평균(Moving Average) 상수로 주로 $\beta_1 = 0.9, \beta_2 = 0.999$를 이용한다[18,19].

여기에 가중치 감소 정규화를 적용하면 다음과 같은 방법으로 파라미터 w가 변화하게 된다.

$$w_{i+1} = w_i - \eta\frac{\widehat{M}_i}{\sqrt{\widehat{V}_i + \varepsilon}},$$

$$\widehat{M}_i = M_i/(1-\beta_1^i),\ \widehat{V}_i = V_i/(1-\beta_2^i),$$

$$M_i = \beta_1 M_{i-1} + (1-\beta_1)\left(\frac{\partial C(w_i)}{\partial w} + \lambda w_i\right) \quad (22)$$

$$V_i = \beta_2 V_{i-1} + (1-\beta_2)\left(\frac{\partial C(w_i)}{\partial w} + \lambda w_i\right)^2 \quad (23)$$

여기서 λ는 decay rate로 주로 10^{-3}정도의 수를 이용한다.

3. 비용함수를 이용한 최적화 알고리즘

1장과 2장에서 이야기했듯이 Momentum 알고리즘은 학습이 진행되면서 변화량이 계속 쌓이는 문제가 있었고, 이를 해결하기 위하여 Adam과 같은 알고리즘이 만들어졌다.

하지만 큰 학습 데이터(ex, 학습 데이터의 개수가 많아지거나, 학습 데이터의 size가 커지는 경우)와 복잡한 인공 신경망을 이용하는 기계학습의 경우에는 Momentum 알고리즘이 Adam 알고리즘보다 더 좋은 결과를 도출하는 경우가 있다. 그래서 우리는 Momentum 알고리즘에 새로운 정규화

$$argmin_w\{C(w) + \lambda_1 \| w \|_2^2 + \lambda_2 \int_{w-\epsilon}^{w+\epsilon} C\} \quad (24)$$

를 적용한 기계학습 최적화 알고리즘을 제안한다.

일반적인 최적화 $argmin_w\{C(w)\}$는 비용함수 $C(w)$를 최소화하는 파라미터 w를 찾는 것이 목적이었고, 가중치 감소 정규화를 적용한 최적화 $argmin_w\{C(w) + \lambda \| w \|_2^2\}$는 비용함수 $C(w)$와 파라미터 w를 최소화하는 w를 찾는 것이 목적이었다.

우리가 제안하는 최적화인 수식 (24)는 기존의 가중치 감소 정규화 과정에 추가적으로 학습을 통하여 얻은 파라미터 w의 근방(neighborhood) $(w-\epsilon,\ w+\epsilon)$에서의 비용함수의 면적이 적은 w를 찾는 과정이다.

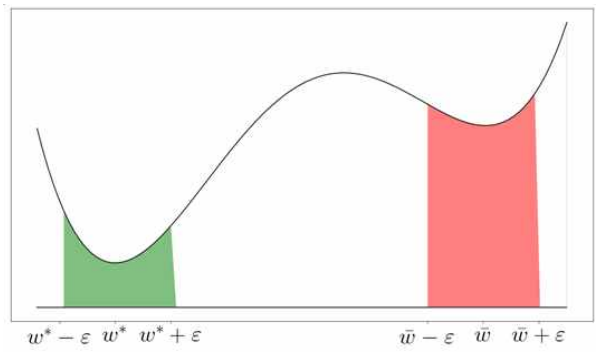


Fig. 2. Mechanism of proposed algorithm

Fig. 2는 우리가 제안하는 최적화 과정의 메커니즘을 나타낸다. Fig. 2에서 녹색 영역은 주어진 함수의 최솟값 w^*의 근방에서의 적분값을 나타내며 붉은 영역은 부분 최솟값 $\overline{w}$ 근방에서의 적분값을 나타낸다. 이것은 최솟값 근방에서의 적분값이 부분 최솟값 근방에서의 적분값보다 작다는 것을 보여준다. 그렇기 때문에 우리는 최적화 과정에 일정 구간에서 비용함수의 적분값 $\int_{w-\epsilon}^{w+\epsilon} C$를 줄이는 효과를 추가하였다.

우리가 제안하는 최적화 과정을 최적화 알고리즘에 적용한 것은 기존의 Momentum에 가중치 감소 정규화를 적용하고 추가로 비용함수의 값을 더한 알고리즘이다. 즉,

$$w_{i+1} = w_i - m_i,$$
$$m_i = \beta m_{i-1} + \eta\left(\frac{\partial C}{\partial w} + \lambda_1 w_i + \lambda_2 C(w_i)\right). \quad (25)$$

여기서 $m_0 = 0$ 이고, λ_1, λ_2는 정규화 상수이다.

과적합 문제를 해결하기 위해 가중치 감소 정규화를 위한 $\lambda_1 w_i$를 추가하였고, 부분 최솟값들 중에서 최솟값을 찾기 위해서 $\lambda_2 C(w_i)$를 추가하였다. 우리가 제안한 방법은 그래디언트 기반 방법에 모두 적용 가능하며, 그 중 Momentum 기반 방법이 가장 효과적이다.

4. 기계학습 수치 실험을 통한 성능 평가

이 장에서는 가중치 감소 정규화를 추가한 기존의 최적화 알고리즘들과 우리가 제안한 알고리즘을 기계학습에 적용하여 성능을 비교한다. 수치 실험에서 이용할 기계학습은 MNIST, CIFAR-10 그리고 CIFAR-100 데이터 집합을 이용한 이미지 분류 문제이다. CIFAR-10 과 CIFAR-100 데이터 집합을 이용한 수치 실험에서는 ResNet이라고 불리는 인공 신경망을 이용하였으며 ResNet 논문에서 가중치 감소 정규화의 decay rate λ를 10^{-4}을 이용했기 때문에 우리도 모든 수치 실험에서 λ를 10^{-4}로 설정하여 실험하였다.

4.1 MNIST

MNIST(Modified National Institute of Standards and Technology) 데이터 집합은 학습에 이용할 6만개의 학습 데이터와 학습이 끝난 후 테스트에 이용할 1만개의 테스트 데이터를 갖는 0부터 9까지의 손글씨 숫자 이미지 데이터 집합이다 (http://yann.lecun.com/exdb/mnist/). 하나의 이미지는 28×28의 크기를 갖는 흑백 이미지이다. Fig. 3은 MNIST 데이터 집합에 포함되는 이미지들의 일부이다.

MNIST 데이터 집합을 학습하는데 이용할 기계학습 인공 신경망은 2개의 합성곱(Convolution) 레이어와 1개의 FC(Fully Connected) 레이어로 이루어진 신경망이며 합성곱 레이어는 각각 (5, 5, 10), (3, 3, 10)의 크기를 갖는 합성곱 필터를 파라미터로 이용한다.

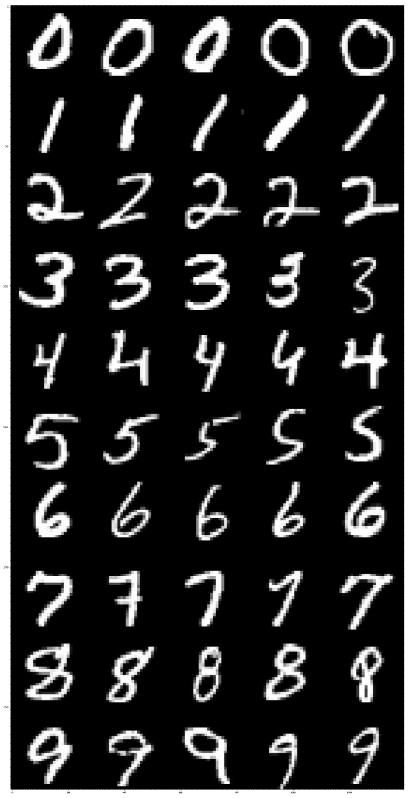
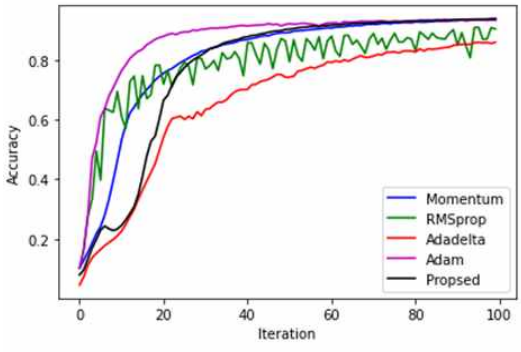


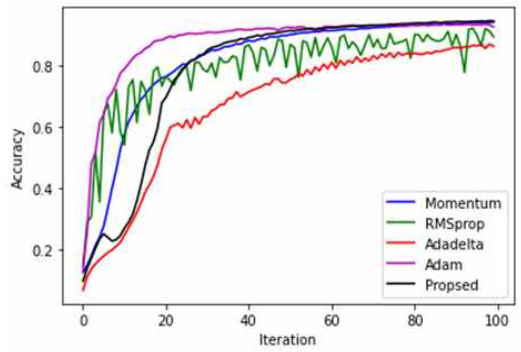
Fig. 3. Examples of MNIST dataset

MNIST를 이용한 최적화 알고리즘 성능 비교에 이용한 알고리즘들은 Momentum, Adadelta, RMSprop, Adam 그리고 이 논문에서 제안하는(Proposed) 알고리즘이다. 실험에 적용한 학습률은 Momentum, Adam, Proposed는 0.1을 이용하였고 Adadelta와 RMSprop는 학습률 0.1에서 결과가 좋지 않았기 때문에 각각 학습률 0.7과 0.05를 적용하여 실험하였고 실험 결과는 Fig. 4에 나타난다.

Fig. 4의 (a)는 학습 데이터를 이용하여 학습하는 도중의 정확도를 측정한 것이고 (b)는 학습하는 도중에 테스트 데이터를 이용하여 정확도를 측정한 것이다. Fig. 4의 그래프에서 파란색, 녹색, 빨간색, 보라색, 검은색은 각각 Momentum, RMSprop, Adadelta, Adam 그리고 제안한 알고리즘의 정확도를 나타낸다. 주어진 상황에서 RMSprop와 Adadelta는 학습이 잘 이루어지지 않는 것을 알 수 있고, Momentum, Adam 그리고 제안한 알고리즘은 비슷한 성능을 보이고 있으며, 이 실험에서의 각 알고리즘별 최대 정확도는 Table 1에 보여진다.



(a) Accuracy of training data for each algorithm



(b) Accuracy of test data for each algorithm.

Fig. 4. Result of subsection 4.1

Table 1. Maximum accuracy for each algorithm.

Method	Training data accuracy	Test data accuracy
Momentum	93.69 %	94.02 %
RMSprop	90.92 %	91.96 %
Adadelta	85.92 %	86.86 %
Adam	93.44 %	93.38 %
Proposed	93.74 %	94.45 %

Table 1은 Fig. 4에서의 정확도 중에서 최댓값을 나타내고 있으며, Momentum, Adam 그리고 제안한 알고리즘이 비슷한 성능을 나타내는 것을 알 수 있다.

MNIST 데이터 집합은 10종류의 흑백 이미지를 분류하는 문제에 이용되기 때문에 분류 문제에서는 간단한 편에 속한다. 다음 실험에는 컬러 이미지의 집합인 CIFAR-10을 이용하여 최적화 알고리즘의 성능을 비교한다.

4.2 CIFAR-10

CIFAR-10(Canadian Institute For Advanced

Research 10)은 비행기, 자동차, 새, 고양이, 사슴, 개, 개구리, 말, 배, 그리고 트럭의 32×32 크기의 10종류 컬러(RGB) 이미지를 갖는 데이터 집합이다(https://www.cs.toronto.edu/~kriz/cifar.html). MNIST 데이터 집합의 이미지는 28×28크기의 흑백 이미지였기 때문에 하나의 이미지가 784(=28×28)개의 디지털 신호를 갖지만 CIFAR-10 데이터 집합의 이미지 하나는 컬러 이미지이기 때문에 3072(=32×32×3)개의 디지털 신호를 갖고 그러기 때문에 기계학습에 난이도가 더 높다.

이 실험에서 CIFAR-10 데이터 집합을 학습시킬 인공신경망은 CNN을 기반으로 하는 ResNet 20을 이용하였다. ResNet은 2015년도 ILSVRC(Imagenet Large Scale Visual Recognition Challenge)에서 우승한 인공 신경망 모델이며 이 실험에서 이용한 ResNet 20은 ResNet의 20개의 레이어를 갖는 모델이다[20]. 3개의 레이어만을 이용하여 학습이 가능한 MNIST와는 다르게 CIFAR-10은 컬러 이미지이기 때문에 더욱 복잡한 인공신경망을 이용하여야 학습이 가능하다.

CIFAR-10의 학습에는 학습의 효과를 높이기 위하여 5만개의 학습 데이터를 128개씩 나누어서 200번 학습시켰다. 따라서 총 학습 횟수는 78,200번(=⌈50000/128⌉×200) 이다.

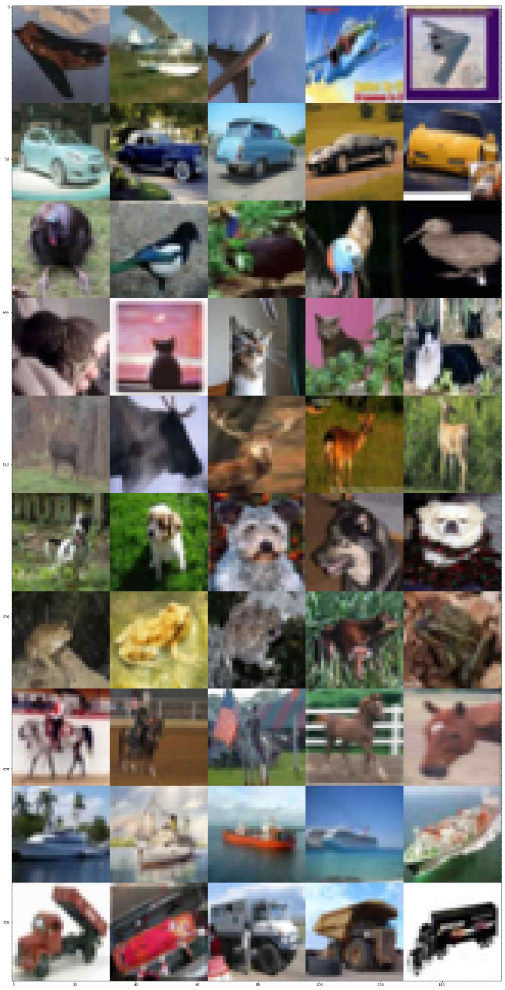
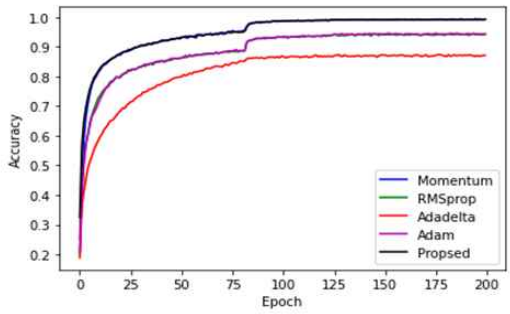


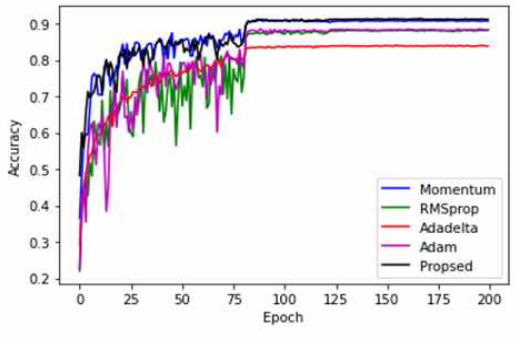
Fig. 5. Examples of CIFAR-10 dataset

또한 이 실험에서는 ResNet 신경망 논문에서 소개하는 것처럼 초기 학습률을 0.1로 설정하고 학습 횟수가 32,000번과 48,000번이 될 때마다 학습률을 10으로 나눈 후 이용한다. 즉, 학습 횟수가 0번부터 31,999번까지는 학습률로 0.1을 사용하고, 학습 횟수가 32,000번부터 47,999번까지는 학습률을 0.01로 사용하며, 학습 횟수가 48,000번 이상일때는 학습률을 0.001로 사용한다. 우리가 제안하는 알고리즘에서는 정규화 상수 λ_1, λ_2를 각각 $10^{-5}, 10^{-6}$으로 설정하여 실험을 하였다.

Fig. 6은 CIFAR-10 데이터 집합을 ResNet 20 신경망을 이용하여 각 최적화 알고리즘으로 학습시킨 결과로 (a)는 학습 데이터로 학습 중의 정확도를 나타내며 (b)는 학습 도중 테스트 데이터를 이용하여 측정한 정확도이다. 학습 초반에 정확도의 변화가 많아서 학습 후반부에서의 각 알고리즘들의 정확도 차이가 Fig. 6에서 보기 어렵기 때문에 Epoch이 75 이후일 때의 그래프를 Fig. 7에 나타내었다.

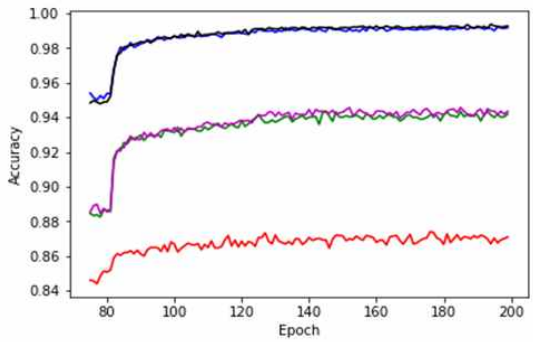


(a) Accuracy of training data for each algorithm.

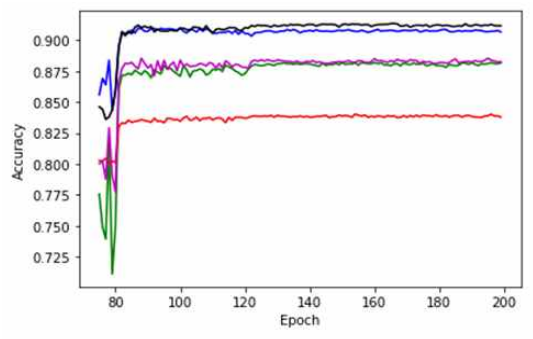


(b) Accuracy of test data for each algorithm.

Fig. 6. Performance on the CIFAR-10 dataset



(a) Accuracy of training data for each algorithm.



(b) Accuracy of test data for each algorithm.

Fig. 7. Performance on the CIFAR-10 dataset after 75 epoch

Fig. 7의 (a)와 (b)는 각각 Fig. 6의 (a)와 (b)에서의 그래프를 75 epoch 이후만 나타낸 그래프이다. Fig. 7에서 보여주듯이 Momentum과 제안한 알고리즘이 가장 높은 정확도를 나타내며, 이 실험에서의 Momentum의 최대 정확도는 91.00%이며 제안한 알고리즘의 최대 정확도는 91.33%이다.

4.1장과 4.2장에서는 각 최적화 알고리즘의 성능을 비교하기 위하여 10종류를 분류하는 기계학습 데이터 집합을 이용하였다. 4.3에서는 4.1과 4.2보다 더 복잡한 데이터 집합을 이용하여 각 최적화 알고리즘의 성능을 비교한다.

4.3 CIFAR-100

이 실험에서는 CIFAR-100 이라고 불리는 100가지 종류의 컬러 이미지를 갖는 CIFAR-10의 확장된 데이터 집합을 이용한다.

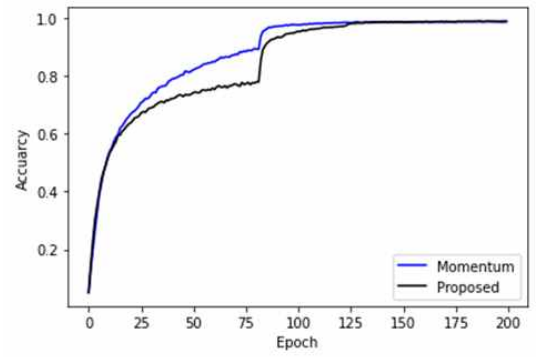
CIFAR-100은 100종류의 이미지가 각각 500개씩 학습 데이터로 주어진다. 그렇기 때문에 10종류의 이미지가 각각 5000개씩 주어진 CIFAR-10보다 더 많은 종류의 이미지를 더 적은 수의 데이터를 이용하여 학습하기 때문에 이전에 사용했던 MNIST나 CIFAR-10보다 학습을 하기가 더 어렵다. 따라서 이 실험에서는 4.2장에서 이용했던 인공 신경망인 ResNet 20의 확장버전인 56개의 레이어를 갖는 ResNet 56을 이용하여 학습을 진행하며 학습률과 학습 횟수는 4.2장에서와 같은 값을 이용한다. 우리가 제안하는 알고리즘에서는 정규화 상수

Fig. 8. Examples of CIFAR-100 dataset

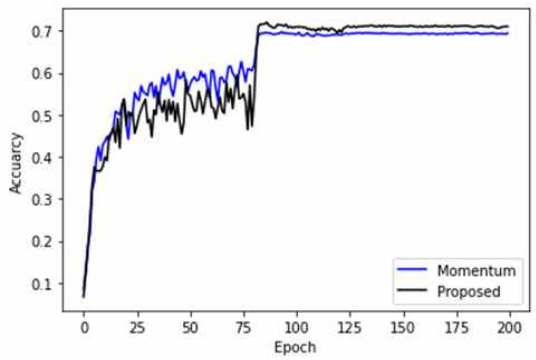
λ_1, λ_2를 각각 $10^{-4}, 10^{-6}$으로 설정하여 실험을 하였다.

Fig. 8은 CIFAR-100의 100가지 종류의 이미지를 하나씩 보여주고 있다.

4.2장의 수치 실험 결과에서 나타나듯이 학습 데이터 정확도와 테스트 데이터 정확도 모두에서 Momentum과 제안한 알고리즘이 가장 좋은 성능을 보이기 때문에 이 장에서는 Momentum과 제안한 알고리즘의 성능만 비교를 한다.



(a) Accuracy of training data for Momentum and Proposed algorithm.



(b) Accuracy of test data for Momentum and Proposed algorithm.

Fig. 9. Result of subsection 4.3

Fig. 9는 CIFAR-100 데이터 집합을 ResNet 56 신경망을 이용하여 Momentum과 제안한 알고리즘으로 학습시킨 결과로 (a)는 학습 데이터로 학습 중의 정확도를 나타내며 (b)는 학습 도중 테스트 데이터를 이용하여 측정한 정확도이다.

Fig. 9에서 (a)의 그래프를 보면 학습에 이용하는 학습 데이터로 학습 후에 정확도를 측정하기 때문에 Momentum과 제안한 알고리즘 모두 100%에 근접한 정확도를 보이고 있다. (b)에서 보여주는 테스트 데이터 정확도는 (a)에서의 학습 데이터 정확도 보다 예측이 어려운데 (b)에서 Momentum의 최대 정확도는 69.78%이고 제안한 알고리즘의 최대 정확도는 72.03%로 제안한 방법이 더 높은 정확도를 보이고 있다.

5. 결론

복잡한 신호의 학습 데이터 일수록 학습하는데 많은 파라미터를 갖는 인공 신경망이 요구된다. 이러한 요구사항은 과적합 문제를 야기한다. 최근에는 많은 연구자들이 과적합 문제를 해결하는 하나의 방법으로 가중치 감소 정규화 방법을 사용하고 있다. 우리는 이 방법을 Momentum 방법에 적용하고 부분 최솟값을 해결하기 위하여 비용함수를 추가하여 이 문제를 해결하였다. 과적합 문제도 해결했으며, 부분 최솟값으로 수렴하는 현상을 방지하였다. 실험 4.1은 레이어 2 정도의 단순한 인공신경망을 이용한 실험에서는 비용함수를 추가한 효과가 크게 나타나지는 않았다. 그러나 데이터와 인공신경망이 복잡한 실험 4.3(레이어 56개, 학습 데이터 5만개, 100가지 종류의 출력값)인 경우에는 비용함수를 추가함으로 정확도가 높게 나타났다. 더 복잡한 인공신경망에서도 우리가 제안한 방법이 더 효과가 있는지 연구해 볼 필요가 있다. 과학 기술의 발달로 매우 높은 화질의 사진과 영상을 찍을 수 있는 장비들이 개발되고 있고 이에 따라 더 복잡하고 큰 디지털 신호를 처리하는 능력이 요구되고 있다. 따라서 이러한 복잡하고 큰 디지털 신호를 처리하는데 우리가 제안한 알고리즘을 이용하여 보다 높은 성과를 얻을 수 있다.

REFERENCES

[1] I. Goodfellow, Y. Bengio, & A. Courville. (2016). *Regularization for deep learning. In Deep Learning.* Cambridge : MIT Press.

[2] H. Zhao, Y. H. Tsai, R. Salakhutdinov, & G. J. Gordon. (2019). Learning Neural Networks with Adaptive Regularization. arXiv:1907.06288v2.

[3] Y. Zheng, R. Zhang, & Y. Mao. (2021). Regularizing Neural Networks via Adversarial Model Perturbation. arXiv:2010.04925v4.

[4] Y. Wang, Z. P. Bian, J. Hou, & L. P. Chau. (2021). Convolutional Neural Networks With Dynamic Regularization. *IEEE Transactions on Neural Networks*

and Learning Systems, 32(5), 2299-2304.

[5] J. Duchi, E. Hazan, & Y. Singer. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res. 12,* 2121-2159.

[6] Y. LeCun, L. Bottou, Y. Bengio, & P. Haffner. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE, 86,* 2278-2324.

[7] R. Pascanu, & Y. Bengio. (2013). Revisiting natural gradient for deep networks. arXiv:1301.3584.

[8] J. Sohl-Dickstein, B. Poole, & S. Ganguli. (2014). Fast large-scale optimization by unifying stochastic gradient and quasi-newton methods. *In Proceedings of the 31st International Conference on Machine Learning (ICML-14),* Beijing, China, 21-26 June, 604-612.

[9] S. Chaudhury, & T. Yamasaki. (2021). Robustness of Adaptive Neural Network Optimization Under Training Noise. *IEEE Access, 9,* 37039-37053.

[10] R. He, L. Liu, H. Ye, Q. Tan, B. Ding, L. Cheng, J. W. Low, L. Bing, & L. Si. (2021). On the Effectiveness of Adapter-based Tuning for Pretrained Language Model Adaptation. arXiv:2106.03164v1.

[11] C. T. Kelley. (1995). *Iterative methods for linear and nonlinear equations.* In Frontiers in Applied Mathematics; SIAM: Philadelphia, PA, USA. Volume 16.

[12] H. Zulkifli. (2018). *Understanding Learning Rates and How It Improves Performance in Deep Learning.* https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10

[13] S. Lau. (2017). *Learning Rate Schedules and Adaptive Learning Rate Methods for Deep Learning. Towards Data Science.* https://towardsdatascience.com/learning-rate-schedules-and-adaptive-learning-rate-methods-for-deep-learning-2c8f433990d1.

[14] G. Aurélien. (2017). *Gradient Descent.* Hands-On Machine Learning with Scikit-Learn and TensorFlow. O'Reilly. pp. 113-124. ISBN 978-1-4919-6229-9.

[15] I. Sutskever, J. Martens, G. Dahl, & G.E. Hinton. (2013). On the importance of initialization and momentum in deep learning. *In Proceedings of the 30th International Conference on Machine Learning (ICML-13),* Atlanta, GA, USA, 16-21 June, 1139-1147.

[16] T. Tieleman, & G.E. Hinton. (2012). *Lecture 6.5—RMSProp, COURSERA: Neural Networks for Machine Learning.* Technical Report, University of Toronto, Toronto, ON, Canada.

[17] M. D. Zeiler. (2012). Adadelta: An adaptive learning rate method. arXiv:1212.5701.

[18] D. P. Kingma, & J. L. Ba. (2015). Adam: A Method for Stochastic Optimization. *In Proceedings of the 3rd International Conference for Learning Representations, ICLR 2015,* San Diego, CA, USA, 7-9 May

[19] M. J. Kochenderfer, & T. A. Wheeler. (2019). *Algorithms for Optimization.* Cambridge: The MIT Press.

[20] K. He, X. Zhang, S. Ren, & J. Sun. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Las Vegas, NV, USA, 27-30 June, 770-778.

지 상 민(Sangmin Ji) [정회원]

· 2017년 2월 : 충남대학교 수학과(이학사)
· 2017년 3월 ~ 현재 : 충남대학교 수학과 대학원 석박통합과정
· 관심분야 : 인공지능, 기계학습, 수치해석, 대수학
· E-Mail : smji@cnu.ac.kr

박 지 은(Jieun Park) [정회원]

· 1998년 2월 : 이화여자대학교 과학교육과(화학전공) (학사)
· 2013년 2월 : 이화여자대학교 과학교육학과 (박사)
· 2015년 3월 ~ 현재 : 대구대학교 성산교양대학 부교수
· 관심분야 : 문제해결력, 표상, 인공지능
· E-Mail : writer2yah@daegu.ac.kr