KIPS Trans. Softw. and Data Eng.
Vol.11, No.1 pp.1~10
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2022.11.1.1

Performance Improvement Method of Fully Connected Neural Network Using Combined Parametric Activation Functions

Young Min Ko† · Peng Hang Li†† · Sun Woo Ko†††

ABSTRACT

Deep neural networks are widely used to solve various problems. In a fully connected neural network, the nonlinear activation function is a function that nonlinearly transforms the input value and outputs it. The nonlinear activation function plays an important role in solving the nonlinear problem, and various nonlinear activation functions have been studied. In this study, we propose a combined parametric activation function that can improve the performance of a fully connected neural network. Combined parametric activation functions can be created by simply adding parametric activation functions. The parametric activation function is a function that can be optimized in the direction of minimizing the loss function by applying a parameter that converts the scale and location of the activation function according to the input data. By combining the parametric activation functions, more diverse nonlinear intervals can be created, and the parameters of the parametric activation functions can be optimized in the direction of minimizing the loss function. The performance of the combined parametric activation function was tested through the MNIST classification problem and the Fashion MNIST classification problem, and as a result, it was confirmed that it has better performance than the existing nonlinear activation function and parametric activation function.

Keywords : Fully Connected Neural Network, Nonlinear Activation Function, Combined Parametric Activation Function, Learning

결합된 파라메트릭 활성함수를 이용한 완전연결신경망의 성능 향상

고 영 민† · 이 붕 항†† · 고 선 우†††

요 약

완전연결신경망은 다양한 문제를 해결하는데 널리 사용되고 있다. 완전연결신경망에서 비선형활성함수는 선형변환 값을 비선형 변환하여 출력하는 함수로써 비선형 문제를 해결하는데 중요한 역할을 하며 다양한 비선형활성함수들이 연구되었다. 본 연구에서는 완전연결신경망의 성능을 향상시킬 수 있는 결합된 파라메트릭 활성함수를 제안한다. 결합된 파라메트릭 활성함수는 간단히 파라메트릭 활성함수들을 더함으로써 만들어낼 수 있다. 파라메트릭 활성함수는 입력데이터에 따라 활성함수의 크기와 위치를 변환시키는 파라미터를 도입하여 손실함수를 최소화하는 방향으로 최적화할 수 있는 함수이다. 파라메트릭 활성함수들을 결합함으로써 더욱 다양한 비선형간격을 만들어낼 수 있으며 손실함수를 최소화하는 방향으로 파라메트릭 활성함수들의 파라미터를 최적화할 수 있다. MNIST 분류문제와 Fashion MNIST 분류문제를 통하여 결합된 파라메트릭 활성함수의 성능을 실험하였고 그 결과 기존에 사용되는 비선형활성함수, 파라메트릭 활성함수보다 우수한 성능을 가짐을 확인하였다.

키워드 : 완전연결신경망, 비선형활성함수, 결합된 파라메트릭 활성함수, 학습

1. 서 론

선형변환을 가지고 Sigmoid와 같이 유계함수와 같은 비선형활성함수를 가지는 하나 이상의 은닉층을 사용하는 순방향 신경망은 임의의 함수를 근사할 수 있다[1]. 비선형활성함수를 가지는 은닉층의 개수가 하나만으로도 노드가 충분히 많으면 임의의 함수를 근사할 수 있으나 보통의 경우 많은 수의 은닉층을 가지는 신경망이 일반화 능력이 개선되고 필요한 파라미터 수를 줄일 수 있다[2].

순방향 신경망의 한 종류인 완전연결신경망(Fully connected neural network)은 모든 은닉층의 모든 노드들이 순방향으로 연결되어 있는 신경망이다. 각 은닉층은 선형변환 단계와 비선형활성함수를 이용한 비선형변환을 포함하고 있다. 비선형활성

† 준 회 원 : 전주대학교 인공지능학과 석사과정
†† 비 회 원 : 전주대학교 인공지능학과 석사과정
††† 정 회 원 : 전주대학교 인공지능학과 교수
Manuscript Received : May 31, 2021
First Revision : July 7, 2021
Accepted : July 28, 2021
* Corresponding Author : Sun Woo Ko(godfriend0@gmail.com)

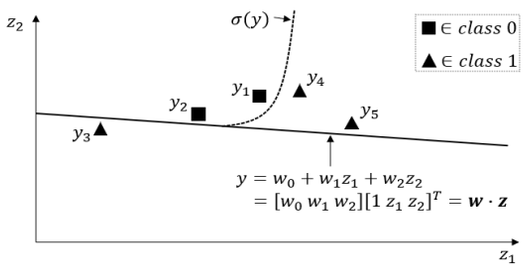


Fig. 1. Nonlinear Classification Problem with 2 Classes (w: Parameter, z: Input Value)

함수를 가지는 완전연결신경망의 은닉층 개수가 많아질 경우, 완전연결신경망의 손실함수는 볼록함수(Convex function)를 보장할 수 없으며 손실함수의 미분계수가 0이 되는 점을 경사하강법을 통해 찾았을 때 전역최솟값(Global minimum)을 보장할 수 없다.

완전연결신경망에서 비선형활성함수는 선형변환 값을 비선형 변환하여 출력하는 함수로써 비선형활성함수의 의미를 보기 위해 2개 클래스를 구분하는 비선형 분류문제 Fig. 1을 생각해 보자. 입력변수 (z_1, z_2)을 통해 구해진 결과값 $y_i, i=1,2,...,5$들이 있고 각 y_i는 클래스 0 또는 클래스 1을 가진다고 했을 때 y_i을 클래스 0 또는 클래스 1에 분류하는 문제이다. z_1, z_2의 선형변환으로 만들어지는 초평면 y가 경사하강법을 통해 y_i들이 속하는 클래스를 최대한 나눈 뒤, 비선형활성함수 σ을 통해 y_i들을 비선형 변환하여 분류성능을 향상시킨다.

비선형활성함수의 중요한 성질은 비선형성이며 본 논문은 파라메트릭 활성함수[3-5]를 결합하여 만든 결합된 파라메트릭 활성함수가 완전연결신경망의 성능을 향상시킬 수 있음을 보인다. 결합된 파라메트릭 활성함수를 구성하고 있는 파라메트릭 활성함수는 입력데이터에 따라 활성함수의 크기와 위치를 변환시키는 파라미터를 도입하여 손실함수를 최소화하는 방향으로 최적화할 수 있는 함수이다[5]. 이런 파라메트릭 활성함수는 기존의 비선형활성함수보다 다양한 비선형변환 값을 입력데이터의 특성에 따라 학습할 수 있지만 만들어낼 수 있는 비선형간격이 제한되어 있다.

더욱 다양한 비선형간격을 만들고 이를 입력데이터에 따라 손실함수를 최소화하는 방향으로 학습할 수 있는 결합된 파라메트릭 활성함수는 파라메트릭 활성함수를 간단히 더함으로써 구현된다. 결합된 파라메트릭 활성함수를 구성하고 있는 각각의 파라메트릭 활성함수의 파라미터는 손실함수를 최소화하는 방향으로 학습되며 기존에 사용되는 비선형 활성함수보다 필요한 은닉층과 노드의 개수를 줄일 수 있다.

2. 관련 연구

완전연결신경망의 성능을 향상시키기 위한 비선형활성함수에 대한 연구는 Sigmoid같은 S자형 함수와 ReLU(Rectified linear unit)[6] 함수의 연구로 크게 2가지로 구분할 수 있다.

S자형 함수에서 대표적으로 Sigmoid와 Tanh이 있으며 각각의 치역은 (0,1), (-1,1)을 갖고 미분가능하며 일대일 대응인 단조함수로써 유용한 수학적 성질을 갖고 있다[7]. Sigmoid와 Tanh의 다양한 함수변형이 있으며 대표적으로 ISigmoid[8], ReLTanh[9], Hexpo[10] 등이 있다.

ISigmoid[8]와 ReLTanh[9]은 각각 Sigmoid와 Tanh에 LReLU(Learky ReLU)[11]를 결합한 형태로써 함수의 중심부분은 Sigmoid와 Tanh을 사용하여 비선형성을 키우고 기울기가 작아지는 함수의 양쪽 끝부분은 LReLU의 직선으로 구성하여 기울기 소실을 방지할 수 있다. ISigmoid는 Sigmoid에서 직선으로 교체되는 구간을 정하는 파라미터 a와 직선의 기울기를 변환시키는 파라미터 α가 있다. ReLTanh도 마찬가지로 직선으로 교체되는 구간을 정하는 파라미터 λ^-, λ^+가 있지만 ISigmoid의 파라미터와 다르게 λ^+와 λ^-은 각각 양수와 음수 구간에서 직선으로 교체되는 구간과 기울기를 결정한다. ISigmoid와 ReLTanh의 파라미터들은 경사하강법을 사용하여 학습할 수 있다.

Hexpo[10]는 기울기 소실 문제를 해결하기 위해 소개된 비선형활성함수로 0점을 기준으로 양수 구간에 대해 기울기 크기를 결정하는 파라미터 a,b와 음수 구간에 대해 기울기 크기를 결정하는 파라미터 c,d을 도입한 함수이다. Hexpo는 4개의 파라미터에 의해 좀 더 다양한 비선형변환을 할 수 있으며 MNIST 분류문제에서 ReLU 보다 학습속도와 손실함수 값에서 우수한 성능을 가짐을 보였다.

ReLU 함수는 0보다 큰 구간에 대해서는 입력 값을 그대로 출력하고 아닌 구간에 대해서는 0을 출력하는 함수로써 근래 신경망에서 보편적으로 사용되는 비선형활성함수이다. ReLU 함수는 양수 구간에 대해서 기울기 값을 1을 가짐으로써 기울기 소실 문제를 완화할 수 있다. 하지만 기울기 폭발 문제[12]가 발생할 수 있으며 Pascanu 등[13]은 기울기 폭발 문제를 해결하기 위해 임계값을 지정하여 기울기가 임계값을 초과하면 기울기를 자르는 Gradient clipping을 제안하였다.

ReLU의 다양한 함수변형이 연구되었으며 대표적으로 LReLU, PReLU(Parametric ReLU)[14], ELU(Exponential linear unit)[15] 등이 있다.

Xu 등[11]이 연구한 LReLU는 0보다 작은 음수 구간은 출력하지 않는 ReLU가 가지는 문제점을 보완하기 위해 음수 구간을 0.01을 곱하여 출력하도록 하는 함수이다.

He 등[14]에 의해 연구된 PReLU는 음수 구간에 0.01을 곱한 LReLU와 다르게 0.01을 학습할 수 있는 파라미터 α로 대체한 함수로써 He 등[14]에 의해 소개된 파라미터 초기화 방법을 함께 사용하여 은닉층 수가 30개가 넘는 구조에 대해서 수렴함을 보였다.

Clevert 등[15]이 제안한 ELU는 입력 값 y을 가지는 ReLU 함수에서 0보다 작은 음수 구간을 $\alpha(e^y-1)$로 대체한 함수로 파라미터 α는 보통 1로 설정한다. LReLU, PReLU와 다르게 0점에서 함수 값이 매끄럽게 변하는 특징이 있다.

마지막으로 공나영 등[3]이 소개한 파라메트릭 활성함수는

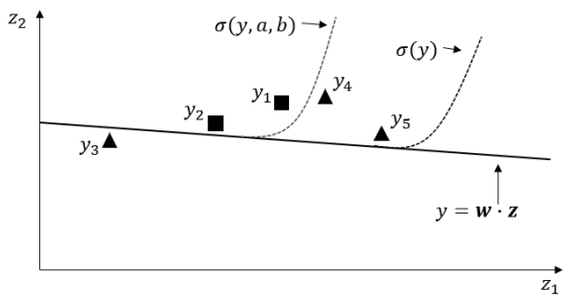


Fig. 2. Parametric Activation Function $\sigma(y,a,b)$ and Activation Function $\sigma(y)$ in Classification Problem

Sigmoid, ReLU 뿐만 아니라 임의의 비선형활성함수에 크기와 위치를 결정하는 파라미터를 적용하여 손실함수를 최소화하는 방향으로 학습할 수 있다.

3. 결합된 파라메트릭 활성함수

3.1 비선형활성함수와 결합된 파라메트릭 활성함수

경사하강법을 사용하여 신경망을 학습하는 것은 손실함수 값을 최소화하는 선형변환층 파라미터 $\boldsymbol{w}$을 계산하는 것으로 Fig. 1과 같이 비선형 분류문제를 구분할 수 있다. 하지만 Fig. 2에서 학습 시에 $\sigma(y)$과 같이 비선형활성함수 σ가 비선형 변환하는 정도는 손실함수와 무관하게 비선형 변환하여 손실함수를 증가시킬 위험이 있고 이로 인해 경사하강법의 학습 횟수를 증가시킬 수 있다.

공나영 등[3,4]과 고영민 등[5]은 이 문제를 개선하기 위해 비선형활성함수의 형태를 손실함수 값이 최소화되는 방향으로 최적화할 수 있는 파라미터를 적용한 파라메트릭 활성함수를 소개하였다. 비선형활성함수의 크기와 위치를 결정하는 파라미터 a,b을 적용한 파라메트릭 활성함수는 Fig. 2에서 $\sigma(y,a,b)$을 나타내며 입력 데이터의 특성을 반영하여 손실함수를 최소화하는 방향으로 최적화할 수 있어 $\sigma(y)$에 비해 분류 성능을 향상시킬 수 있다.

제안된 파라메트릭 활성함수는 임의의 비선형활성함수에 적용할 수 있으며 간단한 예로 Sigmoid와 ReLU에 크기와 위치를 결정하는 파라미터 a,b을 적용한 파라메트릭 활성함수는 Table 1과 같다. 그리고 파라메트릭 활성함수의 파라미터 a,b과 입력 y에 의한 파라메트릭 활성함수의 변화율은 Table 2와 같다.

Table 1에서 파라미터 a는 파라메트릭 활성함수에서 크기를 변환할 수 있고 파라미터 b는 위치를 변환할 수 있다. a,b 파라미터 모두 임의의 실수값을 가지며 Table 2를 사용하여 경사하강법을 통해 학습할 수 있다. Parametric PN Sigmoid (PPNS)와 Parametric PN ReLU(PPNR)에서 PN은 Positive, Negative value를 의미하며 Parametric Sigmoid와 Parametric ReLU 각각의 함수에 $-a/2$을 더한 형태로써 변환된 활성함수 값이 양수값과 음수값을 가지도록 만든 활성함수이다.

Table 1. Various Parametric Activation Functions

	Parametric activation function formula	Note
Parametric activation function	$z = \sigma_{(a,b)}(y)$	Common form
Parametric Sigmoid	$z = \frac{a}{1+e^{-(y-b)}}$	Transform Sigmoid
Parametric PN Sigmoid	$z = \frac{a}{1+e^{-(y-b)}} - \frac{a}{2}$	
Parametric ReLU	$z = \begin{cases} a(y-b), & y \geq b \\ 0, & y < b \end{cases}$	Transform ReLU
Parametric PN ReLU	$z = \begin{cases} a(y-b) - \frac{a}{2}, & y \geq b \\ -\frac{a}{2}, & y < b \end{cases}$	

Table 2. Gradients of Various Parametric Activation Functions

	$\frac{\partial z}{\partial y}$	$\frac{\partial z}{\partial a}$	$\frac{\partial z}{\partial b}$
Parametric Sigmoid	$z(1-\frac{z}{a})$	$\frac{1}{1+e^{-(y-b)}}$	$z(\frac{z}{a}-1)$
Parametric PN Sigmoid	$z(1-\frac{z}{a})$	$\frac{1}{1+e^{-(y-b)}} - \frac{1}{2}$	$z(\frac{z}{a}-1)$
Parametric ReLU	$\begin{cases} a, & y \geq b \\ 0, & y < b \end{cases}$	$\begin{cases} y-b, & y \geq b \\ 0, & y < b \end{cases}$	$\begin{cases} -a, & y \geq b \\ 0, & y < b \end{cases}$
Parametric PN ReLU	$\begin{cases} a, & y \geq b \\ 0, & y < b \end{cases}$	$\begin{cases} y-b-\frac{1}{2}, & y \geq b \\ -\frac{1}{2}, & y < b \end{cases}$	$\begin{cases} -a, & y \geq b \\ 0, & y < b \end{cases}$

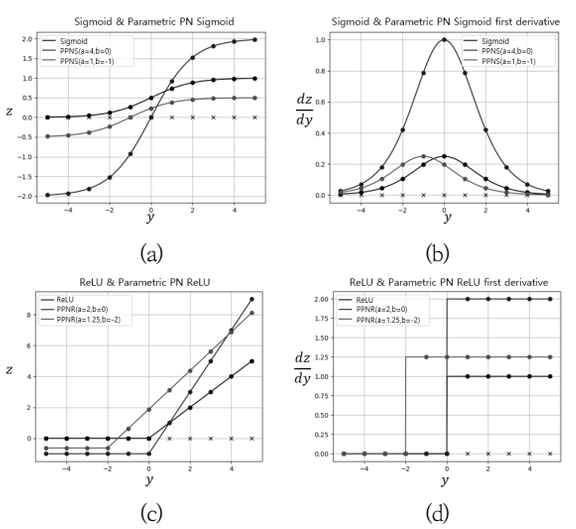
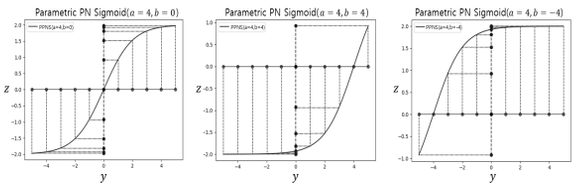


Fig. 3. Comparison of Parametric Activation Function and Activation Function (a),(b) : Sigmoid vs PPNS (c),(d) : ReLU vs PPNR

파라메트릭 활성함수를 보다 자세히 보기 위해 Fig. 3은 기존에 사용되는 활성함수와 파라메트릭 활성함수를 비교한 그림으로 Fig. 3(a)은 선형변환 y값에 대한 비선형활성함수 Sigmoid와 PPNS의 비선형변환 값을, Fig. 3(b)은 Sigmoid와 PPNS을 사용할 때 y에 의한 z의 변화율을 나타낸 그림이다. 마찬가지로 Fig. 3(c)과 Fig 3(d)은 ReLU와 PPNR에 대



(a) (b) (c)

Fig. 4. PPNS z Value According to Equal Interval y
(a) : $\sigma_{(4,0)}(y)$, (b) : $\sigma_{(4,4)}(y)$, (c) : $\sigma_{(4,-4)}(y)$

해서 비교한 그림이다. Fig. 3(a)에서 PPNS의 a,b값이 변함에 따라 고정된 비선형변환 값을 가지는 Sigmoid와 다르게 좀 더 자유로운 비선형변환 값을 가질 수 있음을 알 수 있다. Fig. 3(b)을 보면 $a=4$일 때 dz/dy값이 최대 1임을 볼 수 있으며 b값에 따라 y에 대응하는 dz/dy값이 달라지는 것을 알 수 있다. 즉 파라메트릭 활성함수의 파라미터 a,b은 Fig. 3과 같이 다양한 비선형변환을 할 수 있으며 이때 입력 데이터에 따른 손실함수 값을 최소화하는 방향으로 변환하게 된다.

Fig. 4는 비선형활성함수의 비선형성을 보기 위해 등간격 y값에 대응하는 파라메트릭 활성함수 PPNS z값을 위치를 변환시키는 파라미터 b을 변화시켜 나타낸 것이다. Fig. 4는 크게 0점을 중심으로 3가지 유형의 비선형 패턴이 나타나며 다음과 같다.

1) Fig. 4(a)의 경우 : 0점에서 멀어질수록 간격이 좁아짐.
2) Fig. 4(b)의 경우 : y값이 증가함에 따라 간격이 넓어짐.
3) Fig. 4(c)의 경우 : y값이 증가함에 따라 간격이 좁아짐.

즉 PPNS의 경우, Fig. 4에서 만들어낼 수 있는 z값의 간격이 3가지 유형으로 한정되어 있다. 만약 등간격 y값에 대응하는 z값의 간격이 좀 더 복잡한 패턴을 만들어낼 수 있고 이를 입력데이터 특성에 따라 변환할 수 있다면 필요한 노드 수와 은닉층 수가 줄어들 가능성이 있다.

이 점을 개선하기 위해 결합된 파라메트릭 활성함수를 제안한다. 결합된 파라메트릭 활성함수는 간단히 파라메트릭 활성함수들을 더한 형태로 구해지며(Table 3), 결합된 파라메트릭 활성함수의 파라미터 a_i,b_i 그리고 y에 대한 활성함수의 변화율은 Table 4와 같다.

결합된 파라메트릭 활성함수는 앞서 소개된 파라메트릭 활성함수들의 단순한 합으로 나타나며 각각의 파라메트릭 활성함수의 파라미터 a_i,b_i값들을 가지고 있다. Table 3에 Combined Parametric PN Sigmoid(C PPNS)와 Combined Parametric PN ReLU(C PPNR)은 간단히 각각 k개의 PPNS들의 합, PPNR들의 합으로 이루어져 있다. C PPN(S,R)은 C PPNS와 C PPNR의 합으로 구성되어 있다.

예를 들어 등간격 선형변환 y값에 대응하는 결합된 파라메트릭 활성함수의 그림이 Fig. 5에 나타나있다. Fig. 5(a)는 $k=2$인 C PPNS을 나타낸 그림으로 C PPNS의 파라미터를 $(a_1=-4, b_1=4)$, $(a_2=4, b_2=4)$로 주었을 때 z값의 간격을 보면 다양한 비선형 간격을 만들어낼 수 있다. 즉 Fig. 4에서 하나의 PPNS를 사용하였을 때 비선형 간격의 패턴보다 C PPNS를 사용하는 Fig. 5(a)에서 비선형 z값의 간격을 보면 선형변환 y값이 증가함에 따라 비선형 간격이 증가와 감소를 각각 2번하는 것을 볼 수 있다. 파라메트릭 활성함수 개수를 나타내는 k와 각각의 파라미터 a_i,b_i값에 따라 더욱 다양한 비선형 간격을 만들어 낼 수 있다. Fig. 5(b)는 4가지의 서로 다른 a_i,b_i 값을 가지고 있는 PPNR을 결합하여 만든 C PPNR로 ReLU보다 다양한 비선형성을 만들어낼 수 있음을 볼 수 있다.

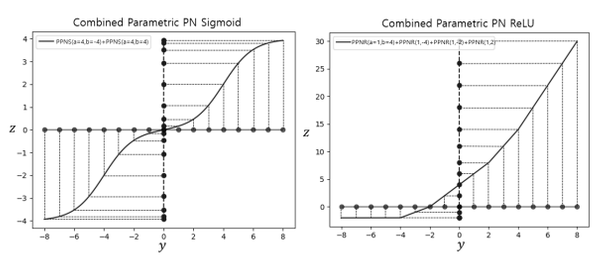
Table 3. Combined Parametric Activation Functions

	Combined parametric activation function formula
Combined parametric activation function (C PPNA)	$z=\sum_{i=1}^{k} z_i, z_i=\sigma_{(a_i,b_i)}(y)$
Combined Parametric PN Sigmoid(C PPNS)	$z=\sum_{i=1}^{k} z_i, z_i=\left(\frac{a_i}{1+e^{-(y-b_i)}}-\frac{a_i}{2}\right)$
Combined Parametric PN ReLU(C PPNR)	$z=\sum_{i=1}^{k} z_i, z_i=\begin{cases} a_i(y-b_i)-\frac{a_i}{2}, & y>b_i \\ -\frac{a_i}{2}, & y\le b_i \end{cases}$
Combined Parametric PN Sigmoid & ReLU (C PPN(S,R))	C PPNS + C PPNR

Table 4. Gradients of Combined Parametric Activation Functions

	$\frac{\partial z}{\partial y}$	$\frac{\partial z}{\partial a_i}$	$\frac{\partial z}{\partial b_i}$
C PPNS	$\sum_{i=1}^{k}\left(z_i\left(1-\frac{z_i}{a_i}\right)\right)$	$\frac{1}{1+e^{-(y-b_i)}}-\frac{1}{2}$	$z_i\left(\frac{z_i}{a_i}-1\right)$
C PPNR	$\sum_{i=1}^{k}\begin{cases} a_i, & y>b_i \\ 0, & y\le b_i \end{cases}$	$\begin{cases} y-b_i-\frac{1}{2}, & y>b_i \\ -\frac{1}{2}, & y\le b_i \end{cases}$	$\begin{cases} -a_i, & y>b_i \\ 0, & y\le b_i \end{cases}$

Table 4와 같이 결합된 파라메트릭 활성함수는 경사하강법을 사용하여 각각의 a_i,b_i값들이 모두 입력 데이터의 특성에 따라 손실함수를 최소화할 수 있게 학습할 수 있다. 이처럼 좀 더 복잡한 비선형 간격을 만들어냄으로써 필요한 신경망의 노드, 은닉층의 개수를 줄일 가능성이 있다.



(a) (b)

Fig. 5. Combined Parametric Activation Functions
(a) : (C PPNS$(k=2)$: $(a_1=4, b_1=4)$,$(a_2=4, b_2=-4)$)
(b) : (C PPNR$(k=4)$: $(a_1=1, b_1=4)$,$(a_2=1, b_2=-4)$, $(a_3=1, b_3=-2)$, $(a_4=1, b_4=2)$)

3.2 결합된 파라메트릭 활성함수의 학습

결합된 파라메트릭 활성함수의 파라미터를 학습하기 위해 먼저 일반적인 완전연결신경망의 구조 Fig. 6을 고려하자.

Fig. 6은 전체 K개의 은닉층과 출력층 그리고 n_0개 입력변수 $\boldsymbol{z}^{(1)}=\left[z_1^{(1)}\cdots z_{n_0}^{(1)}\right]^T$을 가지는 입력층이 있는 일반적인 완전연결신경망이다. l번째 은닉층에서 $(n_l\times 1)$형태의 선형변환한 벡터 $\boldsymbol{y}^{(l)}=\left[y_1^{(l)}\cdots y_{n_l}^{(l)}\right]^T$은 $(n_l\times n_{(l-1)})$형태의 가중치 파라미터행렬 $W^{(l)}=\left[\boldsymbol{w}_1^{(l)}\cdots \boldsymbol{w}_{n_l}^{(l)}\right]^T$과 $(n_{(l-1)}\times 1)$형태의 l번째 은닉층의 입력벡터 $\boldsymbol{z}^{(l)}=\left[z_1^{(l)}\cdots z_{n_{(l-1)}}^{(l)}\right]^T$ 그리고 $(n_l\times 1)$형태의 절편 파라미터 벡터 $\boldsymbol{w}_0^{(l)}=\left[w_{01}^{(l)}\cdots w_{0n_l}^{(l)}\right]^T$을 통해 선형변환된 것이다. 이렇게 변환된 벡터 $\boldsymbol{y}^{(l)}$은 $(n_l\times 1)$형태의 비선형활성함수 벡터 $\boldsymbol{\sigma}^{(l)}=\left[\sigma_1^{(l)}\cdots \sigma_{n_l}^{(l)}\right]^T$와 원소별로 곱해진다. Fig. 6의 모든 선형변환층의 파라미터를 W로 나타내고 출력층의 활성함수 $\boldsymbol{\sigma}^{(K+1)}$은 Softmax, 손실함수 L은 Cross-Entropy로 표기한다.

이상의 표기로 Fig. 6의 완전연결신경망의 출력값은 Equation (1)과 같이 선형변환과 비선형변환의 반복으로 표현할 수 있으며 Equation (1)의 출력값과 레이블 $\boldsymbol{t}$에 의해 정의된 손실함수는 Equation (2)과 같다.

$$\boldsymbol{z}^{(out)}=\boldsymbol{\sigma}^{(K+1)}(\,W^{(K+1)}\boldsymbol{\sigma}^{(K)}(\cdots\boldsymbol{\sigma}^{(1)}(\,W^{(1)}\boldsymbol{z}^{(1)}+\boldsymbol{w}_0^{(1)})\cdots)+\boldsymbol{w}_0^{(K+1)}) \quad (1)$$

$$L(W)=L(\boldsymbol{z}^{(out)},\boldsymbol{t}) \quad (2)$$

l번째 은닉층의 노드 i와 노드 j을 연결하는 선형변환층의 파라미터 $w_{ij}^{(l)}$는 경사하강법을 이용하여 Equation (3)과 같이 최적화한다.

$$w_{ij}^{(l)}=w_{ij}^{(l)}-\rho\nabla L(w_{ij}^{(l)}) \quad (3)$$

ρ는 학습률을 나타내고 $\nabla L(w_{ij}^{(l)})$은 $w_{ij}^{(l)}$에서 손실함수의 변화율을 나타낸다.

Equation (4)은 Fig. 6의 l번째 은닉층에서 선형변환 벡터 $\boldsymbol{y}^{(l)}$의 계산을, Equation (5)은 $\boldsymbol{y}^{(l)}$을 비선형활성함수를 통해 비선형변환 벡터 $\boldsymbol{z}^{(l+1)}$을 계산하는 식을 나타낸 것이다.

$$\boldsymbol{y}^{(l)}=W^{(l)}\boldsymbol{z}^{(l)}+\boldsymbol{w}_0^{(l)},\ l=1,\cdots K,K+1 \quad (4)$$

$$\boldsymbol{z}^{(l+1)}=\boldsymbol{\sigma}^{(l)}(\boldsymbol{y}^{(l)}),\ l=1,\cdots,K,K+1 \quad (5)$$

일반적인 완전연결신경망에 결합된 파라메트릭 활성함수를 적용할 수 있으며 Fig. 6과 같은 신경망에 l번째 은닉층에서 결합된 파라메트릭 활성함수를 적용할 때 그림은 Fig. 7과 같다.

Fig. 7의 l번째 은닉층에서 j번째 선형변환 $y_j^{(l)}$가 결합된 파라메트릭 활성함수를 통해 나온 $z_j^{(l+1)}$는 Equation (6)과 같다.

$$z_j^{(l+1)}=\sum_{u=1}^{k}\sigma^{(l)}_{(a_{j,u}^{(l)},b_{j,u}^{(l)})}(y_j^{(l)})=\sum_{u=1}^{k}z_{j,u}^{(l+1)} \quad (6)$$

결합된 파라메트릭 활성함수를 이용한 신경망의 모든 파라미터들은 경사하강법을 이용하여 학습할 수 있다. 예를 들어

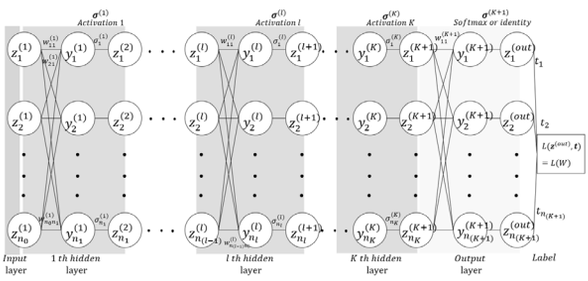


Fig. 6. Structure of Fully Connected Neural Network (The l^{th} Hidden Layer has n_l Nodes and Consists of a Total of K Hidden Layers)

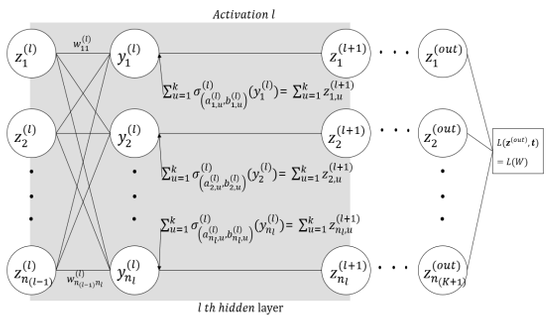


Fig. 7. Combined Parametric Activation Function Process in the l th Hidden Layer

결합된 파라메트릭 활성함수를 C PPNS로 사용할 때 l번째 은닉층에서 $w_{ij}^{(l)}$에 의한 손실함수 $L(W)$의 변화율, l번째 은닉층의 j번째 비선형변환 노드 $z_j^{(l+1)}$의 파라미터 $a_{j,u}^{(l)}$와 $b_{j,u}^{(l)}$에 의한 손실함수(L)의 변화율은 Equation (7), (8), (9)과 같다.

$$\begin{aligned}\nabla L(w_{ij}^{(l)})&=\frac{\partial L}{\partial z_j^{(l+1)}}\frac{\partial z_j^{(l+1)}}{\partial y_j^{(l)}}\frac{\partial y_j^{(l)}}{\partial w_{ij}^{(l)}}\\&=\frac{\partial L}{\partial z_j^{(l+1)}}\times\sum_{u=1}^{k}\left(z_{j,u}^{(l+1)}\left(1-\frac{z_{j,u}^{(l+1)}}{a_{j,u}^{(l)}}\right)\right)\times\frac{\partial y_j^{(l)}}{\partial w_{ij}^{(l)}}\end{aligned} \quad (7)$$

$$\nabla L(a_{j,u}^{(l)})=\frac{\partial L}{\partial z_j^{(l+1)}}\frac{\partial z_j^{(l+1)}}{\partial a_{j,u}^{(l)}}=\frac{\partial L}{\partial z_j^{(l+1)}}\times\left(\frac{1}{1+e^{-(y_j^{(l)}-b_{j,u}^{(l)})}}-\frac{1}{2}\right) \quad (8)$$

$$\nabla L(b_{j,u}^{(l)})=\frac{\partial L}{\partial z_j^{(l+1)}}\frac{\partial z_j^{(l+1)}}{\partial b_{j,u}^{(l)}}=\frac{\partial L}{\partial z_j^{(l+1)}}\times z_{j,u}^{(l+1)}\left(\frac{z_{j,u}^{(l+1)}}{a_{j,u}^{(l)}}-1\right) \quad (9)$$

Equation (7), (8), (9)을 사용하여 Equation (10), (11), (12)과 같이 경사하강법을 사용해 완전연결신경망의 모든 l,i,j,u에 대해 손실함수를 최소화하는 파라미터 $w_{ij}^{(l)}$, $a_{j,u}^{(l)}$, $b_{j,u}^{(l)}$들을 계산할 수 있다.

$$w_{ij}^{(l)}=w_{ij}^{(l)}-\rho_1\nabla L(w_{ij}^{(l)}) \quad (10)$$

$$a_{j,u}^{(l)}=a_{j,u}^{(l)}-\rho_2\nabla L(a_{j,u}^{(l)}) \quad (11)$$

$$b_{j,u}^{(l)}=b_{j,u}^{(l)}-\rho_3\nabla L(b_{j,u}^{(l)}) \quad (12)$$

ρ_1, ρ_2, ρ_3는 학습률을 나타낸다. 이상의 과정에서 결합된 파라메트릭 활성함수를 적용한 완전연결신경망은 Sigmoid와 ReLU같은 활성함수를 적용한 완전연결신경망보다 입력데이터의 특성에 따라 다양한 비선형간격을 만들어내어 필요

한 은닉층의 개수와 노드를 줄일 가능성이 있으며 손실함수를 최소화하는 방향으로 최적화할 수 있다.

4. 결합된 파라메트릭 활성함수의 성능 실험

결합된 파라메트릭 활성함수가 완전연결신경망의 성능을 향상시킬 수 있는지 확인하기 위해 8x8 MNIST 데이터에 대해 크게 두 가지 실험을 진행하였고 추가적으로 일반적인 성능을 확인하기 위해 28X28 MNIST 데이터와 Fashion MNIST 데이터에 대해 실험하였다.

8x8 MNIST 데이터는 보편적으로 사용되는 28x28 MNIST 숫자 이미지보다 사이즈가 작고 데이터 개수 또한 적다. 이 때문에 보다 구분하기 어려운 8x8 MNIST 데이터를 실험에 사용하여 결합된 파라메트릭 활성함수가 학습을 할 수 있는지 확인하기 위함이다.

첫 번째 실험은 결합된 파라메트릭 활성함수와 기존 비선형활성함수 그리고 파라메트릭 활성함수의 성능에 대해 비교하였고 두 번째 실험은 결합된 파라메트릭 활성함수가 기존에 사용되는 비선형활성함수보다 은닉층 수를 줄일 수 있는지 실험하였다.

4.1 결합된 파라메트릭 활성함수 성능 확인을 위한 은닉층 수가 1개인 8x8 MNIST 분류 문제

은닉층 수가 1개이고 노드 개수가 3개인 신경망(Fig. 8 참조)을 이용하여 Table 5와 같이 대표적인 활성함수 Sigmoid와 ReLU에 대해 파라메트릭 활성함수 파라미터를 적용하여 총 8개 함수에 대해 실험하였다. Table 5에서 결합된 파라메트릭 활성함수의 파라미터 개수를 나타내는 $k=2$을 주었다. $k=2$을 준 이유는 두 개의 파라메트릭 활성함수들의 단순한 결합으로도 성능을 입증해보이기 위해서이다. 8x8 MNIST 분류 문제에 노드 개수를 3개를 준 이유는 적은 노드 개수만으로 결합된 파라메트릭 활성함수가 기존 비선형활성함수 혹은 파라메트릭 활성함수보다 더욱 다양한 비선형성을 만들어 손실함수를 최소화할 수 있는지를 확인하기 위함이다.

Table 6은 Table 5의 제시된 파라메트릭 활성함수 파라미터의 초기화를 나타낸 것으로 크기를 결정하는 파라미터 a

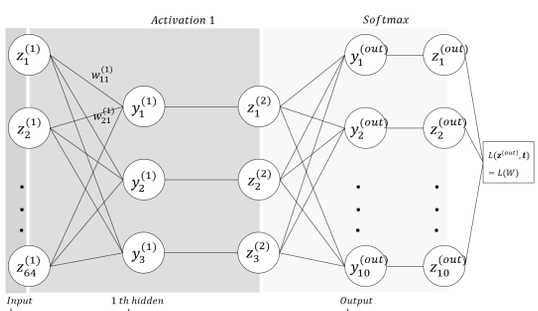


Fig. 8. Neural Network Structure for an 8x8 MNIST with 1 Hidden Layer and 3 Nodes

Table 5. Activation Functions used in the Experiment

	Sigmoid	ReLU
Activation function (A)	$\frac{1}{1+e^{-y}}$	$\begin{cases} y, y>0 \\ 0, y\le 0 \end{cases}$
Parametric activation function (PA)	$\frac{a}{1+e^{-(y-b)}}$	$\begin{cases} a(y-b), y>b \\ 0 \quad\quad , y\le b \end{cases}$
Parametric PN activation function (PPNA)	$\frac{a}{1+e^{-(y-b)}}-\frac{a}{2}$	$\begin{cases} a(y-b)-\frac{a}{2}, y>b \\ -\frac{a}{2} \quad\quad , y\le b \end{cases}$
Combined parametric activation function ($k=2$) (C PPNA)	$\sum_{i=1}^{2}\left(\frac{a_i}{1+e^{-(y-b_i)}}-\frac{a_i}{2}\right)$	$\sum_{i=1}^{2}\begin{cases} a_i(y-b_i)-\frac{a_i}{2}, y>b_i \\ -\frac{a_i}{2} \quad\quad , y\le b_i \end{cases}$

Table 6. Initialization of Activation Function Parameters

	Initialization of parameters	
Activation function	Sigmoid	ReLU
Parametric activation fucntion	$a=4, b=0$	$a=1, b=0$
Parametric PN activation function	$a=4, b=0$	$a=1, b=0$
Combined parametric activation function($k=2$)	$a_1=4, b_1=2,$ $a_2=4, b_2=-2$	$a_1=1, b_1=0.1,$ $a_2=1, b_2=-0.1$

는 학습 초기에 발생할 수 있는 기울기 소실을 방지하기 위함을 목적으로 주었으며 위치를 결정하는 파라미터 b는 적절한 비선형간격을 만들어내기 위해 주었다.

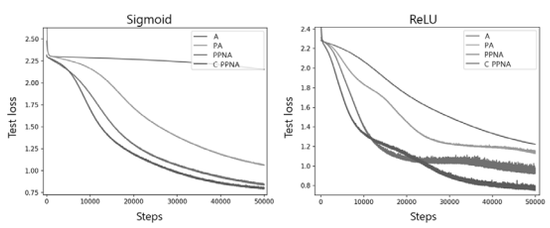
실험조건은 Table 7과 같으며 선형변환층 파라미터 W을 모두 동일한 값으로 초기화하였고 Step마다 Test data에 대한 Test loss를 그린 실험 결과는 Fig. 9와 같다. Fig 9에서 활성함수가 파라메트릭 활성함수 파라미터를 적용함으로써 나타나는 성능을 보다 쉽게 비교하기 위해 Sigmoid 계열(Fig. 9(a))과 ReLU 계열(Fig. 9(b))을 구분하여 나타내었으며 Table 5의 표기를 사용하였다. Fig. 9(a)와 Fig. 9(b)에서 50,000번의 step결과 Sigmoid와 ReLU 모두 Test loss가 A>PA>PPNA>C PPNA 순서로 결합된 파라메트릭 활성함수(C PPNA)가 가장 낮은 Test loss를 가지는 것을 알 수 있다.

Fig. 9의 실험결과에 대해 결합된 파라메트릭 활성함수(C PPNA)가 기존 비선형활성함수(A)보다 손실함수를 최소화하는 파라메트릭 활성함수 파라미터에 의해 다양한 비선형변환 값을 가지는 것을 확인하기 위해 학습이 끝난 후 Test data 540개에 대해 Fig. 8에서 선형변환 $y_1^{(1)}$에 대응하는 비선형변환 $z_1^{(2)}$의 분포를 나타낸 그림은 Fig. 10과 같고 나머지 두 개의 노드 $z_2^{(2)}, z_3^{(2)}$에 대해서도 비슷한 결과를 가졌다.

Fig. 10(a,b)은 Sigmoid와 C PPNS를, Fig. 10(c,d)은 ReLU와 C PPNR에 대해 비교한 그림이다. Sigmoid의 경우

Table 7. Experiment Conditions

Data preprocessing	Min-Max normalization
Train data	1,257
Test data	540
Batch size	128
Algorithm	Gradient descent
Learning rate	0.01
Steps	50,000
Loss function	Cross-Entropy
Initialization of W parameters	Same random normal initialization
Number of hidden layers	1
Number of nodes in each hidden layer	3
Regularization	X



(a) (b)

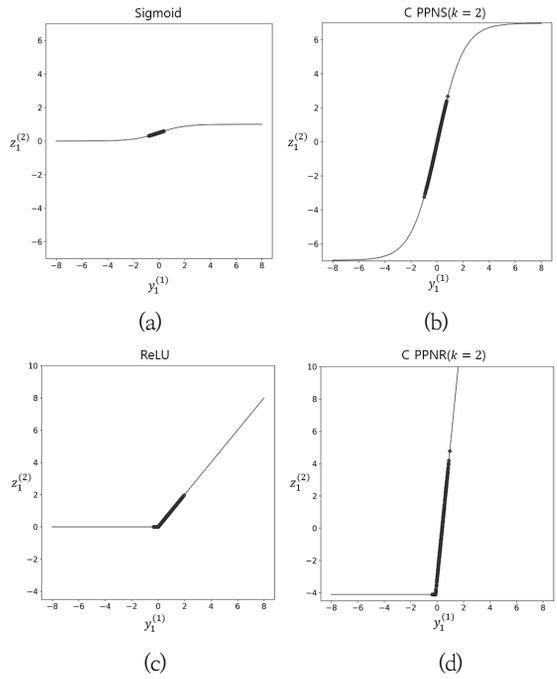
Fig. 9. Test Loss for Steps. A: Activation Function, PA : Parametric Activation Function, PPNA : Parametric PN Activation Function, C PPNA : Combined Parametric PN Activation Function. (a) : Sigmoid Series, (b) : ReLU Series.

$z_1^{(2)}$값이 한정된 범위 0에서 1을 갖는 반면 C PPNS는 약 -7에서 7정도로 Sigmoid보다 넓은 범위의 값을 가지며 그에 따른 Test data의 분포도 퍼져있어 다양한 비선형변환 값을 취할 수 있음을 알 수 있다. 이로 인해 Fig. 9(a)에서 Sigmoid보다 C PPNS가 월등히 낮은 Test loss 값을 가지는 것을 확인할 수 있다. ReLU의 경우도 마찬가지로 C PPNR을 사용한 Test data에 대한 $z_1^{(2)}$값은 -4에서 약 5.5정도로 분포되어 있으며 ReLU보다 다양한 비선형변환 값을 취하여 Fig .9(b)에서 C PPNR이 월등히 낮은 Test loss를 취하는 것을 확인할 수 있다.

C PPNS의 파라미터 초기화는 Table 6과 같이 초기화되었다. 학습 후 파라미터가 초기화된 값에서 얼마나 변화하였는지를 보기 위해 Equation (13)과 같이 Fig. 10(a)의 $z_1^{(2)}$의 계산에서 사용된 파라미터 값을 나타낸 결과는 Table 8과 같다. Table 8에서 $a_{11}^{(1)}$값은 4에서 6.958, $b_{11}^{(1)}$값은 2에서 0.02, 나머지 파라미터도 학습되었음을 볼 수 있다. 이는 결합된 활성함수 파라미터들이 초깃값에서 입력데이터 특성에 따라 손실함수를 최소화하는 값으로 이동하였음을 알 수 있다.

$$z_1^{(2)} = \sum_{i=1}^{2} \sigma^{(1)}_{(a_{1i}^{(1)}, b_{1i}^{(1)})}(y_1^{(1)}) \tag{13}$$

이상의 은닉층 수 1개, 3개의 노드 수를 갖는 8x8 MNIST 실험결과에서 Sigmoid와 ReLU에 대해 결합된 파라메트릭



(a) (b)

(c) (d)

Fig. 10. $z_1^{(2)}$ Distribution Corresponding to $y_1^{(1)}$ for 540 Test Data (a) : Sigmoid, (b) : C PPNS, (c) : ReLU, (d) : C PPNR

Table 8. C PPNS Parameter Values

	Initial value	Trained value
$a_{11}^{(1)}$	4	6.958
$b_{11}^{(1)}$	2	0.020
$a_{12}^{(1)}$	4	6.978
$b_{12}^{(1)}$	-2	0.017

활성함수가 기존의 비선형활성함수, 파라메트릭 활성함수보다 더욱 다양한 비선형변환 값을 가질 수 있고 활성함수의 파라미터들이 손실함수를 최소화하는 방향으로 최적화하여 Test loss가 가장 낮음을 확인하였다.

4.2 결합된 파라메트릭 활성함수를 이용한 은닉층 수 감소 실험을 위한 8x8 MNIST 분류 문제

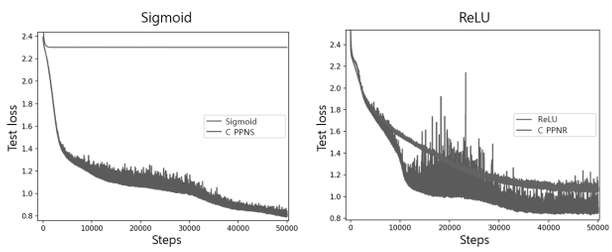
결합된 파라메트릭 활성함수가 기존의 비선형활성함수를 사용했을 때보다 필요한 은닉층 수를 감소시킬 수 있는지를 확인하기 위해 Sigmoid와 결합된 파라메트릭 활성함수 C PPNS 그리고 ReLU와 C PPNR에 대해 비교하였다(Table 9). Table 9는 8x8 MNIST 분류 문제에 대한 은닉층 수와 노드 개수를 활성함수에 따라 나타낸 것으로 Sigmoid와 ReLU는 4개의 은닉층을, C PPNS와 C PPNR은 1개 적은 3개의 은닉층을 사용하였으며 각 은닉층의 노드 수는 동일하다.

C PPNS와 C PPNR에 대한 파라미터 초기화는 Table 6에서 사용된 결합된 파라메트릭 활성함수 파라미터 초기화와 동일하며 실험조건은 Table 7에서 은닉층의 개수를 제외하고 동일하다.

실험결과를 나타내는 Fig. 11은 Step마다 Test data에 대

Table 9. The Number of Hidden Layers and Nodes Corresponding to the Activation Functions

	Activation function (A)		Combined parametric activation function ($k=2$) (C PPNA)	
	Sigmoid	ReLU	C PPNS	C PPNR
Number of hidden layers	4	4	3	3
Number of nodes in each hidden layer	3	3	3	3



(a) (b)

Fig. 11. Test loss for Activation Functions
(a) : Sigmoid, C PPNS, (b) : ReLU, C PPNR

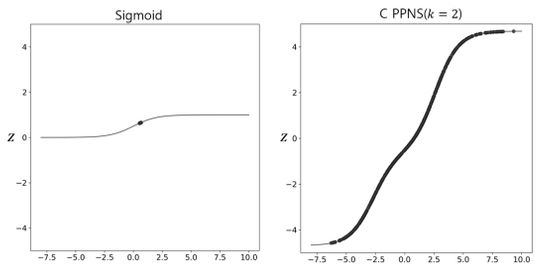
한 Test loss를 그린 것으로 Fig. 11(a)은 Sigmoid와 C PPNS를, Fig. 11(b)은 ReLU와 C PPNR에 대해 나타낸 것이다. 두 그림 모두 결합된 파라메트릭 활성함수가 기존 비선형활성함수보다 은닉층의 개수가 1개 부족함에도 불구하고 성능이 우월할 것을 확인할 수 있다. 특히 Sigmoid의 경우 기울기 소실 문제가 발생하여 손실함수 값이 감소하지 않는 반면 C PPNS은 빠르게 감소하는 것을 볼 수 있으며, ReLU의 경우도 마찬가지로 C PPNR이 손실함수의 감소 속도가 더욱 빠른 것을 볼 수 있다.

결합된 파라메트릭 활성함수가 다양한 비선형 간격을 만들어내는지 보기위해 Sigmoid와 C PPNS에 대해 학습 후 Test data 540개에 대해 3번째 은닉층의 첫 번째 노드의 선형변환 y에 대한 비선형변환 z의 분포를 나타내었으며(Fig. 12) 나머지 노드도 비슷한 결과를 가짐을 확인하였다.

Sigmoid를 나타내는 Fig. 12(a)를 보면 540개의 Test data들이 한 점에 쏠려있는 반면 C PPNS를 나타내는 Fig. 12(b)는 좀 더 유연한 비선형간격에 대해 다양한 비선형변환 값들이 분포되어 있는 것을 볼 수 있다.

Fig. 12(b)에서 결합된 파라메트릭 활성함수를 통해 비선형변환 z의 파라메트릭 활성함수 파라미터가 학습된 값은 Table 10에 나타내었다. Table 10에서 a,b값은 입력데이터의 특성을 반영하여 경사하강법을 통해 학습되었으며 그 결과 Fig. 12(b)와 같이 다양한 비선형간격을 만들어낸 것을 확인할 수 있다.

이상의 8x8 MNIST 분류문제에서 결합된 파라메트릭 활성함수 C PPNS, C PPNR이 Sigmoid, ReLU보다 은닉층 수가 줄어들었음에도 불구하고 우월한 성능을 가짐을 확인하였다.



(a) (b)

Fig. 12. z Distribution Corresponding to y for 540 Test Data (1st Node of 3rd Hidden Layer) (a) : Sigmoid, (b) : C PPNS

Table 10. C PPNS Parameter Values of the 1st Node of the 3rd Hidden Layer

	Initial value	Trained value
a_1	4	5.30
b_1	2	2.62
a_2	4	4.07
b_2	-2	-2.72

4.3 결합된 파라메트릭 활성함수 성능 확인을 위한 28x28 MNIST 분류 문제와 Fashion MNIST 분류 문제

결합된 파라메트릭 활성함수의 좀 더 일반적인 성능을 확인하기 위해 28x28 MNIST 데이터와 Fashion MNIST 데이터에 대해 실험을 진행하였다.

실험에 사용된 활성함수는 $k=2$을 사용한 C PPNS와 C PPNR 그리고 신경망에서 보편적으로 사용되고 있는 함수 ReLU와 Tanh 총 4가지 함수에 대해 비교하였다. 결합된 파라메트릭 활성함수의 초기화를 다양하게 주어 실험하였으며 기울기 소실을 방지하기 위해 C PPNS의 경우 a값을 4, 4.5, 5로 C PPNR의 경우 a값을 1, 1.5, 2로 주었다. b값의 초기화는 Table 6에 결합된 파라메트릭 활성함수 초기화를 사용하였다.

다양한 신경망 구조에서 성능을 확인하기 위해 은닉층 수가 3, 5, 8개인 3가지 경우에 대해 각 경우마다 노드 수를 각각 50, 100, 200개를 주어 총 9가지 경우에 대해 진행하였다. 28x28 MNIST와 Fashion MNIST 모두 같은 실험조건을 사용하였으며 배치 사이즈는 64개, 30에폭, 모멘텀 최적화 알고리즘을 사용하였다.

각 활성함수마다 랜덤 초기화로 5번씩 시행한 시험 데이터의 정확도의 평균과 표준편차를 나타낸 결과는 Table 11과 12와 같다.

28x28 MNIST 실험결과를 나타내는 Table 11에서 은닉층 수가 8개, 각 노드가 50개인 경우를 제외하고 모든 경우에서 C PPNR의 시험 데이터의 정확도가 비교된 활성함수 중 가장 높게나왔다. Fashion MNIST 실험결과에서도 은닉층 수가 8개인 경우를 제외하면 C PPNR의 성능이 가장 좋은 것을 확인할 수 있다.

하지만 C PPNR의 경우 a값이 1.5이상으로 초기화되는 경우, 노드 개수와 은닉층 수가 늘어남에 따라 불안정한 수렴성을 보이며 낮은 평균과 표준편차를 가지는 것을 확인하였다.

Table 11. Test Data Accuracy for the MNIST 28×28 Experiment

Number of nodes in each hidden layer \ Number of hidden layers	3	5	8
50	$ReLU$: 95.78% ± 0.00006 $Tanh$: 94.66% ± 0.0 $CPPNS$ {(a = 4): 92.63% ± 0.00004; (a = 4.5): 92.78% ± 0.0015; (a = 5): 92.83% ± 0.0014} $CPPNR$ {**(a = 1) : 96.63% ± 0.002**; (a = 1.5) : 96.62% ± 0.0006; (a = 2) : 96.42% ± 0.0028}	$ReLU$: 96.1% ± 0.0003 $Tanh$: 95.82% ± 0.0 $CPPNS$ {(a = 4): 93.76% ± 0.0; (a = 4.5): 94.22% ± 0.004; (a = 5): 94.36% ± 0.004} $CPPNR$ {**(a = 1) : 96.18% ± 0.0013**; (a = 1.5) : 96.04% ± 0.001; (a = 2) : 67.46% ± 0.4}	$ReLU$: 96.07% ± 0.00066 **$Tanh$: 96.11% ± 0.00004** $CPPNS$ {(a = 4): 94.02% ± 0.00007; (a = 4.5): 94.32% ± 0.003; (a = 5): 94.68% ± 0.005} $CPPNR$ {(a = 1) : 96.06% ± 0.0009; (a = 1.5) : 52.93% ± 0.4313; (a = 2) : 38.55% ± 0.406}
100	$ReLU$: 96.24% ± 0.0001 $Tanh$: 94.88% ± 0.00004 $CPPNS$ {(a = 4): 92.5% ± 0.0; (a = 4.5): 92.51% ± 0.0001; (a = 5): 92.66% ± 0.002} $CPPNR$ {(a = 1) : 96.97% ± 0.0001; **(a = 1.5) : 97.08% ± 0.001**; (a = 2) : 96.97% ± 0.0018}	$ReLU$: 96.58% ± 0.0003 $Tanh$: 95.7% ± 0.0 $CPPNS$ {(a = 4): 93.5% ± 0.00004; (a = 4.5): 93.47% ± 0.00025; (a = 5): 93.73% ± 0.0037} $CPPNR$ {**(a = 1) : 96.95% ± 0.0005**; (a = 1.5) : 96.64% ± 0.0033; (a = 2) : 67.7% ± 0.409}	$ReLU$: 96.52% ± 0.0003 $Tanh$: 96.33% ± 0.0 $CPPNS$ {(a = 4): 94.5% ± 0.000007; (a = 4.5): 94.71% ± 0.002; (a = 5): 95% ± 0.0044} $CPPNR$ {**(a = 1) : 96.62% ± 0.0015**; (a = 1.5) : 53.21% ± 0.43; (a = 2) : 38.74% ± 0.409}
200	$ReLU$: 96.7% ± 0.00005 $Tanh$: 94.73% ± 0.0 $CPPNS$ {(a = 4): 92.46% ± 0.0; (a = 4.5): 92.44% ± 0.0; (a = 5): 92.4% ± 0.0004} $CPPNR$ {(a = 1) : 97.19% ± 0.0003; **(a = 1.5) : 97.20% ± 0.0005**; (a = 2) : 97.08% ± 0.002}	$ReLU$: 97.08% ± 0.00037 $Tanh$: 95.91% ± 0.00004 $CPPNS$ {(a = 4): 92.41% ± 0.00004; (a = 4.5): 92.6% ± 0.002; (a = 5): 92.87% ± 0.004} $CPPNR$ {**(a = 1) : 97.37% ± 0.0005**; (a = 1.5) : 53.72% ± 0.436; (a = 2) : 39.08% ± 0.412}	$ReLU$: 96.84% ± 0.0004 $Tanh$: 96.55% ± 0.00004 $CPPNS$ {(a = 4): 93.4% ± 0.00004; (a = 4.5): 93.9% ± 0.005; (a = 5): 94.35% ± 0.007} $CPPNR$ {**(a = 1) : 96.9% ± 0.0015**; (a = 1.5) : 53.34% ± 0.435; (a = 2) : 38.83% ± 0.41}

Table 12. Test Data Accuracy for the Fashion MNIST Experiment

Number of nodes in each hidden layer \ Number of hidden layers	3	5	8
50	$ReLU$: 86.5% ± 0.0005 $Tanh$: 85.87% ± 0.0 $CPPNS$ {(a = 4): 85.37% ± 0.00007; (a = 4.5): 85.4% ± 0.003; (a = 5): 85.41% ± 0.0004} $CPPNR$ {(a = 1) : 86.67% ± 0.0004; **(a = 1.5) : 86.78% ± 0.003**; (a = 2) : 86.38% ± 0.006}	$ReLU$: 85.63% ± 0.0011 $Tanh$: 86.4% ± 0.00004 $CPPNS$ {(a = 4): 85.8% ± 0.0; (a = 4.5): 85.76% ± 0.0002; (a = 5): 85.81% ± 0.0006} $CPPNR$ {(a = 1) : 86.21% ± 0.004; **(a = 1.5) : 86.21% ± 0.003**; (a = 2) : 60.8% ± 0.36}	$ReLU$: 85.56% ± 0.0025 **$Tanh$: 86.93% ± 0.00007** $CPPNS$ {(a = 4): 85.92% ± 0.00014; (a = 4.5): 85.77% ± 0.0007; (a = 5): 85.69% ± 0.0013} $CPPNR$ {(a = 1) : 86.18% ± 0.0063; (a = 1.5) : 48.09% ± 0.38; (a = 2) : 35.4% ± 0.359}
100	$ReLU$: 86.87% ± 0.0003 $Tanh$: 85.9% ± 0.00004 $CPPNS$ {(a = 4): 85.27% ± 0.000004; (a = 4.5): 85.13% ± 0.0014; (a = 5): 85.14% ± 0.0006} $CPPNR$ {**(a = 1) : 87.26% ± 0.0003**; (a = 1.5) : 87.15% ± 0.0012; (a = 2) : 86.94% ± 0.003}	$ReLU$: 86.45% ± 0.0005 $Tanh$: 86.85% ± 0.00008 $CPPNS$ {(a = 4): 85.44% ± 0.00004; (a = 4.5): 85.63% ± 0.0018; (a = 5): 85.58% ± 0.0017} $CPPNR$ {**(a = 1) : 87% ± 0.0015**; (a = 1.5) : 86.25% ± 0.007; (a = 2) : 60.83% ± 0.36}	$ReLU$: 85.94% ± 0.00048 **$Tanh$: 86.59% ± 0.00004** $CPPNS$ {(a = 4): 85.8% ± 0.00006; (a = 4.5): 85.78% ± 0.0001; (a = 5): 85.82% ± 0.0007} $CPPNR$ {(a = 1) : 85.9% ± 0.0056; (a = 1.5) : 47.95% ± 0.379; (a = 2) : 35.3% ± 0.3578}
200	$ReLU$: 87.02% ± 0.0008 $Tanh$: 85.68% ± 0.00004 $CPPNS$ {(a = 4): 85.2% ± 0.00006; (a = 4.5): 85.17% ± 0.0004; (a = 5): 85.07% ± 0.0004} $CPPNR$ {**(a = 1) : 87.3% ± 0.0013**; (a = 1.5) : 87.06% ± 0.002; (a = 2) : 86.69% ± 0.005}	$ReLU$: 86.24% ± 0.0008 $Tanh$: 86.62% ± 0.0 $CPPNS$ {(a = 4): 84.89% ± 0.00004; (a = 4.5): 85.2% ± 0.003; (a = 5): 85.22% ± 0.0025} $CPPNR$ {**(a = 1) : 86.78% ± 0.003**; (a = 1.5) : 48.4% ± 0.384; (a = 2) : 35.6% ± 0.36}	$ReLU$: 85.61% ± 0.0032 **$Tanh$: 86.81% ± 0.0** $CPPNS$ {(a = 4): 85.28% ± 0.00004; (a = 4.5): 85.26% ± 0.00025; (a = 5): 85.43% ± 0.0024} $CPPNR$ {(a = 1) : 86.58% ± 0.0037; (a = 1.5) : 48.3% ± 0.38; (a = 2) : 35.52% ± 0.361}

C PPNR과 ReLU의 계산속도 비교를 위해 은닉층 수가 5개, 각 노드 수가 200개인 경우의 MNIST 실험에서 ReLU와 C PPNR 모두 시험 데이터 정확도가 97.08% 까지 도달하는데 걸린 컴퓨팅 시간을 비교해본 결과, C PPNR이 ReLU보다 계산 시간이 약 10% 정도 감소한 것을 확인할 수 있었다.

5. 결 론

완전연결신경망에서 비선형활성함수는 선형변환 값을 비선형 변환하여 출력하는 함수로써 비선형 문제를 해결하는데 중요한 역할을 한다. 결합된 파라메트릭 활성함수는 위치와 크기를 결정하는 파라미터를 적용한 파라메트릭 활성함수들을 간단히 더하여 만들어진 함수로 기존에 사용되는 비선형 활성함수 또는 파라메트릭 활성함수보다 더욱 다양한 비선형 간격을 입력데이터의 특성에 맞게 손실함수를 최소화하는 방향으로 학습시킬 수 있다.

결합 활성함수가 완전연결신경망의 성능을 향상시킬 수 있는지 확인하기 위해 8x8 MNIST 분류문제에 대해 실험하였다. 기존의 비선형활성함수, 파라메트릭 활성함수를 결합된 파라메트릭 활성함수와 비교하였으며 그 결과 은닉층 수가 1개, 노드가 3개인 구조에 대해서 가장 낮은 Test loss를 가지는 것을 확인하였고 결합된 파라메트릭 활성함수의 파라미터 값들이 손실함수를 최소화하는 방향으로 학습되어 다양한 비선형변환 값을 가질 수 있음을 확인하였다.

또한 8x8 MNIST 분류문제를 은닉층 수가 4개인 Sigmoid, ReLU와 은닉층 수가 3개인 결합된 파라메트릭 활성함수 C PPNS와 C PPNR에 대해 비교하여 결합된 파라메트릭 활성함수가 기존에 사용되는 비선형활성함수보다 은닉층 수를 줄

일 수 있는지 실험하였다. 실험결과 Sigmoid와 ReLU 모두에 대해 결합된 파라메트릭 활성함수가 은닉층 수가 줄었음에도 불구하고 더 낮은 손실함수 값을 가짐을 확인하였다.

결합된 파라메트릭 활성함수의 일반적인 성능 검증을 위해 28x28 MNIST 분류문제와 Fashion MNIST 분류문제에 대해 은닉층 수와 노드 수를 다양하게 주어 실험한 결과, CPPNR이 많은 경우, ReLU, Tanh보다 시험 데이터 정확도에 대해서 높은 성능을 가지는 것을 확인하였다.

이상의 MNIST와 Fashion MNIST 실험에 대해 결합된 파라메트릭 활성함수의 파라미터는 다양한 비선형 간격을 만들어내어 입력데이터 특성에 맞게 손실함수를 최소화하는 방향으로 학습하여 기존의 비선형활성함수와 파라메트릭 활성함수보다 우수한 성능을 가짐을 알 수 있었다.

본 논문은 크기 및 위치를 변환시키는 파라메트릭 활성함수들을 간단히 더함으로써 만들어진 결합된 파라메트릭 활성함수가 완전연결신경망의 성능을 향상시킬 수 있음을 Sigmoid와 ReLU을 통해 실험으로 확인하였고 다른 임의의 비선형활성함수에 대해서도 간단히 적용할 수 있음에 의의가 있다.

References

[1] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, Vol.2, Iss.5, pp.359-366, 1989.

[2] Y. Bengio, I. Goodfellow, and A. Courville, "Deep learning," MIT Press, 2017.

[3] N. Y. Kong and S. W. Ko, "Performance improvement method of deep neural network using parametric activation functions," *Journal of the Korea Contents Association*, Vol.21, No.3, pp.616-625, 2021.

[4] N. Y. Kong, Y. M. Ko, and S. W. Ko, "Performance improvement method of convolutional neural network using agile activation function," *KIPS Transactions on Software and Data Engineering*, Vol.9, No.7, pp.213-220, 2020.

[5] Y. M. Ko and S. W. Ko, "Alleviation of vanishing gradient problem using parametric activation functions," *KIPS Transactions on Software and Data Engineering*, Vol.10, No.10, pp.407-420, 2021.

[6] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines," *ICML*, pp.807-814, 2010.

[7] M. Roodschild, J. Gotay Sardinas, and A. Will, "A new approach for the vanishing gradient problem on sigmoid activation," *Springer Nature*, Vol.20, Iss.4, pp.351-360, 2020.

[8] Y. Qin, X. Wang, and J. Zou, "The optimized deep belief networkswith improved logistic Sigmoid units and their application in faultdiagnosis for planetary gearboxes of wind turbines," *IEEE Transactions on Industrial Electronics*, Vol.66, No.5, pp.3814-3824, 2018.

[9] X. Wang, Y. Qin, Y. Wang, S. Xiang, and H. Chen, "ReLTanh: An activation function with vanishing gradient resistance for SAE-based DNNs and its application to rotating machinery fault diagnosis," *Neurocomputing*, Vol.363, pp.88-98, 2019.

[10] S. Kong and M. Takatsuka, "Hexpo: A vanishing-proof activation function," *International Joint Conference on Neural Networks*, pp.2562-2567, 2017.

[11] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolution network," *arXiv:1505.00853*, 2015.

[12] R. Pascanu, T. Mikolov, and Y. Bengio, "Understanding the exploding gradient problem," *arXiv:1211.5063*, 2012.

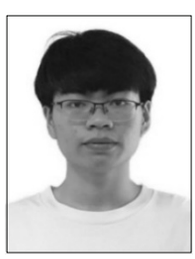
[13] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," *arXiv:1211.5063*, 2013.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *International Conference on Computer Vision*, *arXiv:1502.01852*, 2015.

[15] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units(ELUs)," *arXiv:1511.07289*, 2016.

고 영 민

https://orcid.org/0000-0003-2779-3170
e-mail : gjtrj55@naver.com
2020년 전주대학교 경영학과(학사)
2020년 ~ 현 재 전주대학교 인공지능학과 석사과정
관심분야 : Data Science & Artificial Intelligence



이 붕 항

https://orcid.org/0000-0001-8496-8067
e-mail : 422217887@qq.com
2020년 전주대학교 스마트미디어학과(학사)
2020년 ~ 현 재 전주대학교 인공지능학과 석사과정
관심분야 : Digital Image Processing & Artificial Intelligence

고 선 우

https://orcid.org/0000-0002-6328-5440
e-mail : godfriend0@gmail.com
1985년 고려대학교 산업공학과(학사)
1988년 한국과학기술원 산업공학과(석사)
1992년 한국과학기술원 산업공학과(박사)
2005년 ~ 현 재 전주대학교 인공지능학과 교수
관심분야 : Data Science & Artificial Intelligence