KIPS Tr. Software and Data Eng.
Vol.6, No.12 pp.573~580 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2017.6.12.573

A Deep Neural Network Model Based on a Mutation Operator

Jeon Seung Ho† · Moon Jong Sub††

ABSTRACT

Deep Neural Network (DNN) is a large layered neural network which is consisted of a number of layers of non-linear units. Deep Learning which represented as DNN has been applied very successfully in various applications. However, many issues in DNN have been identified through past researches. Among these issues, generalization is the most well-known problem. A Recent study, Dropout, successfully addressed this problem. Also, Dropout plays a role as noise, and so it helps to learn robust feature during learning in DNN such as Denoising AutoEncoder. However, because of a large computations required in Dropout, training takes a lot of time. Since Dropout keeps changing an inter-layer representation during the training session, the learning rates should be small, which makes training time longer. In this paper, using mutation operation, we reduce computation and improve generalization performance compared with Dropout. Also, we experimented proposed method to compare with Dropout method and showed that our method is superior to the Dropout one.

Keywords : Deep Learning, Generalization, Denoising, Mutation Operation

돌연변이 연산 기반 효율적 심층 신경망 모델

전 승 호† · 문 종 섭††

요 약

심층 신경망은 많은 노드의 층을 쌓아 만든 거대한 신경망이다. 심층 신경망으로 대표되는 딥 러닝은 오늘날 많은 응용 분야에서 괄목할만한 성과를 거두고 있다. 하지만 다년간의 연구를 통해 심층 신경망에 대한 다양한 문제점이 식별되고 있다. 이 중 일반화는 가장 널리 알려진 문제점들 중 하나이며, 최근 연구 결과인 드롭아웃은 이러한 문제를 어느 정도 성공적으로 해결하였다. 드롭아웃은 노이즈와 같은 역할을 하여 신경망이 노이즈에 강건한 데이터 표현형을 학습할 수 있도록 하는데, 오토인코더와 관련된 연구에서 이러한 효과가 입증되었다. 하지만 드롭아웃은 빈번한 난수 연산과 확률연산으로 인해 신경망의 학습 시간이 길어지고, 신경망 각 계층의 데이터 분포가 크게 변화하여 작은 학습율을 사용해야하는 단점이 있다. 본 논문에서는 돌연변이 연산을 사용하여 비교적 적은 횟수의 연산으로 드롭아웃과 동등 이상의 성능을 나타내는 모델을 제시하고, 실험을 통하여 논문에서 제시한 방법이 드롭아웃 방식과 동등한 성능을 보임과 동시에 학습 시간 문제를 개선함을 보인다.

키워드 : 딥 러닝, 일반화, 잡음제거, 돌연변이 연산

1. 서 론

최근 딥 러닝[1]이 지금까지 해결하지 못한 다양한 응용 및 연구 분야에서 성공을 거두며 각광을 받고 있다. 딥 러닝에서 사용하는 심층 신경망이란, 비선형 출력을 내는 수많은 노드를 여러 계층으로 쌓음으로써, 복잡한 분포를 갖는 데이터를 효과적으로 모델링한다. 일반적인 기계학습 알고리즘과 마찬가지로 심층 신경망 또한 주어진 데이터 집합을 가장 잘 표현할 수 있도록 파라미터를 조정하는 학습 과정과 이를 토대로 학습 과정에 사용되지 않는 데이터를 이용해 성능을 측정하는 테스트 과정을 거치게 된다.

학습의 목적은 주어진 데이터 집합을 이용해 입력 데이터와 출력 데이터 사이에 존재하는 관계를 추론하고, 향후 임의의 입력 데이터에 대해 올바른 출력을 내는 파라미터를 도출하는 것이다. 일반적으로 심층 신경망을 포함하는 신경망은 학습에 사용된 데이터에 대해서는 우수한 성능을 보이지만, 학습에 사용되지 않은 임의의 데이터에 대해서는 비교적 떨어지는 성능을 보이게 되는데, 이를 일반화(generalization) 문제라 한다[2, 3]. 즉, 일반화 성능이 높은 모델일수록 임의의 입력 데이터에 대한 예측 성능이 학습 데이터에 대한 예

※ 이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2017-0-00427, 딥 러닝을 이용한 시스템 독립적 악성코드 및 취약점 탐지 기술개발)
† 준 회 원 : 고려대학교 정보보호학과 석사과정
†† 정 회 원 : 고려대학교 전자 및 정보공학과 교수
Manuscript Received : September 1, 2017
Accepted : October 19, 2017
* Corresponding Author : Moon Jong Sub(jsmoon@korea.ac.kr)

측 성능과 비슷한 수준을 보이게 된다. 하지만 일반적으로 가중치, 바이어스 등 신경망 파라미터 학습에 사용되는 Stochastic Gradient Descent(SGD)[4]은 파라미터를 전역 최적해가 아닌 지역 최적해에 빠지게 함으로써 일반화 성능 뿐 아니라 전체적인 성능을 떨어뜨린다. 이에 대한 주요 원인은 다음과 같다. 첫 번째로 항상 경사도가 감소하는 방향으로 학습이 진행되는 점이다. 따라서 학습 전 파라미터의 초기값이 지역 최적해 근처에서 시작하게 될 경우, 경사도가 감소하는 방향으로만 학습이 진행되어 지역 최적해에 수렴하게 된다. 두 번째는 노드 사이의 강한 상관관계이다. 입력 데이터가 신경망을 거쳐 출력을 내는 과정에서 각 노드들은 상위 계층의 모든 노드에 영향을 미치게 되고, 결과적으로 강한 연관성을 가지게 되어 데이터를 표현하기 위한 자유도가 떨어지게 된다.

위와 같은 문제를 해결하기 위해 현재까지 많은 일반화 기법이 제안되었는데, 여기에는 가중치를 규제하는 방식과 데이터에 노이즈를 부여하는 방식이 사용되며, 최근 연구 결과인 드롭아웃[5]은 두 가지 방식을 모두 만족시키는 기법이다. 하지만 드롭아웃은 빈번한 난수 연산과 확률연산으로 인해 학습 속도가 저하되고, 신경망 각 계층의 입력 데이터 분포가 크게 변하기 때문에 작은 학습율(learning rate)을 사용해야하는 문제를 가지고 있다[6].

본 논문에서는 위와 같은 문제점을 개선하고 학습 속도를 향상시키기 위해 유전 알고리즘에서 사용되던 돌연변이 연산[7]의 개념을 도입한 새로운 기법을 제안한다. 이 기법은 적은 수의 가중치를 제어함으로써 연산 횟수를 줄이는 것과 동시에 지속적으로 가중치 공간의 새로운 지역을 탐색하여 학습이 정체되는 것을 방지한다. 또한 이러한 성능을 보이기 위해 심층 신경망 분류기의 일반화 실험과 오토인코더의 특징 추출 실험을 수행했다.

본 논문의 구성은 다음과 같다. 2장에서 심층 신경망의 일반화와 관련된 연구들을 소개하고, 3장에서 제안하는 일반화 기법에 대해 정형적으로 기술하고, 이를 적용한 학습 알고리즘을 제시한다. 4장에서 제안하는 기법의 성능을 측정하기 위한 실험을 보이고, 마지막 5장에서 결론과 앞으로의 연구 방향을 제시한다.

2. 관련 연구

이 장에서는 심층 신경망의 성능을 향상과 특징 추출에 대한 대표적인 관련 연구를 소개하고, 이후 4장에서 본 논문에서 제안하는 방법과의 비교 대상으로 삼는다.

2.1 드롭아웃(Dropout)

일반화 성능을 향상시키는 가장 좋은 방법은 여러 개의 심층 신경망 학습시키고, 각 신경망들이 내는 출력 값의 평균을 사용하는 것이다. 이렇게 여러 모델을 사용하는 방법을 앙상블(ensemble)이라 한다[8]. [5]는 하나의 신경망 모델을 이용해 앙상블의 효과를 내는 드롭아웃이라는 방법을 제시했다. 드롭아웃은 신경망에 존재하는 노드를 제거한다는 의미하며, 이는 학습 과정에만 적용되어 확률적으로 노드를 제거하여 하나의 신경망에서 여러 개의 얇은 신경망(thinned network)을 만들어내어 학습을 진행한다. 드롭아웃은 노드 간의 강한 상관관계를 끊는 효과를 내며 이는 신경망이 데이터의 특징을 잘 학습할 수 있도록 한다. 학습이 끝난 후에는 신경망의 파라미터에 드롭아웃의 확률 값을 곱하여 학습에서 나타난 얇은 신경망이 내는 출력의 평균을 사용하는 것과 같은 효과를 낸다. 드롭아웃은 간단한 방법으로 좋은 일반화 성능을 내는 것으로 알려져 있으나 몇 가지 단점이 있다. 첫 번째는 학습 속도이다. 제거할 노드를 결정하기 위해 모든 노드에 대해 베르누이 분포로부터 샘플링을 해야하기 때문에 학습 시간이 길어진다. 두 번째는 학습율의 선정이다. 드롭아웃은 특성상 신경망 각 계층의 입력 데이터의 분포가 크게 변하게 된다. 이러한 분포의 변화는 신경망 학습이 수렴하는데 까지 긴 시간을 필요하게 하며, 이러한 현상은 신경망이 깊어질수록 심화된다.

2.2 잡음제거 오토인코더(Denoising AutoEncoder, DAE)

오토인코더는 주어진 입력 데이터를 압축하는 인코더 부분과 압축된 데이터를 원래대로 복원하는 디코더 부분으로 이루어있다. 이때 인코더는 미가공 입력 데이터에 대한 특징 추출의 역할을 한다. 오토인코더의 성능은 주어진 데이터를 압축 후 얼마나 원래 데이터에 가깝게 복원하는 가로 결정되는데, 이때 입력 데이터에 노이즈를 섞은 후, 노이즈가 제거된 출력을 내도록 학습하는 것이 오토인코더의 성능을 높이는 것으로 나타났다[9, 10]. 이렇게 노이즈를 이용하는 오토인코더를 잡음제거 오토인코더라 한다. 이때 입력 데이터에 부여하는 노이즈는 일반적으로 가우시안 노이즈와 이진 마스크 노이즈가 사용되는데, 이 중 이진 마스크 노이즈는 드롭아웃의 개념과 일치한다. 따라서 시대적으로, 잡음제거 오토인코더는 드롭아웃보다 먼저 연구되었지만 드롭아웃의 응용 분야라 할 수 있다. 이렇게 오토인코더에 노이즈를 이용할 경우, 노이즈에 대한 저항성을 학습하여 입력 데이터에서 집중해야하는 부분을 파악하고 특징을 추출하게 된다. 이는 결국 입력 데이터의 다양성을 확보하게 해주며, 잡음제거 오토인코더가 추출한 특징은 기본적인 오토인코더가 추출한 특징에 비해 입력 데이터의 특성을 보다 잘 반영하게 된다.

3. 제안하는 방법론

이 장에서는 돌연변이 연산을 이용한 일반화 기법에 대해 정형적으로 기술한다. 이를 위해 일반적인 심층 신경망 모델을 기술하고, 돌연변이 연산을 적용한 학습 알고리즘을 제시한다.

3.1 전통적 신경망 모델

L개의 계층을 가진 심층 신경망은 (1)과 같이 기술할 수 있다.

$$\begin{aligned} &\boldsymbol{a}^{(l)} = W^{(l)}\boldsymbol{y}^{(l-1)} + \boldsymbol{b}^{(l)} \\ &\boldsymbol{y}^{(l)} = f(\boldsymbol{a}^{l}) \\ &l \in \{1,\ldots,L\} \end{aligned} \tag{1}$$

여기에서 l은 계층에 대한 인덱스로 $W^{(l)}$는 l 번째 계층의 가중치 행렬, $\boldsymbol{a}^{l}$은 l 번째 계층의 가중 합(weighted sum) 벡터, $\boldsymbol{b}^{l}$은 같은 계층의 편향 벡터, $\boldsymbol{y}^{(l)}$은 $\boldsymbol{a}^{(l)}$에 활성화 함수 $f(.)$를 적용한 l 번째 계층의 출력을 의미 한다 (이때 $\boldsymbol{y}^{(0)} = \boldsymbol{x}$는 심층 신경망의 입력 값이다). 본 논문에서는 이와 같은 계층을 쌓아 만든 다층 분류 모델과 오토인코더 모델을 사용한다.

3.2 돌연변이 연산

심층 신경망은 유전 알고리즘의 개념에 따라 Fig. 1과 같이 하나의 염색체로 표현 가능하다. 이때 염색체를 구성하는 각 유전자는 신경망의 가중치가 된다. 따라서 심층 신경망의 학습은 하나의 염색체를 학습하는 것과 같은 의미를 갖는다. 유전 알고리즘에서 사용되는 돌연변이 연산은 염색체를 구성하는 유전자 중 하나 혹은 하나 이상을 선택하여 강제로 값을 변경한다. 본 논문에서 제안하는 돌연변이 연산은 유전 알고리즘의 개념에서 착안하여 학습 과정에 확률적으로 수행되어 가중치 값에 강제로 변화를 일으킨다. 따라서 돌연변이 연산을 정의하기 위해 발생 시점, 변경할 가중치 선정, 가중치를 변경하는 기준을 정의해야 한다.

첫 번째로 발생 시점은 전형적인 ϵ-greedy 알고리즘[11]을 사용한다. ϵ-greedy 알고리즘은 ϵ의 확률로 탐색(exploration)을 수행하고, $(1-\epsilon)$의 확률로 활용(exploitation)을 수행한다. 본 논문에서의 활용은 학습 중이던 가중치를 변경 없이 그대로 이용하고, 탐색 수행 시 돌연변이 연산을 수행하는 것을 의미한다. Table 1은 이와 같은 방법으로 돌연변이 연산을 적용한 신경망 학습의 의사코드이다. 의사 코드에서는 신경망의 학습을 위해 SGD Optimizer를 사용했으나 Adam Optimizer[12]나 RmsProp Optimizer[13]와 같은 다른 최적화 알고리즘을 사용할 수 있다. 또한 연산이 드롭아웃과 같이 최적화 알고리즘에 직접 관여하지 않기 때문에 학습율을 조심스럽게 설정하지 않아도 된다는 장점이 있다.

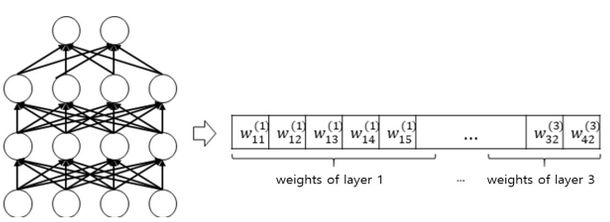

Fig. 1. Chromosomal Representation about DNN

Table 1. Pseudo Code for Training DNN using Mutation Operator

Pseudo code for training DNN

```
while not stopping criterion do
    sample r from uniform(0, 1)
    if r < ϵ then
        do mutation
    end if
    do Stochastic Gradient Descent training
end while
```

두 번째로 변경할 가중치 선정은 계층 별로 이루어지며, 두 가지 선택 방법을 가지고 정의한다. 첫 번째는 절대값 기반 가중치 선정이다. 절대값이 큰 가중치는 다음 계층으로의 데이터 전달에 큰 영향을 미치게 된다. 이로 인해 계층 간 강한 연관관계가 형성되고 이는 심층 신경망이 데이터를 표현하는 방식을 제한하게 된다. 따라서 절대값이 큰 가중치를 선정함으로써 이러한 연관관계를 끊어낼 수 있다. 두 번째 방법은 무작위 선정이다. 이 방법은 자연계에서 발생하는 무작위 돌연변이 방식과 동일한 방법이다. 무작위로 가중치를 선정할 경우, 명시적으로 계층 간 연관관계에 관여하지 않는다. 두 가지 선택방법에 적용되는 가중치의 수 k는 계층을 구성하는 가중치의 수에 비례하도록 한다. k를 설정했을 때 가중치 선정 방법들은 Table 2와 같이 표현할 수 있다.

Table 2. Selecting the Weight to Change

Weight selection method 1

$$\begin{aligned} &\{(i_k,j_k)\}_{k=1}^{K} = topk(W^{(l)}),\ \forall l \in \{1,2,\ldots,L\} \\ &where\ topk(.)\ takes\ an\ array\ of\ weights \\ &and\ return\ K\ indices\ of\ weights\ sorted\ by\ absolute\ value \end{aligned} \tag{3}$$

Weight selection method 2

$$\begin{aligned} &\{(i_k,j_k)\}_{k=1}^{K} = randIndex(W^{l}), \forall l \in \{1,2,\ldots,L\} \\ &where\ randIndex(.)\ takes\ an\ array\ of\ weights \\ &and\ return\ K\ indices\ of\ weights \end{aligned} \tag{4}$$

마지막으로 선정된 가중치에 적용되는 변경값을 정의한다. 변경값은 균등 분포를 이용하여 선정하며, 이때 최저값을 0으로 설정하고 최고값을 변경 전 가중치 값으로 설정한다. 이렇게 함으로써 변경되는 가중치는 변경 전 가중치 보다 작은 절대값을 갖게 되며, 특정 계층의 모든 가중치에 대한 벡터크기(Norm)가 작아져 정규화의 효과를 내게 된다. 이러한 가중치 값의 변경은 아래와 같이 기술할 수 있다.

$$\begin{aligned} &W^{(l)^{new}}[\{(i_k,j_k)\}_{k=1}^{K}] = uniform(0, W^{(l)^{old}}[\{(i_k,j_k)\}_{k=1}^{K}]) \\ &where\ W^{(l)}[\{(i_k,j_k)\}_{k=1}^{K}]\ is\ the\ chosen\ weights\ of\ l_{th}\ layer \end{aligned} \tag{5}$$

이 방법은 최소 범위가 0이기 때문에 변경 후 가중치가 0을 가질 가능성이 있는데, 이는 희소성(sparsity) 측면에서 중요한 의미를 갖는다. 희소성이란, 특정 가중치가 0으로 변경됨으로써 임의의 계층의 입력 데이터가 상위 계층으로 전달되지 못하며, 이러한 연산을 적절하게 수행함으로써 중요한 입력 데이터만 상위 계층으로 전달이 가능하다. 이것을 특징 선택이라 한다.

4. 실 험

이 장에서는 3장에서 제안한 돌연변이 연산의 성능을 측정하기 위해 수행한 실험 결과를 제시한다. 이 실험은 돌연변이 연산에 대해 객관적인 성능 평가를 위해 연산의 유무 실험과 선행 연구와의 비교 실험을 수행했다. 실험은 분류기의 일반화 성능과 오토인코더의 특징 추출 성능에 대해 수행했으며, 주어진 데이터 집합에 대해 최적화된 신경망을 찾는 것이 아닌 같은 초기값 조건에서의 돌연변이 연산에 대한 성능 평가를 목표로 한다. 또한 드롭아웃에 비해 적은 수의 가중치 제어로 향상된 성능과 효율성을 실험적으로 입증한다.

4.1 실험 환경

심층 신경망과 관련된 실험은 많은 연산량을 필요로 한다. 따라서 일반적인 CPU 기반 연산 대신 GPU 기반 연산을 사용한다. Table 3은 본 논문에서 제안하는 기법의 성능을 측정하기 위해 사용한 실험 환경이다.

Table 3. Experiment Specification

Component	Specification
CPU	Intel(R) Core(TM) i5-6600 3.30GHz
RAM	Samsung DDR4 16G PC4-17000 2개
Mainborad	ASUS PRIME B250-PLUS STCOM
Power	Micronics Performance 2 HV 850W Bronze
VGA	ZOTAC Geforce GTX 1080 8G
SSD	Samsung 750 EVO 120G

4.2 성능 비교 실험

돌연변이 연산의 성능을 비교 분석하기 위해 MNIST 손글씨 데이터 집합[14]을 이용하여 세 가지 실험을 실시했다. MNIST 데이터셋은 60,000장의 학습 데이터와 10,000장의 검증 데이터로 구성되어 있다. 첫 번째 실험은 돌연변이 연산의 하이퍼 파라미터를 변경하며 분류에 대한 일반화 성능을 측정했다. 두 번째 실험은 드롭아웃과 비교하여 전체적인 분류 성능 및 일반화 성능을 비교 했으며, 마지막으로 오토인코더를 사용하여 특징 추출 성능을 측정하였다. 실험은 학습 경과에 따른 추이를 확인하기 위해 학습 에러(training error)와 검증 에러(validation error)를 측정했다. 학습 에러는 모델을 학습시키기 위한 데이터 집합을 이용해 측정한 것이며, 검증 에러는 학습 데이터에 포함되지 않은 데이터 집합을 이용해 측정한 것이다. 이때 학습 에러는 미니배치(mini-batch) 단위의 에러를 측정했으며, 검증 에러는 검증 데이터 전체에 대한 에러를 측정했다.

1) 일반화 성능 측정 실험

a) 실험에 사용한 모델

이 실험에서는 돌연변이 연산의 일반화 및 하이퍼 파라미터에 따른 연산의 일반화 성능을 측정했다. 사용한 모델은 784개의 노드로 구성된 입력 계층과 각각 1,024개의 노드를 가지는 은닉 계층 3개, 10개의 노드를 가지는 출력 계층으로 구성된 다층 전방향 신경망(Multi-Layered Feedforward Perceptron, conventional) 모델로 설계하였다. 실험에 사용한 모델에서는 활성화 함수로 Rectified Linear Unit(ReLU)[15]을 사용했다. 신경망의 최적화를 위해 SGD를 이용했으며, 10,000회의 파라미터 갱신을 수행했다. 실험의 비교 대상은 Table 4와 같다.

Table 4. Performance Comparison Model based on Hyper-parameters

Model	Prob. Of Occurrence	Selection Ratio	Selection Method
conventional			
mutation1	5%	0.01%	방법2
mutation2	10%	0.01%	방법2
mutation3	5%	0.01%	방법1
mutation4	10%	0.01%	방법1

Method1 : Absolute value based weight selection
Method2 : Random weight selection

b) 결과 분석

Fig. 2는 conventional 모델과 mutation3 모델 학습의 전반적인 일반화 성능 및 분류 성능을 측정한 결과이다. conventional 모델의 경우, 학습이 진행됨에 따라 오버피팅이 발생하는 데에 반해 mutation3 모델은 학습 에러와 검증 에러의 차이가 conventional 모델에 비해 적다. 이는 돌연변이 연산에 의해 지속적으로 오버피팅에서 벗어나려 하기 때문이다. 이러한 효과는 학습이 안정화될수록 두드러지게 나타난다. 이러한 양상

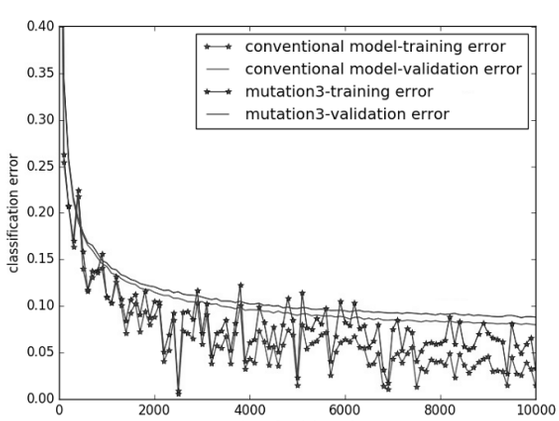

Fig. 2. Generalization and Classification Performance

을 보이는 이유는 학습 초기에는 가중치가 격렬하게 변동하기 때문에 임의로 가중치에 변화를 가해도 그 효과가 뚜렷하게 나타나지 않지만, 학습이 후반에 들어설수록 작은 가중치 변화도 학습의 안정화를 방해하기 때문이다.

Fig. 3은 돌연변이 연산을 적용한 경우의 학습 마지막 2000회에서의 돌연변이 연산을 적용한 모델 사이의 일반화 성능을 측정한 결과이다. 실험 결과가 나타내듯이 일반적으로 절대값 기반으로 가중치를 선정했을 경우 무작위 선정에 비해 좋은 일반화 효과를 나타낸다. 이는 절대값이 큰 가중치를 강제로 조정함으로써 해당 가중치에 의해 상위 계층으로 전달되는 영향력이 감소되어 계층 간 강한 상관관계가 만들어 지는 것을 방지하기 때문이다. 하지만 영향력이 큰 가중치를 조정하기 때문에 무작위 선정에 비해 전체 신경망에 가해지는 변화가 크다. 따라서 절대값 기반으로 가중치를 선정할 경우, 무작위 선정에 비해 전체적인 에러 수준이 높게 나타난다.

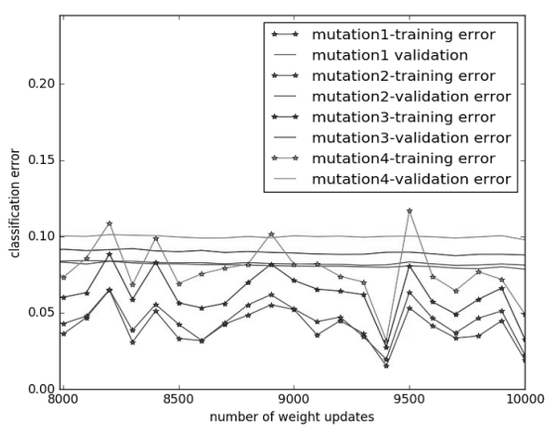

Fig. 3. Generalization Performance based on Hyper-parameters

2) 드롭아웃과의 비교 실험

a) 실험에 사용한 모델

이 실험에서는 드롭아웃과 비교하여 돌연변이 연산의 전체적인 분류 및 일반화 성능을 비교했다. 사용한 모델은 일반화 성능 측정 실험에서 사용했던 다층 전방향 신경망과 동일한 조건이다. 또한 드롭아웃의 확률은 입력 계층에 0.8, 은닉 계층에 0.5를 적용하였다. 이는 [5]에서 제시한 일반적으로 최적에 가까운 수치이다. 이 실험에서는 순수한 드롭아웃과 돌연변이 연산의 성능을 비교하기 위해 학습율 감소(learning rate decaying)이나 기타 정규화 기법은 사용하지 않았다. 실험의 비교 대상은 Table 5와 같다.

Table 5. Generalization Performance Comparison to Dropout

Model	Input layer		Hidden layer
dropout	0.8		0.5
Model	Prob. of occurrence	Selection ration	Selection method
mutation1	5%	0.01%	방법1
Method1 : Absolute value based weight selection			

b) 결과 분석

Fig. 4는 학습의 마지막 2,000회에서 드롭아웃과 돌연변이 연산의 일반화 성능을 비교한 결과이다. 두 가지 모델 모두 평균적으로 유사한 수준의 일반화 성능을 보였다. 하지만 돌연변이 연산을 사용했을 경우 더 낮은 수준의 학습 및 검증 에러를 보였으며, 학습 에러의 변동이 적다. 이는 드롭아웃에 비해 돌연변이 연산을 사용하는 것이 안정적인 학습이 가능하다는 것을 의미한다. 뿐만 아니라 드롭아웃은 학습 중 많은 노드가 변화하지만 돌연변이 연산은 매우 적은 수의 가중치를 제어하기 때문에 드롭아웃보다 효율적 연산이 가능하다.

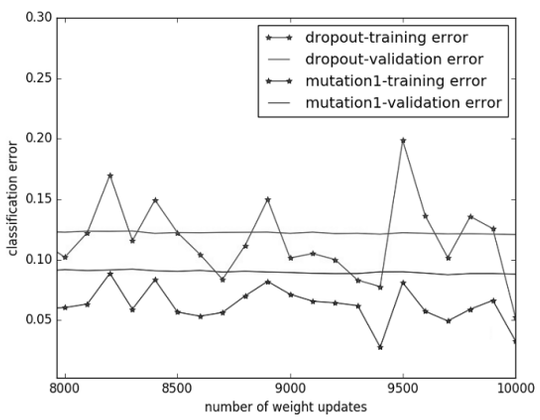

Fig. 4. Generalization Performance Compared to Dropout

3) 특징 추출 성능 실험

a) 실험에 사용한 모델

이 실험에서는 오토인코더를 이용해 특징 추출 성능을 비교했다. 원본 MNIST 데이터에 대해 오토인코더의 인코더를 이용해 특징을 압축 및 추출하고, 이를 이용해 분류 성능을 측정하였다. 실험에 사용한 conventional 모델은 784개의 노드를 가지는 입력 계층과 256개의 노드를 가지는 인코더로 설계했으며, 분류에 사용된 모델은 일반화 성능 측정 실험에서 사용한 모델과 동일하다. 오토인코더는 5,000,000회를 학습하고, 분류 모델은 10,000회를 학습했다. 실험에는 conventional 오토인코더, 잡음제거 오토인코더(DAE), 인코더에 돌연변이 연산을 적용한 오토인코더를 사용했으며, 분류 모델에는 어떠한 추가적인 기법도 적용되지 않았다. Table 6은 실험에 사용한 비교 대상이다.

Table 6. Models for Feature Extraction Comparison

Model	Prob. of occurrence	Selection ration	Selection method
conventional			
mutation1	1%	0.02%	방법1
mutation2	5%	0.02%	방법1
mutation3	5%	0.02%	방법2
method1 : absolute value based weight selection method2 : random weight selection			
Model	Binary mask noise ratio		
DAE	50%		

b) 결과 분석

Fig. 5는 학습의 마지막 2,000회에서 conventional 모델과 각 mutation 모델을 이용해 특징을 추출한 후, 이를 입력 데이터로 사용하여 분류 성능을 측정한 결과이다. 실험 결과 돌연변이 연산을 적용했을 경우, conventional 모델에 비해 높은 분류 성능을 나타내는 것을 확인했다. 이는 연산이 입력 데이터를 왜곡시켜 노이즈와 유사한 역할을 수행하고, 학습에 의해 노이즈가 제거됨으로써 인코더가 노이즈에 강한 특징을 추출하게 되기 때문이다. 이와 같은 개념은 잡음제거 오토인코더와 유사하다.

Fig. 6은 잡음제거 오토인코더와 돌연변이 연산을 이용해 특징을 추출한 후, 검증 데이터 집합에 대한 분류 성능을 측정한 결과이다. 실험에 사용한 잡음제거 오토인코더는 이진 마스크 노이즈를 이용했으며 드롭아웃을 이용해 구현했다. 실험 결과 돌연변이 연산을 적용했을 때 더 낮은 에러 수준을 확인할 수 있었다. 이는 돌연변이 연산을 사용했을 경우, 잡음제거 오토인코더에 비해 좋은 특징을 추출했음을 의미한다. 잡음제거 오토인코더는 노이즈에 강한 특징을 추출하기 위해 인코더의 각 노드가 입력 데이터의 특정 지역에 집중하도록 학습된다. 이에 비해 돌연변이 연산을 사용할 경우, 입력 데이터의 전체 영역에서 특징을 추출한다. Fig. 7은 잡음제거 오토인코더와 돌연변이 연산을 적용한 오토인코더의 인코더 가중치를 시각화 한 것이다. 또한 이와 같은 특징 추출 효과는 인코더의 희소성을 통해 확인할 수 있다. Fig. 8은 비교 대상 인코더의 평균 활성화 정도를 측정함으로써 모델들의 희소성을 확인한 것이다.

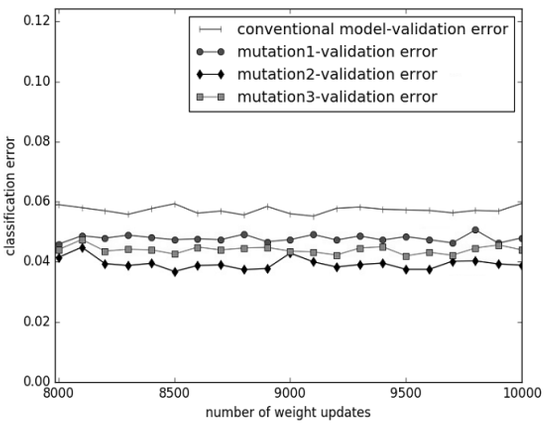

Fig. 5. Classification Performance Comparison between Models

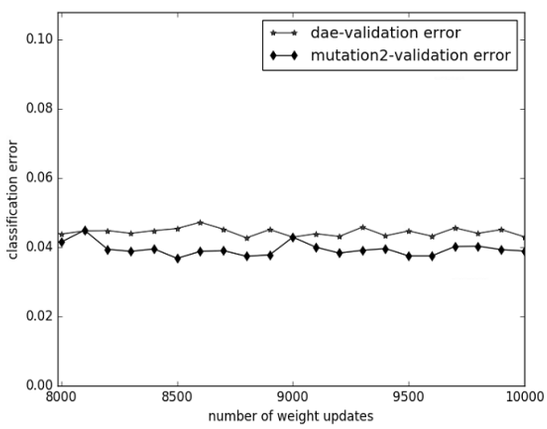

Fig. 6. Classification Performance Comparison to DAE Feature Extraction

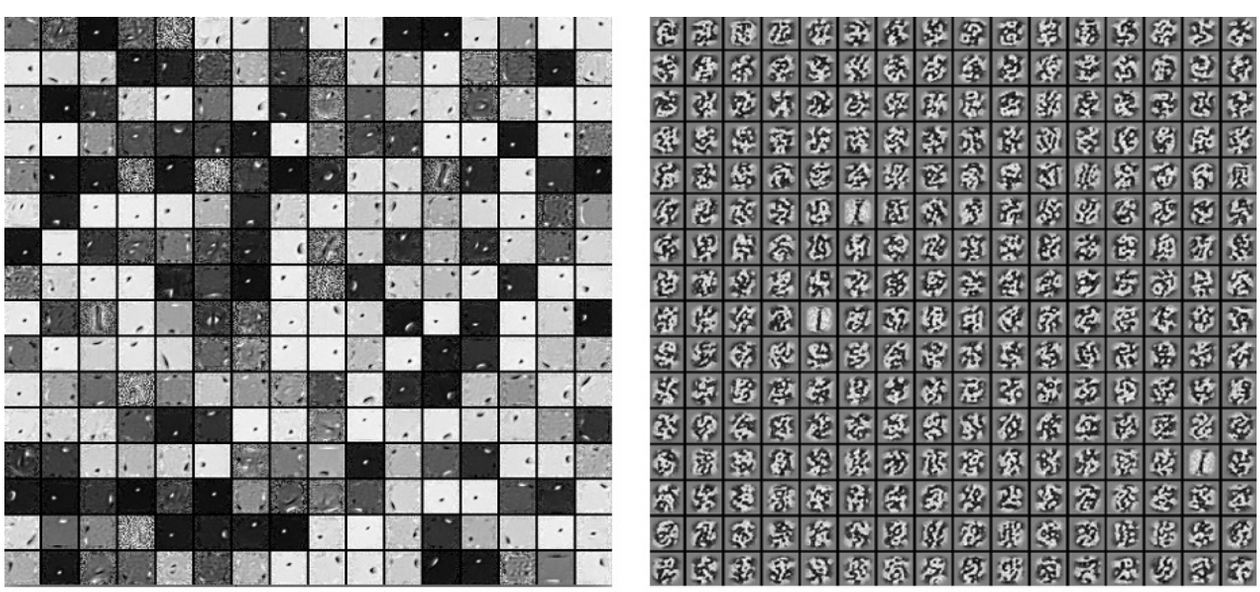

Fig. 7. Encoder Weights Visualization

(a): Denoising AutoEncoder (b): Mutation operation-applied AutoEncoder

잡음제거 오토인코더의 인코더 활성화는 대부분 0 근처에 위치한다. 이는 활성화되는 노드의 수가 적어 인코더의 출력이 희소해지는 것을 의미한다. 반면 기본형 모델의 경우, 0.5 근처에 평균이 위치하며, mutation2 모델은 잡음제거 오토인코더와 conventional 모델의 중간 정도 활성화를 보이고 있다. 이는 돌연변이 연산에 의해 인코더가 적절한 희소성 수준을 학습하기 때문이다.

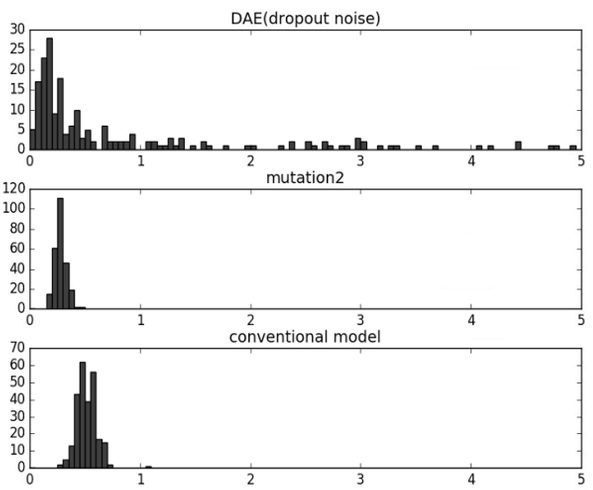

Fig. 8. Average Activation Rate in Encoder

5. 결 론

본 논문에서는 유전 알고리즘에서 사용되던 돌연변이 연산을 이용해 심층 신경망의 성능 향상을 위한 새로운 기법을 제안하고, 이를 입증하기 위한 실험을 수행했다. 이 방식은 드롭아웃과 같이 신경망에 많은 변화를 주지 않고도 동등 이상의 수준의 일반화 성능을 낼 수 있음을 증명했으며, 오토인코더 실험을 통해 잡음제거 오토인코더에 비해 노이즈에 강한 특징을 추출할 수 있음을 보였다. 이러한 성능은 돌연변이 연산의 하이퍼 파라미터를 적절하게 설정함으로써 제어가 가능하며, 적은 수의 가중치를 제어함으로써 드롭아웃에 비해 효율적인 연산이 가능하다.

돌연변이 연산은 정의에 따라 기존 가중치보다 절대 값이 작은 값을 새로운 가중치 값으로 설정한다. 이 과정에서 학습에 정규화 효과를 낼 수 있도록 하며, 새로운 가중치가 가질 수 있는 값의 범위를 제어함으로써 다양한 정규화를 구현할 수 있다. 따라서 돌연변이 연산과 정규화의 관계 규명이 향후 연구가 될 것이다.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep Learning,” *Nature*, Vol.521, pp.436-444, 2015.

[2] E. Levin, N. Tishby, and S. Solla, “A statistical approach to learning and generalization in layered neural networks,” *Proceedings of the IEEE*, Vol.78, No.10, pp.1568-1574, 1990

[3] C. M. Bishop, “Neural Network for Pattern Recognition,” Oxford University Press, pp.332-380, 1995.

[4] L. Bottou, “Stochastic gradient learning in neural networks,” *Proceedings of Neuro-Nimes*, Vol.91, No.8, 1991.

[5] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *Journal of Machine Learning Research*, Vol.15, No.1, pp.1929-1958, 2014.

[6] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” In *International Conference on Machine Learning*, pp.448-456, 2015.

[7] M. Mitchell, “Genetic Algorithms: An Overview,” *Complexity*, Vol.1, No.1, pp.31-39, 1995.

[8] L. K. Hansen and P Salamon, “Neural Network Ensembles,” *IEEE Trans, Pattern Analysis and Machine Intelligence*, Vol. 12, No.10, pp.993-1001, 1990.

[9] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and Composing Robust Features with Denoising Autoencoders,” In *International Conference on Machine Learning*, pp.1096-1103, 2008.

[10] P. Vincent, H. Larochelle, L. Lajoie, Y. Bengio, and P. A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” In *International Conference on Machine Learning*, pp.3371-3408, 2010.

[11] R. S. Sutton and A. G. Barto, “Reinforcement Learning: An Introduction,” Cambridge: MIT Press, pp.25-42, 2012.

[12] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[13] Tieleman, Tijmen and G. Hinton, “Lecture 6.5: RMSProp-Divide the gradient by a running average of its recent magnitude,” In COURSERA: Neural Networks for Machine Learning, 2012.

[14] Y. LeCun, C. Cortes, and C. J. C. Burges, “The Mnist Database of handwritten digits” [Internet], http://yann.lecun.com/exdb/mnist/.

[15] X. Glorot, A. Bordes, and Y. Bengio, “Deep Sparse Rectified Neural Networks,” In *Conference on Artificial Intelligence and Statistics*, 2011.

전 승 호

http://orcid.org/0000-0002-7116-6062
e-mail : ohgnu90@korea.ac.kr
2016년 명지대학교 컴퓨터공학과(학사)
2017년~현 재 고려대학교 정보보호학과 석사과정
관심분야 : 딥 러닝, 시스템 보안

문 종 섭

http://orcid.org/0000-0002-6457-4316
e-mail : jsmoon@korea.ac.kr
1981년 서울대학교 계산통계학과(학사)
1983년 서울대학교 계산통계학과(석사)
1991년 Illinois Institute of Technology 전산학(박사)
2002년~현 재 고려대학교 전자 및 정보공학과 교수
관심분야 : 정보보호, 전자공학, 통신공학