유전 알고리즘을 이용한 신경망 구조 최적화

Neural Network Architecture Optimization using Genetic Algorithm

요 약

신경망 학습에 있어서 하이퍼파라미터를 설정하는 것과 신경망의 구조를 설계하는 것은 신경망의 성능에 있어서 매우 핵심적인 부분이다. 하지만 사람이 하이퍼파라미터와 신경망의 구조를 어떻게 정해주어야 신경망의 성능이 좋을지 알기도 어려울뿐더러 많은 시간이 요구된다. 현재 대부분의 신경망 구조는 전문가의 경험적 지식에 의존하여 직관적인 판단에 의해 설계된다. 이에 본 논문에서는 유전 알고리즘을 이용하여 하이퍼파라미터와 신경망 구조를 최적화하는 방법에 대해 소개한다.

1. 서 론

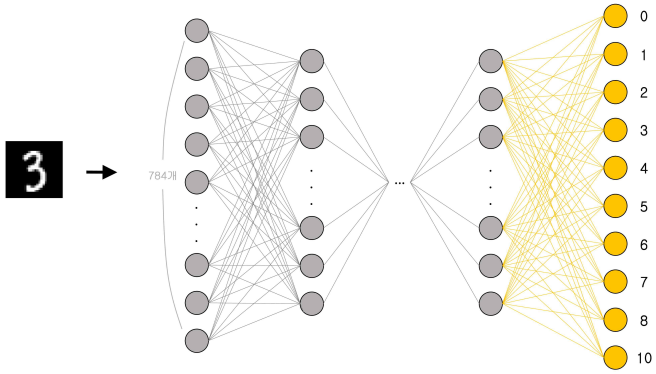
현재 신경망(Neural Network) 학습에 있어서 하이퍼파라미터를 설정하는 것과 신경망의 구조를 설계하는 것은 신경망의 성능에 있어서 매우 핵심적인 부분을 차지한다. 여기서 하이퍼파라미터란 훈련 데이터와 학습 알고리즘에 의해 자동 획득되는 매개변수와 달리, 사람이 직접 설정해야 하는 매개변수이다[1]. 대부분의 신경망을 학습시킬 때 많은 하이퍼파라미터들의 설정은 사람이 직접 설정하는 경우가 많다. 하지만 사람이 하이퍼파라미터와 신경망 구조를 어떻게 정해야 신경망의 성능이 좋을지 알기 어렵고 많은 시간을 필요로 한다. 본 연구는 유전 알고리즘을 이용해 신경망 구조 최적화가 사람에 비해 적은 시간을 소비하면서도, 유사하거나 더 좋은 결과를 도출할 수 있음을 보이는 것을 목표로 한다.

2. DNN (Deep Neural Network)

인공 신경망은 데이터를 입력받는 입력층, 최종 결과를 출력하는 출력층, 입력층과 출력층 사이에 존재하며 사람 눈에 보이지 않는 은닉층으로 이루어져 있다[2]. 각 층은 가중치를 가진 선들로 연결되어 있으며, 이 가중치를 더해 계산한 값으로 노드는 활성화 함수를 통해 임곗값을 넘는지 확인하고 다음 층으로 입력을 보낼지 결정한다.

여기서 2개 이상의 은닉층으로 이루어진 신경망을 DNN(Deep Neural Network)이라고 한다. 각 은닉층은 특정 요소, 특징을 나타내며 상위 은닉층으로 갈수록 높은 특징을 나타낸다. 따라서, DNN은 높은 수준의 특징을 필요로 하는 복잡한 작업을 수행하는데 있어서 탁월한 성능을 갖는다[2]. DNN은 특징 추출(feature extraction)이 자동으로 수행되는 것과 복잡한 비선형 관계를 표현할 수 있다는 장점이 있다. DNN은 경사 하강법(Gradient Descent)을 통해 적절한 가중치를 갱신하는 오차역전파법(Backpropagation)을 사용한다. 이때, 은닉층의 개수가 늘어날수록, 출력층의 오차가 입력층까지 전달되지 않는 경사 사라짐 문제(Vanishing Gradient)와 반대로 갱신 값이 극단적으로 커지는 경사 폭발 문제(Exploding Gradient)가 발생하는 단점이 있다[3].

2-1. MNIST 데이터베이스



<그림 1. MNIST DNN 구조>

본 논문은 MNIST 데이터를 사용하는 DNN 구조를 최적화한다. MNIST 데이터를 사용하는 이유는 복잡하지 않고 가장 먼저 시도해보기 좋은 데이터셋 이라고 판단하였기 때문이다.

MNIST 데이터셋를 사용해 만든 DNN을 MNIST DNN이라고 한다[4]. MNIST란 손글씨 숫자 이미지 집합이고, MNIST 이미지 데이터는 28x28로 이루어져 있기 때문에 입력층의 노드 개수는 모든 픽셀의 수인 784개이다. MNIST 데이터는 0부터 9까지의 숫자를 나타내기 때문에 출력층의 노드 개수는 10개이다. 은닉층의 가중치와 활성화 함수를 통해 임의의 계산을 해 입력된 이미지가 0부터 9중에 어떤 숫자인지 판단한다.

3. 유전 알고리즘

유전 알고리즘이란 생물의 진화 과정을 모방한 것으로, 선택, 교차, 변이 등의 연산을 통해 최적에 가까운 해를 찾아주는 방법론이다[5]. 유전 알고리즘의 구조는 초기 세대의 염색체 구성, 초기 세대의 유전염색체들에 대한 적합도 계산, 적합도가 높은 염색체 선택, 자손 생성의 반복으로 이루어진다. 이 과정에서 교차, 돌연변이 연산을 통해 유전자풀의 다양성을 확보할 수 있다.

유전 알고리즘은 해를 정형화된 유전표현으로 나타내어야 하고, 각각의 개체를 평가하기 위한 적합도 함수를 정의해주어야 한다[6]. 본 논문은 DNN의 구조와 하이퍼파라미터를 최적화시키는 것이 목표이므로 본 논문의 시스템에서의 유전표현은 learning rate, epochs, hidden layers, optimizer, 활성화 함수, 가중치 초깃값으로 이루어진다. 유전 알고리즘으로 유전표현들을 최적화시키기 전에 위 유전표현들이 DNN의 성능에 직접적으로 영향을 주는지 확인하기 위해 사전 실험을 진행했다.

<표 1. 각 요소 값 변화에 대한 정확도 변화>

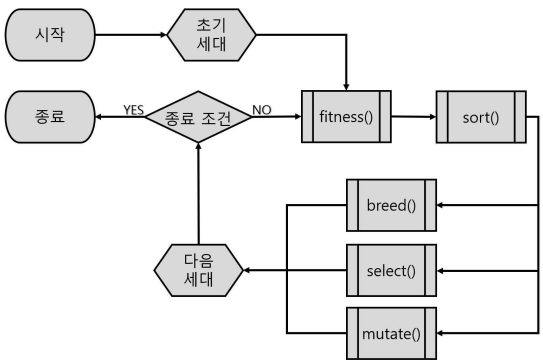
	Activation Function	Optimizer	Weight Init	Hidden Layer Dept	Learning Rate	Epochs	Accuracy
Original	sigmoid	adam	zero	2	0.001	15	0.9737
A	sigmoid	adam	zero	2	0.001	5	0.6628
B	sigmoid	adam	zero	2	0.01	15	0.098
C	sigmoid	adam	zero	6	0.001	15	0.4627
D	sigmoid	adam	random	2	0.001	15	0.9492
E	sigmoid	SGD	zero	2	0.001	15	0.1135
F	relu	adam	zero	2	0.001	15	0.1135

위 실험 결과를 통해 신경망 구조에 있어서 하이퍼파라미터값과 신경망의 구조가 신경망의 성능에 영향을 미친다는 사실을 알 수 있다.

각각의 개체를 평가하기 위한 적합도 함수는 학습된 DNN의 정확도로 설정하였다.

4. 시스템 설계

본 논문은 유전 알고리즘을 사용해 신경망의 구조 및 파라미터를 최적화하는 것을 목표로 한다. 따라서 본 논문의 유전 알고리즘의 염색체 유전정보에는 은닉층의 개수, 은닉층의 노드 개수, 활성화 함수의 종류, 에폭 수, 최적화 함수의 종류, learning rate, 가중치 초깃값의 종류를 넣어준다. 이 유전정보의 형태에 맞게 초기 세대를 정의해주어야 하는데 이때 일정 범위 내의 무작위 값들을 넣어서 정의해준다. 본 논문의 시스템에서는 100개의 개체를 유전 알고리즘의 입력층으로 넣어준다. 이후 초기 세대에 대한 적합도를 구해주어야 한다. 이 유전 알고리즘의 적합도는 신경망의 구조, 파라미터가 유전정보로 주어졌을 때 그 유전정보로 이루어진 DNN의 성능이다. 그러므로 적합도 함수는 DNN의 정확도를 구해주는 유전정보를 받아 생성된 DNN으로 설정해준다. 이렇게 임의로 생성된 초기 세대에서 적합도를 구하고 선택, 교배, 변이 연산을 통해 다음 세대로 넘어갈 개체들을 정해준다. 이 과정을 반복하여 최적 해에 가까운 적절한 하이퍼파라미터 값과 신경망의 구조를 찾을 수 있다.



<그림 2. 시스템 구조도>

4-1. 적합도 함수

본 유전 알고리즘의 적합도는 유전자에 의해 만들어진 DNN의 성능이다. 따라서 유전 알고리즘을 통해 생성된 하이퍼파라미터와 신경망 구조의 정보를 담고 있는 개체가 입력으로 들어가게 되고 그 유전정보를 통해 만들어진 DNN에 MNIST 데이터를 학습시켜 얼마나 좋은 성능을 내는지 정확도를 출력으로 설정하였다. 이렇게 구한 적합도를 sort() 함수를 통해 적합도가 높은 개체부터 내림차순으로 재배열을 해준다.

4-2. 선택

선택 연산에서는 적합도 함수를 거쳐 내림차순으로 정렬된 적합도 값들을 기반으로 다음 세대로 보낼 개체들을 우선적으로 선택한다. 본 시스템에서는 다음 세대로 보낼 개체들을 선택하는 방법으로 상위 30개의 개체를 우선 선택한 후 남은 70자리를 교배와 변이 연산을 통해 채운다.

4-3. 변이

변이 연산에서는 유전형의 값에 변화를 주어 더 다양한 유전자풀을 갖게 해주는 연산을 수행한다. 본 시스템

의 변이 연산에서는 선택 연산에서 선택된 30개의 개체를 받아와 각 개체에 대해 최소 1개에서 3개의 유전형의 값에 대해 변화를 준다. 이때, 사용자가 각 요소 값들에 대한 범위를 설정하여 각 요소 값들이 비이상적인 값으로 변화하지 않도록 한다. 이렇게 설정한 범위 안에서 무작위로 값을 받아와 더 다양한 유전자풀을 갖게 해준다. 이후 남은 40자리는 교배연산을 통해 채워진다.

4-4. 교배

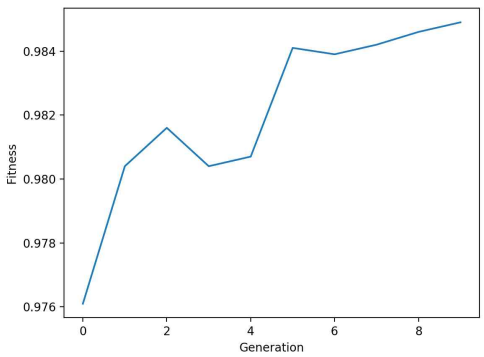
교배연산에서는 두 개체의 염색체를 교차시켜 새로운 개체를 만들어 더 다양한 유전자풀을 갖게 해주는 연산을 수행한다. 본 시스템의 교배연산에서는 선택 연산과 변이 연산을 통해 생성된 개체들 중 두 개의 개체를 뽑아 무작위로 분리점을 정해 염색체를 분리시킨 후 분리된 염색체를 다른 염색체의 것과 서로 바꾸어 새로운 개체를 만든다.

이렇게 선택, 변이, 교배연산을 통해 생성된 100개의 개체가 다음세대로 보내진다.

5. 실험결과

본 논문의 실험에서는 10개의 개체를 10세대동안 유전알고리즘을 적용시켜 최적화를 해보았다. 실험결과는 표 2와 같았다.

<표 2. 실험결과>



본 실험에서 생성된 첫세대중 가장 성능이 좋았던 개체는 [0.9, xavier, Adagrad, tanh, 2]의 유전자를 가진 개체로 정확도는 0.9761이었다. 이 10개의 개체가 10번의 유전알고리즘을 거쳐 정확도 0.9849를 기록하였다.

이 때의 최적의 개체는 [0.15, xavier, Adagrad, relu, 3]의 유전자를 가졌으며 위 유전자로 이루어진 DNN의 신경망의 구조가 최적에 근접한 구조라는 결론을 내릴 수 있다.

표 중간에 정확도가 떨어지는 현상을 발견할 수 있는데 이는 변이와 교차연산을 통해 나온 구조의 정확도가 이전세대의 가장 우수한 구조의 정확도 보다 높지 않아 이전세대의 가장 우수한 구조로 한번더 신경망을 실행하였을 때 이전의 실행보다 정확도가 떨어지는 결과가 나와 발생하는 현상이다.

6. 결 론

본 논문의 시스템은 신경망의 구조 및 하이퍼파라미터를 최적화 해준다. 시스템이 잘 작동한다면 신경망의 구조와 하이퍼파라미터를 사람이 직접 정해주지 않아도 기존에 최적이라고 생각되었던 하이퍼파라미터 세팅에 대한 정확도의 값에 필적하는 결과가 나올 것이다.

본 논문의 실험은 DNN의 구조를 최적화 해보았는데 MNIST DNN의 경우 난이도가 높지 않은 문제였기 때문에 눈에띄는 만큼의 큰 최적을 보여주지는 못하였다. 본 논문의 결과를 통해 유전알고리즘으로 신경망의 구조를 최적화 할 수 있다는 결론을 얻었으며 이후에는 좀더 난이도가 높고 복잡한 CNN의 구조를 최적화 해주는 실험을 진행해볼 것이다.

7. 참고문헌

[1] 사이토 고키, Deep Learning from Scratch, 한빛미디어, 2017.

[2] 김인중, Deep Learning: 기계학습의 새로운 트랜드, 한국통신학회지(정보와통신) 31(11), 52-57(6page), 2014.10.

[3] Hochreiter S., Bengio Y., Frasconi P., and Schmidhuber J. "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press, 2001.

[4] Y. LeCun, and C. Cortes., MNIST handwritten digit database, http://yann.lecun.com/exdb/mnist/, 2010

[5] M. Srinivas, Lalit M. Patnaik, Genetic algorithms, A Survey. Computer, Vol. 27, No. 6 (June 1994), 17-26, 1994.

[6] 주형주, 박정은, 전지윤, 김철연, 유전 알고리즘을 이용한 CNN 내 가중치의 최적화, 한국정보과학회 학술발표논문집, 1944-1946, 2019.