CNN 에서의 DropOut 과 DropConnect 에 대한 성능 비교*

장윤석 임현일
경남대학교 컴퓨터공학부
e-mail: ysjang0318@gmail.com hilim@kyungnam.ac.kr

Performance Comparison of DropOut and DropConnect in CNN

Yun-Seok Jang Hyun-il Lim
Dept. of Computer Engineering, Kyungnam University

요 약

CNN 은 합성곱 연산을 사용하는 인공신경망의 한 종류이다. 이러한 인공 신경망에서는 훈련 데이터에 대한 과도한 학습으로 인해 시험 데이터에 제대로 반응하지 못하는 오버피팅이 발생할 우려가 있다. 이를 해결하기 위해 DropOut 과 DropConnect 를 사용할 수 있다. 본 논문에서는 DropOut 과 DropConnect 를 통한 학습 정도를 실험을 통해서 비교해보고, 인공 신경망에서 이 방법의 효과를 살펴본다.

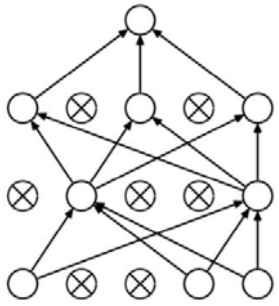
1. 서론

최근의 데이터 분석 기술에 대한 지배적인 트렌드는 AI 와 머신 러닝을 통한 분석이다. 머신 러닝은 다양한 모델을 통해 학습 모델을 만드는데, CNN 은 합성곱 연산을 사용하여, 이미지 인식 분야 등에서 널리 쓰이는 머신 러닝 모델이다. 다수의 합성 곱 계층으로부터 특징을 추출하고 서브 샘플링으로 단순화한 후, 마지막으로 완전 연결 계층으로 이전 계층의 처리결과를 연결하여 이미지를 분류하게 된다. CNN 에서 학습 횟수를 늘려가면 훈련 정확도가 100%에 가깝게 증가하지만, 시험 정확도 값은 오히려 낮아지는 문제가 생길 수 있는데, 이는 과도한 학습으로 인한 과적합 문제이다. 훈련 데이터에 너무 과하게 학습되어 새로운 데이터에 대한 예측을 잘못하는 것이다.

이를 해결하기 위한 방법으로는 훈련 데이터를 많이 수집하거나, 활성 함수의 개수를 줄이는 방법, 그리고 가중치가 너무 커지지 않도록 정규화하는 방법 등이 있다[2]. 과적합을 줄이기 위해 정규화 하는 방법중 DropOut 과 DropConnect 에 대해 살펴보고 이 방법이 학습에 어떻게 영향을 미치는지 실험을 통해 비교해본다.

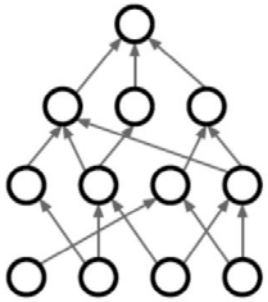
2. DropOut 과 DropConnect

DropOut 은 완전 연결된 일반적인 신경망의 학습 과정에서 특정 비율로 임의의 노드들을 작동하지 않게 하고 학습을 적용시키는 방법이다. 이는 너무 많은 노드로 인해 필요 이상으로 학습데이터에 최적화된 학습 결과를 가져오는 과적합 문제를 줄일 수 있는 방법이다.



(그림 1) DropOut 개념도[2]

(그림 1)은 DropOut 의 개념도를 보여주고 있다. DropOut 은 학습과정에서 오버피팅을 줄이기 위해 각 층에서 일부 노드를 무작위로 제외시키고 학습을 하는 방법이다.



(그림 2) DropConnect 개념도[1]

DropConnect 는 DropOut 을 일반화한 형태로 학습과정에서 노드를 작동하지 않게 하는 것이 아닌 노드 사이에 전달되는 가중치를 무작위로 차단하는 방법이다.

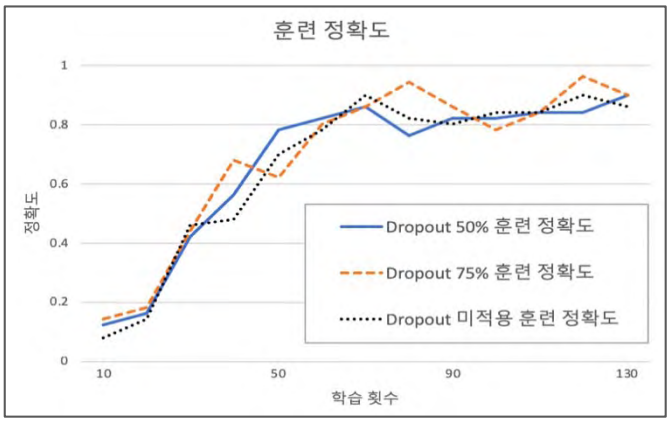
* 이 논문은 2017 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2017R1D1A1B03034769).

모든 노드는 학습에 참여하지만, 다음 층으로 가는 일부 가중치를 0 으로 차단한다. 본 논문에서는 이 두 방식이 실제 학습 과정에서 어떤 차이를 보이는지 실험을 통해 비교해 보았다.

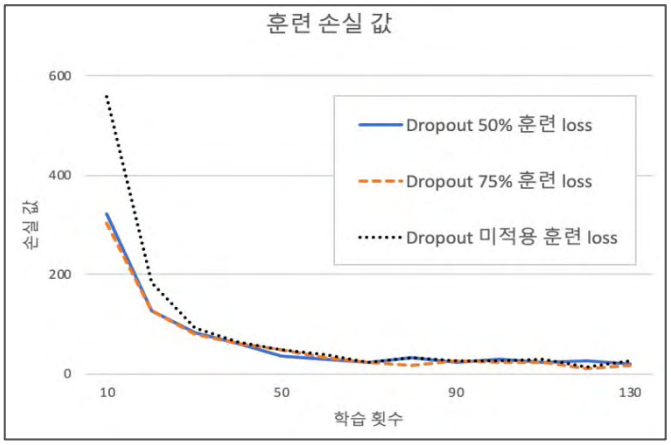
3. Dropout 적용한 모델의 비교 실험

본 실험에서 사용한 모델은 TensorFlow[3]를 이용하여 구현한 CNN 을 이용하였고, MNIST 를 학습 데이터로 사용하였다. 본 실험에서는 DropOut 을 적용시킬 때 학습에서 활성화 또는 비활성화 하는 노드의 비율에 따른 학습 과정의 변화를 실험하였다. 이 실험에서는 DropOut 확률 50%, 75% 그리고 미적용된 모델을 비교 하여 실험하였다.

그림 3 과 4 에서 실험 과정에서 변화하는 정확도와 훈련 손실 값을 보여준다. DropOut 을 적용한 모델의 학습 속도가 그렇지 않은 모델보다 속도가 높았다. 학습 초반에 보다 빨리 훈련 손실 값을 줄일 수 있음을 알 수 있으며, 학습 정확도는 35 회 이후부터 노드 특성에 따른 차이를 보여준다.

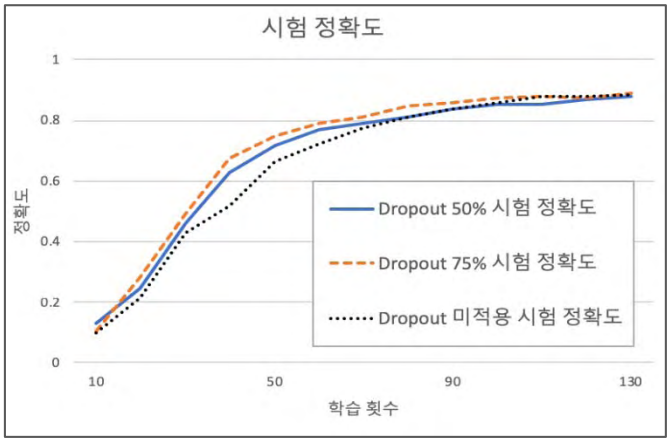


(그림 3) DropOut 적용에 따른 훈련 정확도의 변화

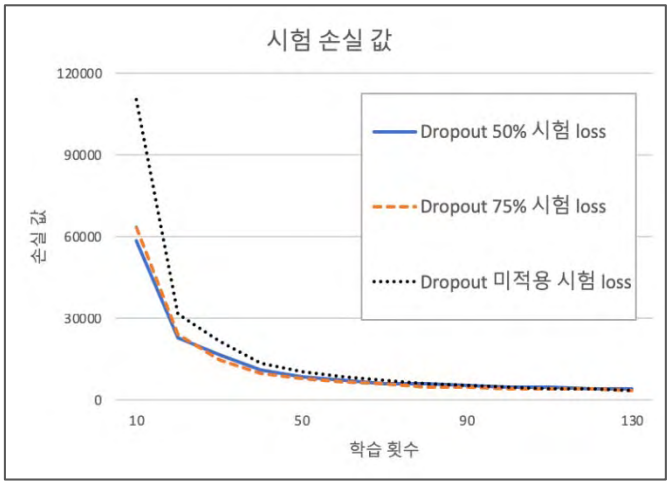


(그림 4) DropOut 적용에 따른 훈련 손실 값의 변화

그림 5 와 6 은 시험 정확도와 손실 값의 변화를 보여준다. DropOut 을 적용시킨 모델은 미적용 모델에 비해 초반 학습 정확도가 약간 높고 손실 값도 낮았다. 그리고 50%의 노드만 활성화시킨 모델보다 75% 의 노드를 활성화시킨 모델의 성능이 조금 더 좋았다. 따라서 DropOut 을 하는 경우에는 비율을 어떻게 정하는지에 따라 학습 성능이 달라질 수 있음을 보여준다.

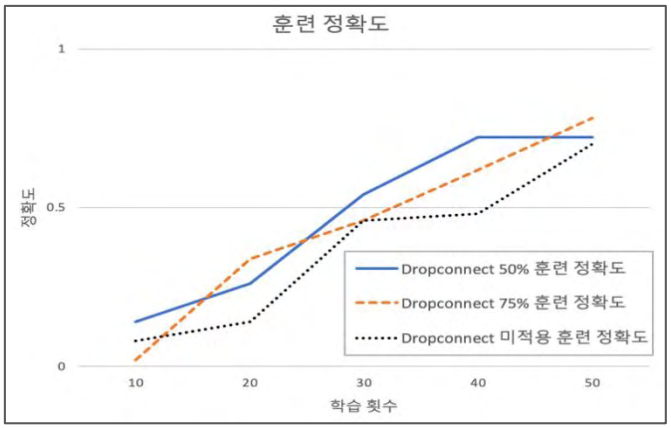


(그림 5) DropOut 적용에 따른 시험 정확도의 변화

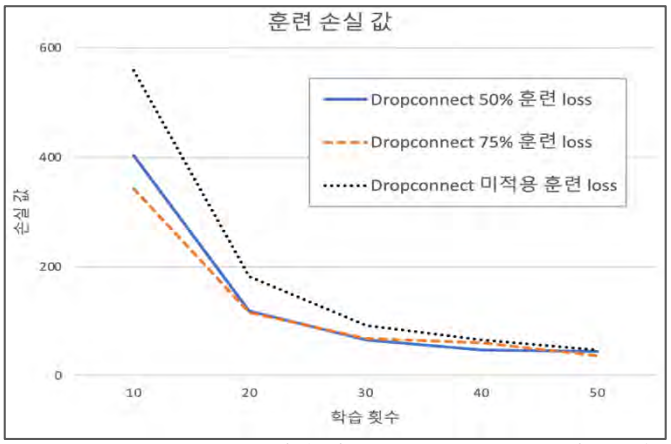


(그림 6) DropOut 적용에 따른 시험 손실 값의 변화

4. Dropconnect 적용 모델과 미적용 모델 비교



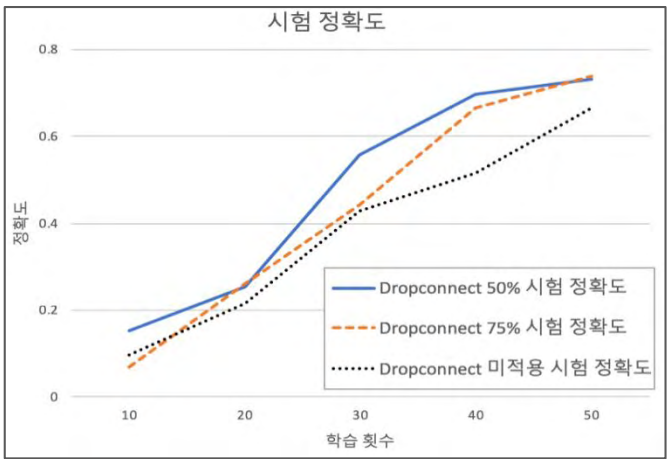
(그림 7) DropConnect 적용에 따른 훈련 정확도 변화



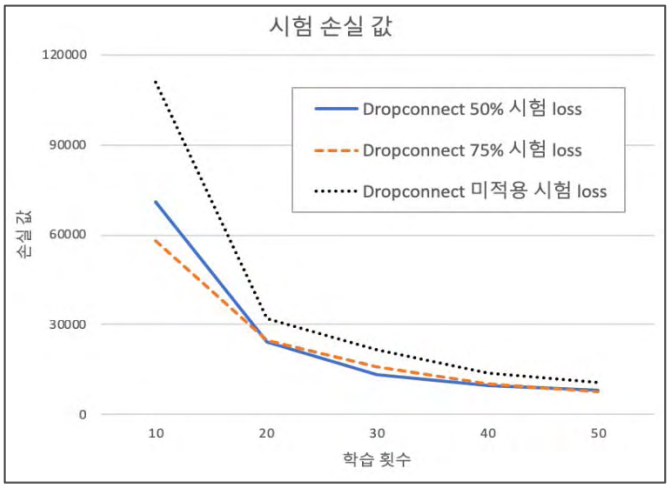
(그림 8) DropConnect 적용에 따른 훈련 손실 값 변화

본 실험에서는 DropConnect 적용에 따른 학습의 변화 과정을 실험하였다. 그림 7 과 8 은 훈련 정확도와 훈련 손실 값을 보여주고 있다. DropConnect 을 적용하는 경우 미적용의 경우보다 정확도가 높게 유지되고, 손실 값도 작다는 것을 알 수 있다.

그림 9 와 10 은 DropConnect 적용에 따른 시험 정확도와 손실 값에 대한 비교 실험 결과를 보여준다.



(그림 9) DropConnect 적용에 따른 시험 정확도 변화



(그림 10) DropConnect 적용에 따른 시험 손실값 변화

DropConnect 를 적용시킨 모델과 일반 모델의 시험 정확도와 손실 값을 비교해 보면 DropOut 과 마찬가지로 시험 정확도가 높고 손실 값도 낮은 값에서 시작하는 것을 확인할 수 있다. 따라서 초반에 보다 빨리 학습된다는 것을 알 수 있다.

neuron	model	error(%) 5 network	voting error(%)
relu	No-Drop	1.62 ± 0.037	1.40
	Dropout	1.28 ± 0.040	1.20
	DropConnect	1.20 ± 0.034	**1.12**
sigmoid	No-Drop	1.78 ± 0.037	1.74
	Dropout	1.38 ± 0.039	**1.36**
	DropConnect	1.55 ± 0.046	1.48
tanh	No-Drop	1.65 ± 0.026	1.49
	Dropout	1.58 ± 0.053	1.55
	DropConnect	1.36 ± 0.054	**1.35**

(표 1) MNIST 에서의 활성 함수 종류, 모델에 따른 오류율[1]

(표 1)은 [1]에서 제시한 DropConnect 의 효율이 Dropout 보다 우수하다는 실험 결과를 보여주고 있다. 본 실험의 모델에서 학습시켜본 실험 결과에서 두 모델은 큰 차이를 보이지 않았다. 실험 결과로부터 신경망 모델의 성능은 실험 데이터 및 모델의 설계 방법이 중요하다는 것을 알 수 있다.

5. 결론

본 논문에서는 인공신경망에서 DropOut 과 DropConnect 를 적용하는 경우 학습 과정에서 어떤 변화를 보여주는지 실험을 통해 살펴보았다. DropOut 과 DropConnect 모두 일반 모델보다 높은 시험 정확도와 낮은 손실 값을 보여주었다.

본 실험에서는 MNIST 라는 10 개 클래스의 작은 데이터를 가지는 작은 규모의 데이터를 이용한 실험이라 실험 결과가 제한적일 수도 있다. 향후 실험을 확대해서 CIFAR-10 과 같은 좀 더 큰 규모의 훈련 데이터와 시험 데이터를 이용해 학습 과정을 살펴본다면 보다 명확한 차이를 보일 수 있을 것이라 예상된다.

감사의 글

이 논문은 2017 년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2017R1D1A1B03034769).

참고문헌

[1] Li Wan, Matthew Zeiler, Sixin Zhang, Yann LeCun, Rob Fergus “Regularization of Neural Networks using DropConnect” PMLR 28(3):1058-1066, 2013.

[2] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov “Dropout: A Simple Way to Prevent Neural Networks from Overfitting” JMLR 15(1):1929-1958, 2014.

[3] TensorFlow. https://www.tensorflow.org/