



Dropout과 마찬가지로 overfitting을 완화시켜주며, Dropout이 학습 과정에서 지정해준 비율만큼의 뉴런(unit)을 0으로 만들어서 사용을 하지 않게 되어, 이를 통해 과도하게 학습하는 것을 막아 과적합을 방지하는 반면, Dropconnect는 Dropout에서 뉴런(unit)의 connection(weight)을 0으로 만들어버리는 것이다. 이를 통해 뉴런은 이전 layer에서 랜덤한 weight를 받게 된다.

어떠한 뉴런을 0으로 만드는 것은 그 뉴런에 연결된 connection 모두를 0으로 만들어버리는 것과 같으므로 Dropconnect는 Dropout의 일반화라고 볼 수 있다. 그렇기 때문에 connection만 생략을 하는 Dropconnect에서 학습 때마다 Dropout보다 더 많은 모델을 훈련시키는 것과 같은 결과를 낼 수 있다.

Dropout을 적용했을 때의 신경망의 출력 수식

$$S_i(I) = \sum_{j=1}^n w_{ij} \delta_j I_j \quad \text{for } i = 1, \dots, k$$

δ : 뉴런의 생략에 관련 베르누이 확률값, w : 가중치, I : 입력데이터

DropConnect를 적용했을 때의 신경망의 출력 수식

$$S_i(I) = \sum_{j=1}^n \delta_{ij} w_{ij} I_j \quad \text{for } i = 1, \dots, k$$

δ : 뉴런이 아닌 connection 생략 관련 베르누이 확률값, w : 가중치, I : 입력데이터

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

<http://www.koreascience.or.kr/article/CFKO201924664106359.pdf> 논문에서의 실험 결과를 따르면, 기본적으로 Dropout, Dropconnect 모두 정확도가 높고 손실값이 낮으며, relu, sigmoid, tanh 3가지 함수를 적용하여 실험을 했을 때, relu와 tanh 함수를 사용하였을 때 Dropconnect가 Dropout보다 조금 더 나은 성능을 보였음을 알 수 있다.

Tensorflow 4 | dropconnect layer : <https://pypi.org/project/dropconnect-tensorflow>