



🌙 Type something...

01) Dropout

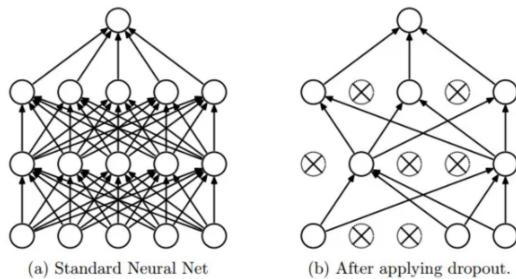
01_dropout_forward(x, dropout_param)

1. Inline Question 1

02_dropout_backward(dout, cache)

2. Inline Question 2

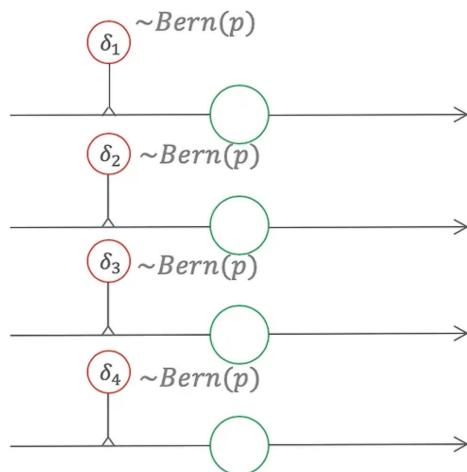
01) Dropout



- Dropout은 랜덤한 뉴런의 output을 0으로 만드는 것이다.
- 이를 통해서 overfitting을 막을 수 있다.

01_dropout_forward(x, dropout_param)

- dropout ratio = p : 살리는 비율 ex) $p = 0.7 \rightarrow 30\%$ 만 날리는 것.
- dropout의 원리는 각 layer에 있는 노드들에 베르누이 분포를 따르는 변수를 곱하여 각 노드를 random하게 선택하는 것이다.
- 0이 나온 확률변수가 곱해진 노드는 자연스럽게 weight 가 0이 되어 사라지는 효과를 볼 수 있다.
- test일 때는 모든 데이터를 사용해야 하므로 dropout을 사용하지 않는다.



```

if mode == "train":
    #####
    # TODO: Implement training phase forward pass for inverted dropout. #
    # Store the dropout mask in the mask variable. #
    #####
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    #
    # mask를 구현해서 곱해주는 방식으로 drop을 한다.
    # np.random.rand(shape) 사용 0~1사이의 값을 임의로 끄냄
    mask = (np.random.rand(*x.shape) < p) / p
    out = x * mask

    # *****END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    #
    # END OF YOUR CODE
    #####
elif mode == "test":
    #####
    # TODO: Implement the test phase forward pass for inverted dropout. #
    #####
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    #
    out = x

```

- np.random.rand를 사용해서 0~1사이의 값을 임의로 끄내는데 x.shape 모양으로 끄내주고 p보다 작은 값들은 1로 아닌 값들은 0으로 변환해준다.
- p로 나눠야 하는 이유는 밑에서 설명하겠다.

1. Inline Question 1

Inline Question 1:

What happens if we do not divide the values being passed through inverse dropout by p in the dropout layer? Why does that happen?

여기서 p 를 나누어 주는 이유는 총 output에서 $(1-p)$ 만큼을 날려서 output의 전체 비율이 p 로 줄어들었기 때문이다. 예시로

$[[1, 1, 1],$

$[1, 1, 1]]$

의 output 크기는 6이다. 이 output에서 0.5를 dropout하면

$[[1, 0, 0],$

$[0, 1, 1]]$

이렇게 되어 output의 크기 3으로 줄어들게 된다. output의 scale을 기존값과 같게 맞춰주기 위해 output에 0.5를 나누어 주면

$[[2, 0, 0],$

$[0, 2, 2]]$

가되어 output의 크기가 6으로 기존 output과 같게 scaled 된다. 그리고 test때는 dropout 하면 안되므로 그대로 보내준다.

- output에서 1-p만큼을 날려 output의 전체 비율이 p 로 줄어들었기 때문이다.
- 따라서 output의 scale을 기존값과 같게 맞춰 주기 위해서 p 로 나누어서 한다.



02_dropout_backward(dout, cache)

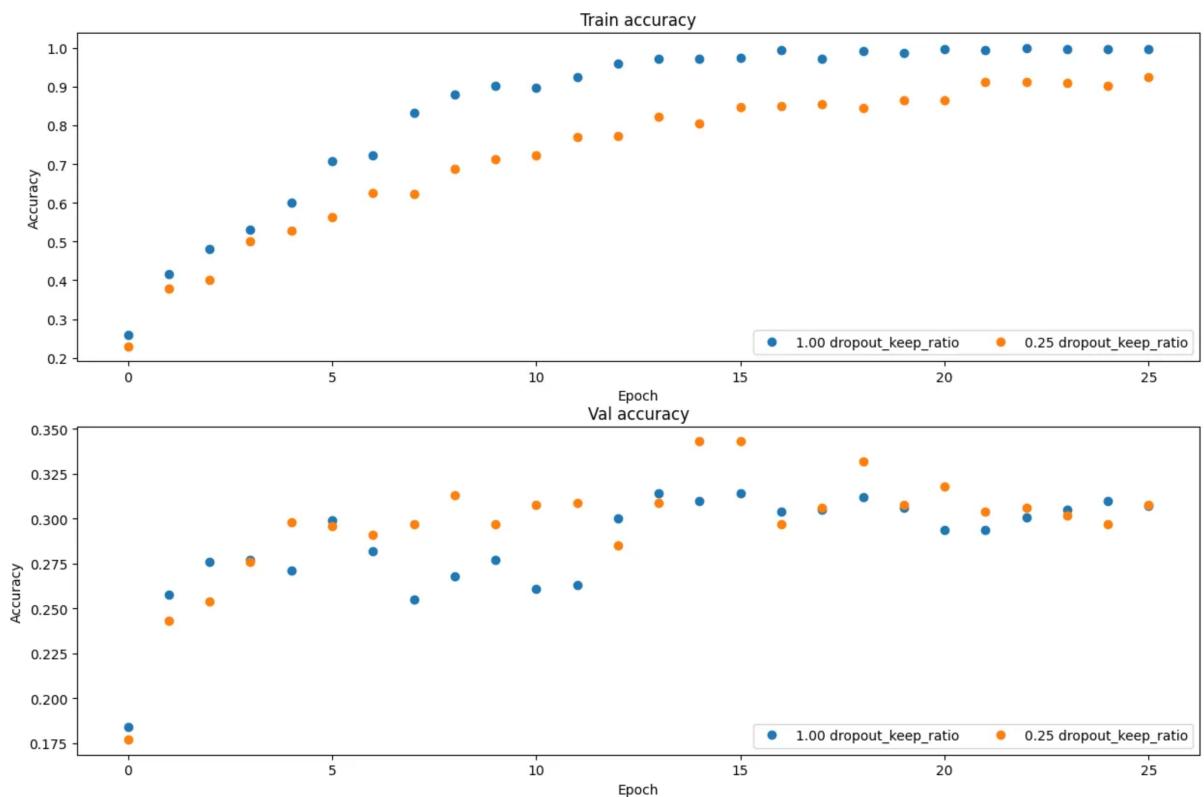
```
dx = None
if mode == "train":
    #####
    # TODO: Implement training phase backward pass for inverted dropout #
    #####
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    dx = dout * mask

    # *****END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    #
    # END OF YOUR CODE
    #####
elif mode == "test":
    dx = dout
return dx
```

- train일 때는 x 로 미분했을 때 mask가 남으로 `dout * mask`로 `dx`를 구한다

- test일 때는 `dx = dout` 으로 반환해준다.

- Layer들의 순서는 `affine -> batch norm -> relu -> dropout` 순서로 진행된다.



2. Inline Question 2

Inline Question 2:

Compare the validation and training accuracies with and without dropout – what do your results suggest about dropout as a regularizer?

- dropout을 사용하지 않았을 때가 train accuracy가 더 높은 것을 확인할 수 있다.
- 반대로 dropout을 사용했을 때가 val accuracy가 더 높다.
- 좋은 regularizer로 잘 작동했다고 볼 수 있다.

