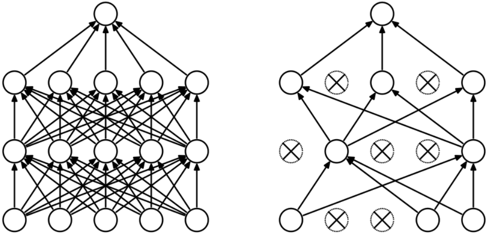
Dropout

Victor Kitov

Dropout idea

Each node in the neural network is removed with probability $1-p$ independently from decisions about other nodes:

Comparison neural net without/with dropout



(a) Standard Neural Net (b) After applying dropout.

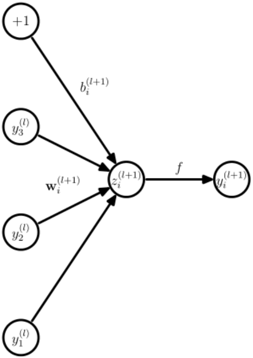
- Output layer nodes are never removed.
- Recommended parameters:
    - $p = 0.5$ for inner layer nodes
    - $p = 0.8$ for input layer nodes (feature subsampling)

Dropout motivation

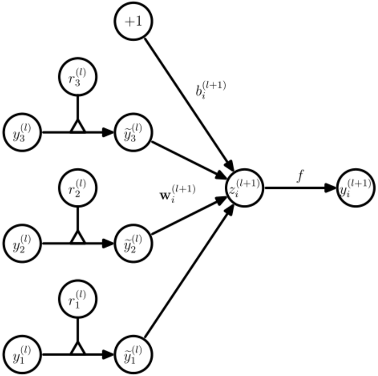
- Motivation from genetic theory of evolution:
  - sexual reproduction involves taking half the genes of one parent and half of the other.
  - best fit genes get mixed with 0.5 probabilities
  - best genes should learn "by themselves", not relying on complex outer gene structure
    - less ovefitting
- In dropout network:
  - nodes rely less on outputs of other nodes
  - try more to learn something by themselves
  - behave in a more robust way
  - resulting network becomes less overfitted.

Dropout algorithm

Comparison of usual and dropout network for one node



(a) Standard network



(b) Dropout network

Definitions

Define:

- $f(x)$ - an activation function.
- y^l - vector of outputs at layer l
- z^l - vector of inputs to layer l
- $a * b$ defines element-wise product of elements.
- L - number of layers in neural network
- $y^{(0)} = x$ - input feature vector
- $Bernoulli(p)$ returns a vector of independent Bernoulli random variables with parameter p.

Forward propagation algorithm

We need to repeat forward propagation recurrently for $l = 0, 1, ...L-1$.

1. Usual feed-forward neural network:

$$
\begin{aligned}
z_i^{(l+1)} &= w_i^{(l+1)} y^l + b_i^{(l+1)} \\
y_i^{(l+1)} &= f(z_i^{(l+1)})
\end{aligned}
$$

2. Feed-forward network with dropout:

$$
\begin{aligned}
r_j^{(l)} &\sim Bernoulli(p) \\
\tilde{y}^l &= r^{(l)} * y^{(l)} \\
z_i^{(l+1)} &= w_i^{(l+1)} \tilde{y}^l + b_i^{(l+1)} \\
y_i^{(l+1)} &= f(z_i^{(l+1)})
\end{aligned}
$$

Application of dropout

- **Learning**
  - while weights not converge:
    1. sample random subnetwork ("thinned network") with dropout
    2. apply one step of stochastic gradient descent to thinned network

  Comment: due to weights sharing across all thinned networks the number of parameters is the same as in original network.
- **Prediction**
  - use full networks with all nodes, but multiply each weight by p.
  - such scaling will yield the same output as average thinned network.

Conclusion

- Dropout behaves similar to generating 2^W networks and taking weighted average of their predictions (W is the number of weights in the original neural network).
- Properties:
  - number of parameters is the same
  - training complexity is reduced
  - complexity of prediction is the same
- Dropout provides accuracy improvement in many domains.
- More details in: *"Dropout: A Simple Way to Prevent Neural Networks from Overfitting". Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. Journal of Machine Learning Research 15 (2014) 1929-1958.*