Section 4 : Regularization and Normalization

- L_1 and L_2 regularization
- Dropout: +) in the training phases, for a specific hidden layer, for each training sample, for each iteration, randomly disable a fraction p_1 of neurons. It force the neural network to learn more robust features that are useful in conjunction with many different random subsets of other neurons.
  +) Dropout doubles the number of iterations required to converge. However, training time for each epoch is less, since some neurons are disabled
  +) At traing time, assume the neural is kept with a probability of p. Then, at test time, dropout is not applied, so we want the output at test time to be identical to their expected output at training time

$$\text{Neural Output (Dropout)} = px + (1-p).0 = px$$

So at test time, we must scale x by px to get the same result

- Another way is using Inverted Dropout; at training time, divide the activations by the keeping probability p.

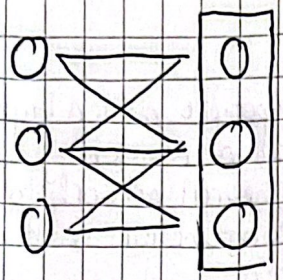
- Drop Connect (Dropweight) : the only difference with drop out is that we drop the weights rather than drop the neuron outputs

- Normalization : transferring all the data points to have the same range

$$\text{Min-max normalization: } x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- Standardization: $x = \frac{x - \text{mean}}{\text{std}} \rightarrow x \sim N(0, 1)$

• Batch Normalization

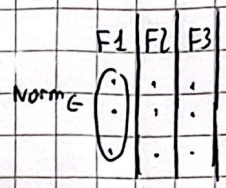


← Normalize them

Input: Values of x over a mini-batch: $B = \{x_1, \dots x_m\}$.

Parameters to be learn: γ, β

Output: $\{y_i = BN_{\gamma,\beta}(x_i)\}$



$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$$

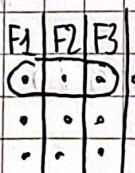
γ, β: learnable parameter.

- This is a layer, it learns the best distribution through γ, β

because $N(0,1)$ performs poorly for some activation functions, ex Sigmoid

- Benefit: + Converges faster and reduced the need for Dropout.

• Layer Normalization: Normalize each feature in a layer according to the features in that layer



• Group Normalization: The input channels are seperated into num-groups groups, each containing num_channels / num-groups channels.

If we put all feature into a group, it equivalent with LayerNorm