PadhAI: Batch Normalization and Dropout

One Fourth Labs

Learning Mu and Sigma

But by normalising the activations are we enforcing some constraints?

1. By forcing the activation function outputs to lie within a particular range, are we imposing some constraints on the model?
2. Batch normalization has a solution for that, in the form of the γ and β terms.
3. Consider the matrix H:

$$H = \begin{pmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1d} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{m1} & h_{m2} & \cdots & \cdots & h_{md} \end{pmatrix}$$

   a. $h_{ij}^{(norm)} = \frac{h_{ij} - \mu_j}{\sigma_j}$ (1)
   b. $h_{ij}^{(final)} = \gamma_j \,.\, h_{ij}^{(norm)} + \beta_j$ (2)
   c. **γ and β are learned parameters** for each column of H. They are learned just like the weights and biases, using an update rule like SGD, Adam, NAG etc.
   d. Introduction of learned parameters **γ and β** ensures we are not locked to a **$\mu = 0$ and $\sigma = 1$**
      i. Let's see what this means
      ii. Suppose the network learns the values **$\gamma_j = \sigma_j$** and **$\beta_j = \mu_j$**
      iii. So, equation (2) can be rewritten: $h_{ij}^{(final)} = \sigma_j \,.\, h_{ij}^{(norm)} + \mu_j$
      iv. Rearranging equation (1): $h_{ij} = \sigma_j . h_{ij}^{(norm)} + \mu_j$
      v. From the above two equations, we can see that if there is a particular set of values of **σ** and **μ** that cause the loss to decrease, then the network will learn **$\gamma_j = \sigma_j$** and **$\beta_j = \mu_j$** so that $h_{ij}^{(final)} = h_{ij}$
      vi. This allows us to the network to opt out of normalization and use the original h_{ij} value in cases where normalization does not decrease the loss.

4. Another use of Batch Normalization is that it acts as a form of regularization
   a. Here, **μ and σ** are computed from a mini-batch of size k, thus they are very likely to be noisy (as they are not calculated using the entire dataset).
   b. Introducing noise to the data leads to better regularization.
   c. Hence, the network is less likely to overfit the training data, thereby making the network more robust.