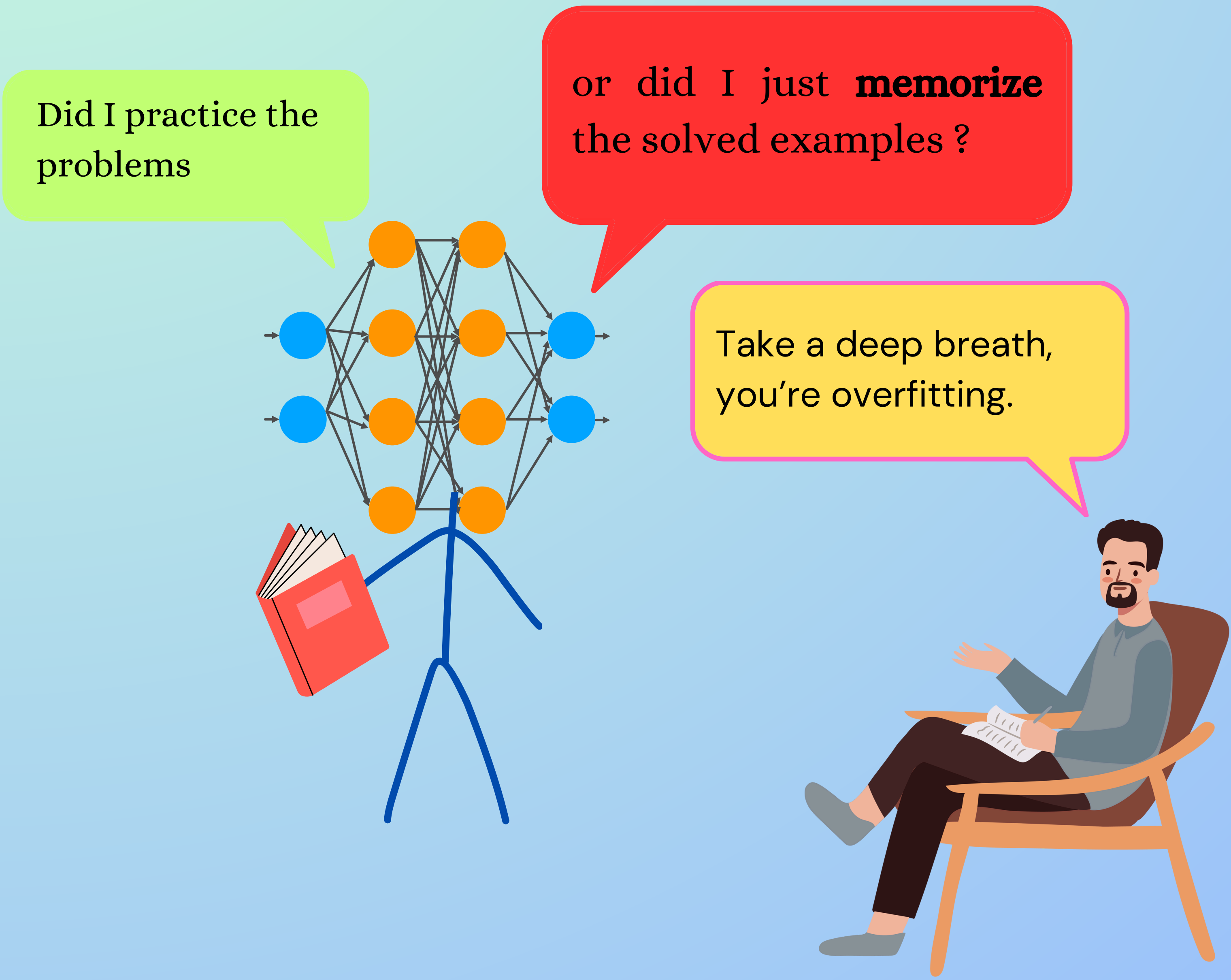
FUNDAMENTALS OF DEEP LEARNING

Regularization for Neural Networks



Shivang Kainthola

Regularization in Neural Networks

⟶ When a neural network or machine learning model performs too well on the training data but fails to generalize to the testing data, the problem is called overfitting.

⟶ Overfitting is a common issue while training deep neural networks or any machine learning models.

Overfitting in a neural network also manifests itself in or with other problems like :

1) Internal Covariate Shift
2) Co-Adaptation
3) Large Weights

⟶ To counter overfitting in neural networks, we use some regularization techniques :

1) Batch Normalization
2) Dropout
3) L1 Regularization
4) L2 Regularization

1) BATCH NORMALIZATION

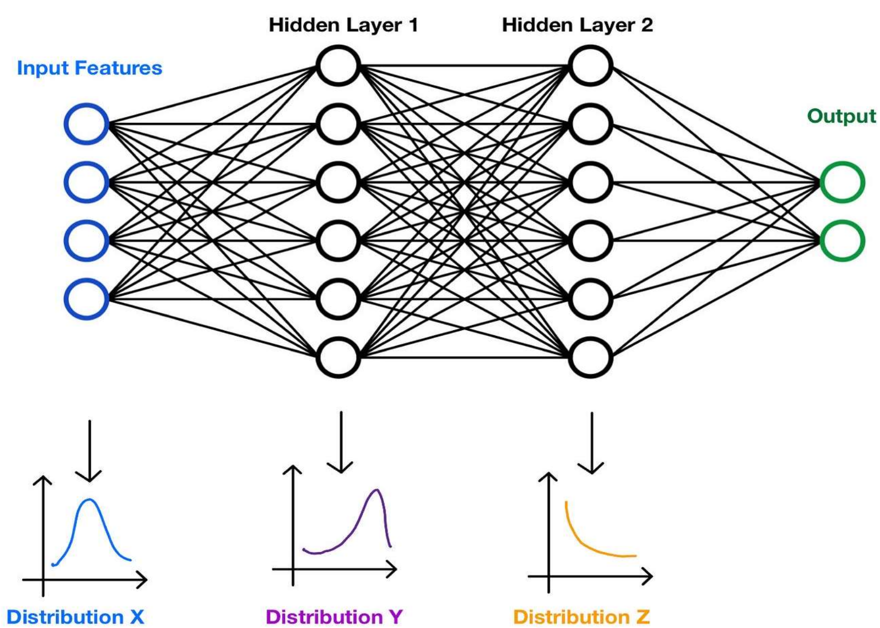
Problem : Internal Covariate Shift

During the training of a neural network by backpropagation, the parameters (weights and biases) are updated based on the error calculated by the loss function.

⟶ The activations of each layer depend on the inputs and parameters, both of which are changing during training.

⟶ Since the parameters as well as the inputs to each layer are constantly being updated, so there is a shift in the distribution of inputs to each layer which is called internal covariate shift.

Internal covariate shift can slow down training and lead to instability.



https://kwokanthony.medium.com/batch-normalization-in-neural-network-simply-explained-115fe281f4cd

Solution : Batch Normalization

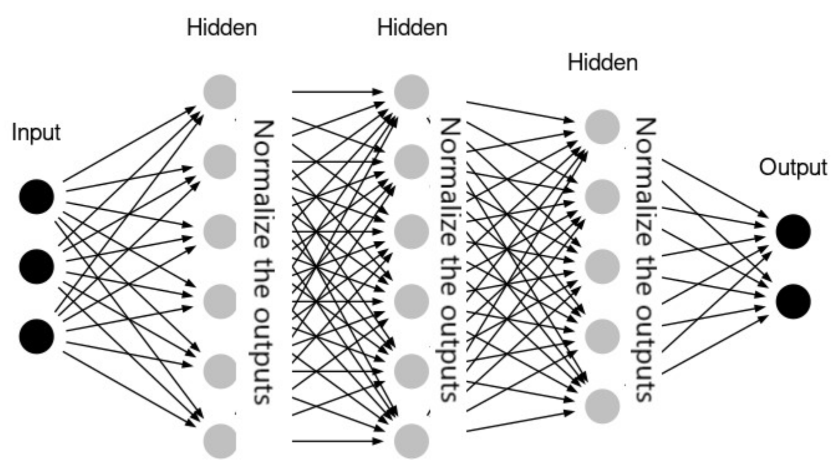
⟶ A batch or mini-batch is a collection of samples that will be passed through the network at one time for the weights update.

⟶ Batch Normalization is a regularization technique - where we normalize the inputs to a layer for every mini-batch.

⟶ Normalizing the data involves transforming it to have mean = 0 and standard deviation = 1.

⟶ Besides tackling internal covariate shift, it makes the gradient descent better.

https://medium.com/@abheerchrome/batch-normalization-explained-1e78f7eb1e8a



⟶ In a convolutional neural network, Batch Normalization is carried out with a Normalization Layer placed after the convolution layer.

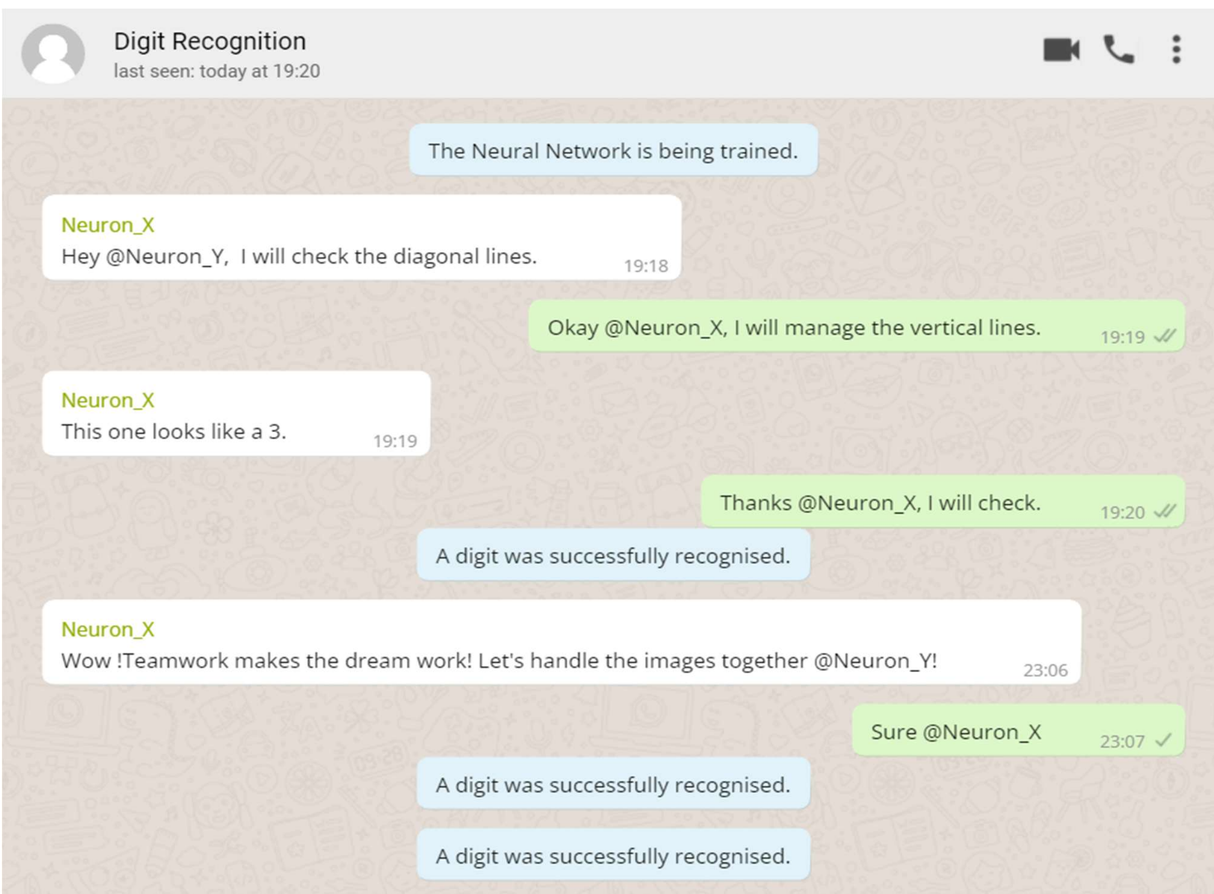
2) Dropout

Problem : Co-adaptation relationships

⟶ Co-adaptation is a situation when neurons become excessively reliant on one another.

⟶ To work properly, the neurons rely on the input of other co-adapted neurons.

⟶ With this co-adaptation relationship, the neurons also tend to be more generalized to training data, may underperform on testing data.



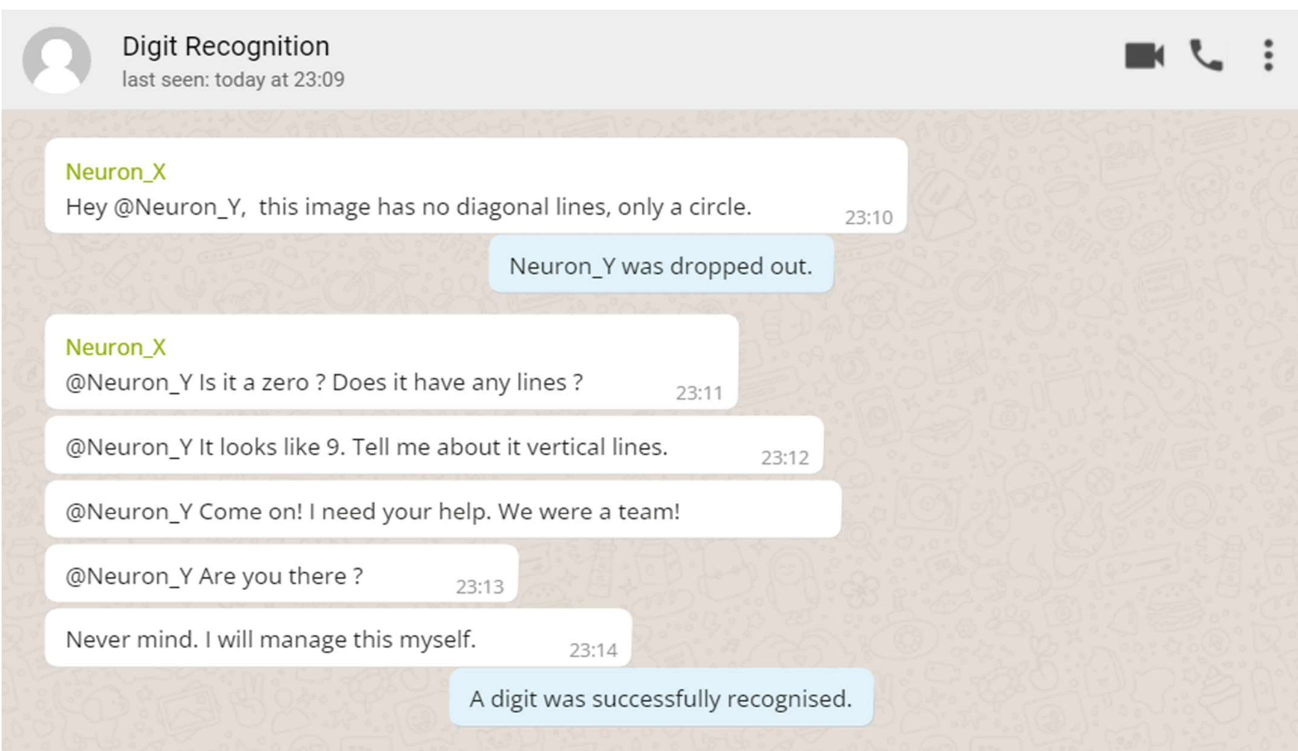
The co-adapted neurons X and Y can become dependent on each other.

Solution : Dropout

⟶ The 'dropout' method works **by randomly setting a fraction of the input units (neurons) in a layer to zero**, i.e. dropping them out, during each training iteration.

⟶ This dropped out fraction of neurons does not take part in forward pass (activation and gradient computation) or backpropagation (gradient updates) during training.

⟶ With dropout, the neurons are denied the convenience of making co-adaptations and relying on other neurons, since they must learn more robust features on their own.



⟶ Dropout method is applied to a neural network as a Dropout( ) layer, which takes the fraction of neurons to be dropped out as input.

3) L2 Regularization

Problem : Large weights

⟶ L2 Regularization is a popular regularization technique which penalizes models with large parameter (weight) values.

⟶ It works by **adding L2 regularization term to the loss function**, and when the loss function is minimized (by gradient descent - which seeks the global minima), it steers the network away from having large weights.

⟶ The loss function L of a neural network with L2 regularization term will be :

||w||^2 represents the squared L2 norm of the weights (sum of squares of all elements in the weight vector)

$$\mathbf{L}_{\text{L2 REGULARIZED}}(y, \hat{y}) = \mathbf{L}(y, \hat{y}) + \frac{1}{2}\lambda \, \|w\|^2$$

⟶ It is controlled by the parameter lambda λ, and regularization penalties are applied on a per layer basis.

4) L1 Regularization

⟶ L1 regularization penalizes large weights by adding the L1 regularization term to the loss function, similar in working to L2 regularization.

⟶ The loss function L of a neural network with L1 regularization term will be :

||w||_1 represents the L1 norm of the weights, which is the sum of the absolute values of all elements in the weight vector (w)

$$\mathbf{L}_{\text{L1 REGULARIZED}}(y, \hat{y}) = \mathbf{L}(y, \hat{y}) + \lambda \, \|w\|$$

⟶ It can be combined with L2 regularization, often known as Elastic Net regularization.

Thank You !

If you found this article helpful, please do leave a like and comment any feedback you feel I could use.

While I'm working on the next project, I'll be over at :

www.linkedin.com/in/shivang-kainthola-2835151b9/

shivang.kainthola64@gmail.com

https://shivangkainthola28.medium.com/

https://github.com/HeadHunter28