









Neural networks are a class of models that are built with layers. Commonly used types of neural networks include convolutional and recurrent neural networks.

1 DATA PROCESSING

1.1 Data Augmentation

Deep learning models usually need a lot of data to be properly trained. It is often useful to get more data from the existing ones using data augmentation techniques. The main ones are summed up in the table below. More precisely, given the following input image, here are the techniques that we can apply:

Original	Flip	Rotation	Random crop
			
- Image without any modification	- Flipped with respect to an axis for which the meaning of the image is preserved	- Rotation with a slight angle - Simulates incorrect horizon calibration	- Random focus on one part of the image - Several random crops can be done in a row
Color shift	Noise addition	Information loss	Contrast change
			
- Nuances of RGB is slightly changed - Captures noise that can occur with light exposure	- Addition of noise - More tolerance to quality variation of inputs	- Parts of image ignored - Mimics potential loss of parts of image	- Luminosity changes - Controls difference in exposition due to time of day

1.2 Batch Normalization

This step involves the use of hyperparameters γ and β to normalize the batch $\{x_i\}$. By considering μ_B and σ_B^2 as the desired mean and variance for the batch correction, the procedure is executed as follows:

$$x_i \leftarrow \frac{\gamma}{\sqrt{\sigma_B^2 + \epsilon}}(x_i - \mu_B) + \beta$$

This normalization step is typically applied after a fully connected or convolutional layer and before a non-linearity layer. Its objective is to facilitate the utilization of higher learning rates and decrease the heavy reliance on initializations.

2 TRAINING A NEURAL NETWORK

2.1 Epoch

An epoch in machine learning refers to a complete iteration through the entire training dataset. During each epoch, the model's param-

eters are updated based on the calculated gradients from the entire training set.

2.2 Minibatch Gradient Descent

Minibatch Gradient Descent is an optimization algorithm used for training machine learning models. It involves dividing the training dataset into smaller subsets called minibatches. The model's parameters are updated based on the average gradient computed from the gradients of the loss function with respect to the parameters calculated over the minibatch.

2.3 Loss Function

In order to quantify how a given model performs, the loss function L is usually used to evaluate to what extent the actual outputs y are correctly predicted by the model outputs z .

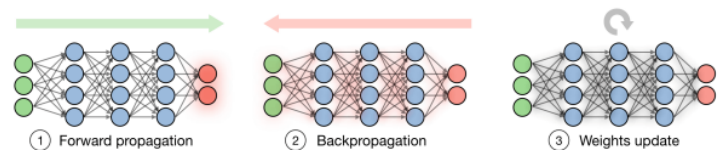
2.4 Cross Entropy Loss

In the context of binary classification in neural networks, the cross entropy loss $L(z, y)$ is commonly used and is defined as follows:

$$L(z, y) = -(y \log(z) + (1 - y) \log(1 - z))$$

2.5 Back Propagation

Back Propagation is a process used to calculate the gradients of the loss function with respect to the model's parameters. It involves propagating the error backward through the network, layer by layer, and computing the gradients using the chain rule.



2.6 Updating Weights

In the context of training neural networks, updating weights refers to adjusting the weights of the model's connections based on the calculated gradients. This adjustment aims to minimize the loss function and improve the model's performance during training. In a neural network, weights are updated as follows:

1. Take a batch of training data and perform forward propagation to compute the loss.
2. Backpropagate the loss to get the gradient of the loss with respect to each weight.
3. Use the gradients to update the weights of the network.




3 PARAMETER TUNING

3.1 Xavier initialization

Instead of initializing the weights in a purely random manner, Xavier initialization enables to have initial weights that take into account characteristics that are unique to the architecture.

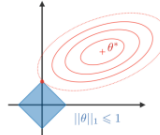
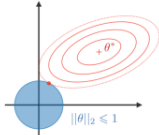
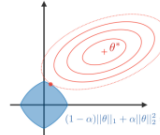
3.2 Transfer learning

Training a deep learning model requires a lot of data and more importantly a lot of time. It is often useful to take advantage of pre-trained weights on huge datasets that took days/weeks to train, and leverage it towards our use case. Depending on how much data we have at hand, here are the different ways to leverage this:

Training size	Illustration	Explanation
Small		Freezes all layers, trains weights on softmax
Medium		Freezes most layers, trains weights on last layers and softmax
Large		Trains weights on layers and softmax by initializing weights on pre-trained ones

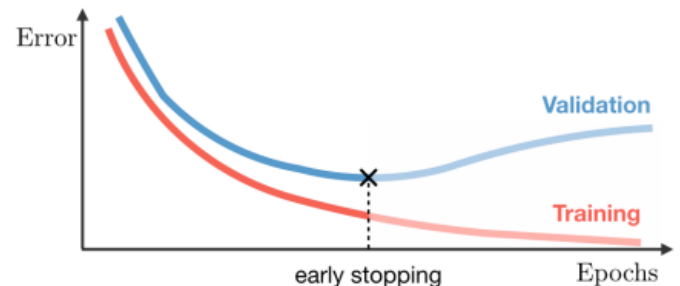
3.3 Learning rate

The learning rate, often noted α or sometimes η , indicates at which pace the weights get updated. It can be fixed or adaptively changed. The current most popular method is called Adam, which is a method that adapts the learning rate.

LASSO	Ridge	Elastic Net
- Shrinks coefficients to 0 - Good for variable selection	Makes coefficients smaller	Tradeoff between variable selection and small coefficients
		
$\dots + \lambda \theta _1$ $\lambda \in \mathbb{R}$	$\dots + \lambda \theta _2^2$ $\lambda \in \mathbb{R}$	$\dots + \lambda \left[(1 - \alpha) \theta _1 + \alpha \theta _2^2 \right]$ $\lambda \in \mathbb{R}, \alpha \in [0, 1]$

4.3 Early stopping

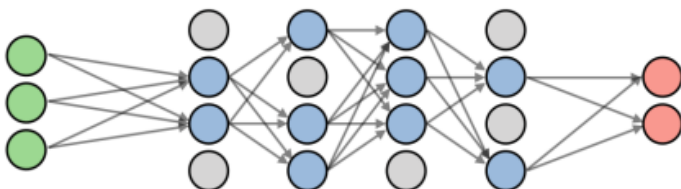
This regularization technique stops the training process as soon as the validation loss reaches a plateau or starts to increase.



4 REGULARIZATION

4.1 Dropout

Dropout is a technique used in neural networks to prevent overfitting the training data by dropping out neurons with probability $p > 0$. It forces the model to avoid relying too much on particular sets of features



4.2 Weight regularization

In order to make sure that the weights are not too large and that the model is not overfitting the training set, regularization techniques are usually performed on the model weights. The main ones are summed up in the table below: