Deep Learning

Abhijit Amrendra Kumar

August 2023

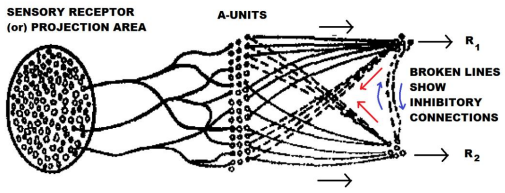
Chapter 1

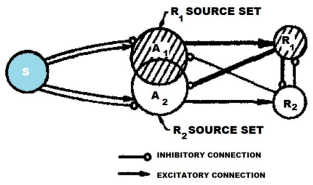
Perceptrons

1.1 Introduction

1.1.1 What is a Perceptron ?

Perceptron is a probabilistic model for information storage and organization in the brain.





Mathematically, a perceptron is a function that takes input signals and produces an output signal based on the weighted sum of the inputs. These weights are adjusted during learning to enable the perceptron to make accurate classifications or decisions.

The learning rule is inspired by Hebbian learning, a biological learning principle that suggests that synaptic connections between neurons strengthen when those neurons are activated together. In the context of the perceptron, the learning rule updates the weights to minimize the error between the perceptron's output and the desired target output.

Frank Rosenblatt demonstrated that the perceptron can successfully learn linearly separable patterns and achieve convergence within a finite number of iterations. Therefore, the perceptron model can be used for binary classification tasks, where it learns to separate input data points into two classes by finding an optimal decision boundary. Although this model is not used in practice nowadays, it serves as a building block for the modern neural network architectures.

However, Rosenblatt also highlighted the limitations of the perceptron model. He showed that the perceptron is limited to learning linearly separable patterns and cannot handle more complex patterns that require nonlinear decision boundaries. This limitation, known as the **perceptron convergence theorem**, sparked subsequent research into developing more advanced neural network architectures capable of handling nonlinear problems.

1.1.2 Key Assumptions

- Stimuli which are similar will tend to form pathways to same sets of response cells
- Stimuli which are dissimilar will tend to form pathways to different sets of response cells
- Application of positive and negative reinforcements may facilitate or hinder the formation of connections.
- Similarity of stimuli is a dynamically evolving attribute.

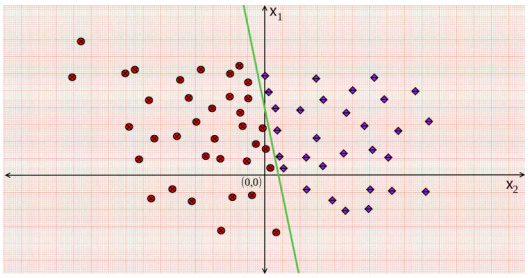
1.2 Perceptron Training

1.2.1 Definitions

- **Margin of a hyperplane**: The distance between the hyperplane and the data point nearest to it.
- **Margin of a dataset** γ: The maximum margin possible for that dataset using any weight vector w.

1.2.2 Geometric Intuition

Given a set of points on a n-dimensional plane, each having one out of two labels $\{+1, -1\}$.
For visualization, we can use the following figure.



We want to use the perceptron model to determine a hyperplane that can separate the points based on their labels, i.e. $\{+1, -1\}$.

1.2.3 Assumptions

Based on the above problem, for such a hyperplane to exist, the dataset must satisfy some assumptions.

- There must exist atleast 1 hyperplane which separates the points in the dataset based on their labels.
  - An example of a 2D dataset which doesn't satisfy the above condition is given below.

- The margin of the dataset $\gamma > 0$.
  - Empirically, γ should be large enough for a adequately fast training speed.

1.2.4 The Model

Consider the training dataset $D = \{(x^i, y^i)\}_{i=1}^n$. To solve the above problem, our aim is to train a function/model $h : X \to Y$. For testing, given $\hat{x}$, the model will be able to predict $\hat{y} = h(\hat{x})$.

Considering the perceptron model, the input is of the form $x = (x_1, x_2, x_3, ...x_d) \in \mathbb{R}^d$, the associated weights are of the form $w = (w_1, w_2, w_3, ...w_d) \in \mathbb{R}^d$, and the output is of the form $y \in \{+1, -1\}$.

The prediction rule is

$$h(x) = \begin{cases} +1 & \text{if } w^T x \geq \theta \\ -1 & \text{if } w^T x \leq \theta \end{cases}$$

For notational purposes, we can define the following

$$\tilde{x} = (1, x) = (1, x_1, x_2, ...x_d) \in \mathbb{R}^{d+1}$$
$$\tilde{w} = (1, w) = (-\theta, w_1, w_2, ...w_d) \in \mathbb{R}^{d+1}$$

This implies the following modified prediction rule

$$H(\tilde{x}) = \begin{cases} +1 & \text{if } \tilde{w}^T\tilde{x} \geq 0 \\ -1 & \text{if } \tilde{w}^T\tilde{x} \leq 0 \end{cases}$$

1.2.5 The Algorithms

Algorithm 1 Algorithm to train a perceptron with a large dataset

$\tilde{w}^1 \leftarrow 0$ ▷ Initialize weights
for $t \leftarrow 1, 2, 3, ...N$ do
  receive $(x^t, y^t), x^t \in \mathbb{R}^d, y^t \in \{+1, -1\}$
  transform x^t into $\tilde{x}^t = (1, x^t) \in \mathbb{R}^{d+1}$
  $\hat{y} = \text{perceptron}(\tilde{x}^t, \tilde{w}^t) = sign(\langle \tilde{x}^t, \tilde{w}^t \rangle)$
  if $\hat{y} \neq y^t$ then
    $\tilde{w}^{t+1} = \tilde{w}^t + y^t\tilde{x}^t$
  else
    $\tilde{w}^{t+1} = \tilde{w}^t$

Algorithm 2 Algorithm to train a perceptron with a small dataset

$\tilde{w}^1 \leftarrow 0$ ▷ Initialize weights
for $t \leftarrow 1, 2, 3, ...$ do
  for $j \leftarrow 1, 2, 3, ...N$ do
    receive $(x^j, y^j), x^j \in \mathbb{R}^d, y^j \in \{+1, -1\}$
    transform x^j into $\tilde{x}^j = (1, x^j) \in \mathbb{R}^{d+1}$
    $\hat{y} = \text{perceptron}(\tilde{x}^j, \tilde{w}^j) = sign(\langle \tilde{x}^j, \tilde{w}^j \rangle)$
    if $\hat{y} \neq y^j$ then
      $\tilde{w}^{j+1} = \tilde{w}^j + y^j\tilde{x}^j$
    else
      $\tilde{w}^{j+1} = \tilde{w}^j$

1.2.6 The Procedure

Let's try to mathematically formulate the **linear separability assumption**.

Linear Separability Assumption

Let $D = \{(x^t, y^t)\}_{t=1}^{\infty}$ denote the training data where $x^t \in \mathbb{R}^d, y^t \in \{+1, -1\}\ \forall t = 1, 2, ...$
Let $\gamma > 0$ denote the **margin of the dataset**. Then $\exists\ w^* \in \mathbb{R}^d, w^* \neq 0$, such that

$$\langle w^*, x^t \rangle > \gamma, \text{ where } \gamma^t = 1 \tag{1.1}$$
$$\langle w^*, x^t \rangle < -\gamma, \text{ where } \gamma^t = -1 \tag{1.2}$$
$$(1.1), (1.2) \implies y^t\langle w^*, x^t \rangle > \gamma \tag{1.3}$$

Claim: If the linear separability assumption holds, the perceptron makes a finite number of mistakes during the run of the algorithm.

Proof: Consider a round t where the perceptron made a mistake. By definition, w^* depends on the dataset, so it is essentially constant in our analysis. The problem lies in figuring out the value of w^*. However, let's consider the difference in the dot products $\langle w^*, w^{t+1} \rangle$ and $\langle w^*, w^t \rangle$.

$$\langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle = \langle w^*, w^t + y^t x^t \rangle - \langle w^*, w^t \rangle \tag{1.4}$$
$$= \langle w^*, y^t x^t \rangle \tag{1.5}$$
$$= y^t \langle w^*, x^t \rangle \tag{1.6}$$
$$\tag{1.7}$$

Using the **linear separability assumption**,

$$(1.3) \implies y^t \langle w^*, x^t \rangle > \gamma \tag{1.8}$$

$$\implies \langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle > \gamma \tag{1.9}$$

Now, let's try doing a summation over all the dataset points.

$$\sum_{i=1}^{N} \langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle$$

For the rounds having mistakes, (1.9) holds, while for rounds having no mistakes,

$$w^{t+1} = w^t \implies \langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle = 0$$

Let's say the total number of rounds with mistakes is M. Hence, our summation changes into

$$\sum_{i=1}^{N} \langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle = \sum_{\text{mistakes}} \langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle \tag{1.10}$$

$$\implies \sum_{i=1}^{N} \langle w^*, w^{t+1} \rangle - \langle w^*, w^t \rangle > M\gamma \tag{1.11}$$

$$\implies \langle w^*, w^{N+1} \rangle - \langle w^*, w^1 \rangle > M\gamma \tag{1.12}$$

$$w^1 = 0 \implies \langle w^*, w^{N+1} \rangle > M\gamma \tag{1.13}$$

$$\implies M\gamma < \langle w^*, w^{N+1} \rangle \leq ||w^*||.||w^{N+1}|| \tag{1.14}$$

For the rounds having mistakes, let's consider $||w^{t+1}||^2$.

$$||w^{t+1}||^2 = ||w^t + y^t x^t||^2 \tag{1.15}$$

$$= (w^t + y^t x^t)^T (w^t + y^t x^t) \tag{1.16}$$

$$= ||w^t||^2 + ||y^t||^2.||x^t||^2 + 2y^t \langle w^t, x^t \rangle \tag{1.17}$$

We know that the last term is the product of y^t and the prediction value $\hat{y}$. Since this round has the prediction mistake, we can conclude that the last term is **negative**. Also, we know that $||y^t||^2 = 1$.

$$||w^{t+1}||^2 - ||w^t||^2 = ||y^t||^2.||x^t||^2 + 2y^t \langle w^t, x^t \rangle \tag{1.18}$$

$$< ||x^t||^2 \tag{1.19}$$

Performing a telescopic summation over all rounds with mistakes, and using $w^1 = 0$ yields

$$||w^{N+1}||^2 < M||x^t||^2 \tag{1.20}$$

Realistically, all the points in the dataset can be bounded within a d-dimensional ball, whose radius R is the max L2-norm among all the feature vectors $x^i, i = 1, 2, 3, ...N$.

$$||w^{N+1}||^2 < M||x^t||^2 < MR^2 \tag{1.21}$$

$$\implies ||w^{N+1}|| < R\sqrt{M} \tag{1.22}$$

Using (1.14), (1.22), we obtain

$$M\gamma < ||w^*||.||w^{N+1}|| < R\sqrt{M}||w^*|| \tag{1.23}$$

$$\implies M < ||w^*||^2 \cdot R^2/\gamma^2 \tag{1.24}$$

To summarize, the following inequality, also known as the **Perceptron Mistake Bound**, implies an upper bound on the number of mistakes M occuring during the training of a perceptron.

$$\boxed{M < ||w^*||^2 \cdot R^2/\gamma^2}$$

1.2.7 Moving away from Perceptrons — Dealing with the XOR problem

- Multi-layer perceptron
- Replacing the $sign(\langle w, x \rangle)$ with other activation functions, like the ReLU function or the sigmoid function.
- n_k^l denotes the kth neuron at the lth layer.
- $w_{i,j}^l$ denotes the weight connecting the neuron in the $(l-1)th$ layer to the ith neuron in the lth layer.
- z_k^l denotes the the inner product computed at n_k^l.
- a_k^l denotes the activation of n_k^l.

$$z^1 = \begin{bmatrix} z_1^1 \\ z_2^1 \end{bmatrix} = \begin{bmatrix} w_{11}^1 & w_{12}^1 \\ w_{21}^1 & w_{22}^1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{1.25}$$

Loss function: $l : Y \times Y \to [0, \infty)$
Some commonly used loss functions are

$$l(y, \hat{y}) := (y - \hat{y})^2 \tag{1.26}$$

$$l(y, \hat{y}) := |y - \hat{y}| \tag{1.27}$$

$$\tag{1.28}$$

Total loss: $L = \sum_{i=1}^{N} \text{loss}(\hat{y}^i, y^i)$
We want to minimise the total loss.