Notes for computational core course Perceptron theory lecture

Shaul Druckmann

Note: these notes are just an adapted version of the presentation found in Hertz et alii, "Introduction to the Theory of Neural Computation"

1 Perceptron Learning

1.1 Definitions

The perceptron learning problem is defined by a set of tuples: patterns and the output that we *require* the perceptron to achieve for each pattern $\{(S^1, \sigma^1), (S^2, \sigma^2), ..., (S^P, \sigma^P)\}$. The end result of successful perceptron learning is a *vector* of weights $\vec{J}$ that defines a perceptron that has the correct (required) output for each pattern in the set. We will primarily consider here a *binary* perceptron:

$$y(\vec{S}) = sign(\sum_j J_j S_j) = sign(J \cdot S) \tag{1}$$

Before discussing the nature of perceptron learning it is useful to consider a geometrical interpretation of the input-output transformation that a perceptron embodies. Examining equation (1) we see that the output of the perceptron is equal to the sign of the dot product of the vectors J and S. The geometric interpretation of that is that the output of the neuron is positive if the vector S lies on the positive side of the hyperplane defined by J and the output negative if the vector S lies on the other side of the hyperplane defined by J.

For instance let us consider an $N = 2$ dimensional binary input pattern. There are four possible configurations: $\{(+1+1), (+1-1), (-1+1), (-1-1)\}$. We can require for each of these configurations a certain output, either $+1$ or -1. By choosing different outputs for the four possible configurations we can embody different input-output transformations such as AND or XOR, see Figure 1. Let us consider the left part of Figure 1, the AND function. We need to output $+1$ for the pattern $(+1+1)$ and -1 for the rest. Thus, if we choose J to be the vector marked in red on the figure, we see that indeed the points that are required to have positive output lie on the positive side of the hyperplane (in two dimensions it is just a line) defined by J and that the points required to have negative output lie on the other side (the ambiguity of a point lying exactly on the line is resolved by adding a very small threshold to the perceptron causing an input of 0 to result in a negative output). The vector J sets the input to the correct values and we say that it solves the learning problem. Thus, the geometric interpretation of the learning problem is finding a vector J that defines a hyperplane so that all positive output points lie on one side and all negative points on the other. Note that such a J does not always exist. For instance one cannot find a vector J that correctly assigns the XOR outputs, Figure 1 right hand side.

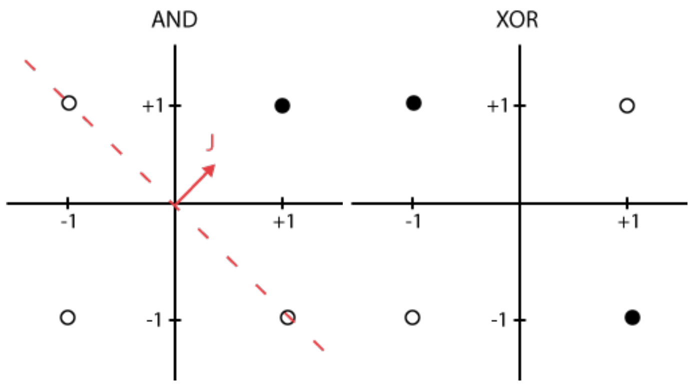

Figure 1: And and Xor - Two different transformations for N=2 perceptron

1.2 Perceptron learning rule

We are now ready to define the perceptron learning algorithm. We will discuss the *batch* learning algorithm. Batch algorithms refer to algorithms that learn by repeated presentation of a set of examples (in contrast to *online* algorithms in which each example is shown only once). The learning rule is defined as follows:

$$J_j^{new} = J_j^{old} + \Delta J_j$$

$$\Delta J_j = \begin{cases} 2\eta \xi_j^\mu \sigma^\mu & \text{if } y^\mu \neq \sigma^\mu \\ 0 & \text{if } y^\mu = \sigma^\mu \end{cases} \tag{2}$$

Before discussing the utility of this learning rule we will again consider a geometric interpretation. Recall that we showed that our problem can be interpreted as finding a hyper-plane that divides the inputs so that all those with a positive sign will be on one side and those with a negative sign on the other. It behooves us to make one more change of variable: multiplying every input ξ_j^μ by the *required* sign of the output. If the required sign of the output is positive we will not change anything, if the sign is negative we will multiply the pattern by

-1, or, geometrically speaking, reflect it relative to the origin. Following this transformation, the patterns in the space of the new variables x_j^μ are required to lie on the same side of the hyper-plane. See Figure 2.

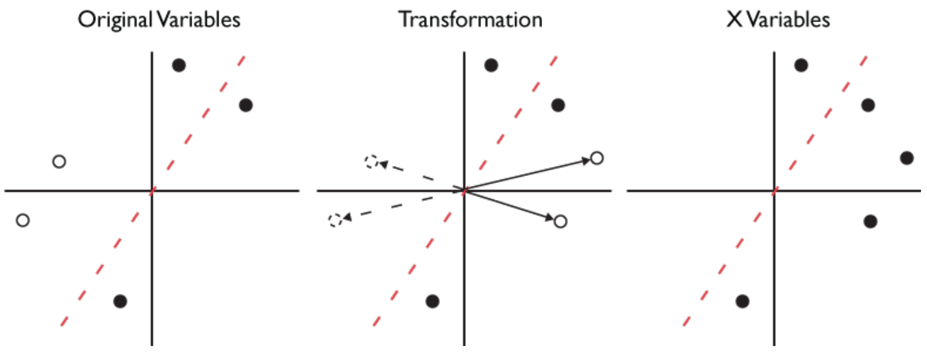

Figure 2: Transformations to X variables

There is also a useful geometric interpretation of our learning rule. If the pattern obeys the inequality, we do not change the weight at all. If it does not, we add to the weight vector the vector $\eta\vec{x}^\mu$, thereby turning the vector in the direction of $\vec{x}^\mu$ (which explains why this point is now more likely to be on the positive side of the hyperplane). The strength of the change is scaled by the size of η, our learning parameter. See Figure 3.

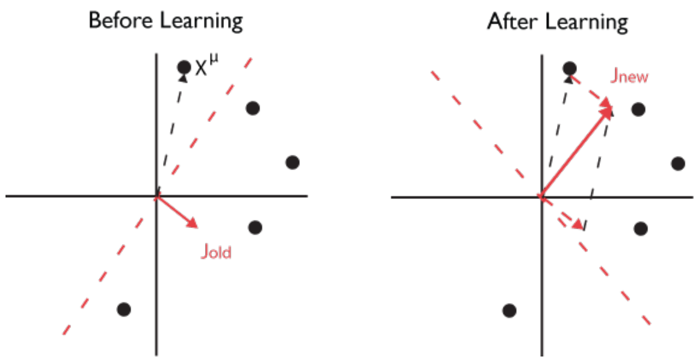

Figure 3: Weight change

1.3 Proving Convergence of Batch Perceptron Learning

We will know prove the convergence of the perceptron learning rule defined above for batch learning (i.e. repeated presentations of the set of patterns). Specifically, we will show that if the problem has a solution, that is a hyperplane that correctly separates the points *exists* (this is sometimes referred to in the literature as the problem being linearly separable), the learning procedure will converge in a finite number of steps to a weight vector that correctly assigns the required outputs. We will begin by defining a measure of the difficulty of the problem, D, (see Figure 4) defined in the following fashion:

$$D(J) = \frac{1}{|J|} min_{\mu} \left[\vec{J} \cdot \vec{x}^{\mu} \right] \tag{3}$$

A useful result of this definition is that if one can find at least one vector J^* for which $D(J^*) > 0$ then the problem is linearly separable.

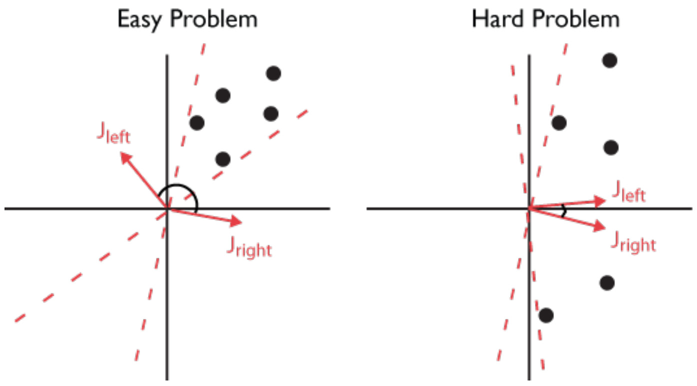

Figure 4: Problem difficulty

We are now ready to begin proving the perceptron learning conversion theorem. Let us assume that the problem is linearly separable, that is $\exists J^*$ so that $D(J^*) > 0$. Let us define the size M^{μ} as the number of times the pattern μ has been used to update the weights. Assuming that the size of the initial (before learning) weight vector is zero ($|J_0| = 0$) we obtain:

$$J = J_0 + \sum \Delta J = 0 + \eta \sum_{\mu} M^{\mu} x^{\mu} \tag{4}$$

Were the last transition is simply the explicit term for the sum of all the updates to the vector, see equation (2). If the learning rule has converged then by definition there are now more updates to the weight vector. Thus, if we define $M = \sum M^{\mu}$ then if M is finite the learning has converged. Our proof will proceed by showing that M must indeed be finite.

The main idea of our proof will be to show that the expression $\frac{J \cdot J^*}{|J|}$ on one hand increases with M but on the other hand must be finite as it is a (partially) normalized dot product between two finite size vectors. Thus, M must be finite.

Let us begin:

$$J \cdot J^* = (\eta \sum_\mu M^\mu x^\mu) \cdot J^* \geq \eta M min(x^\mu \cdot J^*) = \eta M |J^*| \frac{1}{|J^*|} min(x^\mu \cdot J^*)$$
$$= \eta M |J^*| D(J^*) \tag{5}$$

We find that $J \cdot J^*$ increases proportionally to M. Let us now consider $|J|$. We shall assume that at a given time the pattern x^α failed to produce the right output and was thus used to update the weight vector. The change in the length of J due to this update is:

$$\Delta |J|^2 = (J + \eta x^\alpha)^2 - J^2 = J^2 + 2J\eta x^\alpha + \eta^2 (x^\alpha)^2 - J^2 = \eta^2 (x^\alpha)^2 + 2J\eta x^\alpha$$
$$= \eta^2 |x^\alpha| + 2J\eta x^\alpha = \eta^2 N + 2J\eta x^\alpha \tag{6}$$

Were in the first transition on the second line we plugged in the fact that x^{alpha} is an N dimensional vector of ± 1 value and thus the square of its size is N. Since the pattern x^{alpha} failed to produce the right output we know that it failed to match the inequality we request $J \cdot x^\mu > N\kappa$. Plugging this in:

$$\Delta |J|^2 = \eta^2 N + 2J\eta x^\alpha \leq \eta^2 N + 0 = N\eta^2$$
$$|J|^2 = M \Delta |J|^2 \leq MN\eta^2 \tag{7}$$

Note that the first transition on the second line was justified as following our calculations we eliminate the dependency on the specific pattern x^α that caused the update. We find that $|J|^2$ increases proportionally to at most $\sqrt{M}$. Thus:

$$\frac{J \cdot J^*}{|J|} \propto \frac{M}{\sqrt{M}} = \sqrt{M} \tag{8}$$

As we have previously mentioned, the expression in the l.h.s must be finite. Since we find it increases proportionally to $\sqrt{M}$ then $\sqrt{M}$ must be finite and thus of course also M, thereby proving our theorem.