NEURAL POLYHEDRA

(Beta version 1.0)

Daniel Crespin
Facultad de Ciencias
Universidad Central de Venezuela

Abstract

This paper presents a method to construct a three layer perceptron neural network equal to the product of the characteristic functions of given polyhedra. The method is extended to generalized polyhedra defined by polynomial and other inequalities.

1.- **Introduction.** A set is *neural* if its characteristic function is given by a neural network. It is known that the characteristic map of a polyhedron can be realized as a three layer linear perceptron neural network, see [7], which can be paraphrased as "all polyhedra are neural". The present paper presents a detailed version and generalizations of this result. It will be applied in [5] to prove the existence of efficient methods to decide the architecture and initial weights for perceptrons that perform a given pattern recognition task.

Recall that linear half-spaces are defined by linear inequalities, their finite intersections are linear (convex) cells and finite unions of linear cells are linear polyhedra. A formalism of indices will be developed to describe these intersections and unions in an algorithmically convenient way. Then, given a polyhedron index, a three layer linear perceptron neural network equal to the characteristic map of the polyhedron will be explicitly constructed. This is Theorem 2 below.

Cells and polyhedra are members of the algebra of subsets of euclidean space generated by half-spaces and, conversely, any member of this algebra is a polyhedron. In the more general Theorem 1, euclidean space is generalized to an arbitrary set X, half-spaces to arbitrary subsets of X, cells to finite intersections of these subsets and polyhedra to finite unions of these cells. It

then follows that "generalized polyhedra are neural". In particular, it is possible to consider subsets of euclidean space defined by polynomial inequalities, their intersections, and unions of these intersections; these are the algebraic half-spaces, algebraic cells and algebraic polyhedra respectively. Similarly for analytic, C^1 or continuous inequalities. In all cases these generalized polyhedra can be explicitly realized with three layer generalized perceptron neural networks, as stated in Theorem 3.

Many pattern recognition problems reduce to the construction of a neural network that performs certain recognition task. As already mentioned, an application of the results presented here is the formulation of efficient algorithms to determine, directly and without training, a convenient architecture and weights for a neural network to perform the desired task.

The theorems below have converses, resumed in "perceptron neural networks are polyhedral", and can be found in [3]. It follows that the most general task to be expected from a perceptron is the determination of polyhedra. Therefore the direct construction of the neural network along the lines proposed here is a reasonable general alternative to training, and in particular to training by backpropagation.

2.- **Preliminaries.** The following terminology and notation will be used in this paper. Map and function are synonymous, and C^1 means continuously differentiable. A *characteristic indicator*, or simply *indicator* is a product of characteristic functions. Linear forms are in general non-homogeneous and non-constant $f : \mathbf{R}^m \to \mathbf{R}$, $f(x_1, \ldots, x_m) = w_0 + w_1 x_1 + \cdots + w_m x_m$. The coefficients w_i are in some cases specific real numbers and in other cases will be considered as literal parameters. The linear form f defines half-spaces $H = H(f) \subseteq \mathbf{R}^m$. These can be open half-spaces $H = H^> = \{x | f(x) > 0\}$, $H = H^< = \{x | f(x) < 0\}$ or closed ones $H = H^\geq = \{x | f(x) \geq 0\}$, $H = H^\leq = \{x | f(x) \leq 0\}$. The complement of an open half-space is a closed one and viceversa. A *Heaviside function* h can be either the *open Heaviside function* $\hat{h} : \mathbf{R} \to \{0, 1\} \subseteq \mathbf{R}$, $\hat{h}(t) = 1$ for $t > 0$ and $\hat{h}(t) = 0$ for $t \leq 0$; or the *closed Heaviside function* $\bar{h} : \mathbf{R} \to \{0, 1\} \subseteq \mathbf{R}$, $\bar{h}(t) = 1$ for $t \geq 0$ and $\bar{h}(t) = 0$ for $t < 0$. The characteristic map of a half-space $H = H(f)$ is $\chi_H = h \circ f$ with $h = \hat{h}$ if the half-space is open and $h = \bar{h}$ if it is closed. By definition the *linear perceptron processing unit* p with

m inputs $x = (x_1, \ldots, x_m)$ and parameters (weights) $w = (w_0, w_1, \ldots, w_m)$ is the map $p = p(w, x) : \mathbf{R}^{m+1} \times \mathbf{R}^m \to \{0, 1\} \subseteq \mathbf{R}$ given by $p(w, x) = h(w_0 + w_1 x_1 + \cdots + w_m x_m)$. Hence, p is a parametrization of a family of half-spaces in $\mathbf{R}^m$. Again, the w_i can be arbitrary numbers (all the half-spaces), can be restricted to a subset (some half-spaces) or can take specific real values (single half-space). Will denote by $p_w : \mathbf{R}^m \to \{0, 1\} \subseteq \mathbf{R}$ the map $x \to p(w, x)$. Also, the perceptron can be an *open perceptron*, $p = \hat{p} = \hat{h} \circ f$, in which case the parametrized half-spaces are open; or it can be a *closed perceptron*, $p = \bar{p} = \bar{h} \circ f$, parametrizing closed half-spaces.

Consider now vectors $w_1 = (w_{10}, w_{11}, \ldots, w_{1m}), \ldots, w_n = (w_{n0}, w_{n1}, \ldots, w_{nm})$, $w_j \in \mathbf{R}^{m+1}$. If $p_1 = p_1(w_1, x), \ldots, p_n = p_n(w_n, x)$ are perceptrons with m inputs their *parametric product* is the map $p = p_1 \hat{\times} \cdots \hat{\times} p_n : \mathbf{R}^{m+1} \times \cdots \times \mathbf{R}^{m+1} \times \mathbf{R}^m \to \{0, 1\}^n \subseteq \mathbf{R}^n$ defined by $p(w_1, \ldots, w_n, x) = (p_1(w_1, x), \ldots, p_n(w_n, x))$. Equivalently, $p_{(w_1, \ldots, w_n)} = p_{w_1} \times \cdots \times p_{w_n}$. By definition, a *linear perceptron layer* with m inputs and n units is this parametric product. It is a parametrized family of ordered n-tuples of linear half-spaces in $\mathbf{R}^m$. If specific numerical values are given to the weights then the parametric product is the usual cartesian product.

Let $p : \mathbf{R}^{(m+1)n} \times \mathbf{R}^m \to \mathbf{R}^n$, $p' : \mathbf{R}^{(n+1)q} \times \mathbf{R}^n \to \mathbf{R}^q$ be perceptron layers. Their *parametric composition* is the map $p' \hat{\circ} p : \mathbf{R}^{(m+1)n} \times \mathbf{R}^{(n+1)q} \times \mathbf{R}^m \to \mathbf{R}^q$ given by $(p' \hat{\circ} p)(w, w', x) = p'(w', p(w, x))$. Equivalently, $(p' \hat{\circ} p)_{(w, w')} = p'_{w'} \circ p_w$. Similarly for more layers, $p^{(r)} \hat{\circ} \cdots \hat{\circ} p^{(1)}_{(w_1, \ldots, w_r)} = p^{(r)}_{w_r} \circ \cdots \circ p^{(1)}_{w_1}$. By definition, the parametric composition of r linear perceptron layers is a *linear perceptron neural network*. If the weights are specific numbers then the perceptrons reduce to single maps of their inputs and the parametric composition becomes the usual composition of functions. For more details on this neural formalism see [1]. The contents of the present paper, formulated over the real number system $\mathbf{R}$, is in general valid over the field $\mathbf{Q}$ of rational numbers.

3.- **Cells.** Consider the n-dimensional logical cube $\{0, 1\}^n = \{0, 1\}^{\{1, \ldots, n\}} \subseteq \mathbf{R}^n$ and the power set $\mathcal{P}_n = \mathcal{P}(\{1, \ldots, n\}) = \{I | I \subseteq \{1, \ldots, n\}\}$. For each binary vector $b = b^{(n)} \in \{0, 1\}^n$ denote by $I(b)$ the *support* of b, $I(b) = \{i | b_i \neq 0\}$. And for each $I \in \mathcal{P}_n$ let $b_I = b^{(n)}_I \in \{0, 1\}^n$ be the characteristic map of

I so that for any $b \in \{0,1\}^n$ one has $b = b_{I(b)}$ and for any $I \in \mathcal{P}(\{1,\ldots,n\})$, $I = I(b_I)$ holds. The correspondence $b_I \leftrightarrow I$ is one-to-one between $\{0,1\}^n$ and $\mathcal{P}_n$. By definition a *cell index* is a pair $\Sigma = (I_0, I_1)$ with $I_i \subseteq \{1,\ldots,n\}$, $i = 0,1$. Let $s_0 = |I_0|$ and $s_1 = |I_1|$. Will denote I_0 and I_1 also by $I_0 = I_0(\Sigma)$ and $I_1 = I_1(\Sigma)$.

Let X be a set with subsets $H_1,\ldots,H_n$ having characteristic functions χ_{H_1}, $\ldots,\chi_{H_n}$ and indicator $\chi_{H_1\cdots H_n} = \chi_{H_1} \times \cdots \times \chi_{H_n}$. A cell index Σ determines a *generalized cell* which is the subset of X defined as $C_\Sigma = (\bigcap_{i\in I_0}(X - H_i)) \cap (\bigcap_{i\in I_1} H_i)$. If $I_0 \cap I_1 \neq \emptyset$ it follows trivially that $C_\Sigma = \emptyset$ so, in what follows *it will be assumed that* $I_0 \cap I_1 = \emptyset$. Note that $x \in C_\Sigma \Leftrightarrow \forall i \in I_0, \chi_{H_i}(x) = 0$ and $\forall i \in I_1, \chi_{H_i}(x) = 1$.

A cell index $\Sigma = (I_0, I_1)$ also determines a (non-homogeneous) linear form $f_\Sigma : \mathbf{R}^n \to \mathbf{R}$ given by $f_\Sigma(x) = (b_{I_1} - b_{I_0})\cdot x - s_1(\Sigma) - \frac{1}{2}$. The corresponding linear perceptrons have the form $p_\Sigma = h \circ f_\Sigma : \mathbf{R}^n \to \{0,1\} \subseteq \mathbf{R}$ with $h : \mathbf{R} \to \{0,1\} \subseteq \mathbf{R}$ a Heaviside function.

Furthermore, a cell index Σ determines a set $B_\Sigma \subseteq \{0,1\}^n$ defined by $B_\Sigma = \{b = (b_1,\ldots b_n) | b_i = 0 \text{ for } i \in I_0 \text{ and } b_i = 1 \text{ for } i \in I_1\}$. It then follows that for $b \in \{0,1\}^n$, $p_\Sigma(b) = 1 \Leftrightarrow b \in B_\Sigma$. But now, for $x \in X$ one has $p_\Sigma \circ \chi_{H_1\cdots H_p}(x) = 1 \Leftrightarrow \chi_{H_1\cdots H_p}(x) \in B_\Sigma \Leftrightarrow \chi_i(x) = 0$ for $i \in I_0$ and $\chi_i(x) = 1$ for $i \in I_1 \Leftrightarrow x \in C_\Sigma$. Therefore the characteristic map of the cell C_Σ equals the composition of the indicator $\chi_{H_1\cdots H_p}$ and the perceptron p_Σ.

4.- **Polyhedra.** A *polyhedron index* is a collection $T = \{\Sigma_1,\ldots,\Sigma_q\}$ of cell indices. With the hypothesis and notation of previous section, if a polyhedron index $T = \{\Sigma_1,\ldots,\Sigma_q\}$ is given it defines linear perceptrons $p_{\Sigma_1},\ldots,p_{\Sigma_q}$ and their product $p_{\Sigma_1} \hat{\times} \cdots \hat{\times} p_{\Sigma_q}$. The index T also determines a *generalized polyhedron* which is the subset of X given by $P_T = \bigcup_{\Sigma\in T} C_\Sigma$. Denote by g the linear form $g : \mathbf{R}^q \to \mathbf{R}$ given for $z = (z_1,\ldots,z_q)$ as $g(z) = z_1 + \cdots + z_q - \frac{1}{2}$, and having perceptron $p_g = h \circ g$. Let $x \in X$, $y = (\chi_{H_1}(x),\ldots,\chi_{H_n}(x))$, $z = (p_{\Sigma_1}(y),\ldots,p_{\Sigma_q}(y))$. Then $p_g \circ (p_{\Sigma_1} \hat{\times} \cdots \hat{\times} p_{\Sigma_q}) \circ (\chi_{H_1} \hat{\times} \cdots \hat{\times} \chi_{H_n})(x) = 1 \Leftrightarrow z_1 + \cdots + z_q \geq 1 \Leftrightarrow \exists j$ s.t. $p_{\Sigma_j}(y) = 1 \Leftrightarrow \exists j$ s.t. $(\chi_{H_1}(x),\ldots,\chi_{H_n}(x)) \in B_{\Sigma_j} \Leftrightarrow x \in \bigcup_{\Sigma\in T} C_\Sigma$. Hence, the characteristic map of the polyhedron P_T equals the composition of the indicator $\chi_{H_1} \hat{\times} \cdots \hat{\times} \chi_{H_n}$ and the two layer linear perceptron $p_g \circ (p_{\Sigma_1} \hat{\times} \cdots \hat{\times} p_{\Sigma_q})$.

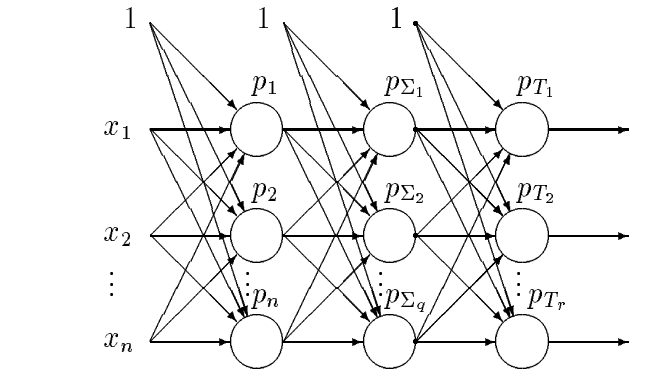
5.- **Collections of polyhedra.** Let $T_1, \ldots, T_r$ be polyhedron indices and consider $T = T_1 \cup \cdots \cup T_r$. Assume that $T = \{\Sigma_1, \ldots, \Sigma_q\}$ and that subsets $J_1, \ldots, J_r$ of $\{1, \ldots, q\}$ are given such that $T_k = \{\Sigma_k | k \in J_k\}$. In particular the corresponding linear perceptrons $p_{\Sigma_1}, \ldots, p_{\Sigma_q}$ and the indicator $p_{\Sigma_1} \widehat{\times} \cdots \widehat{\times} p_{\Sigma_q} : \mathbf{R}^n \to \mathbf{R}^q$ are specified. Also, for $k = 1, \ldots, r$ the linear forms $f_{T_k} : \mathbf{R}^q \to \mathbf{R}$ given by $f_{T_k}(z_1, \ldots, z_q) = (\sum_{j \in J_k} z_j) - \frac{1}{2}$, and the linear perceptrons $p_{T_k} = h \circ f_{T_k} : \mathbf{R}^q \to \{0, 1\} \subseteq \mathbf{R}$ can be specified. These play the role of the g and p_g of the previous section. As before, $p_{T_k} \circ (p_{\Sigma_1} \widehat{\times} \cdots \widehat{\times} p_{\Sigma_q}) \circ (\chi_{H_1} \widehat{\times} \cdots \widehat{\times} \chi_{H_n})(x) = 1 \Leftrightarrow x \in P_{\Sigma_k}$, therefore the characteristic indicator $\chi_{P_1} \widehat{\times} \cdots \widehat{\times} \chi_{P_r}$ of the polyhedra equals the composition of the indicator $\chi_{H_1} \widehat{\times} \cdots \widehat{\times} \chi_{H_n}$ with the two layer linear perceptron $(p_{T_1} \widehat{\times} \cdots \widehat{\times} p_{T_r}) \circ (p_{\Sigma_1} \widehat{\times} \cdots \widehat{\times} p_{\Sigma_q})$. The discussion implies

Theorem 1. *The indicator of a finite collection of generalized polyhedra can be realized as a three layer neural network.*

In this theorem the second and third layers are linear perceptrons. Suppose now that $X = \mathbf{R}^m$, and that the H_i's are half-spaces defined by linear perceptrons $p_1, \ldots, p_n$. Then the cells are usual convex cells in euclidean space and the polyhedra are the usual polyhedra. The previous arguments prove:

Theorem 2. *The indicator of a finite collection of polyhedra in euclidean space can be realized as a three layer linear perceptron neural network.*

In more detail, given $T_1, \ldots, T_r$ and $T = T_1 \cup \cdots \cup T_r = \{\Sigma_1, \ldots, \Sigma_q\}$ as above, the neural network is the one shown in the figure below.



Neural network for $\chi_{T_1 \cdots T_r}$

A brief description of the network is the following. The first layer has one bias, m variable inputs and n perceptron units $p_1, \ldots, p_n$. The arrows entering the unit p_i have weights equal to the coefficients of the linear form that defines the half-space H_i. The second layer has one bias, n variable inputs and q perceptron units $p_{\Sigma_1}, \ldots, p_{\Sigma_q}$; the weights for the arrows entering p_{Σ_i} are, from top to bottom, the bias weight $-s_1(\Sigma_i) - \frac{1}{2}$ and the components of the vector $b_{J_1(\Sigma_i)} - b_{J_0(\Sigma_i)}$. The third layer has one bias, q variable inputs and r perceptron units, $p_{T_1}, \ldots, p_{T_r}$; the weights on the arrows entering p_{T_k} are the bias weight $-\frac{1}{2}$ and the q components of the vector b_{J_k}.

The network can also be described in terms of weight matrices.
FIRST LAYER: Let m=number of components of the initial input vectors, n=number of perceptrons; these are independent of the indices. The weight matrix, with n rows and $m+1$ columns, has the form

$$W^{(1)} = \begin{bmatrix} w_{10}^{(1)} & w_{11}^{(1)} & \cdots & w_{1m}^{(1)} \\ \vdots & \vdots & \cdots & \vdots \\ w_{n0}^{(1)} & w_{n1}^{(1)} & \cdots & w_{nm}^{(1)} \end{bmatrix}$$

SECOND LAYER: Let q be the total number of cell indexes composing the

various polyhedra, as indicated in $T = T_1 \cup \cdots \cup T_r = \{\Sigma_1, \ldots, \Sigma_q\}$. In the second layer the weight matrix, with q rows and $m+1$ columns, is

$$W^{(2)} = \begin{bmatrix} -s_1(\Sigma_1) - \frac{1}{2} & b_{J_1(\Sigma_1)} - b_{J_0(\Sigma_1)} \\ \vdots & \vdots \\ s_1(\Sigma_q) - \frac{1}{2} & b_{J_1(\Sigma_q)} - b_{J_0(\Sigma_q)} \end{bmatrix}$$

This matrix depends on the indexes. Recall that b_J was defined as the only binary vector with support J, hence, all entries outside the first column are 1, 0 or -1.

THIRD LAYER: If the total number of polyhedron indices is r, the weight matrix for the third layer has r rows and $q+1$ columns

$$W^{(3)} = \begin{bmatrix} -\frac{1}{2} & b_{T_1} \\ \vdots & \vdots \\ -\frac{1}{2} & b_{T_r} \end{bmatrix}$$

with all entries outside the first column equal to 0 or 1.

6.- **Examples of generalized polyhedra.** Let $f = f(x_1, \ldots, x_m) : \mathbf{R}^m \to \mathbf{R}$ be a polynomial, $h : \mathbf{R} \to \{0,1\} \subseteq \mathbf{R}$ a Heaviside function, then $p = h \circ f : \mathbf{R}^m \to \{0,1\} \subseteq \mathbf{R}$ is an (open or closed) *algebraic perceptron* with weights the coefficients of f. The *degree* of p is, by definition, the degree of f. For example, perceptrons of degree one are the same as linear perceptrons. Algebraic perceptrons are characteristic functions of parametrized families of *algebraic half-spaces* $H = H^{>} = \{x | p(x) > 0\}$ ($h = \hat{h}$ open) or of $H = H^{\geq} = \{x | p(x) \geq 0\}$ ($h = \bar{h}$ closed). Suppose that $p_1, \ldots, p_n$ are algebraic perceptrons with algebraic half-spaces $H_1, \ldots, H_n$ and let Σ be a cell index, then the corresponding cell C_Σ is an *algebraic cell* (or *basic semialgebraic set*). If furthermore $T = \{\Sigma_1, \ldots, \Sigma_q\}$ is a polyhedron index then $P_T = C_{\Sigma_1} \cup \cdots \cup C_{\Sigma_q}$ is an *algebraic polyhedron.* Let us state explicitly

Theorem 3. *The indicator of a finite collection of algebraic polyhedra in euclidean space can be realized as a three layer algebraic perceptron neural network with second and third layers of degree one.*

Analogous concepts and results can be obtained if analytic C^1 or continuous maps are used instead of polynomials. In this way one obtains analytic

half-spaces, analytic cells, analytic polyhedra, C^1 half-spaces, C^1 cells, C^1 polyhedra, and continuous half spaces, cells and polyhedra. Analytic, C^1 and continuous indicators are realizable, as in Theorem 3, with a three layer analytic or C^1 perceptron network.

7.- **Training.** Linear perceptrons, and the algebraic, analytic, C^1 and continuous perceptrons of the previous section, are not C^1 and not even continuous functions of their inputs. Given $p = h \circ f$, with f a C^1 map, to obtain a C^1 function replace the discontinuous Heaviside function h by the sigmoid $\sigma : \mathbf{R} \to (0,1) \subseteq \mathbf{R}$, $\sigma(t) = (1 + e^{-t})^{-1}$. The resulting object, $\sigma \circ f$ will be called *σ-perceptron* or *generalizedσperceptron*. These σ-perceptrons are C^1 functions of their inputs and the σ-neural networks that result can be trained using backpropagation rules. For a generalized formulation of backpropagation see [2]. As already mentioned, the combined results of [3] and of sections 3 and 4 above, provide a direct and efficient method to specify the architecture and weights for a linear or generalized (including continuous) perceptron neural network that performs a given recognition task. At this point standard training of the network (in the C^1 case) could fine tune the weights and improve its performance, something not yet explored.

REFERENCES

[1] Crespin, D. Neural Network Formalism. To appear.
[2] Crespin, D. Generalized Backpropagation. To appear.
[3] Crespin, D. Geometry of Perceptrons. To appear.
[4] Crespin, D. Neural Polyhedra (this present paper).
[5] Crespin, D. Pattern recognition with untrained perceptrons.
[6] Crespin, D. Feature Extraction. To appear.
[7] Lippmann, D. Introduction to Neural Networks.

Daniel Crespin
dcrespin@euler.ciens.ucv.ve
Caracas, November 29, 1995.