Optimization in Machine Learning*

O. L. Mangasarian†

Mathematical Programming Technical Report 95-01
January 1995

1 Introduction

Optimization has played a significant role in training neural networks [23]. This has resulted in a number of efficient algorithms [22, 3, 5, 29, 31] and practical applications in medical diagnosis and prognosis [34, 35, 27]. Other applications of neural networks abound [12, 30, 18, 13] . In this brief work we focus on a number of problems of machine learning and pose them as optimization problems. Hopefully this will point to further applications of optimization to the burgeoning field of machine learning.

2 Misclassification Minimization

A fundamental problem of machine learning is to construct (train) a classifier to distinguish between two or more disjoint point sets in an n-dimensional real space. A key factor in determining the classifier is the measure of error used in constructing the classifier. We shall propose two error measures: one will merely count the number of misclassified points, while the other will measure the average distance of misclassified points from a separating plane. We will show that the first leads to an LPEC (linear program with equilibrium constraints) [24, 20] while the second leads to a single linear program [21, 4]. However, the problem of minimizing the number of misclassified points turns out to be NP-complete [11, 17], but we shall indicate effective approaches [24, 2] that render it more tractable.

For the sake of simplicity we shall limit ourselves to discriminating between two sets, although optimization models apply readily to multicategory discrimination [6, 7]. Let $\mathcal{A}$ and $\mathcal{B}$ be two disjoint point sets in R^n with cardinalities m and k respectively. Let the m points of $\mathcal{A}$ be represented by the $m \times p$ matrix A, while the k points of $\mathcal{B}$ be represented by the $k \times p$ matrix B. The integer p represents the dimensionality of the real space R^p into which the points of $\mathcal{A}$ and $\mathcal{B}$ are mapped by $F : R^n \to R^p$, before their separation is attempted. In the simplest model $p = n$ and F is the identity map. However, more complex separation, say by quadratic surfaces [21], can be effected if one resorts to more general maps. (Note that complex separation, like fitting with high degree polynomials, is not always desirable, since it may lead to merely "memorizing" the training set.) The simplest and one of the most effective classifiers in R^p is the plane

$$xw = \theta \tag{1}$$

where $w \in R^p$ is the normal to the plane, $|\theta|/\|w\|_2$ is the distance of the plane to the origin in R^p, $x \in R^p$ is a point belonging to $F(\mathcal{A})$ or $F(\mathcal{B})$, and $\|\cdot\|_2$ denotes the 2-norm. The problem of training a linear classifier consists then of determining $(w, \theta) \in R^{p+1}$ so as to minimize the error criterion chosen. We note immediately that if the sets $F(\mathcal{A})$ and $F(\mathcal{B})$ are strictly linearly separable in R^p, then there exist $(w, \theta) \in R^{p+1}$

*This material is based on research supported by Air Force Office of Scientific Research Grant F49620-94-1-000036 and National Science Foundation Grant CCR-9322479.

†Computer Sciences Department, University of Wisconsin, 1210 West Dayton Street, Madison, WI 53706, email: *olvi@cs.wisc.edu*.

such that

$$\begin{aligned} Aw &\geqq e\theta + e \\ Bw &\leqq e\theta - e \end{aligned} \tag{2}$$

where e is a vector of ones of appropriate dimension. Since, in general (2) is not satisfiable, we attempt its approximate satisfaction by minimizing the chosen error criterion.

2.1 Minimization of Number of Misclassified Points

Let $s : R \to \{0, 1\}$ determine the step function that maps nonpositive numbers into $\{0\}$ and positive numbers into $\{1\}$. When applied to a vector $z \in R^p$, s returns a vector of zeros and ones in R^p, corresponding respectively to nonpositive and positive components $z_i,\ i = 1, \dots p$, of z. The problem of minimizing the number of misclassified points then reduces to the following unconstrained minimization problem of a discontinuous function:

$$\begin{aligned} \min_{(w,\theta)\in R^{n+1}} &\|s(-Aw + e\theta + e)\| \\ &+\|s(Bw - e\theta + e)\| \end{aligned} \tag{3}$$

where $\|\cdot\|$ denotes some arbitrary, but fixed norm, on R^m or R^k. The sets $F(\mathcal{A})$ and $F(\mathcal{B})$ are linearly separable in R^p, if and only if the minimum of (3) is zero, and no points are misclassified, otherwise the minimum of (3) "counts" the number of misclassified points if the 1-norm is used. In [24] it was shown that (3) with the 1-norm is equivalent to the following LPEC:

$$\begin{array}{lll} \underset{w,\theta,r,u,s,v}{minimize} & er + es & \\ & u + Aw - e\theta - e \geqq 0 & \\ & r \geqq 0 & \\ & r(u + Aw - e\theta - e) = 0 & \\ & -r + e \geqq 0 & \\ & u \geqq 0 & \\ \text{subject to} & u(-r + e) = 0 & (4) \\ & v - Bw + e\theta - e \geqq 0 & \\ & s \geqq 0 & \\ & s(v - Bw + e\theta - e) = 0 & \\ & -s + e \geqq 0 & \\ & v \geqq 0 & \\ & v(-s + e) = 0 & \end{array}$$

It turns out that problem (4) is extremely difficult to solve. In fact, almost every point $(w, \theta) \in R^{p+1}$ is a stationary point, since a small perturbation of a plane $xw = \theta$ in R^p that does not contain points of either $F(\mathcal{A})$ or $F(\mathcal{B})$ will not change the number of misclassified points. In order to circumvent this difficulty, a parametric implicitly exact penalty function was proposed for solving (4) in [24] and implemented successfully in [2] by an approach that also identifies outlying misclassified points. A fast hybrid algorithm for approximately solving the misclassification minimization problem is also given in [11].

Another approach to solving (3) is by utilizing the highly effective smoothing technique [9, 10] that has been used to solve many mathematical programs and related problems. In this approach, the step function $s(\zeta)$ is replaced by the classical sigmoid function of neural networks [18]:

$$s(\zeta) \cong \sigma(\zeta, \alpha) := \frac{1}{1 + e^{-\alpha\zeta}} \tag{5}$$

where α is a positive real number that approaches $+\infty$ for more accurate representation of the step function. With this approximation, the unconstrained discontinuous minimization problem is reduced to an unconstrained continuous optimization problem, that is however nonconvex. By letting α grow judiciously, effective computational schemes for tackling the NP-complete problem can be utilized. An important application of the misclassification error (3), is its use in constructing the more complex nonlinear neural network classifier of Section 3 below.

2.2 Minimization of Average Distance of Misclassifications from Separating Plane

As early as 1964 [8, 21], the distance of misclassified points from a separating plane was utilized to generate a linear programming problem for obtaining a separating plane (1) that approximately satisfied (2) by minimizing some measure of distance of misclassified points from the plane (1). Unfortunately, all these attempts [22, 16, 15] contained *ad hoc* ways for excluding the null solution ($w = 0$) that plagued a linear programming formulation for linearly inseparable sets. However, the robust model proposed in [4], which consists of minimizing the average of the 1-norm of the distances of misclassified points from the separating plane, completely overcame this difficulty. The linear program [4] proposed is this:

$$\begin{array}{lll} \underset{w,\theta,y,z}{minimize} & \frac{ey}{m} + \frac{ez}{k} & \\ & Aw + y \geqq e\theta + e & \\ \text{subject to} & Bw - z \leqq e\theta - e & \\ & y, z \geqq 0 & \end{array} \qquad (6)$$

The key property of (6) is that it gives the null solution $w = 0$ if and only if $\frac{eA}{m} = \frac{eB}{k}$, in which case $w = 0$ is guaranteed to be not unique. Computationally, the LP (6) is very robust, rarely giving rise to the null solution, even in contrived examples where $\frac{eA}{m} = \frac{eB}{k}$. In the parlance of machine learning [18], the separating plane (1) is referred to as a "perceptron", "linear threshold unit" or simply "unit", with threshold θ and incoming arc weight w. This is in analogy to a human neuron which fires if the input $x \in R^p$, scalar-multiplied by the weight $w \in R^p$, exceeds the threshold θ.

3 Neural Networks as Polyhedral Regions

A neural network can be defined as a generalization of a separating plane in R^p, and can be thought of as a nonlinear map: $R^p \to \{0, 1\}$. One intuitive way to generate such a map is to divide R^p into various polyhedral regions, each of which containing elements of $F(\mathcal{A})$ or $F(\mathcal{B})$ only. In its general form, this problem is again an extremely difficult and nonconvex problem. However, greedy sequential constructions of the planes determining the various polyhedral regions [22, 25, 1] have been quite successful in obtaining very effective algorithms for training neural networks much faster than the classical online (that is training on one point at a time) backpropagation (BP) gradient algorithm [32, 18, 26]. Online BP is often erroneously referred to as a descent algorithm, which it is not.

In this section of the paper we relate the polyhedral regions into which R^p is divided, to a neural network with one hidden layer of linear threshold units. It turns out that every such neural network can be related to a partitioning of R^p into polyhedral regions, but not the conversely. However, any two disjoint point sets in R^p can be discriminated between by <u>some</u> polyhedral partition that corresponds to a neural network with one hidden layer with a sufficient number of hidden units [19, 25].

We describe now precisely when a specific partition of R^p by h separating planes

$$xw^i = \theta^i, \ i = 1, \ldots, h, \qquad (7)$$

corresponds to a neural network with h hidden units. The h separating planes (7) divide R^p into at most t polyhedral regions, where [14]

$$t := \sum_{i=0}^{p} \begin{pmatrix} h \\ i \end{pmatrix}. \qquad (8)$$

We shall assume that $F(\mathcal{A})$ and $F(\mathcal{B})$ are contained in the interiors of two mutually exclusive subsets of these regions. Each of these polyhedral regions can be mapped uniquely into a vertex of the unit cube in R^h,

$$\{z | z \in R^h, \ 0 \leqq z \leqq e\} \qquad (9)$$

by using the map:

$$s(xw^i - \theta^i), \ i = 1, \ldots, h \qquad (10)$$

where s is the step function defined earlier, and x is a point in R^p belonging to some polyhedral region. If the r polyhedral regions of R^p constructed by the h planes (7) are such that vertices of the cube (9) corresponding to points in $\mathcal{A}$, are linearly separable in R^h from the vertices of (9) corresponding to points in $\mathcal{B}$ by a plane

$$zv = \tau, \qquad (11)$$

then the polyhedral partition of R^p corresponds to a neural network with h hidden linear threshold units (with thresholds θ^i, incoming arc weights w^i, $i = 1, \ldots, h$) and output linear threshold unit (with threshold τ and incoming arc weights v_i, $i = 1, \ldots, h$ [23]) . This condition is necessary and sufficient for the polyhedral partition of R^p in order for it to correspond to a neural network with one layer of hidden units. For more detail and graphical depiction of the neural network, see [23]. "Training" a neural network consists of determining $(w^i, \theta^i) \in R^{p+1}$, $i = 1, \ldots, h$, $(v, \tau) \in R^{h+1}$, such that the following nonlinear inequalities are satisfied as best as possible:

$$\begin{aligned} \sum_{i=1}^{h} s(Aw^i - e\theta^i)v_i &\geqq e\tau + e \\ \sum_{i=1}^{h} s(Bw^i - e\theta^i)v_i &\leqq e\tau - e \end{aligned} \tag{12}$$

This can be achieved by minimizing the number of misclassified points in R^h by solving the following unconstrained minimization problem

$$\begin{aligned} &\min_{w^i,\theta^i,v,\tau} \|s(-\sum_{i=1}^{h} s(Aw^i - e\theta^i)v_i + e\tau + e)\| \\ &+\|s(\sum_{i=1}^{h} s(Bw^i - e\theta^i)v_i - e\tau + e)\| \end{aligned} \tag{13}$$

where the norm is some arbitrary norm. If the square of the 2-norm is used in (13) instead of the 1-norm, and if the step function s is replaced by the sigmoid function in (13), we obtain an error function similar to the error function that BP attempts to find a stationary point for, and for which a convergence proof is given in [26], and stability analysis in [33]. We note that the classical exclusive-or (XOR) example [28] for which F is the identity map and $A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$, gives a zero minimum for (13) with the following solution:

$$\begin{aligned} &(w^1, \theta^1) = ((2 \;\; -2),\ 1),\ (w^2, \theta^2) = ((-2 \;\; 2),\ 1) \\ &(v, \tau) = ((2 \;\; 2),\ 1) \end{aligned} \tag{14}$$

It is interesting to note that the same solution for the XOR example is given by the greedy multisurface method tree (MSMT) [1]. MSMT attempts to separate as many points of $\mathcal{A}$ and $\mathcal{B}$ as possible by a first plane obtained by solving (6), and then repeats the process for each of the ensuing halfspaces, until adequate separation is obtained. For this example, the first plane obtained [4] is $(w^1, \theta^1) = ((2 \quad -2),\ 1)$, which separates $\{(1,0)\}$ from $\{(0,0),(0,1),(1,1)\}$. The second plane obtained is $(w^2, \theta^2) = ((-2 \;\; 2),\ 1)$, separates $\{(0,1)\}$ from $\{(0,0),(1,1)\}$, and the separation is complete between $\mathcal{A}$ and $\mathcal{B}$. These planes correspond to a neural network that gives a zero minimum to (13), which of course is not always the case. However, MSMT frequently gives better solutions than those generated by BP and is much faster than BP.

4 Conclusion

Various problems associated with neural network training have been cast as mathematical programs. Effective methods for solving these problems have been briefly described. For more details, the reader is referred to [3, 4, 23].

References

[1] K. P. Bennett. Decision tree construction via linear programming. In M. Evans, editor, *Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society Conference*, pages 97–101, Utica, Illinois, 1992.

[2] K. P. Bennett and E.J. Bredensteiner. A parametric optimization method for machine learning. Department of Mathematical Sciences Report No. 217, Rensselaer Polytechnic Institute, Troy, NY 12180, 1994.

[3] K. P. Bennett and O. L. Mangasarian. Neural network training via linear programming. In P. M. Pardalos, editor, *Advances in Optimization and Parallel Computing*, pages 56–67, Amsterdam, 1992. North Holland.

[4] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.

[5] K. P. Bennett and O. L. Mangasarian. Bilinear separation of two sets in n-space. *Computational Optimization & Applications*, 2:207–227, 1993.

[6] K. P. Bennett and O. L. Mangasarian. Multicategory separation via linear programming. *Optimization Methods and Software*, 3:27–39, 1993.

[7] K. P. Bennett and O. L. Mangasarian. Serial and parallel multicategory discrimination. *SIAM Journal on Optimization*, 4(4):722–734, 1994.

[8] A. Charnes. Some fundamental theorems of perceptron theory and their geometry. In J. T. Lou and R. H. Wilcox, editors, *Computer and Information Sciences*, pages 67–74, Washington, D.C., 1964. Spartan Books.

[9] Chunhui Chen and O. L. Mangasarian. Smoothing methods for convex inequalities and linear complementarity problems. Technical Report 1191, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, November 1993. Mathematical Programming, to appear. Available from ftp://ftp.cs.wisc.edu/tech-reports/reports/93/tr1191.ps.Z.

[10] Chunhui Chen and O. L. Mangasarian. A class of smoothing functions for nonlinear and mixed complementarity problems. Technical Report 94-11, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, August 1994. Computational Optimization and Applications, to appear. Available from ftp://ftp.cs.wisc.edu/math-prog/tech-reports/94-11.ps.Z.

[11] Chunhui Chen and O. L. Mangasarian. Hybrid misclassification minimization. Technical Report 95-05, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, February 1995. Advances in Computational Mathematics, submitted. Available from ftp://ftp.cs.wisc.edu/math-prog/tech-reports/95-05.ps.Z.

[12] L. DeSilets, B. Golden, Q. Wang, and R. Kumar. Predicting salinity in the Chesapeake Bay using backpropagation. *Computers & Operations Research*, 19:277–285, 1992.

[13] S.I. Gallant. *Neural Network Learning and Expert Systems.* MIT Press, Cambridge, Massachusetts, 1993.

[14] G.M. Georgiou. Comments on hidden nodes in neural nets. *IEEE Transactions on Circuits and Systems*, 38:1410, 1991.

[15] F. Glover. Improved linear programming models for discriminant analysis. *Decision Sciences*, 21:771–785, 1990.

[16] R.C. Grinold. Mathematical methods for pattern classification. *Management Science*, 19:272–289, 1972.

[17] David Heath. *A geometric Framework for Machine Learning.* PhD thesis, Department of Computer Science, Johns Hopkins University–Baltimore, Maryland, 1992.

[18] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley, Redwood City, California, 1991.

[19] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[20] Z.-Q. Luo, J.-S. Pang, D. Ralph, and S.-Q. Wu. Exact penalization and stationarity conditions of mathematical programs with equilibrium constraints. Technical Report 275, Communications Research Laboratory, McMaster University, Hamilton, Ontario, Hamilton, Ontario L8S 4K1, Canada, 1993. Mathematical Programming, to appear.

[21] O. L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1965.

[22] O. L. Mangasarian. Multi-surface method of pattern separation. *IEEE Transactions on Information Theory*, IT-14:801–807, 1968.

[23] O. L. Mangasarian. Mathematical programming in neural networks. *ORSA Journal on Computing*, 5(4):349–360, 1993.

[24] O. L. Mangasarian. Misclassification minimization. *Journal of Global Optimization*, 5:309–323, 1994.

[25] O. L. Mangasarian, R. Setiono, and W. H. Wolberg. Pattern recognition via linear programming: Theory and application to medical diagnosis. In T. F. Coleman and Y. Li, editors, *Large-Scale Numerical Optimization*, pages 22–31, Philadelphia, Pennsylvania, 1990. SIAM. Proceedings of the Workshop on Large-Scale Numerical Optimization, Cornell University, Ithaca, New York, October 19-20, 1989.

[26] O. L. Mangasarian and M.V. Solodov. Serial and parallel backpropagation convergence via nonmonotone perturbed minimization. *Optimization Methods and Software*, 4(2):103–116, 1994.

[27] O. L. Mangasarian, W. Nick Street, and W. H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. Technical Report 94-10, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin 53706, 1994. Operations Research, to appear. Available from ftp://ftp.cs.wisc.edu/math-prog/tech-reports/94-10.ps.Z.

[28] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry.* MIT Press, Cambridge, Massachusetts, 1969.

[29] S. Mukhopadhyay, A. Roy, and S. Govil. A polynomial time algorithm for generating neural networks for pattern classification: Its stability properties and some test results. *Neural Computation*, 5:317–330, 1993.

[30] K.E. Nygard, P. Juell, and N. Kadaba. Neural networks for selecting vehicle routing heuristics. *ORSA Journal on Computing*, 4:353–364, 1990.

[31] A. Roy, L.S. Kim, and S. Mukhopadhyay. A polynomial time algorithm for the construction and training of a class of multilayer perceptrons. *Neural Networks*, 6:535–545, 1993.

[32] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing*, pages 318–362, Cambridge, Massachusetts, 1986. MIT Press.

[33] M.V. Solodov and S.K. Zavriev. Stability properties of the gradient projection method with applications to the back-propagation algorithm. Computer Sciences Department, Mathematical Programming Technical Report 94-05, University of Wisconsin, Madison, Wisconsin, June 1994. SIAM Journal on Optimization, submitted.

[34] W. H. Wolberg and O. L. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the National Academy of Sciences,U.S.A.*, 87:9193–9196, 1990.

[35] W. H. Wolberg, W. N. Street, and O. L. Mangasarian. Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates. *Cancer Letters*, 77:163–171, 1994.