NOTE Communicated by Richard Lippmann

A Short Proof of the Posterior Probability Property of Classifier Neural Networks

Raúl Rojas
Institut für Informatik, Freie Universität Berlin, Takustr. 9, 14195 Berlin, Germany

It is now well known that neural classifiers can learn to compute a posteriori probabilities of classes in input space. This note offers a shorter proof than the traditional ones. Only one class has to be considered and straightforward minimization of the error function provides the main result. The method can be extended to any kind of differentiable error function. We also present a simple visual proof of the same theorem, which stresses the fact that the network must be perfectly trained and have enough plasticity.

It is now well known that neural networks trained to classify an n-dimensional input x in one out of M classes can actually learn to compute the Bayesian a posteriori probabilities that the input x belongs to each class. Several proofs of this fact, differing only in the details, have been published (Bourlard and Morgan 1993; Richard and Lippmann 1991), but they can be simplified. In this note we offer a shorter proof of the probability property of classifier neural networks.

Figure 1a shows the main idea of the proof. Points in an input space are classified as belonging to a class A or its complement. This is the first simplification: we do not have to deal with more than one class. In classifier networks, there is one output line for each class C_i, $i = 1, \ldots, M$. Each output C_i is trained to produce a 1 when the input belongs to class i and otherwise a 0. As the expected total error is the sum of the expected individual errors of each output, we can minimize the expected individual errors independently. This means that we need to consider only one output line and when it should produce a 1 or a 0.

Assume that input space is divided into a lattice of differential volumes of size dv, each one centered at the n-dimensional point v. If at the output representing class A the network computes the value $y(v) \in [0, 1]$ for any point x in the differential volume $V(v)$ centered at v, and denoting by $p(v)$ the probability $p[A \mid x \in V(v)]$, then the total expected quadratic error is

$$E_A = \sum_V \{p(v)[1 - y(v)]^2 + [1 - p(v)]y(v)^2\}dv$$

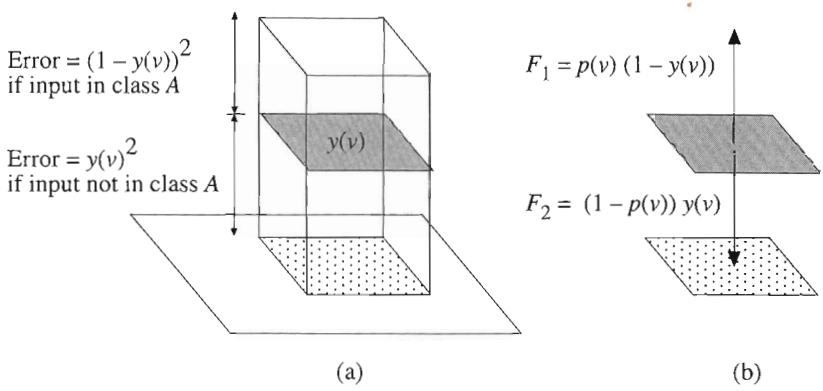

Figure 1: The output $y(v)$ in a differential volume.

where the sum runs over all differential volumes in the lattice. Assume that the values $y(v)$ can be computed independently for each differential volume. This means that we can independently minimize each of the terms of the sum. This is done by differentiating each term with respect to the output $y(v)$ and equating the result to zero

$$-2p(v)[1 - y(v)] + 2[1 - p(v)]y(v) = 0$$

From this expression we deduce $p(v) = y(v)$, that is the output $y(v)$ which minimizes the error in the differential region centered at v is the a posteriori probability $p(v)$. In this case the expected error is

$$p(v)[1 - p(v)]^2 + [1 - p(v)]y(v)^2 = p(v)[1 - p(v)]$$

and E_A becomes the expected variance of the output line for class A.

Note that extending the above analysis to other kinds of error functions is straightforward. For example, if the error at the output is measured by $\log[1 - y(v)]$ when the desired output is 1 and $\log[y(v)]$ when it is 0, then the terms in the sum of expected differential errors have the form

$$p(v)\log[1 - y(v)] + [1 - p(v)]\log[y(v)]$$

Differentiating and equating to zero we again find $y(v) = p(v)$.

This short proof also strongly underlines the two conditions needed for neural networks to produce a posteriori probabilities, namely *perfect training* and *enough plasticity* of the network, so as to be able to approximate the patch of probabilities given by the lattice of differential volumes and the values $y(v)$, which we optimize independently of each other.

It is still possible to offer a simpler visual proof "without words" of the Bayesian property of classifier networks, as is done in Figure 1b. When training to produce 1 for the class A and 0 for A^c, we subject the function produced by the network to an "upward force" proportional to the derivative of the error function, i.e., $[1-y(v)]$, and the probability $p(v)$, and a downward force proportional to $y(v)$ and the probability $[1-p(v)]$. Both forces are in equilibrium when $p(v) = y(v)$.

References

Bourlard, H., and Morgan, N. 1993. *Connectionist Speech Recognition.* Kluwer Academic, Boston.

Richard, M. D., and Lippmann, R. P. 1991. Neural network classifiers estimate *a posteriori* probabilities. *Neural Comp.* 3(4), 461–483.

Received August 19, 1994; accepted April 5, 1995.