6-15-1993

Putting Humpty-Dumpty together again: Reconstructing functions from their projections.

Anil Ravindran Menon
Syracuse University

Kishan Mehrotra
Syracuse University, mehrotra@syr.edu

Chilukuri K. Mohan
Syracuse University, ckmohan@syr.edu

Sanjay Ranka
Syracuse University

SU-CIS-93-28

Putting Humpty-Dumpty together again: Reconstructing functions from their projections

Anil Menon, Kishan Mehrotra,
Chilukuri Mohan, and Sanjay Ranka

June 15, 1993

School of Computer and Information Science
Syracuse University
Suite 4-116, Center for Science and Technology
Syracuse, New York 13244-4100

Putting Humpty-Dumpty together again: Reconstructing functions from their projections.

Anil Menon
armenon@top.cis.syr.edu

Kishan Mehrotra
kishan@top.cis.syr.edu

Chilukuri Mohan
mohan@top.cis.syr.edu

Sanjay Ranka
ranka@top.cis.syr.edu

Syracuse University
Neural Network Group
School of Computer & Information Science, 4-116 CST
Syracuse, NY 13244-4100

ABSTRACT:
We present a problem decomposition approach to reduce neural net training times. The basic idea is to train neural nets in parallel on marginal distributions obtained from the original distribution (via projection), and then reconstruct the original table from the marginals (via a procedure similar to the *join* operator in database theory). A function is said to be reconstructible, if it may be recovered without error from its projections. Most distributions are non-reconstructible. The main result of this paper is the *Reconstruction theorem*, which enables non-reconstructible functions to be expressed in terms of reconstructible ones, and thus facilitates the application of decomposition methods.

Main Category: Algorithms & Architectures,
Sub-Category: Constructive & Pruning Algorithms.

Putting Humpty-Dumpty together again: Reconstructing functions from their projections.

Anil Menon
armenon@top.cis.syr.edu

Kishan Mehrotra
kishan@top.cis.syr.edu

Chilukuri Mohan
mohan@top.cis.syr.edu

Sanjay Ranka
ranka@top.cis.syr.edu

Syracuse University
Neural Network Group
School of Computer & Information Science, 4-116 CST
Syracuse, NY 13244-4100

1 Introduction

Suppose a neural network is to be taught a complicated, real valued multivariable function $f : \Re^n \rightarrow \Re$, represented as a table. Real-time calculation of f is assumed to be infeasible or undesirable. The decomposition-reconstruction problem is the problem of reconstructing a function from its projections. Metaphorically, it is the problem of putting Humpty-Dumpty together again, without having to call in all the King's horses or men. We present a solution to the decomposition-reconstruction problem, using tools and techniques from database theory.

The main idea underlying our solution is quite simple. We start with a multivariable distribution and obtain a set of marginals. Obviously, it is not always possible to reconstruct the original distribution from the marginals. One may construct a "best" reconstruction in many different ways. A particularly appealing criterion is to obtain a reconstruction that has the maximum Shannon entropy. In probabilistic database theory such a maximum entropy reconstruction is referred to as the *join* distribution. It is possible then, to compute an *error* distribution; basically nothing more than the difference between the original and the join distributions. It may be shown

(the Reconstruction theorem) that after some minor processing, the error distribution is reconstructible. In other words, the error distribution after some modification, is also a maximum entropy reconstruction of a set of marginals. The original distribution may then be expressed as a linear combination of the join distribution and the error distribution. The marginals of each of these two distributions are taught to a disjoint set of neural nets in parallel. For other approaches to the problem of reducing the training time, of neural networks, (for example, by reducing the size of the network), see [5, 15, 17].

The decomposition-reconstruction problem has of course, already been studied in great depth from various points of view, including Reconstructability theory, Database theory, Statistics and Information theory. Many of the fundamental issues involved were first worked out by Ross Ashby in the early 60's, in a Systems theoretic context [1]. The joint work of George Klir and Roger Cavallo laid the foundations for the notion of system reconstructability [6, 7], much of it destined to be rediscovered in the then emerging field of relational database theory. In recent years there has been an increasing interest in a generalization of relational databases, the theory of probabilistic databases [2, 8].

We believe probabilistic database theory offers a fresh, simple, powerful and elegant approach to the problem of training neural networks on tables of data. It is possible to develop a much more systematic and logical approach to this problem, free of the rampant *ad hockery* present in current approaches.

Section 2 defines the basic tools underlying this approach, drawn from the theory of probabilistic databases[8]. In Section 3 we present the Reconstruction theorem, the central theorem of this paper. Section 4 is concerned with how ideas from probabilistic database theory may be used to advantage in neural network theory. Finally, in Section 5, we illustrate these ideas by applying them to a control problem. There are many *kinds* of neural networks. For concreteness, the semilinear feedforward network may be assumed to be the basic underlying neural architecture for this paper (though the results are architecture independent).

2 Probabilistic Systems

Definition 2.1 A *probabilistic system* (PS) is a 2-tuple $D = (V, \mathbf{p})$ where, $V = \{v_1, \ldots, v_n\}$, referred to as the *scheme* of the PS, is a non-empty set

of variables, each v_i taking values from a *finite* set S_i, and,

$$\mathbf{p} : \mathcal{T}(V) \rightarrow [0,1]$$
$$\sum_{t \in \mathcal{T}(V)} \mathbf{p}(t) = 1;$$

where, *the finite product space* $\mathcal{T}(V) = \prod_i S_i$, is the set of *tuples* of the system D. A *model* of a scheme V is a set $X = \{V_1, \ldots, V_n\}$ such that $\cup_1^n V_j \subseteq V$ and $V_i \not\subseteq V_j, \forall i,j \in \{1, \ldots, n\}$. A model X, of a scheme V, is *non-trivial* iff $X \neq \{\emptyset\}$ and $X \neq \{V\}$. A collection of probabilistic systems $D_1, \ldots, D_k$ is a *probabilistic database* (PD). A collection of probabilistic systems $D_i = (V_i, \mathbf{p}_i)$ together define a *probabilistic database* (PD). ∎

Notation:
The tuples of a PS will be referred to by small greek letters α, β, etc. The kth component of a tuple α, will be denoted by α_k. Given two schemes V and V' with $V' \subseteq V$, and tuple $\alpha \in \mathcal{T}(V)$ then $\alpha[V']$ is the restriction of α to variables in V'. ∎

Probabilistic systems closely resemble contingency tables. However, viewing them as generalizations of relational databases leads to important conceptual and computational advantages. It is fortunate that the development of probabilistic database theory was largely motivated by problems peculiar to database theory. A wealth of new ideas (at least, most of them are only a decade old) lie waiting to be interpreted in contexts and applications, other than database theory. We will consider this issue in greater detail in Section 4. The next two definitions formalize the notions of decomposition and reconstruction.

Definition 2.2 (Projections): Let $D = (V, \mathbf{p})$ be a probabilistic system. Let $V = \{v_1, \ldots v_n\}$, where variables v_i take values from finite sets S_i, Let $V' = \{v_{i_1}, \ldots v_{i_k}\} \subseteq V$, $k \leq n$ and $\mathcal{T}(V') = \prod_{j=1}^{k} S_{i_j}$. The *projection* of $\mathbf{p}$ onto V' yields a new distribution, $\mathbf{p}' = \pi_{V'}(\mathbf{p})$ such that,

$$\mathbf{p}' : \mathcal{T}(V') \rightarrow [0,1]$$
$$\mathbf{p}'(\beta) = \sum_{\alpha \in \mathcal{T}(V), \alpha[V]=\beta} \mathbf{p}(\alpha)$$

The projection of a distribution $\mathbf{p}$ onto a model, $X = \{V_1, \ldots, V_k\}$ is, $\pi_X(\mathbf{p}) = \{\pi_{V_1}(\mathbf{p}), \ldots, \pi_{V_k}(\mathbf{p})\}$. ∎

Definition 2.3 (Extensions): Let P_V denote the set of all probability distributions over the finite product space $\mathcal{T}(V)$. Then if V' is a scheme with distribution $\mathbf{p}$, $V' \subseteq V$, the *extension* of $\mathbf{p}'$ to the scheme V, is the set $E^V(\mathbf{p}')$ of all preimages of $\mathbf{p}'$ under the mapping $\pi_{V'}$. Formally, $E^V(\mathbf{p}') = \{\mathbf{p} \in P_V \mid \pi_{V'}(\mathbf{p}) = \mathbf{p}'\}$. Similarly, the *extension* of a *set* of distributions $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_r\}$, is defined to be the intersection of the extensions of its constituent distributions. Formally, $E^V(P) = \bigcap_{\mathbf{p} \in P} E^V(\mathbf{p})$. In particular, if X is a model for a scheme V, the *extension* of V relative to X, $E^V(\pi_X(\mathbf{p}))$, is seen to be the set $E^V(\pi_X(\mathbf{p})) = \{\mathbf{p}' \in P_V \mid \pi_X(\mathbf{p}') = \pi_X(\mathbf{p})\}$ ∎

Example 2.1 Consider the PS $D = (V = \{v_1, v_2, v_3\}, \mathbf{p})$:

v_1	v_2	v_3	$\mathbf{p}(.)$
0	0	0	$0.12 = p_0$
0	0	1	$0.04 = p_1$
0	1	0	$0.32 = p_2$
0	1	1	$0.11 = p_3$
1	0	0	$0.01 = p_4$
1	0	1	$0.00 = p_5$
1	1	0	$0.29 = p_6$
1	1	1	$0.11 = p_7$

Choose a model X of V to be $\{\{v_1, v_2\}, \{v_2, v_3\}, \{v_1, v_3\}\}$. Then $P = \pi_X(\mathbf{p}) = \{\pi_{\{v_1,v_2\}}, \pi_{\{v_2,v_3\}}, \pi_{\{v_1,v_3\}}\}$ is:

v_1	v_2	$\pi_{\{v_1,v_2\}}$
0	0	0.16
0	1	0.43
1	0	0.01
1	1	0.40

v_2	v_3	$\pi_{\{v_2,v_3\}}$
0	0	0.13
0	1	0.04
1	0	0.61
1	1	1.22

v_1	v_3	$\pi_{\{v_1,v_3\}}$
0	0	0.44
0	1	0.15
1	0	0.30
1	1	0.11

It is clear that there are many such $\mathbf{p}$'s that would have produced precisely the same sub-tables. In fact any solution to the set of equations below:

$$\begin{array}{lll} \mathbf{p}_0 + \mathbf{p}_1 = 0.16 & \mathbf{p}_0 + \mathbf{p}_4 = 0.13 & \mathbf{p}_0 + \mathbf{p}_2 = 0.44 \\ \mathbf{p}_2 + \mathbf{p}_3 = 0.43 & \mathbf{p}_1 + \mathbf{p}_5 = 0.04 & \mathbf{p}_1 + \mathbf{p}_3 = 0.15 \\ \mathbf{p}_4 + \mathbf{p}_5 = 0.01 & \mathbf{p}_2 + \mathbf{p}_6 = 0.61 & \mathbf{p}_4 + \mathbf{p}_6 = 0.30 \\ \mathbf{p}_6 + \mathbf{p}_7 = 0.40 & \mathbf{p}_3 + \mathbf{p}_7 = 0.22 & \mathbf{p}_5 + \mathbf{p}_7 = 0.11 \end{array}$$

will be a member of $E^V(P) = E^V(\pi_X(\mathbf{p}))$. The above twelve equations in eight unknowns characterize E^V completely. Note that the system of equations is overdetermined; it is by no means guaranteed that solutions will always exist. When solutions do exist, the set of projections are said to be *consistent*. Equivalently, $E^V(P) = E^V(\pi_X(\mathbf{p})) \neq \emptyset$. ∎

A model X of V, partitions P_V into classes $E^V(\pi_X(\mathbf{p}))$ equivalent with respect to projections onto X. The polyhedral structure of E^V, can be used to show that there exists a unique representative for each class, the maximum entropy distribution, or the join distribution [13]. Formally:

Definition 2.4 (Join): Let $\{D_i = (V_i, \mathbf{p}_i)\}$ be a set of probabilistic systems. Let $V = \cup_i V_i$ and $X = \{V_1, \ldots, V_n\}$ be a model for V. Let $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$. If $E^V(P) \neq \emptyset$ (the distributions are consistent), then define the *join* of the systems $\{D_i\}$, $D_\bowtie = (V, \mathbf{p}_\bowtie)$, such that,

$$H(\mathbf{p}_\bowtie) = \max\{H(\mathbf{p}') \mid \mathbf{p}' \in E(\pi_X(P))\}$$

where $H()$ is the Shannon entropy function. In particular, $\mathbf{p}_\bowtie$ will be referred to as the *probabilistic join* of the distributions $\{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ and is denoted by $\mathbf{p}_\bowtie = \bowtie(\{\mathbf{p}_1, \ldots, \mathbf{p}_n\})$. If $\mathbf{p} = \bowtie(\pi_X(\mathbf{p}))$ then $D = (V, \mathbf{p})$ is said to be *reconstructible* relative to X. ∎

Remark 2.1 To reconstruct via the join operator, is to reconstruct adopting a maximum entropy philosophy. The pros and cons of this approach have been argued extensively in the literature [13, 14, 18, 19]. The $\bowtie$ operator may be thought of as a binary operator acting on a pair of probabilistic systems, and defined in a manner analogous to the classical relational database join operator. It is a remarkable fact that the generalization of the database join operator to probabilistic systems is equivalent to maximum entropy reconstruction.

3 Function Reconstruction

A relational database is a set $RD = \{D_1, D_2, \ldots, D_n\}$ of *relational systems* $D_i = (V_i, r_i)$, where the range of each r_i is the set $\{0, 1\}$. These functions are relatively trivial to reconstruct owing to the fact that, at the cost of introducing one additional variable, for *any* relational system, a non-trivial model exists such that, the system is reconstructible relative to that model, with respect to relational join. Formally,

Theorem 3.1 Let $D = (V, \mathbf{r})$ be a relational system and $X = \{V_1, V_2\}$ be a model for the scheme V. Let v_g be a variable such that $v_g \notin V$, and let v_g take values from a finite set S_g. For the scheme $V' = V \cup \{v_b\}$, let $X' = \{V_1 \cup \{v_b\}, V_2 \cup \{v_b\}\}$ be a model. Then, there exists a relational system $D' = (V', \mathbf{r}')$ such that,

$$\begin{aligned} \mathbf{r}_1{}' &= \pi_{\{V_1\}}(\mathbf{r}') = \pi_{\{V_1\}}(\mathbf{r}) = \mathbf{r}_1 \\ \mathbf{r}_2{}' &= \pi_{\{V_2\}}(\mathbf{r}') = \pi_{\{V_2\}}(\mathbf{r}) = \mathbf{r}_2 \\ \mathbf{r}' &= \bowtie (\mathbf{r}_1{}', \mathbf{r}_2{}') \end{aligned}$$

∎

A proof for the above theorem is given in Appendix A. We adopt the position that $0-1$ functions are intrinsically easy to learn and are interested primarily in the general case of real valued functions. It is assumed that efficient methods exist for the computation of a given $0-1$ function. It may even be the case that a methodology other than that of neural networks be more appropriate for computing functions that take only either of two values.

The situation for probabilistic systems is embodied in the Reconstruction theorem. This theorem asserts that the error distribution, obtained as a difference between the original distribution and the maximum entropy reconstruction of a set of marginals, is always reconstructible. This is done by separating the magnitude of the error distribution ($\mathbf{p}_\epsilon$) from its sign ($\mathbf{p}_\sigma$). The former is reconstructible with respect to a non-trivial model, and the latter is converted to a simple $0-1$ function and is trivially learnable. The practical implications are that we may always parallelize the training process by projecting the original function onto some appropriate model. The sum of the pre-calculated error and the join enables us to retrieve the original distribution. The following definition aids the statement of the theorem. A proof of Theorem 3.2 is given in Appendix B.

Definition 3.1 Let $D = (V, \mathbf{p})$ be a PS, $|V| \geq 2$, X be a model for V, $\mathcal{T}$ be the set of tuples of D, and let $\mathbf{p}_\bowtie = \bowtie (\pi_X(\mathbf{p}))$. $err : \mathcal{T} \rightarrow \Re$ is the *error function of* D, where $err(\alpha) = \mathbf{p}(\alpha) - \mathbf{p}_\bowtie(\alpha)$. Further, if $K = \sum_{\gamma \in \mathcal{T}} |\mathbf{p}(\gamma) - \mathbf{p}_\bowtie(\gamma)| = \sum_{\gamma \in \mathcal{T}} |err(\gamma)|$, define $\mathbf{p}_\epsilon : \mathcal{T} \rightarrow [0,1]$, and $\mathbf{p}_\sigma : \mathcal{T} \rightarrow \{0,1\}$ such that, $\mathbf{p}_\epsilon(\alpha) = |err(\alpha)|/K$, and $\mathbf{p}_\sigma(\alpha) = 0$ if $err(\alpha) \leq 0$, otherwise $\mathbf{p}_\sigma(\alpha) = 1$. ∎

Theorem 3.2 (Reconstruction theorem) Let $D = (V, \mathbf{p})$ be a PS, $|V| \geq 2$, X be a model for V, $\mathbf{p}_{\bowtie} = \bowtie(\pi_X(\mathbf{p}))$, and let $\mathbf{p}_\epsilon$, $\mathbf{p}_\sigma$, and K be as defined above. Then, $\mathbf{p}_\epsilon = \bowtie(\pi_X(\mathbf{p}_\epsilon))$ (i.e. $\mathbf{p}_\epsilon$ is reconstructible). Hence $\mathbf{p}_\epsilon$ is reconstructible and $\mathbf{p}(\alpha) = \mathbf{p}_{\bowtie}(\alpha) + K(2\mathbf{p}_\sigma(\alpha) - 1) \times \mathbf{p}_\epsilon(\alpha) \; \forall \alpha \in \mathcal{T}$. ∎

When is a function reconstructible? An answer to this query is provided by the concept of functional and multivalued dependency. If one variable y, *functionally* dependent on another x, then given x, there is no uncertainity regarding the y. On the other hand if x multidetermines y, knowledge of x may *not* remove all our uncertainity regarding y, but it is definitely the case that knowledge of any other variable, say z, will not remove any *further* uncertainity regarding y. Formally,

Definition 3.2 For a probabilistic system $D = (V, \mathbf{p})$, with $X, Y \subseteq V$, and with the distributions over X and Y obtained by projection of $\mathbf{p}$, we say that X is functionally dependent on Y, denoted $X \rightarrow Y$ iff. $H(Y \mid X) = 0$. Additionally, we say that X multidetermines Y, denoted $X \rightarrow\!\!\rightarrow Y$, iff. $H(Y \mid X) = H(Y \mid V - (Y \cup X))$. Let $Z = V - (X \cup Y)$. The *degrees* of functional and multivalued dependencies of Y on X, $FD(D; X, Y)$, $MVD(D; X, Y)$ respectively, are given by:

$$FD(D; X, Y) = \begin{cases} 1 & \text{if } H(Y) = 0 \\ \dfrac{H(Y) - H(Y \mid X)}{H(Y)} & \text{otherwise} \end{cases}$$

$$MVD(D; X, Y) = \begin{cases} 1 & \text{if } H(Y \mid X) = 0 \\ \dfrac{H(Y \mid \{X \cup Z\})}{H(Y \mid X)} & \text{otherwise} \end{cases}$$

∎

Clearly, if $X \rightarrow Y$ then, $X \rightarrow\!\!\rightarrow Y$. In general however, the converse is *not* true. The answer to our query about the reconstructability of a function with respect to a model, lies in Theorem 3.3, first proved for relational databases by Fagin [9], for probabilistic databases by Cavallo and Pittarelli [8]. and may be generalized for models other than binary ones.

Theorem 3.3 Let $D = (V, \mathbf{p})$ be a PS, $X, Y \subseteq V$ and $X \rightarrow\!\!\rightarrow Y$. Then $\mathbf{p}$ is reconstructible with respect to the binary model $\{X, Y\}$. Formally,

$$\mathbf{p} = \bowtie(\pi_X(\mathbf{p}), \pi_Y(\mathbf{p}))$$ ∎

4 Choosing a Model

It is clear that given a PS $D = (V, \mathbf{p})$, a model X for V has to be chosen judiciously, so that effective use can be made of parallelization. We illustrate how basic concepts from database theory suggest guidelines for such a choice.

A model X may be viewed as a reduced *hypergraph*, basically a collection of some of the incomparable subsets of a set [4]. an important class of reduced hypergraphs is that of the *acyclic* hypergraphs. Hypergraph acyclicity is a non trivial generalization of the familiar concept of a tree (in a graph), and has deep connections many other fields [16]. In hypergraphs, there are three *degrees* of acyclicity, α, β and γ [11]. The main guideline for choosing a model, is to select one that is also acyclic (α, β or γ) [3]. The main reason for choosing an acyclic model is that its presence guarantees many nice properties for the join operator such as *monotonicity* and *consistency*, [10, 11] and *convergence* [7, 18].

Faced with two or more acyclic models, a choice may be made on the basis of their entropies relative to a standard distribution, such as the uniform distribution. Given two distributions $\mathbf{p}_1$ and $\mathbf{p}_2$, define $H(\mathbf{p}_1 \,\|\, \mathbf{p}_2)$ to be the *directed divergence* (cross-entropy, relative entropy) of $\mathbf{p}_1$ with respect to $\mathbf{p}_2$ [12]. Then, if we have two acyclic models, X_i and X_j, and $\mathbf{p}_i = \bowtie (\pi_{X_i}(\mathbf{p}))$, $p_j = \bowtie (\pi_{X_j}(\mathbf{p}))$, select X_i over X_j iff, $H(\mathbf{p}_i \,\|\, u) \geq H(\mathbf{p}_j \| u)$, where u is the uniform distribution. Choosing the smaller value ensures that information loss is minimized in the decomposition process.

For a multivariable function, it may be the case that some variables group together naturally into interdependent blocks. Models that give decompositions along these "natural fissures" are better than those that ignore such interdependencies. These qualitative statements may be made precise by using the dependency measures defined in the last section.

1. If $X, Y \subseteq V$, and if $X \twoheadrightarrow Y$, then decompose the system $D = (V, \mathbf{p})$ onto the model $\{X, Y\}$.

2. If such a multivalued dependency does not exist for a scheme V, then choose a decomposition of V into subsets X and Y, such that $MVD(D;\, X, Y)$ is maximum.

5 Example

The methods outlined earlier are particularly suited for control problems. For example, consider the well known Gantry Crane system control problem. A gantry crane is used to move large parts and assemblies from one location to another on a factory floor; the control system is responsible for the horizontal motion of crane and load. Four key variables may be identified, (see Figure 5), x_1, x_2, the positions of the load and crane, with respect to

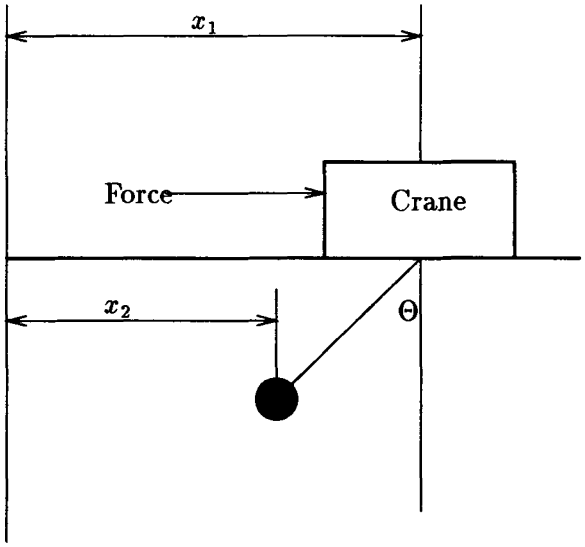

Figure 1: Gantry Crane System

the origin, and v_1 and v_2, the velocities of the load and crane respectively. The problem is to compute the value of the force, for a given set of values of x_1, x_2, v_1 and v_2. It is assumed that it is infeasible to compute F in real-time using the above two equations. Essentially two data sets (one for testing and the other for training, were generated, using the system control equations. All values were normalized to lie in closed-open interval $[0, 1)$. The domains of the input variables were quantized into ten states, by using the following rule for classification:

$$x/10 <= y < (x+1)/10, \quad x \in \{0, \ldots, 9\}$$

i.e. y was said to belong to "class x", if it satisfied the above equation for the given x. Thus if $x_1 = 0.56$ then x_1 would belong to class 0 etc. Note that quantization was done for the input variables alone and *not* for F. In what follows, the training table will be referred to as the *master table* or Table 0. A typical set of entries in the master table is shown below.

x_1	v_1	x_2	v_2	F
1	3	1	1	0.0068925912
1	3	1	1	0.0073849196
.	.	.	.	...
.	.	.	.	.
6	3	6	5	0.0046771155
6	4	6	5	0.0085336845
.	.	.	.	...

The training procedure is schematically outlined in Figure 5. Table 0 was decomposed into Tables 1,2,5,7 and 8. A separate 1-2-1 layer backpropagation network was trained to learn the relation in each of these tables. The testing procedure involved the following,

- For each input (x_1, v_1, x_2, v_2), in testing sample do:

    1. Present (x_1, v_1, x_2) to the Table 1 and Table 7 networks[1]; present (x_2, v_2, v_1) to the Table 2 and Table 8 networks; and similarly present (x_1, v_1, x_2, v_2) to the Table 0 and Table 5 networks.
    2. Compute $\mathbf{p}_r = \mathbf{p}_{\bowtie} + K(2\mathbf{p}_\sigma - 1) \times \bowtie (\mathbf{p}'_\epsilon, \mathbf{p}''_\epsilon)$, the reconstructed output, where K is a normalization factor obtained while training. This gives the output obtained after decomposition.
    3. For each network, the deviation with the actual output value is noted.

The results are shown below. The last row lists the net mean square error of the reconstructed output compares very favorably with that for the Table 0 network.

[1] By "Table 1 " network, we mean the "the network originally trained on Table 1" etc.

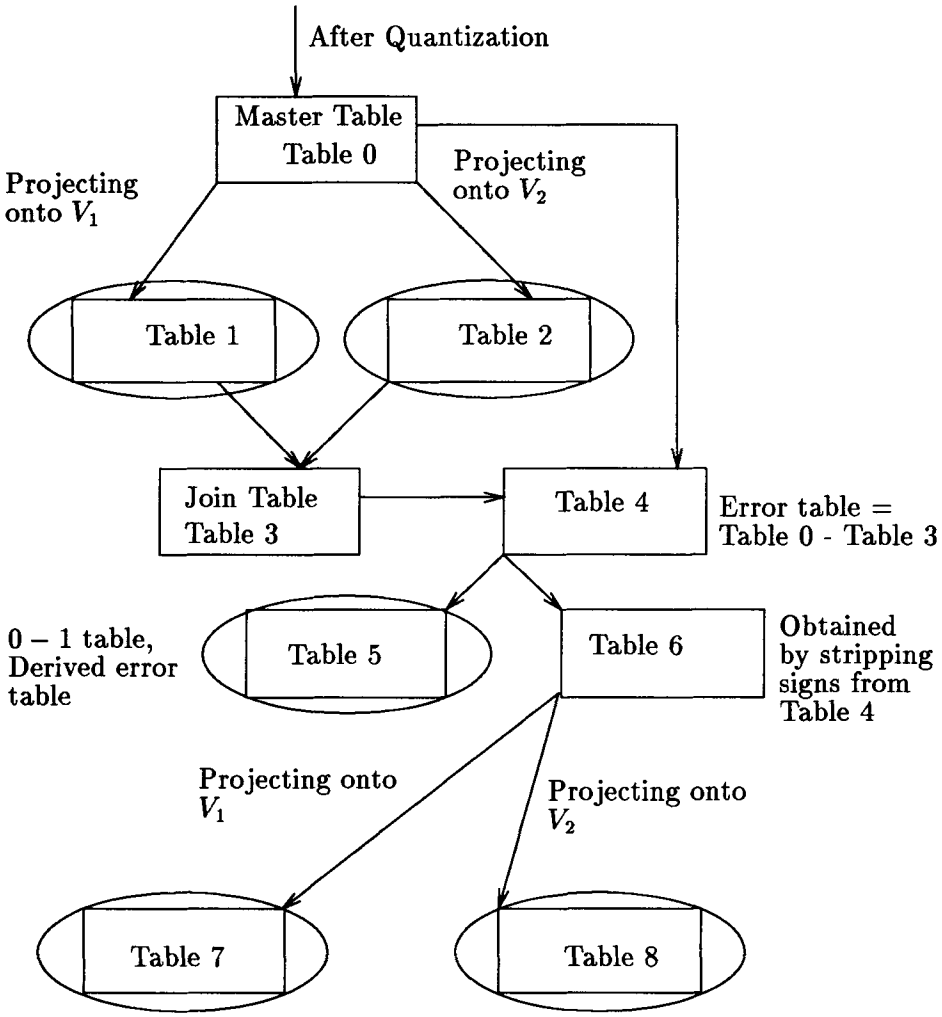

Figure 2: Tables Generated during Training Phase

Tables Trained/Tested	Training		Testing	
	Samples	M.S.E	Samples	M.S.E
Table 0 = $(V, \mathbf{p})$	120	0.000005	481	0.0000759
Table 1 = $(V', \mathbf{p}')$	120	0.000043	110	0.0001470
Table 2 = $(V'', \mathbf{p}'')$	322	0.000003	376	0.0000043
Table 7 = $(V', \mathbf{p}'_\epsilon)$	120	0.000013	94	0.0003080
Table 8 = $(V'', \mathbf{p}''_\epsilon)$	222	0.000008	312	0.0000082
Reconstructed table = $(V, \mathbf{p}_r)$			846	0.0000101

6 Conclusion

The Divide-and-Conquer paradigm has long been recognized as a useful tool for conquering complexity. Our work address the question of when problem decomposition is possible. The Reconstruction theorem (Section 3), asserts that any non reconstructible function may be expressed as a linear combination of reconstructible functions. This suggests a natural input space decomposition method for training neural networks on functions expressed as large tables, drawing on ideas from probabilistic database theory. This approach will be most useful in solving problems characterized by a large number of inputs.

Acknowledgements : This work was supported in part by NSF under CCR-9110812.

Appendix A

Theorem 4.1 Let $D = (V, \mathbf{r})$ be a relational system and $X = \{V_1, V_2\}$ be a model for the scheme V. Let v_g be a variable such that $v_g \notin V$, and let v_g take values from a finite set S_g. For the scheme $V' = V \cup \{v_b\}$, let $X' = \{V_1 \cup \{v_b\}, V_2 \cup \{v_b\}\}$ be a model. Then, there exists a relational system $D' = (V', \mathbf{r}')$ such that,

$$\mathbf{r}_1{}' = \pi_{\{V_1\}}(\mathbf{r}') = \pi_{\{V_1\}}(\mathbf{r}) = \mathbf{r}_1$$
$$\mathbf{r}_2{}' = \pi_{\{V_2\}}(\mathbf{r}') = \pi_{\{V_2\}}(\mathbf{r}) = \mathbf{r}_2$$
$$\mathbf{r}' = \bowtie (\mathbf{r}_1{}', \mathbf{r}_2{}')$$

∎

Proof:
The basic idea is to extend the set of variables V, by one, and construct a reconstructible relational system $D' = (V', \mathbf{r}')$, which yields the same projections as D, including D itself (i.e. $\pi_{\{V\}}(\mathbf{r}') = \mathbf{r}$).

Let $D_1 = (V_1, \pi_{V_1}(\mathbf{r}))$ and $D_2 = (V_1, \pi_{V_1}(\mathbf{r}))$ be the projections of D onto the schemes V_1 and V_2 respectively, and let $D_{\bowtie}$ be their relational join. Also, let $\mathcal{T}$, $\mathcal{T}_1$, $\mathcal{T}_2$, and $\mathcal{T}_{\bowtie}$ be the set of tuples in D, D_1, D_2 and $D_{\bowtie}$ respectively. If $D = (V, \mathbf{r})$ *is* reconstructible with respect to $X = \{V_1, V_2\}$, then there is nothing to prove and we are done. So we may assume that D is not reconstructible with respect to X, relative to relational join. From the definition of relational join, it then follows that $\mathcal{T}_1 \subseteq \mathcal{T}_2$. Let $\Delta\mathcal{T}$ denote the set of such tuples found in $\mathcal{T}_{\bowtie}$, but not in $\mathcal{T}$.

Every tuple t in $\Delta\mathcal{T}$ is formed by the join of a pair, consisting of one tuple, say $t_1 \in \mathcal{T}_1$, from D_1, and one from D_2, say $t_2 \in \mathcal{T}_2$. Tuples t_1 and t_2 agree on the set of attributes belonging to the set $V_1 \cap V_2$. If they didn't so agree, then the tuple t would not have been created while taking the join of D_1 and D_2. This suggests a simple method to prevent the creation of unwanted tuples in $D_{\bowtie}$.

Construct the systems $D_1' = (V_1 \cup \{v_g\}, \mathbf{r}_1')$ and $D_2' = (V_2 \cup \{v_g\}, \mathbf{r}_2')$ and let $D' = (V \cup \{v_g\}, \mathbf{r}')$ be the join of the systems D_1' and D_2'. Let $t \notin \Delta\mathcal{T}$, i.e. $t \in \mathcal{T}_{\bowtie} \cap \mathcal{T}$. Now $t = t_1 \bowtie t_2$ for some $t_1 \in \mathcal{T}_1$, and $t_2 \in \mathcal{T}_2$. Now, for every such t_1 and t_2, there exists $t_1' \in \mathcal{T}_1'$, $t_2' \in \mathcal{T}_2'$ with $t_1'[V_1] = t_1$, $t_2'[V_2] = t_2$, and $t_1'[\{v_g\}] = t_2'[\{v_g\}]$ (see Section 2 for notation). In other words, the tuples t_1' and t_2' agree on the value of v_g, and t_1' agrees with t_1 on all other variables. Similarly, t_2' agrees with t_2 on all variables

other than v_g. By choosing a different value for v_g for each other pair t_1 and t_2, we can ensure that, $\pi_{\{V_1\}}(\mathbf{r}'_1) = \mathbf{r}_1$, $\pi_{\{V_2\}}(\mathbf{r}'_2) = \mathbf{r}_2$ and $\pi_{\{V\}}(\mathbf{r}') = \mathbf{r}$ ∎.

Example A.1 This example illustrates the procedure described in the proof. Consider the relational database $D = (V, \mathbf{r})$ given by the table below.

x	y	z	$\mathbf{r}$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Upon projecting the above system onto the model $X = \{V_1 V_2\}$, where $V_1 = \{x, y\}$ and $V_2 = \{y, z\}$ we get,

x	y	$\mathbf{r}_1$
0	0	1
0	1	1
1	0	1
1	1	1

y	z	$\mathbf{r}_2$
0	0	1
0	1	0
1	0	1
1	1	0

The relational join of $D_1 = (\{x, y\}, \mathbf{r}_1)$, and $D_2 = (\{y, z\}, \mathbf{r}_2)$ yields,

x	y	z	$\mathbf{r}_{\bowtie}$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1 ←
1	1	1	1

The tuple marked with an arrow is not present in the original system D. Hence D is not reconstructible. We note that the problem stems from the fact that the tuple $t_1 = 11$ in D_1 joins with tuple $t_2 = 10$ in D_2. All the following combinations however are required viz. ,

x	y		y	z
0	0	$\times$	0	0
0	1	$\times$	1	0
0	1	$\times$	1	1
1	1	$\times$	1	1

Introduce a new variable w, such that,

x	y	w	w	y	z
0	0	0	0	0	0
0	1	1	1	1	0
0	1	2	2	1	1
1	1	3	3	1	1

Or, defining two new systems $D_1' = (\{x, y, w\}, \mathbf{r}_1')$ and $D_2' = (\{y, z, w\}, \mathbf{r}_2')$,

x	y	w	$\mathbf{r}_1'$
0	0	0	1
0	1	1	1
0	1	2	1
1	1	3	1

w	y	z	$\mathbf{r}_2'$
0	0	0	1
1	1	0	1
2	1	1	1
3	1	1	1

We have shown only the nonzero tuples in both D_1' and D_2'. The join of D_1' and D_2', $D_{\bowtie}' = (\{x, y, w, z\}, \mathbf{r}_{\bowtie}')$, yields,

x	y	w	z	$\mathbf{r}_{\bowtie}'$
0	0	0	0	1
0	1	1	0	1
0	1	2	1	1
1	1	3	1	1

Again all other tuples α, have $\mathbf{r}'(\alpha) = 0$. It is easy to see that $D' = D_1' \bowtie D_2'$ yields D when projected onto $\{V\}$ and also yields D_1 and D_2 when projected onto V_1 and V_2. By construction D' is reconstructible and may be used in place of D. Note that w here takes values from a set $\{0, 1, 2, 3\}$. It is not too difficult to see that in this case, we could have done with w taking values from the smaller set $\{0, 1, 2\}$. It is fairly straightforward to extend the procedure. ∎.

Appendix B

Theorem 5.1 (**Reconstruction theorem**) Let $D = (V, \mathbf{p})$ be a PS, $|V| \geq 2$, X be a model for V, $\mathbf{p}_{\bowtie} = \bowtie (\pi_X(\mathbf{p}))$, and let $\mathbf{p}_\epsilon$, $\mathbf{p}_\sigma$, and K be the derived error functions, and the normalization factor of D respectively (see Definition 3.1). Then,

$$\mathbf{p}_\epsilon = \bowtie (\pi_X(\mathbf{p}_\epsilon))$$

i.e. $\mathbf{p}_\epsilon$ is reconstructible with relative to X. In particular,

$$\mathbf{p}(\alpha) = \mathbf{p}_{\bowtie}(\alpha) + K(2\mathbf{p}_\sigma(\alpha) - 1) \times \mathbf{p}_\epsilon(\alpha) \qquad \forall \alpha \in \mathcal{T}$$

∎

Proof:
From Theorem 3.1, we know that $\mathbf{p}_\sigma$, being a simple $0-1$ relational function is reconstructible, although an additonal variable may have to be introduced to effect this. The issue at hand is whether $\mathbf{p}_\epsilon$ is also reconstructible. This is quite easy to demonstrate. Suppose it is the case that,

$$\bowtie (\pi_X(\mathbf{p}_\epsilon)) = \mathbf{p}'_\epsilon \neq \mathbf{p}_\epsilon \tag{6.1}$$

Since $\mathbf{p}'_\epsilon$ is the join of the probabilistic database $\pi_X(\mathbf{p}_\epsilon)$, it must satisfy the consistency requirement that,

$$\pi_X(\mathbf{p}'_\epsilon) = \pi_X(\mathbf{p}_\epsilon) \tag{6.2}$$

This in turn implies that there exists at least one distribution $\mathbf{p}' \neq \mathbf{p}_{\bowtie}$, such that,

$$\mathbf{p}'_\epsilon = k_1 |\mathbf{p}' - \mathbf{p}| \tag{6.3}$$

where $k_1 = \sum_{\alpha \in \mathcal{T}} |\mathbf{p}'(\alpha) - \mathbf{p}(\alpha)|$. Since $\bowtie (\pi_X(\mathbf{p}'_\epsilon)) = \mathbf{p}'_\epsilon$, we have (from Eqns. 6.1,6.3),

$$\bowtie (\pi_X(\mathbf{p}_\epsilon)) = \bowtie (\pi_X(\mathbf{p}'_\epsilon)) = \bowtie (\pi_X(k_1|\mathbf{p}' - \mathbf{p}|)) \tag{6.4}$$

Noting that, (1) $\mathbf{p}_\epsilon = k_2 |\mathbf{p}_{\bowtie} - \mathbf{p}|$, ($k_2 = \sum_{\alpha \in \mathcal{T}} |\mathbf{p}_{\bowtie}(\alpha) - \mathbf{p}(\alpha)|$), and also that, (2) $\mathbf{p}'_\epsilon$ is a maximum entropy distribution (from Eqns. 6.1, 6.2), we get,

$$H(k_1|\mathbf{p}' - \mathbf{p}|) > H(k_2|\mathbf{p}_{\bowtie} - \mathbf{p}|) \tag{6.5}$$

It is well known that [12], if $\mathbf{q}$ is an arbitrary finite probability distribution of say n variables, and $f()$ a function, with domain and range the set of all finite probability distributions on n variables (so that $f(\mathbf{q})$ is also a distribution on n variables), then,

$$H(\mathbf{q}) = H(f(\mathbf{q})) + H(\mathbf{q}|f(\mathbf{q}))$$

(Roughly, this translates to the statement that in the absense of no new observation, uncertainity cannot be removed by mere transformation of the distribution alone).

Define $f_i(\mathbf{q}) = |k_i(\mathbf{q} - \mathbf{p})|$ where $\mathbf{q}$ is some arbitrary distribution on n variables, $\mathbf{p}$ is the original non reconstructible function, and k_i is a normalizing factor and depends on both $\mathbf{q}$ as well as $\mathbf{p}$. Then,

$$\begin{aligned} H(\mathbf{p}') &= H(f_1(\mathbf{p}')) + H(\mathbf{p}'|f_1(\mathbf{p}')) \\ &= H(|k_1(\mathbf{p}' - \mathbf{p}|) + H(\mathbf{p}'|(k_1|\mathbf{p}' - \mathbf{p}|)) \end{aligned} \tag{6.6}$$

$$\begin{aligned} H(\mathbf{p}_M) &= H(f_2(\mathbf{p}_M)) + H(\mathbf{p}_M|f_2(\mathbf{p}_M)) \\ &= H(|k_2(\mathbf{p}_M - \mathbf{p}|) + H(\mathbf{p}_M|(k_2|\mathbf{p}_M - \mathbf{p}|)) \end{aligned} \tag{6.7}$$

But knowledge of $k_1|\mathbf{p}' - \mathbf{p}|$ determines $\mathbf{p}'$ ($\mathbf{p}$ is fixed and $\mathbf{p}'$ has to be non negative),

$$H(\mathbf{p}'|f_1(\mathbf{p}')) = 0 \tag{6.8}$$

and similarly,

$$H(\mathbf{p}_M|f_2(\mathbf{p}_M)) = 0 \tag{6.9}$$

Then Eqns. 6.6,6.7 reduce to,

$$\begin{aligned} H(\mathbf{p}') &= H(f_1(\mathbf{p}')) = H(|k_1(\mathbf{p}' - \mathbf{p})|) \\ H(\mathbf{p}_M) &= H(f_2(\mathbf{p}_M)) = H(|k_2(\mathbf{p}_M - \mathbf{p})|) \end{aligned} \tag{6.10}$$

Comparing Eqn 6.10 with Eqn. 6.5, it is seen that,

$$H(\mathbf{p}_M) < H(\mathbf{p}')$$

This immediately implies a contradiction, since we must have:

$$H(\mathbf{p}_M) > H(\mathbf{p}')$$

(because of the definitions of $\mathbf{p}_M$), Hence such a distribution $\mathbf{p}'$ cannot exist and the theorem follows. ∎

References

[1] W. R. Ashby. *Constraint Analysis of Many-Dimensional Relations*, volume 2. 1965.

[2] H. Barabará, H. Garcia-Molina, and D. Porter. The management of probabilistic data. IEEE Trans. Know. and Data Eng. (to appear).

[3] C. Beeri, R. Fagin, D. Maier, and M Yannakis. On the desirability of acyclic database schemes. *J. of the ACM*, 30(3):1983, 479-513.

[4] C. Berge. *Graphs and Hypergraphs.* North-Holland, Amsterdam, 1973.

[5] Martin Bischsel and Peter Seitz. Minimum class entropy: A maximum information approach to layer networks. *Neural Networks*, 2:133–141, 1989.

[6] R. Cavallo and G. Klir. Reconstructability analysis of multi-dimensional relations: a theoretical basis for computer-aided determination of acceptable system models. *Int. J. General Systems*, 5(3):143–171, 1979.

[7] R. Cavallo and G. Klir. Reconstructability analysis: Evaluation of reconstruction hypotheses. *Int. J. General Systems*, 7(1):7–32, 1981.

[8] R. Cavallo and M. Pittarelli. The theory of probabilistic databases. *Proc. of the 13th VLDB Conf.*, pages 71–81, 1987.

[9] R. Fagin. Mutlivalued dependencies and a new formal form for relational databases. *ACM Trans. on Database Systems*, 2(3):262–278, 1977.

[10] R. Fagin. *Acyclic Database Schemes (of Various Degrees): A Painless Introduction*, volume LNCS 159 of *Trees in Algebra and Programming*, pages 65–89. Springer-Verlag, 1983.

[11] R. Fagin. Degrees of acyclicity for hypergraphs and relational database schemes. *J. of the ACM*, 30(3):514–550, 1983.

[12] M. Robert Gray. *Entropy and Information Theory.* Springer-Verlag, New York, 1990.

[13] E. T. Jaynes. Prior information and ambiguity in inverse problems. In D. McLaughlin, editor, *Inverse Problems*, pages 151–166. SIAM-AMS, 14, 1984.

[14] J. MacQueen and J Marschak. Partial knowledge, entropy and estimation. *Proc. Natl. Acad. Sci.*, 72:3819–3824, 1975.

[15] M. Marchand, M. Golea, and P. Ruja'an. A convergence theorem for sequential learning in two layer perceptrons. *Europhysics Letters*, 11:487–492, 1990.

[16] A. Menon. Database schemes: Formalisms and applications. Master's thesis, SUNY Inst. of Technology, 1991.

[17] M. Me'zard and J.P. Nadal. Learning in feedforwad layered networks: The tiling algorithm. *Journal of Physics A*, 22:2191 – 2204, 1989.

[18] M. Pittarelli. Uncertainity and estimation in reconstructability analysis. *Int. J. General Systems*, 15:1–58, 1988.

[19] T. Seidenfeld. Entropy and uncertainity. *Philosophy of Science*, 53:467–491, 1986.