Journal of Official Statistics, Vol. 11, No. 4, 1995, pp. 391–411

Editing Statistical Records by Neural Networks

Svein Nordbotten[1]

Editing records to ensure quality in statistical surveys is an expensive and time-consuming process. Since the introduction of computers in statistical processing, development and use of automated editing have been important objectives. In this article, editing is reformulated as a neural network problem. The use of such a network is demonstrated and a preliminary evaluation is presented.

Key words: Editing; imputation; neural networks; synthetic data generation.

1. Introduction

The aim of editing in statistical surveys is to check each record to determine whether it is acceptable and if not to take action to make it acceptable. When computers are used in the process, this is called automated editing.

Programmed electronic computers were introduced in official statistics more than 40 years ago. The first international group for exchange of experience and cooperation was the UN/ECE Working Group on EDP. One of the first problems discussed was automated editing (Nordbotten 1963). Three steps in editing were identified, (1) determine which records are unacceptable, (2) determine which values of such records need to be corrected and (3) correct the values. Different approaches such as unconditional and conditional editing, cold deck, hot deck, and functional correction were discussed.

Fellegi and Holt (1976) formalized editing and imputation problems. They showed that any edit statement could be expressed as a series of edits in the normal form and proposed a method which could identify the minimal set of values in a record which need to be changed to satisfy all edits. This work also developed methods for sequential and joint imputation of values for the minimum set and has provided a framework for the development and implementation of a number of methods for automated editing of survey data during the last two decades.

Fellegi and Holt assumed that the edit specification would be given explicitly by subject matter experts, that as few values as possible should be changed and that the imputation is of the hot deck type. Recent progress in the field of artificial neural

[1] Department of Information Science, University of Bergen, N-5020 Bergen, Norway.
Acknowledgment: This research was carried out in 1993 during a sabbatical year at the Department of Information and Computer Sciences, University of Hawaii at Manoa, funded by the University of Bergen. I am grateful to both universities for their support. I also wish to thank my wife, Associate Professor Joan C. Nordbotten, and an anonymous referee for valuable advice.

networks and continued development in capacity and speed of computers indicate that editing can be reformulated as neural networks which can be trained to edit and impute from a sample of records edited by experts rather than use of explicit edit and imputation rules.

The purpose of this article is to reformulate editing as a neural network problem, explore the application of neural networks as an approach to automated editing and present preliminary conclusions about the feasibility of using such methods in survey processing.

2. New Paradigms

Work on what has later become known as artificial intelligence also started in the early 1950s. Two different paradigms emerged (Sowa 1984).

According to the first paradigm, Artificial Intelligence (AI), a powerful inference engine or computer was envisaged based on the principles of symbolic logic. The inference engine should process problems based on logical principles and human knowledge represented in a machine accessible knowledge base, frequently as rules. For these systems it was an essential requirement that they be able to present an explanation for the results generated. Around 1970, implemented application versions of these systems became known as expert systems. An acknowledged bottleneck problem for practical application of expert systems was, and still is, how to capture the human knowledge needed as rules in the knowledge base.

The second paradigm, Artificial Neural Networks (ANN) used the human brain as a starting point. A simple model of neurons was the basis of this approach, which showed how different tasks could be solved by connecting a number of neurons in networks. Knowledge in these networks is represented as the strengths of connections between neurons, and the processing is considered distributed in parallel to a number of simple processing units rather than to a complex central unit. Compared to the models of AI, Artificial Neural Networks do not represent knowledge in a form which can easily be interpreted, and an ANN cannot readily explain the rationality of its results. Development of neural networks has made great progress during the last decade. Learning algorithms for neural network models were developed early and reduced the knowledge acquisition bottleneck. Important milestone contributions in the development of ANN are collected in two impressive volumes by Anderson and Rosenfeld (1988) and Anderson, Pellionisz, and Rosenfeld (1990).

The common aim of both the above paradigms is to capture the knowledge of experts and represent it in computerized systems to solve non-trivial mental tasks requiring expert knowledge. This is exactly the problem in the control and imputation of individual survey records. If the knowledge of an editing expert can be "implanted" into control and imputation systems, then automated editing of individual statistical records may be carried out with the same quality as if scrutinized by a human expert.

Attempts have also been made to apply methods similar to those of artificial intelligence in automated editing. Early in the 1960s, the automated editing of the U.S. Census of Agriculture was specified in the form of decision tables which was a

kind of rule based system approach. The present author used a similar rule based system in simulating automated editing of individual records for enterprise statistics (Nordbotten 1965).

More recently, Creezy, Masand, Smith, and Waltz (1992) have reported application of an approach related to ANN for classification of units in the U.S. census, while Schafer, Khane, and Ezzali-Rice (1993) reported on multiple imputation of missing data by an approach related to ours.

A highly relevant paper in the present context is a proposal by Roddick (1993) about an edit system based on neural networks. The ideas proposed by Roddick are similar to those in the present article.

It is obvious that the control and imputation of individual records can be modelled as an ANN and benefit from recent developments in this field. In the following sections, editing as a neural network problem is discussed and some of the potential of this approach for statistical processing investigated.

3. Reformulation

3.1. The problem

Let us consider a statistical survey comprising M statistical units with A attributes observed and recorded. Each attribute has a finite number of mutually exclusive class categories, C_a. A record for each unit is represented as a binary vector with $N = \Sigma C_a$ binary elements, one for each category of the attributes.

Assume that the records which we ideally want to obtain can be represented by rows in the "true" matrix $\mathbf{T} = \{t[1, 1], \ldots, t[M, N]\}$. In collection and pre-processing of data, elements may be lost or incorrectly recorded for a number of reasons. The observations obtained can be depicted by the matrix $\mathbf{R}$ of actual raw data. We assume that $\mathbf{R}$ has the same dimensionality as $\mathbf{T}$.

Human editing experts have some knowledge about structural relations between the columns of $\mathbf{T}$. They know that some row patterns are more frequent than others, while some patterns cannot occur because of logical or other reasons. Experts utilize this knowledge to transform the matrix $\mathbf{R}$ to the desired matrix $\mathbf{T}$. At this stage, we discard the fact that even the best expert knowledge is incomplete and that the transformation of $\mathbf{R}$ can at best be an approximation to $\mathbf{T}$.

We assume that $\mathbf{R}_1$ and $\mathbf{R}_2$ are two random and exhaustive partitions with sizes M_1 and M_2, from the rows of $\mathbf{R}$ such that $M_1 + M_2 = M$ and large enough to produce plausible results of the tests introduced. Assume that the row samples $\mathbf{R}_1$ and $\mathbf{R}_2$ both are edited by expert editors with matrices $\mathbf{T}_1$ and $\mathbf{T}_2$ as the final results. The set of pairs of corresponding rows in $\mathbf{R}_1$ and $\mathbf{T}_1$ is called the training set while the set of pairs of corresponding rows in $\mathbf{R}_2$ and $\mathbf{T}_2$ will be called the test set.

The tasks we want to investigate are:

Can a computer implemented model of a human expert be constructed as an ANN? Can such a model be trained from the pair of matrices $\mathbf{R}_1$ *and* $\mathbf{T}_1$ *to transform matrix* $\mathbf{R}_2$ *to a new matrix* $\mathbf{O}_2$ *which will not differ significantly from matrix* $\mathbf{T}_2$*?*

3.2. *The neural network paradigm*

A large number of different ANN models have been developed. We concentrate on a class called *feed-forward* models. Recurrent models are another class which we do not discuss in this connection. This should not lead to the false conclusion that recurrent models are considered irrelevant for the problem considered.

Figure 1 illustrates the main components in a simple, *single-layer* feed-forward neural network model. We will look at this model in relation to our editing problem. The model consists of a set of input sources through which the network gets the input vectors. An input vector corresponds to a row of **R** and is denoted $\mathbf{r} = \{r[1], \ldots, r[i], \ldots, r[N]\}$. Processing is carried out in parallel by a layer of simple processing units called neurons which produce an output vector $\mathbf{o} = \{o[1], \ldots o[j], \ldots, o[N]\}$, a row in the output matrix **O**.

All input sources are connected to all neurons. The strengths of the connections are represented by the weights, $\mathbf{w}[i,j]$, between the input source i and the neuron j. The total input to a neuron, called the net input, is the scalar product sum of the input vector multiplied with the weight vector of the connections to the respective neuron

$$net[j] = \sum_i w[i,j] * r[i].$$

Each neuron can be considered a small, simple processor. Its processing is represented by a transfer function which takes the net input as its argument and provides the value of the function as the output of the neuron to the output vector. We use the sigmoid function which has many attractive properties as a transfer function

$$o[j] = 1/(1 + e^{-net[j]}).$$

An ANN model is used in two modes. In the learning mode, the model adjusts itself to the input and true target vectors of the training matrices in an iterative cycling through the rows of the matrices. In the second mode, the trained model is used to compute output vectors as a function of input vectors which were not used in training. If a set of true output vectors also exists for this set of input vectors, the trained model can be tested and evaluated by comparing the computed output vectors with the corresponding true vectors. Typically, the training of ANN models may be very

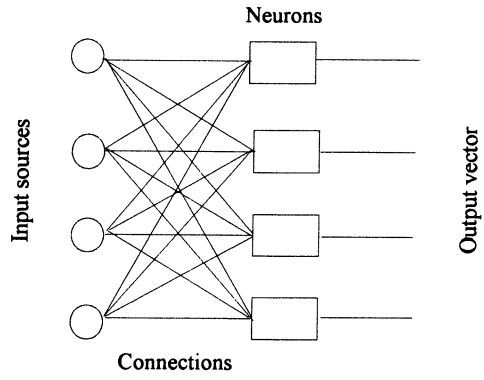

Fig. 1 Single-layer neural network

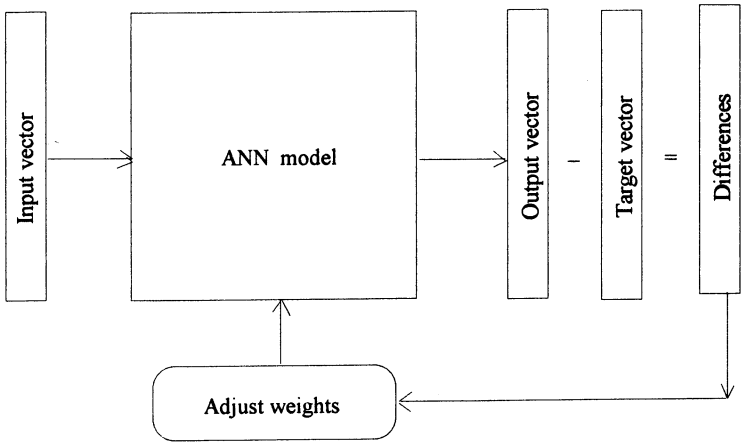

Fig. 2 A learning cycle in the ANN model

resource and time-consuming because of the adjustment calculations and the many iterations frequently required. On the other hand, the use of a trained model is fast and requires relatively limited resources per input vector.

Our editing model learns by observing the examples of pairs of records before and after expert editing, i.e., rows from $\mathbf{R}_1$ and the corresponding from $\mathbf{T}_1$ matrix as indicated in Figure 2. For each row of the matrices, the connection weights are adjusted based on the differences between computed output and true records. The adjustment is done in such a way that the differences are reduced. This adjustment is carried out for each row in the training set, and repeated for the whole set until some stopping condition is satisfied. The weights represent the memory of the model. Before training starts, the weights are given small random initial values. During training, they approach values which represent knowledge contained in the training set. It is difficult, a priori, to state the conditions for successful learning, and, if a model has been trained with success, to give an interpretation to the individual weights with reference to the empirical world.

The adjustment is done according to learning functions. We use learning function expressed by

$$w[i,j]' = w[i,j] + \rho^* \Delta[j]^* r[i]$$

where ρ is a pre-set learning rate which may be kept constant or systematically changed during the iterations. It usually has a value between 0 and 1; $w[i,j]$ and $\mathbf{w}[i,j]'$ are weight values before and after an adjustment based on a presentation of a training pair of records and

$$\Delta[j] = (t[j] - o[j]).$$

After each completed iteration, an evaluation of all outputs from the training input records is carried out. If the result is within pre-set tolerance limits, the learning is considered complete and terminated.

Because of the limited possibilities of a single-layer model (as indicated in Figure 1) to adjust to training sets of complicated records, we shall use a *two-layer* model in this

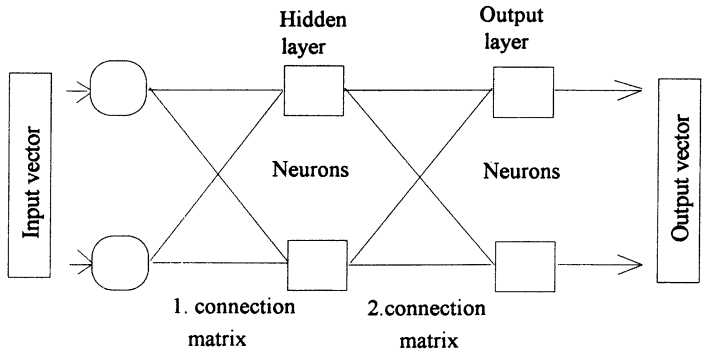

Fig. 3 Two-layer network

investigation. Two-layer ANN models are formed by considering the output vector of one layer to be the input source of a second layer. A multi-layer model has great ability to learn, but usually requires a considerable learning time. Layers of neurons in a multi-layer model, except for the last producing the final output, are called hidden layers, because their output cannot be compared with observations in the application world. A typical two-layer model is indicated in Figure 3.

The two-layer model also has a second connection weight matrix between the output of the neurons in the hidden layer of neurons, and the second layer of neurons. Useful learning algorithms for these multi-layer models have been available for less than 10 years. The Back Propagation (BP) learning algorithm which is outlined and used below, was introduced by Rumelhart and McClelland (1986), but the model was suggested by Werbos in a paper on generalized multivariate regression (Werbos 1974).

The learning functions of the BP algorithm for the two-layer model are generalizations of the single-layer functions described above. They can be presented as

$$\Delta[j] = o[j]^*(1 - o[j])^*(t[j] - o[j]) \quad \text{for adjusting the second matrix}$$

and

$$\Delta[j] = o[j]^*(1 - o[j])^*\Sigma w[j,k]^*\Delta[k] \quad \text{for adjusting the first matrix.}$$

In adjusting the weights of the first matrix, the index k refers to the neurons of the second layer and $\Delta[k]$ to the deltas used for changing the weights of the second matrix.

The above can easily be generalized to higher-layer networks. Empirical experience indicates, however, that a two-layer network performs as well as a higher level network for many applications. We concentrate therefore on a two-layer model because of reduced computational resource requirements.

Three problems remain, specification of initial values for the weights, learning rate and the number of neurons of the hidden layer. At the present stage of development, little supporting theory exists. Selections of values used in this study have been made mainly based on experience and experimentation.

The ANN must justify its use. First, we need to test that the specified model is able

to extract and learn editing knowledge from the training set. Second, the trained model has to prove that it can also carry out satisfactory editing of survey records that it has not previously been presented with.

A relative tolerance contrast V between 0 and 1, must be specified. If the algebraic difference between the true target value and the computed output value for any element of $\mathbf{R}_1$ is less than V after a cycle of presentations of the training records, the model is considered to be trained satisfactorily. Frequently a model is unable to learn the training set completely. Still, it may be useful.

The main test of a model is its ability to edit a new and unknown file of records. For editing models, it is reasonable to compare statistical tables produced from the raw, edited and true matrices in a final evaluation.

4. Empirical Experiments

The editing performance of the model introduced in the previous section can be studied in two different ways. One approach requires access to both raw records and matching records edited by experts from a real survey. This may be impossible for several reasons. The records may not exist in a form which is suitable or there may be legal reasons which prevent a researcher access to the data, and so on. Finally, uncertainty about the quality of the edited records may prevent a researcher from using available edited records.

The second approach to studying the editing performance of the model is to use synthetic data generated in such a way that they represent characteristics similar to the survey objects for which the model is considered. Concern about accessibility and quality of edited individual records is circumvented in this approach.

Since the aim of this article is to introduce and demonstrate a new model for editing rather than testing a method for a particular survey or census, we used the second approach. The data generated have not been derived from any particular survey, but were designed with a demographic or labour force survey in mind. Final decisions about applying models of the type discussed in real survey work should therefore require more specific testing on real data from target surveys. The process outline for the experiments is shown in Figure 4.

4.1. Generation of raw and edited data records

To generate the required synthetic data, an imaginary structure of a person with nine *attributes* was first defined. Each attribute was specified as a set of mutually exclusive categories. Attributes with continuous value ranges were approximated by categories as for the attributes age and income shown below

1. Sex: {male, female}
2. Age: {−19, 20–49, 50–69, 70+}
3. Marital status: {unmarried, married}
4. Region: {city, coast, inland, mountains}
5. Children born: {0, 1, 2, 3, 4+}
6. Education: {7, 8–12, 13+}

7. Industry: {None, agriculture, fishery, manufacturing, trade, services}
8. Employment status: {employed, unemployed}
9. Income: {−99, 100–249, 250–499, 500+}

The properties of an individual were represented by a record with one binary field for each category. In our application, there were 32 different categories. Each attribute must have two or more exclusive categories of which only one can be marked by 1, the remaining must all be 0.

The attributes of an individual are not independent of each other. A set of structural relations exists among the attributes. It is acquired knowledge about this structure which makes it possible for editing experts to scrutinize and correct identified errors in the individual record.

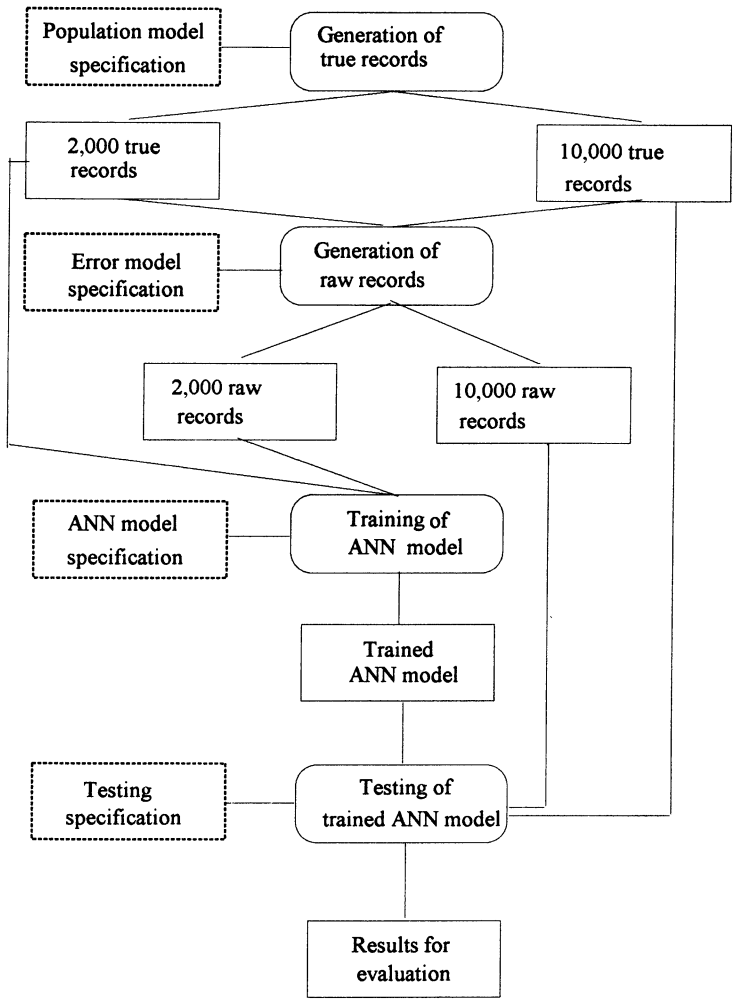

Fig. 4 Outline of the experiments

The knowledge about the structure was expressed as a stochastic model consisting of a set of probability rules. A rule states the conditions, if any, for assigning probabilities to the individual categories. The rules used for generating synthetic, individual records are listed in List 1 in the Appendix.

In real surveys, raw records must be transformed to true records applying some background knowledge about the general structure of the objects concerned and about the error risks associated with collection and pre-processing. In our experiments we went the opposite way. From the generated true records, we introduced random errors to obtain the raw files. Two error types, *non-response* and *interchange* errors, were introduced. The risk probabilities for the two types of errors were specified by rules similar to those used for generating the true population records. These rules, the stochastic error model, are presented in List 2 in the Appendix.

The reason for introducing only these two error types is that they are supposed to be the most frequent. The non-response error identifies itself by leaving all category fields for an attribute blank with an obvious need for imputation. The interchange error, on the other hand, leaves no unique identification. The error identification, as well as the correction, must in this case rely completely on background knowledge. A third type of error, *overlap* error not considered in this article, is characterized by more than one category mark for an attribute. It is similar to non-response error because it identifies itself. It is different from interchange error because one of the marked categories may be true.

The two models, the statistical unit and the error generation models, were implemented in a computer program developed for generation of these individual records. We wanted to study the effects of each of the two error types separately as well as their joint effects. Three different sets of raw records, *S*-, *C*- and *R-files*, were therefore generated. The first set, the *S-files*, included records which were subjected only to the risks of non-response errors, while the second set, the *C-files* were subjected only to interchange error risks. The last set, comprising the *R-files*, included records which were exposed to both non-response error and interchange error risks. Each of the three sets has two files, one with 2,000 and a second independent file with 10,000 records.

The *S-files* were generated from the true *T-files* by exposing each set of attribute fields in all records of the true files to the risk of non-response according to the error model. A random process determined whether a non-response error would occur, and if so all the category fields of the current attribute were marked 0 in the *S-file* being generated. Similarly, the *C-files* were generated with the difference that if the random process gave an error outcome, the marked category of the current attribute was set equal to 0, and another category was randomly chosen and marked 1.

To obtain *R-files* containing both errors, all attributes in records of the generated *T-files* were exposed to a risk for non-response. If the outcome of the random process was a non-response error, all categories of the attribute being processed were set equal to 0 in the *R-file*. In this case, an interchange error could obviously not occur. If the attribute was not determined to have non-response error, the attribute was exposed to a risk of an interchange error. If the random process resulted in such an error, the

mark of the true attribute category was substituted with the mark of another randomly selected category in the corresponding attribute of the *R-file*.

4.2. *Editing experiments*

The purpose of this study was to investigate whether a neural network model could be used for automated editing of individual records from a statistical survey. It is important to bear in mind that manual editing of records without access to the corresponding real statistical units is possible only if the editing experts possess some background knowledge, not about each individual object, but about some general structure shared by the units, and about the general mechanisms of errors occuring during data capture.

Three different experiments were carried out:

Experiment I
The objective of the first experiment was to investigate a situation in which only non-response errors were present. How well could the ANN model learn the background structure from the training files and apply this knowledge for imputing missing values in the test file? In the experiment, the 2,000 pairs of S_1- and the T_1-records were used to train the ANN model to impute missing attribute values. The trained model was then used to impute non-response in the 10,000-record S_2-file using the T_2-file as a control reference.

Experiment II
A situation with interchange errors presented a greater challenge than the non-response errors. In the experiment we wanted to investigate whether the model could be trained to learn from the C_1- and T_1-files to recognize and correct interchange errors. Even a highly experienced editing expert cannot be expected to carry out this task without mistakes. There will be identical records, some of which are correct while others are the results of interchange errors. The trained model was subsequently tested on the 10,000 C_2-records to see how well it performed on an unfamiliar record set.

Experiment III
The third experiment focused on a situation in which both non-response and interchange errors occur. The model was trained to recognize the relationships between the raw records of R_1-file and the corresponding true records of T_1-file. The result of the training was tested against the 10,000 records of R_2-file and checked against the corresponding records of T_2-file.

The editing model was implemented using a standard commercial program for neural networks. Before running this program, the data generated (Section 4.1) were standardized. All attributes were allocated 6 category fields even though not all attributes used all fields. For example, the first attribute, sex, only needed the first 2 of the assigned 6 fields. The remaining 4 were always set equal to 0. For attribute 7, industry, on the other hand, all 6 fields were needed and used. Each standardized record of the *S*-, *C*-, *R*-, and *T*-files then contained 54 fields.

There exists little theoretical support for selection of model type, number of layers, learning rate, neurons in hidden layers, tolerances and maximum number of training cycles. The back propagation algorithm has been in general use for nearly 10 years and some heuristic knowledge has been acquired and made available. Our specifications were based on experience reported in the literature and on personal experience in using this type of network.

We used changing learning rates in all experiments. The rate ρ changed linearly from 1.0 to 0.1 as the percentage of records correctly learned by the models varied from 0% to 100%. The tolerance factor V was set to 0.2 during the training, and the number of neurons in the hidden layer was specified to 300 in all experiments. The stopping rules for repeating presentations were that all training records had been correctly learned or that the number of cycles reached 300.

With 54 element input and output vectors and 300 hidden neurons, up to 32,000 connections had to be evaluated, and adjusted if necessary for each training record presentation. In each training cycle of 2,000 training records, about 65 million connection weights had to be evaluated and recalculated. Each experiment ran for 300 cycles and required up to 18,000 million weight evaluations/recalculations.

5. Numerical Results

5.1. Files generated

Two files, T_1 and T_2, with 2,000 and 10,000 true records were the backbone of all our experiments. Statistics from the T_2-file are summarized in Table 1, and were used as a basis for comparison of the results.

Table 2 displays the differences between statistics based on 10,000 records in the non-response S_2-file and the T_2-file with true records. A count that 2,902 or almost 30% of the records of S_2-file had one or more non-response attributes. Compared with observations made in labor force surveys, this seems to be a high non-response rate (Thomsen and Siring 1980). The table also indicates that the partial non-response varies among the attributes from 116 for employment to 636 for marriage.

Table 3 gives the main net deviations of statistics using the C_2-file with interchange errors from those in Table 1. As can be expected, all figures in this table are smaller than the numbers in Table 2 for several reasons. First, the risk probabilities are

Table 1. Population of 10,000 persons by attribute categories in T_2

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6
Sex	5031	4969				
Age	3002	3961	2003	1034		
Marriage	6954	3046				
Region	4048	2511	2455	986		
Children	7161	1344	945	360	190	
Education	2998	5958	1044			
Industry	4247	474	283	1171	1814	2011
Employment	5329	4671				
Income	5087	3297	1522	94		

Table 2. Differences between the records in the S_2- and the T_2-files

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Total
Sex	−260	−245					−505
Age	−160	−200	−53	−27			−440
Marriage	−488	−148					−636
Region	−100	−54	−55	−19			−228
Children	−103	−36	−25	−15	−4		−183
Education	−109	−218	−38				−365
Industry	−127	−15	−6	−31	−50	−60	−289
Employment	−57	−59					−116
Income	−394	−111	−44	−7			−556

Table 3. Differences between the records in the C_2- and the T_2-files

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Total
Sex	3	−3					0
Age	−52	−117	36	133			0
Marriage	−109	109					0
Region	−47	−28	−14	89			0
Children	−119	−7	18	38	70		0
Education	27	−212	185				0
Industry	−38	−5	−2	−4	11	38	0
Employment	4	−4					0
Income	−68	−82	33	117			0

Table 4. Differences between the records in the R_2- and the T_2-files

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Total
Sex	−212	−313					−525
Age	−227	−332	11	113			−435
Marriage	−536	−82					−618
Region	−140	−81	−80	55			−246
Children	−219	−54	−22	11	83		−201
Education	−120	−407	136				−391
Industry	−163	−23	−17	−36	−37	−32	−308
Employment	−72	−28					−100
Income	−502	−161	−23	98			−588

Table 5. Deviations between edited S_2-records and true T_2-records

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Total
Sex	36	−36					
Age	36	−7	−23	−22			−16
Marriage	19	−19					
Region	17	−2	−8	−16			−9
Children	−15	−26	31	−15	4		−21
Education	−37	50	−13				
Industry	13	−7	−6	−16	34	−24	−6
Employment	10	−12					−2
Income	−12	37	−37	−7			−19

different. Second, the interchange errors are randomly distributed among categories of each attribute and the net deviations will approach zero in large populations. The count of erroneous records in this file was 2,201.

Table 4 summarizes statistics based on the *R-file* including records which were exposed to the risks of both types of errors compared with Table 1. The count of records with one or more errors in this file was almost 4,500 records. Editing this file will obviously be highly challenging for our model.

The 8 different files established and used in the experiments were:

T_1: File with 2,000 records with true attribute categories.
T_2: File with 10,000 records with true attribute categories.

S_1: File with 2,000 records with raw attribute categories with only non-response errors.
S_2: File with 10,000 records with raw attribute categories with only non-response errors.

C_1: File with 2,000 records with raw attribute categories with only interchange errors.
C_2: File with 10,000 records with raw attribute categories with only interchange errors.

R_1: File with 2,000 records with raw attribute categories with both types of errors.
R_2: File with 10,000 records with raw attribute categories with both types of errors.

5.2. *Results from model editing*

It has already been pointed out that the model training was a resource and time-consuming process. The experiments required 300 training cycles and in total about 50 hours on a 486/25 desktop computer were spent training the models.

After 300 training cycles in Experiment I, the model was able to edit correctly most of the records in the training file. The real test of the trained model was how well the statistical results of the 10,000 edited records matched the corresponding true records. Table 5 gives the answer to this question. The model did the imputation surprisingly well. The negative figures in the Total column indicate that there were still records for which the non-response remained unresolved.

The model in Experiment II was trained for identifying and correcting the interchange errors and then tested against the file of 10,000 true records. Table 6 gives the results from this experiment. For all, but two attributes, there were units unclassified. The most difficult to resolve seemed to be Age for which 78 records were unclassified. Still, compared to the figures in Table 3, Table 6 indicates a substantial overall improvement. However, the editing model made two distributions, sex and employment, worse.

The R_2-*file* represented the most challenging task with both non-response and interchange errors in the records. In Experiment III, the model was trained by means of the raw records $\mathbf{R}_1$ and the corresponding true records in $\mathbf{T}_1$. The model learned the first 50% of the training records fast, but continued slowly and ended up having

Table 6. Deviations between edited C_2-records and true T_2-records

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Total
Sex	−57	57					
Age	9	−89	39	−37			−78
Marriage	37	−37					
Region	−2	−17	−3	−19			−41
Children	−18	10	−12	1	−45		−64
Education	34	−72	−23				−61
Industry	3	−2	7	6	−31	−31	−48
Employment	46	−58					−12
Income	−16	−26	−16	−12			−70

Table 7. Deviations between edited R_2-records and true T_2-records

Attribute	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Total
Sex	96	−97					−1
Age	0	−83	21	−45			−107
Marriage	136	−136					
Region	−2	−23	−12	−40			−77
Children	−31	20	−27	−24	−44		−106
Education	90	−88	−70				−68
Industry	−4	−14	−18	2	−30	−26	−90
Employment	42	−50					−8
Income	−58	55	−95	−27			−125

learned to recall correctly only about 1,600 of the 2,000 records of the training files during the 300 cycles it was allowed to run. However, when the trained model was applied to the R_2 test file, the results were encouraging. The edited results compared with the T_2 file, and are shown in Table 7. The trained model failed to handle all records and the table indicates, for example, that 125 records were not categorized by Income.

6. Discussion

All models behind the reported results were trained using 300 cycles. Additional experiments with up to 500 cycles were carried out without significant improvement. Other experiments with 200 neurons in the hidden layer instead of the 300 used in the reported experiments required less computing time per cycles but gave unacceptable results. Other experiments with 400 neurons in the hidden layer indicated that slightly better results might have been obtained at the price of higher training time.

Two main factors influencing the editing results are the *errors* in the raw files to be edited and the editing *power* of the trained editing mode. We used the numbers in Tables 2, 3, and 4 squared and summed, as a general metric for the errors in the files.

The *error* measures for the three raw test files used were:

S_2 raw file: 2,336,095
C_2 raw file: 168,914
R_2 raw file: 3,598,117.

The exponential nature of the metric should be born in mind when interpreting these measures.

The effect of the interchange errors in C_2 was less than the effect of the non-response errors in S_2. As already pointed out, interchange errors may quite well produce a raw record which looks acceptable. Because the metric used is squared differences of aggregates, the effect of the interchange errors has a tendency to approach zero when the population increases. The relative effect of the interchange errors is, for example, more serious in the training file with 2,000 than in the test file with 10,000 records. For the R_2-*file*, the joint effect of the non-response and the interchange errors is greater than the sum of the two. The explanation is the effect of squaring the differences.

We used the same metric on the *edited* files:

S_2 edited file: 16,547
C_2 edited file: 32,920
R_2 edited file: 150,501.

The error measures show a significant reduction in all experiments. The editing models gave the best results for the records with non-response errors only. As could be expected, the records influenced by both types of errors had the highest error measures after editing.

The figures cannot, however, be compared directly for evaluation of the models' abilities to edit raw statistical records. As a metric to express the *power* of the editing models, the error measures of the edited files were considered in relation to the error measures of the corresponding raw files. The ratios obtained were:

S_2 edited/S_2 raw file: 0.007
C_2 edited/C_2 raw file: 0.195
R_2 edited/R_2 raw file: 0.042.

The first measure indicates that the model trained to impute non-response errors had the power to reduce the error measure of the edited file to about 1/100 of the error measure of the raw file. Even though much less, the model trained for correcting interchange errors gave a reduction of the error measure for the edited file to about 1/5 of the error measure in the raw file. Most encouraging is, however, the editing power of the model trained to correct mixed types of errors. It reduced the size of the error measure for the edited file to about 1/25 of the error measure in the raw file.

7. Concluding Remarks and Future Tasks

The purpose of this article was to present a reformulation and investigation of the automated editing problem in statistical surveys by means of the ANN paradigm. Compared with editing methods based on the Fellegi-Holt framework and many others, the ANN approach has the advantage that we do not need to specify detailed editing rules and explicit assumptions about imputation functions prepared by subject matter experts. The ANN editing models can be trained to adjust themselves

to the record structures represented in a sample of records in both raw and edited forms. The editing of the sample must of course be done by experts.

The results of this study indicate the ANN editing models can be trained with success, and perform well with records distorted with frequent and mixed types of errors. The software and hardware resources required are commercially available and at prices which should not be prohibitive for statistical agencies. Still there are many methodological aspects that need further study.

We have not discussed how training and editing performance of ANN models depend on the size of the training sample of records. Because of the complex structure of the models, no function has yet been established for calibrating the training set. The decision about the size of the training set must therefore be approached empirically. When practical experience is gained in the different subject matter fields, it may be possible to establish some approximate rules to be used as a guidance for calibrating the training samples.

In the experiments reported only a limited number of categorical attributes were used. The ANN approach can also be used for editing records with both categorical and continuous attributes. An important future research task will be to study how well ANN editing models perform on records with mixed types of data. The second still unanswered question is how efficient ANN editing will be when the number of attributes increases to a size of hundred or more. These questions are currently being studied on real survey data.

The choice of a good ANN architecture for an editing model, including type of network, number of hidden layers, number of neurons in hidden layers, type of transfer function in the neurons, learning rate, tolerances and many other parameters not mentioned in this article need to be studied further on a varied set of editing applications.

The experiments reported in this article were based on synthetic data. All results rely on the validity of the assumptions made for the data and error generating models. If our assumptions are not upheld by situations met in editing of real survey data, our conclusions fail. It is therefore extremely important to test the ANN editing approach on real data records.

Because no adequate data have been available, comparison between the performances of ANN editing and alternative methods has not been possible. If we acknowledge that even human experts make errors, an objective test would be to apply all alternative editing methods on the same set of synthetic data with raw and true versions included.

Appendix

LIST 1: Rules for generating population records.

1. Sex:

'sex = male', probability = 0.50,
'sex = female', probability = 0.50.

2. Age:

'age = −19 years' probability = 0.30,
'age = 20–49 years', probability = 0.40,
'age = 50–69 years', probability = 0.20,
'age = 70+ years', probability = 0.10.

3. Marital status:

if 'age = −19 years' then:
'marital status = unmarried', probability 0.90,
'marital status = married', probability 0.10.

if 'age = 20+ years' then:
'marital status = unmarried', probability 0.60,
'marital status = married', probability 0.40.

4. Region:

'region = city', probability = 0.40,
'region = coast', probability = 0.25,
'region = inland', probability = 0.25,
'region = mountains', probability = 0.10.

5. Children born:

if 'sex = male' then:
'children = 0', probability = 1.00
'children = 1', probability = 0.00
'children = 2', probability = 0.00
'children = 3', probability = 0.00
'children = 4+', probability = 0.00.

if 'sex = female' AND 'age = −19' then:
'children = 0', probability = 0.85,
'children = 1', probability = 0.10,
'children = 2', probability = 0.05,
'children = 3', probability = 0.00,
'children = 4+', probability = 0.00.

if 'sex = female' AND 'age = +20' then:
'children = 0', probability = 0.25,
'children = 1', probability = 0.35,
'children = 2', probability = 0.25,
'children = 3', probability = 0.10,
'children = 4+', probability = 0.05.

6. Education:

if 'age = −19' then:
'education = 7', probability = 0.30,
'education = 8–12', probability = 0.70,
'education = 13+', probability = 0.00.

if 'sex = male' AND 'age = 20+' then:
'education = 7', probability = 0.30,
'education = 8–12', probability = 0.50,
'education = 13+', probability = 0.20.

if 'sex = female' AND 'age = 20+' then:
'education = 7', probability = 0.30,
'education = 8–12', probability = 0.60,
'education = 13+', probability = 0.10.

7. Industry:

if 'age = −19' OR 'age = 70+' then:
'industry = none', probability = 0.90,
'industry = agriculture', probability = 0.00,
'industry = fisheries', probability = 0.00,
'industry = manufacturing', probability = 0.00,
'industry = trade', probability = 0.00,
'industry = services', probability = 0.10.

if 'sex = male' AND 'age = 20–69, AND 'region = (coast OR city), then:
'industry = none', probability = 0.10,
'industry = agriculture', probability = 0.00,
'industry = fisheries', probability = 0.15,
'industry = manufacturing', probability = 0.20,
'industry = trade', probability = 0.25,
'industry = services', probability = 0.30.

if 'sex = male' AND 'age = 20–69, AND 'region = (inland OR mountain), then:
'industry = none', probability = 0.10,
'industry = agriculture', probability = 0.30,
'industry = fisheries', probability = 0.00,
'industry = manufacturing', probability = 0.15,
'industry = trade', probability = 0.30,
'industry = services', probability = 0.15.

if 'sex = female' AND 'age = 20–69' then:
'industry = none', probability = 0.10,
'industry = agriculture', probability = 0.05,
'industry = fisheries', probability = 0.00,
'industry = manufacturing', probability = 0.20,
'industry = trade', probability = 0.35,
'industry = services', probability = 0.30.

8. Employment status:

if 'age = −19 years' then:
'employment status = employed', probability = 0.20,
'employment status = unemployed', probability = 0.80.

if 'sex = male' AND 'age = 20–69 years' AND 'education = −12' then:
'employment status = employed', probability = 0.90,
'employment status = unemployed', probability = 0.10.

if 'sex = female' AND 'age = 20–69 years' AND 'education = −12' then:
'employment status = employed', probability = 0.60,
'employment status = unemployed', probability = 0.40.

if 'age = 20–69' AND 'education = 13+' then:
'employment status = employed', probability = 1.00,
'employment status = unemployed', probability = 0.00.

if 'age = 70+ years' then:
'employment status = employed', probability = 0.10,
'employment status = unemployed', probability = 0.90.

9. Income:

if 'employment status = unemployed' then:
'income = −99', probability = 1.00,
'income = 100–249', probability = 0.00,
'income = 250–499', probability = 0.00,
'income = 500+', probability = 0.00.

if 'employment status = employed' AND 'education = −12' then:
'income = −99', probability = 0.10,
'income = 100–249', probability = 0.70,
'income = 250–499', probability = 0.20,
'income = 500+', probability = 0.00.

if 'employment status = employed' AND 'education = 13+' then:
'income = −99', probability = 0.00,
'income = 100–249', probability = 0.20,
'income = 250–499', probability = 0.70,
'income = 500+', probability = 0.10.

LIST 2: Probabilities for non-response and random errors in different attributes.

Attribute	Error risk probability	
	Non-response	Interchange[1]
	1	2
1. Sex	0.05	0.03
2. Age		
if 'sex = female' AND 'age = −49 years' then:	0.07	0.05
if 'sex = female' AND 'age = 50+ years' then:	0.02	0.04
if 'sex = male' then:	0.04	0.06
3. Marital status:		
if 'age = −19' then:	0.10	0.01
if 'age = 20+' then:	0.05	0.05
4. Region:	0.02	0.01.
5. Children born:		
if 'sex = male' then:	0.01	0.01.
if 'sex = female' then:	0.03	0.03.
6. Education:		
if 'sex = male' AND 'age = −49 years'	0.05	0.07
if 'sex = male' AND 'age = 50+ years'	0.01	0.01
if 'sex = female'	0.04	0.03
7. Industry:	0.03	0.01
8. Employment status:	0.01	0.01.
9. Income:		
if 'income = −99' OR 'income = 500+' then:	0.08	0.02
if 'income = 100–499' then:	0.03	0.03.

[1] Random interchange errors appear only in the absence of non-response error

8. References

Anderson, J.A. and Rosenfeld, E. (eds.) (1988). Neurocomputing. Foundation of Research. Cambridge, MA: The MIT Press.

Anderson, J.A., Pellionisz, A., and Rosenfeld, E. (eds.) (1990). Neurocomputing 2. Directions for Research. Cambridge, MA: The MIT Press.

Creecy, R.H., Masand, B.M., Smith, S.J., and Waltz, D.L. (1992). Trading MIPS and Memory for Knowledge Engineering. Communications of the ACM, 35, 48–63.

Fellegi, I.P. and Holt, D. (1976). A Systematic Approach to Automatic Editing and Imputation. Journal of the American Statistical Association, 71, 17–35.

Nordbotten, S. (1963). Automatic Editing of Individual Statistical Observations. Statistical Standards and Studies, Handbook No. 2, New York: United Nations.

Nordbotten, S. (1965). The Efficiency of Automatic Detection and Correction of Errors in Individual Observations as Compared with Other Means of Improving the Quality of Statistics. Proceedings from the 35th Session of the International Statistical Institute, Belgrade 1965.

Roddick, L. H. (1993). Data Editing Using Neural Networks. Ottawa: Statistics Canada.

Rumelhart, D.E. and McClelland (1986). Parallel Distributed Processing-Explorations in Microstructure of Cognition. Vol. 1: Foundation. Cambridge, MA: The MIT Press.

Schafer, J.L., Khane, M., and Ezzati-Rice, T.M. (1993). Multiple Imputation of Missing Data in NHANES III. Proceedings of the U.S. Bureau of the Census Annual Research Conference, U.S. Department of Commerce, 459–487.

Sowa, J.F. (1984). Conceptual Structures, Information Processing in Mind and Machine. Readings, MA: Addison-Wesley.

Thomsen, I. and Siring, E. (1980). On the Causes and Effects of Non-Response. Norwegian Experiences. ARTIKLER No. 121, Oslo: Central Bureau of Statistics.

Werbos, P. (1974). Beyond Regression: New Tools for Prediction and Analysis in Behaviour Sciences. Ph.D. dissertation, Harvard University, Cambridge, MA.

Received December 1993
Revised October 1995