

**AUTOMATIC CODING OF HANDPRINTED RESPONSES
IN STATISTICAL SURVEYS**

Svein Nordbotten

University of Bergen
N-5020 Bergen
Norway

ABSTRACT

In mail surveys, it is frequently necessary to ask the respondents to describe themselves by their place of birth, profession, education, etc. in written form. The answers must subsequently be coded by the statistical agency. Modern optical scanning equipment can transcribe the answers to machine-readable form. Because of distorted characters and unusual spelling, automatic coding may not always be an easy solution. This paper outlines an automatic model for handling distorted characters and spelling errors. The model was tested on US city names by means of a prototype system and the results are discussed.

KEYWORDS

Automatic coding, Neural networks, Hybrid systems, Character recognition.

1. Problem description and proposed approach for solution

In mail surveys, it is frequently necessary to ask the respondents to write their activities and status by *descriptors* which later

¹. This paper has been prepared as part of the *SIS* Statistical Information System project at the Department of Information Science, University of Bergen.

can be used for classifying the responses to category identifiers or keywords. If the respondents are required to spell out the descriptors, character by character in boxes, a usual procedure is to use optical scanning techniques for automatic transcription to machine readable form. Subject to successful interpretation of the scanned responses, the descriptors can be translated by search on a list or a thesaurus to *keywords* of classification used.

One problem is to obtain a sufficiently high rate of correctly recognized descriptors. There may be several reasons for not recognizing to which keyword a descriptor should be mapped. One reason may be that characters forming the descriptor are written in a non-standard way. Even capital letters can be written by respondents with many variations. A second reason may be that the descriptor is spelled with characters omitted, or added to the standard keyword spelling. Finally, the descriptor may of course, be outside the vocabulary known to the system. The last situation is not discussed in this paper since the solution is obviously to extend the thesaurus with more synonyms and/or rely on other attributes given by the respondents to impute the keyword.

The approach discussed in this paper is use of a hybrid system of two modules as outline in Figure 1. One module of the system is an *Artificial Neural Network*, ANN, evaluating each character image of the descriptor after scanning. A second system module, called the *Decision Analysis* module, DA, compares the strings of recognized characters produced by the ANN, with the keywords of the classification, and determines the most likely keyword to which the descriptor should be mapped based on a set of decision rules.

It is assumed that each respondent is asked to declare himself by means of a descriptor with a limited length. In this study, it is assumed that the respondents will declare their relationships to a set of US cities. The respondents should 'print' the descriptors character by character in a pre-designed row of boxes on the questionnaire, using capital fonts from the English alphabet. An optical scanner is assumed to produce a pattern of each character in a grid.

2. The coding system

The purpose of the ANN module is to recognize the characters represented by character patterns deviating from the standard fonts because of the individual variation in printing capital characters. Character recognition is a classical field of ANN

applications, and a large number of different types of network models have been used for this purpose. The network used in the hybrid system discussed here is a so-called feed forward model with two layers of neurons or processing units arranged as indicated in Figure 2.

The grid used for character representation is an 8*8 pixel grid, and the standard character fonts are shown in Figure 3. The ANN module trained for recognizing character patterns, reads each character grid as a vector of 64 (=8*8) binary elements in which white pixels are represented by 0 and black pixel by 1, processes the input vector and produces an output vector with 26 continuous variables with range from 0 to 1. Each variable in the output vector corresponds to one of the possible characters of the alphabet. An output variable with a high value is considered more likely to indicate the correct character than a variable with a lower output value.

The 64 input sources corresponding to the pixels of the character grid, are interconnected to each of 100 neurons, in the hidden layer of neurons². The connections between signals from the input vector and hidden neurons, and from the hidden neurons to the output neurons are represented in two weight matrices. The binary values of the input vector are multiplied by the first weight matrix to obtain the inputs to the hidden neurons, while the output of the hidden neurons are multiplied by the second weight matrix to obtain the input to the second layer of neurons.

Each neuron transforms its weighted input sum to an output by means of a sigmoid function. The output value of this function will always be in the range between 0 and 1. The character associated with the output neuron which is producing the highest output value exceeding 0, is considered recognized and added to ANN keyword estimate of recognised characters. If all output values are less than 0.5, the character of the pattern investigated is considered unrecognized and the corresponding string position left blank.

The success of the ANN module in recognizing character patterns depends on how well the weight matrices are determined. The numerical values of the weights in the ANN module applied in this study were derived by an iterative learning or adjustment process. The algorithm used for calculating the weights was the well-known Backpropagation Algorithm.

². To be quite correct, 65 and 101 are the numbers of input sources and hidden neurons. To make the model more general an extra source and hidden neuron are added. These are always emitting 1's.

The aim for the DA module is to analyze the string of ANN keyword estimate strings. These strings may, in addition to correctly recognized characters, have incorrectly recognized characters, blanks because of unrecognized characters, as well as extra and missing characters because of respondent spelling errors. The DA module compares each ANN estimate with each keyword string in the classification list. An outline of the algorithm is shown in Figure 4.

Depending on the differences in a pair of ANN keyword estimate string and keyword, the estimate string is modified according to a set of rules. If there is no difference in length between an ANN estimate and a keyword, the number of matching characters is used as a measure similarity. If the length of the estimate string exceeds the length of the keyword string, non-matching characters in the ANN estimate may subject to certain conditions be disregarded to increase the similarity measure for the pair. On the other hand, if the ANN estimate is shorter than the keyword string, characters may be inserted in the estimate to increase the similarity measure.

After all keyword strings have been compared to an ANN estimate and assigned a similarity value, the keyword with the maximum similarity is considered the DA estimate of the descriptor processed.

3. Experimental design

3.1 Data for testing the system

Four distinct files of data needed for testing the system described above:

1. A standard set of font patterns for the capital characters in the alphabet.
2. A classification list with keywords and codes

3. A list of respondent descriptors for each test of the experiment.

4. An implemented system with an ANN and a DA module.

The standard set of font patterns was introduced in Figure 3 and recorded for the use as an alphabet pattern file.

The classification used was cities of about 50,000 or more residents in the USA. A total of 368 different city names with names up to 17 characters were recorded. Each city name was considered a *keyword* or name for a category in a geographical classification by which each respondent should be classified.

Several tests were designed and are described below. For each test of the experiment, a list of descriptors was needed. A descriptor file was generated in the following way. From the keyword file, *descriptor* records were generated to simulate the answers given in a survey to the question about city of birth, assuming a survey was limited to people in the 368 cities.

Each descriptor was obtained in four steps. First, a city name was drawn at random from the classification list, and the name transferred to the description record to be used for final evaluation. Second, each descriptor was subjected to a specified risk for spelling errors. Third, between each character of the city name, a random character might be inserted subject to a pre-specified character insertion probability for those descriptors drawn for spelling errors. Similarly, characters might be deleted from the same descriptors by a pre-defined deletion probability. Insertion or deletion could be made mutually exclusive within each descriptor. The pre-defined insertion and deletion probabilities characterized spelling differences and errors committed by the respondents. Finally, generated descriptors for some of the tests were distorted to simulate the different ways the respondents printed standard characters. The distortion was implemented sequentially for each character pattern of the descriptors, as a random process by which each black pixel of the standard character font pattern was exposed to a risk for being erased by a white pixel, while each white pixel in the standard pattern was exposed to be transformed to a black pixel.

The *network weights*, were computed by a Backpropagation training algorithm. The training was carried out based on the 520 patterns representing the 20 slightly distorted versions of the standard character alphabet. These versions were generated from the standard alphabet fonts, by a distortion similar, but not equal to the distortion used for the descriptor files. The distortion used for

the 20 versions of the alphabet were based on risk probabilities 0.05 for both white and black noises. On the basis of previous experience, this specification gave a variation of character fonts suitable for training the ANN.

The training also required specification of some parameters for the training algorithm. In our study, we chose an *adjustment rate* determined by a linear function varying from 1.0, when all patterns were unrecognized, to 0.1 when all were correctly recognized. A *smoothing factor* was set to 0.9. The *tolerance threshold* was specified to 0.1, which means that the computed output values will be considered as correct only if they do not deviate more than ± 0.1 from the target values. Specifications of these parameters were done based on previous experimentation and experience³.

3.2 Alternative experimental tests.

The coding system was subjected to six different simulated tests. All tests were represented by separate descriptor files with 100 descriptors each, and specified to evaluate different capabilities of the system.

Test 1: Recognition of descriptors with distorted characters.

The first test was to evaluate the system's capability to recognize correctly spelled city names which were printed with varying type fonts simulating the respondents' different handprinting. The descriptors used in this test were generated by introducing noise in the character patterns of city names written with standard fonts. The probabilities for white and black noises used were specified to 0.125 and 0.075, respectively.

In an 8 by 8 pixel grid, the average standard character has about 20 black pixels. The probability for a character pattern not being distorted was insignificant with the specified distortion probabilities. Each character pattern could be considered as an individual and unique version of the standard character pattern.

³. For further explanation of the training parameters, the reader is referred to the general literature about Backpropagation training algorithms.

Test 2: Recognition of descriptors with extra characters.

In this test, we wanted to evaluate the system's ability to recognize city names which were extended by one or more extra characters. A probability for inserting characters in front of each existing character, and after the last, was specified to 0.1 in generation of the descriptor file for this test. Descriptors generated with no extra characters were discarded. The test file thus comprised 100 descriptors with one or more extra characters. To investigate the system's capability to manage spelling errors, the characters in this and the next two tests were not distorted.

Test 3: Recognition of descriptors which are abbreviated

The third test can be considered a complement to the previous test. The aim of this test was to investigate the system's capability to recognize keywords with one or more deleted characters were evaluated. The probability for deleting a character was set also to 0.1, and only descriptors generated with one or more characters deleted were included in the test file.

Test 4: Recognition of descriptors with both characters inserted and deleted.

The existence of both spelling types of errors in a descriptor is obviously a more difficult recognition task than only recognising descriptors with one type of error. In the fourth test the system was evaluated for its ability to recognize words with a mixture of added and deleted characters. The probabilities for character insertion and deletion were as in the two previous tests, i.e. 0.1 for both insertion and deletion. Only descriptors with both types of spelling errors were included in the test file of 100 descriptors.

Test 5: Recognition of descriptors with types both spelling errors and character distortion.

In this last test, we wanted to explore how well the system managed to map descriptors with both spelling errors and character distortion to the correct keywords. For the generation of the descriptors for this test, the same specifications were used for spelling errors as in test 4 and for character distortion as in

test 1. Also, in this test file, only descriptors with at least one spelling error were included.

Test 6: Recognition of descriptors in a 'real' survey situation.

Finally, we specified a scenario which may be approximately corresponding to a 'real' survey situation. For generation of the test file of this test, we assumed that not all respondents were inclined to make spelling errors and that the risk probability for a spelling error in the record was 0.2. The insertion and deletion probabilities for in these records were both 0.05, and the two types of spelling errors were not assumed to appear within the same descriptor. Also, it was felt that the variations of character fonts used in test 1 and 5, were extreme. All characters in this test were therefore distorted with probabilities for white and black noises 0.075 and 0.050, respectively.

3.3 Computer implementation

The computer programs in the prototype testing system used were:

1. Program for generating different versions from standard alphabet.
2. Backpropagation learning program for computing the weights of ANN module.
3. Program for creating descriptors simulating respondents' answers.
4. Program for the coding the descriptors

The first two were previously developed programs for character distortion and backpropagation training. The last two programs were developed for this study. They were developed as a prototype system with a number of features for specifying error probabilities and relationships among the errors. All programs were developed in the C language to obtain portability between different hardware platforms.

3.4 Summary of the results.

The main results of the experiment outlined above are summarized in Table 1.

Table 1: Main results from recognizing experiments

Test: Correctly recognized keywords

Size of	descriptor file	
Test 1:	99	100
Test 2:	100	92
Test 3:	90	100
Test 4:	80	100
Test 5:	60	100
Test 6:	99	100

The only keyword of the first test which was not correctly recognized was one instance of PORTLAND, the characters of which were recognized as RORT AN by the ANN module, and mapped to DANBURY by the DA module. See Figure 5.

The incorrectly mapped descriptors in test 2, 3,4 and 5 are shown in Table 2, Table 3, Table 4 and Table 5, respectively.

In test 6 one descriptor was not correctly mapped to the keyword. The correct keyword was BOULDER which the ANN module recognized as BONUEDE and the DA module finally mapped to the keyword BOISE.

4. Discussion of experimental results

The ANN module learned to assign the correct characters to the 520 different training patterns, in 92 cycle. The Root Mean Square metric between target vectors and computed output vectors was after training equal to 0.012 which indicates a good adjustment to relationships between the training set patterns and the characters they represent. A better ANN might have been obtained with more than 20 different versions, but the price to be paid would have been a substantially longer training period.

A classification of 368 categories was used. The size of the classification is of course, a very important factor when

evaluating the results. Obviously, the probability for success of automatic coding decreases when the number of categories increases. The number of categories in official statistical standards is frequently 1000 or more, and the size chosen may be small compared with some of the classifications used. The number of characters in the keywords is a factor influencing the probability in the opposite direction. The longer the average keyword is in a classification, the more differences are expected among the keywords, and the higher the probability for successful recognition. The city names are probably quite representative for the length of keywords in real classifications.

It should be noted that the distortion of the character patterns of in this file was stronger than the distortion used for calculating the weight for the ANN module used for recognizing distorted characters. The neural network module was therefore required to recognize character patterns which were outside its training range, and some of the characters were difficult to recognize, even for the human eye and mind. Still, the results from test 1 indicated that the character distortion alone was no serious problem. Figure 5 illustrates the distorted characters of the only descriptor not recognized correctly by the system. It is even difficult for the human eye and mind to recognize the keyword represented by the descriptor in this Figure. Inspections of other descriptors indicate that the distortions of the individual characters in this descriptor were not atypical. To simulate a real situation, lower distortion probabilities should be used.

The alphabet of 26 characters may be adequate for the problem discussed in which the respondents can be instructed to 'print' their responses using capital letters. The corresponding 26 lower case letters could however, be added in a real application to take care of those who more easily print their answers with lower case letters. It is not expected that the learning of 52 different characters would present any learning problems except for the time required. With twice as many characters, the success of recognition might however, be reduced.

Test 2 and 3 results indicated that the rather simple decision algorithm used was capable of discovering errors of the two spelling error types and make satisfactory corrections for the inserted and deleted characters as long as errors of the two error types didn't appear simultaneously in the same descriptor. Table 2 and 3 illustrate again the very serious 'spelling errors' resulting from the spelling generation models used. They are likely to be far beyond the spelling errors expected that respondents in most environments would commit. Both tests indicated that the system with the error specifications used in this study still corrected more than 90 per cent of the incorrectly spelled

descriptors.

However, Table 4 confirms the suspicion that the spelling problem was worse to solve when insertion and deletion of characters were superimposed on each other. The system managed to correct about 80 per cent of the descriptors incorrectly spelled with both insertion and deletion errors. A brief inspection of the cases recorded in Table 4, indicates that most of the descriptors not recognized correctly were spelled in completely unrecognizable forms.

The results of the fifth test, in which both mixed spelling errors and distorted characters are assumed in each descriptor, are reported in Table 5. The results indicate that the system was only able to correct about 60 per cent of the descriptors. This may, at a first glance, seem to be a very discouraging result. It should however, be born in mind that the description file for this test is extreme and that no real survey population would produce a set of descriptors all of which would comprise at least one insertion and one deletion error and be strongly distorted.

On the basis of the results of the above discussed tests, a scenario for an approximately 'real' survey situation was specified as test 6. The results of this test showed that the system managed to recognize 99 per cent of the descriptors from the file generated. It can be discussed how realistic a situation is represented by this descriptor file, and how statistically significant is the obtained rate of correct recognition. The result should however, support the conclusion that automatic coding by means of a hybrid system of the type used in this study, may be a promising approach.

Many objections can be raised to the assumptions made and build into the system for generating respondents' answers. It can be pointed out that the respondents were supposed to give answers from a standard keyword list, in our study a list of city names, and not by a freely chosen descriptor from their own vocabulary. This is true, but in a real application system, the keyword list would be substituted by a thesaurus with usual synonyms combined with well-known parsing and root extracting algorithms to handle a wider spectrum of responses. The basic problem on which this study was focusing, was how to recognize written answers which are distorted, abbreviated, and/or extended versions from a set of keywords.

Some readers may object that only use of data from a real survey should be used for testing systems of this kind. It must be admitted that real data would have been desirable for the type of tests carried out in the present study. However, real data for testing are very rare for several reasons. First, they may not

exist in a form which can easily be used because some of the data reside on questionnaires, others on OCR output or on computer media. Another reason is that a real survey situation very seldom permits the registration of the 'true' keywords, but must rely on some in-house expertise to decide what is finally going to be considered as the correct keyword. Generated synthetic data have the great advantage that the experiment can always be kept under complete control [Nordbotten 1992].

It can be also be argued that the distortion of the characters in keywords used, and the random spelling errors introduced, are different from the individual ways of writing capital letters and spelling names. Even though true, the distortions used in this study were likely to give more variations than the individual character variations in a real situation, and the spelling errors introduced were much more difficult and general than would be expected in real data. The results give a strong indication that a similar system applied on real life variations in responses would also might be successful.

Objections may also be made that non-response and correctly printed and spelled, but false responses are important aspects not been discussed. This problem has been discussed in a previous paper on automatic editing [Nordbotten 1996]. The approach discussed in that paper can easily be adjusted and added to a coding system as discussed in the present study.

Further research in this area will be needed. The ANN module should be extended and trained to handle a wider set of characters including lower case fonts and script types of character fonts used in ordinary handwriting. The DA module, as used for this study, has its rules embedded in the decision algorithm. To obtain greater flexibility and wider generality, the DA module should be substituted with an Inference Module associated to a separate Rule Base which can easily be modified and extended to fit specific tasks.

5. Acknowledgement

This paper has been prepared during the author's stay with the Department of Information and Computer Sciences, University of Hawaii at Manoa, funded by the University of Bergen. I wish to thank Associate Professor Joan C. Nordbotten and an anonymous reviewer for many useful comments on a previous version of this paper.

6. References

Nordbotten, S.: GENERATING SYNTHETIC IMAGES - THE IMAGE5 SYSTEM,
Neural Network World, Vol.2, No.2, 1992, pp-149-173.

Nordbotten, S.: EDITING STATISTICAL RECORDS BY NEURAL NETWORKS,
Journal of Official Statistics, No.3, 1996.

Figure 3: Standard font patterns used

Figure 5: The unrecognized descriptor for the keyword PORTLAND

Table 2: Incorrectly mapped descriptors in test 2.
Descriptors with inserted characters in all descriptors.

Descriptor for Incorrectly mapped to keyword	ANN	Recognized by keyword
--	-----	--------------------------

THOUSAND OAKS

ORANGE

OVERLAND PARK

APPLETON

NORTH LITTLE ROCK

ONTARIO

GLENDALE

THOUSAND OAKS

ORDHANGGEXM

RVERLAND PARK

APOGPVLETON

NOXRTN LITTLE XOT

OONKVARIOLW

NELGWTOGN

HOLLYWOOD

FRAMINGHAM

CLEVELAND

BLOOMINGTON

NORTH CHARLESTON

ANN ARBOR

ALLENTOWN

**Table 3: Incorrectly mapped descriptors in test 3.
Deleted characters in all descriptors.**

Descriptor for Incorrectly mapped to keyword	ANN	Recognized by	keyword
INDEPENDENCE	INPENDENE	GLENDALE	
JACKSON	CKSN	AKRON	
DEARBORN HEIGHTS	DEARRN HIGHS	CHERRY HILL	
MINNEAPOLIS	MNAPOLIS	MEMPHIS	
KENDALL	NDALL	EWA	
GREENSBORO	GREBORO	FREMONT	
KALAMAZOO	KAMAZO	CAMDEN	
SAN FRANCISCO	SAN RANCCO	SAN LEANDRO	
FLINT	F	EWA	
SPRINGFIELD	PINGELD	LINCOLN	

**Table 4: Incorrectly mapped descriptors in test 4.
 Inserted and deleted characters in all descriptors.**

Descriptor for Incorrectly mapped to keyword	ANN	Recognized by keyword
WAKLTHAM	APLHAM	DURHAM
EL CAJON	E CAJOWN	EDISON
PARMA	PAMI	MIAMI
CINCINNATI	CNXNCINNAI	CANTON
HARFORD	HARDHFTFJBRD	ALHAMBRA
ECONDIDO	PECONDIDOH	PREORIA
WEST HARFORD	WES HARTFORZD	NEW BEDFORD
FULLERTON	FULLRTONY	APPLETON
BURBANK	URNK	ERIE
CASPER	OCASPR	DECATUR
MIAMI BEACH	MIMI BEANCH	LONG BEACH
RICHMOND	ICKHMOD	AKRON
SIOUX FALLS	SIUXATLSK	DUNDALK
NEW HAVEN	EW HUAVEN	EL CAJON
WEST PALM BEACH	WESDTPALM BEFACH	REDONDO BEACH
ALAMEDA	TALMEEA	TAMPA
LIVONIA	LINIA	ELYRIA
ARLINGTON HEIGHTS	ARLINGTONJ HEIHT	ARLINGTON
OAK LAWN	AK EOLAWN	ALLERTOWN

**Table 5: Incorrectly mapped descriptors in test 5.
Distorted characters. Inserted and deleted characters in all
descriptors.**

Descriptor for Incorrectly mapped to keyword	ANN	Recognized by keyword
DEARBORN HEIGHTS	DEARBORN KNIGN	DEARBORN
SPRINGFIELD	S AQRING IE	LANSING
BATON ROUGE	AT HRORYUGE	ANCHORAGE
SANTA CLARA	ATA RTA	ATALNTA
LARGO	ULTMRG	UTICA
WESTMINSTER	WEESTMINSTR	WESTLAND
ESCODIDO	ERSCONDIDC	EUCLID
MIAMI	I NBI	LYNN
WHITTIER	HIE X	ERIE
SKOKIE	L KIE	ERIE
ORLANDO	R AOT	ARVADA
ELGIN	YELGN	AKRON
WAUKEGAN	AUKGAN E	AKRON
KENDALL	K M A	TAMPA
PORTSMOUTH	FORTS U	FARGO
LAWRENCE	GU WRENE	ABILENE
REDFORD	E FOIRA	AURORA
CINCINNATI	N IN NENI	ABILENE
SALINAS	AJLNA	DALLAS
GREENVILLE	GRENVI LEA	GLENDALE
GLENDALE	GKNDALVD	DUNDALK
HUNTINGTON	HUSNTIJTQN	AUSTIN
SILVER SPRING	USR LXVE JPRIS	ABILENE
PORT ARTHUR	PRKT HUR	ERIE
HOLLYWOOD	H LY EIO	DALY CITY
BILLINGS	V I INGS	IRVING
SANTA CLARA	SAT COKV A	SANTA MONICA
BEAUMONT	VBELHAUNONT	KOOLAUPOKO
BRISTOL	BMIS LP	BOISE
NORWALK	NRWIAA K	NEWARK
LAKE CHARLES	LA KE H RE	LAKWOOD
CARSON	CAARQ	GARY
MINNEAPOLIS	INDNEAPDOLIS	INDIANAPOLIS
BELLEVUE	CELLEVE	MCALLEN
TYLER	LEJR	ELGIN
TAYLOR	AYO UW	BAYONNE
THOUSAND OAKS	H UCSAN A S	TUCSON

LAKWOOD
PENN HILLS
PORTSMOUTH

LEWSDO DX
HDPX E N HWI
PORTS KT

LAREDO
LS
PORTLAND

APPLETON