Decision Criteria for Using Automated Coding in Survey Processing

Daniel W. Gillman, Bureau of Labor Statistics, Washington, DC 20212

Abstract

Some survey data are classified into pre-specified categories during a process known as coding. If a computer assigns codes without human interaction, then this is called automated coding. Manual coding, computer-assisted manual coding, and interactive coding all require some level of human interaction. The decision to employ automated coding in survey processing is not simple. There are many options and expenses to consider. Some of these are as follows: 1) who develops the software; 2) what is an acceptable error rate; 3) how will errors be controlled; 4) what percent of the cases must the automated coder classify; 5) how much maintenance will a production system require; and 6) what new resources must be developed to build an automated coder. These criteria and others are described in this paper. A cost model is developed along with a description of the interactions between the criteria. Finally, some examples are given to show how the model might be employed by a survey organization.

Key Words: classification, cost model, error control

1.0 Introduction

Some survey data are classified into pre-specified categories during a process known as coding. If a computer assigns codes without human interaction, then this is called automated coding. Automated coding never assigns codes to 100% of the cases, so some type of manual coding is required in addition. However, the use of automated coding can dramatically decrease the amount of manual coding required to finish the job. The decrease in the number of cases requiring manual coding saves time and money, and it increases the consistency in the codes assigned (Appel and Scopp, 1983; and Scopp and Tornell, 1990).

Manual coding operations increasingly make use of computers to aid the clerks during the coding process. This enhanced process is known as computer-assisted manual coding. Some of these systems are able to suggest codes to the clerks. Interactive coding is a computerized process used during data collection. An automated coder assigns the codes it can, and then it provides the interviewer-coder with candidate codes for the cases it cannot decide. For the purposes of this paper, we will restrict the discussion to automated coding only, however, many of the requirements for developing automated coders apply to the interactive case as well.

The decision to employ automated coding in survey processing is not simple. There are many options and expenses to consider. Each of the options has an impact on the quality of the automated coding operation. These considerations and others will be addressed in this paper.

The paper is roughly organized in the following way. After the introduction, there is a section on definitions of terms, followed by a description of the major elements of an automated coding system, and then a set of criteria are given for deciding whether building an automated coder makes sense. An overview of the workflow for the development of an automated coder is in the next section, and the last section contains a model of the costs associated with building and maintaining an automated coder. Costs that are part of any coding system are separated from the costs that apply to automated coding specifically.

2.0 Definitions

The following terms apply to the rest of the discussion in this paper. We assume the reader knows what coding, codes, and classification are.

- **Acceptable Code**: a code assigned by an automated coder that is deemed acceptable by an error control algorithm
- **Expert Coding**: coding by human experts, considered the "truth" when developing coding systems
- **Error Control**: an algorithm for controlling the estimated error rate produced by an automated coder
- **Error Rate**: the ratio of the number of correctly coded cases divided by the number with acceptable codes
- **Feedback Loop**: mechanism for improving an automated coder based on the results of coding test data
- **Production Rate**: the ratio of the number of cases with acceptable codes assigned divided by the total number of cases
- **Quality Control**: a system for estimating the error rate of an automated coder during processing
- **Test Data**: expert coded data used to test the effectiveness an automated coder during training

- **Training Data**: expert coded data used to develop, or train, an automated coder
- **Validation Data**: expert coded data used to test the effectiveness of an automated coder after training is complete

3.0 Elements

The decision to build and implement an automated coder requires careful consideration of the costs. In this section, we provide a description of the required elements for building and implementing an automated coder. In a later section, the costs associated with these elements are analyzed.

The major elements are classifications; training, test, and validation data; software development; and error and quality control methods. A discussion of each of these elements is contained in the remainder of this section. We ignore the production environment, because inserting an automated coder into an existing production system is relatively easy. Software development covers the requirements of getting the system ready for production.

3.1 Classifications

Statistical offices use classifications to categorize and classify some types of survey data. Typical examples of classifications in the US are the North American Industrial Classification (NAICS), the Standard Occupational Classification (SOC), product classifications, and time use activity classifications. Many other examples exist, too.

An automated coder is used to classify survey responses into categories determined by a classification. For instance, the 2000 Census Autocoder was used to classify responses from the 2000 Census long form into industries and occupations from the 2000 Census Industry and Occupation Classification. The 2000 Census Industry and Occupation Classification was derived from NAICS and the SOC.

For an automated coder to work properly, careful construction and maintenance of the relevant classifications are required. An automated coder cannot accurately classify survey responses if classifications on which it depends are poorly constructed. Good software cannot make up for design flaws. Conversely, improvements in the design of the relevant classifications can greatly improve the performance of an automated coder. Examples are removing overlap between concepts or reorganizing concepts. Other strategies also exist.

Classifications also change over time. For example, NAICS is scheduled for revision every 5 years in order to reflect changes in the industrial situation within North America. An automated coder must be revised whenever the relevant classifications are. Otherwise, the automated coder becomes increasingly irrelevant, and all development costs are lost.

3.2 Training/Test/Validation Data

Three data sets of expertly coded cases are required to develop, test, and validate an automated coder. Each of these data sets needs to satisfy the following criteria:

- the cases are current
- the cases are representative of each code category, at east 150 cases per code category (Chen, *et al*, 1993)
- cases are sampled with known probabilities of selection and stratified by code category, so proper weights can be assigned
- cases for each data set should come from a different source, such as different surveys (if the automated coder will be used for multiple surveys), time frames, or modes of data collection

Training is the term used by researchers in the machine learning community for using data to "teach" the software which categories to assign to cases. So, the software developers use the training set for building the system. The proper codes are attached to each case so the developers know to which category each case belongs.

The test data are used periodically by the software developers to independently verify that the training exercise is working. This is known as a "feedback loop". Feedback loops are often run daily, and a well-designed process will provide excellent guidance for the further development of the software (Appel, 1983). It is best if the developers do not know the expert codes for the test data cases, and the evaluation of the test data is done independently. This blind testing increases reliability of the results.

Test data taken from the same source as the training data biases the results (Gillman, 1993). Data from the same source are more similar than data taken from different sources. So, care must be taken to ensure proper interpretation of testing during the development phase. It is best if the test set is drawn independently of the training set.

The validation set is used after the development is complete, and the automated coder is ready for production use. The validation set is used for the

final test before production; the data for the set are usually taken from the initial survey data collection. Again, the independence of the validation set from the training and test sets increases the accuracy of the results.

3.3 Software

There are many algorithms in use or that might be used successfully for automated coding (Lyberg and Dean, 1990; Creecy, *et al*, 1992; Speizer and Buckley, 1998; Sebastiani, 2002). The choice depends on many factors, but often the most important factor is the complexity of the problem. Usually text responses are coded in survey operations. Some responses are very simple, and others are much more complex. Usually, the more text fields in a response, the more categories in a classification, the more classifications, or the more dependencies between multiple classifications for a coding application, the more difficult it is to automate. These observations are meant as guidelines, as there are exceptions to them.

Simple problems can often be solved using exact matching against a list of responses with their codes. New responses not found in the list are added after they are manually coded. Most applications have more complex responses. For these situations, more advanced techniques must be used. The best gauge for whether a particular algorithm or system will work is whether it has worked in the past for other similar applications. However, due to the unique nature of some automated coding problems, survey organizations find it useful to develop their own system.

Many survey organizations have developed application-specific and general-purpose software for automated coding, there are companies that build automated coding systems for survey organizations, and there are several companies that are using machine learning techniques for new types of classifiers. These new classifiers may be adapted to automate coding applications.

Training, test, and validation data are used during the software development life cycle to help build any automated coding system. The requirements for the coverage contained in the data sets goes up as the complexity of the software and coding requirements go up.

3.4 Error and Quality Control

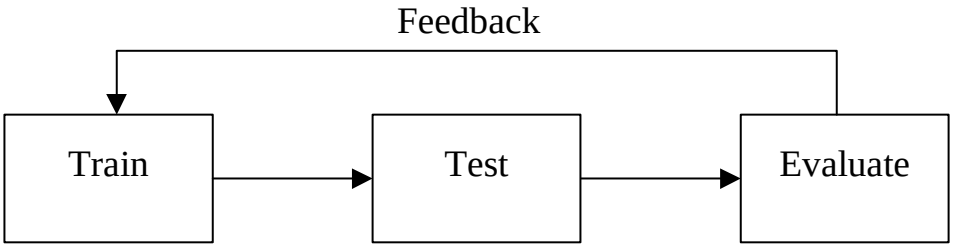
It is not sufficient to estimate the errors for the test or validation data overall, but one must estimate the errors for each code category separately. An error control algorithm is used to control the estimated errors an automated coder makes.

A useful approach (Chen, Creecy, and Appel, 1993) is to pick an overall acceptable error rate and use that rate to control the errors for each code category. This way, the estimated errors for each code category will not exceed the overall estimate. Alternatively, an acceptable error rate may be assigned to each code category independently (Gillman, Appel, and Jablin, 1993). In this case, the overall error rate is harder to estimate, but the automated coder performs as well as the clerks for each code category separately..

Quality control (QC) estimates the error rate during production (Biemer and Caspar, 1994) . Most QC techniques estimate the overall errors, but they do not measure the errors for each code category. This is a shortcoming, but it is mostly due to cost constraints that organizations do not measure errors at the category level.

4.0 Workflow

This section contains a description of the workflow in the development of an automated coder. It is a more detailed description of ideas described in the previous sections. The basic idea is illustrated in the following diagram and explained below in the text:



Development takes place in a series of steps. These steps are performed interatively until the softeware performs at an acceptable level. A level of ascceptable performance is determined by the estimated error and production rates of the system. When the error rate is low enough and the production rate is high enough, the development process ends. Then, implementation of the system begins.

At the beginning of each development iteration, the automated coding software is subjected to testing against cases with known codes: the training data set. The software and system parameters are updated as the results of coding the cases are analyzed. The error and production rates are estimated using this data, but one should expect that these results are biased as the estimates are not indpendent of the training. Use of an independent test data set results in unbiased estimates for error and production.

At regular intervals, or whenever the developers feel enough changes have been made to the automated coding software, the test data set is processed through the system. The test data set is drawn independently of the training data set to eliminate bias. Also, the test phase produces detailed results of coding efficiency for each code category. These results are used in the next phase.

The results of coding the test data set are used to evaluate the results of the training step. There are several important features of this:

- An independent unbiased estimate of errors and productivity is obtained
- Error rates for each code category are provided, and this allows system developers to improve the software where it needs it most
- Effort can be directed towards those (large) code catgories that are expected to have many cases assigned to them during actual survey proessing
- Imporovements to large code categories upgrade overall productivity and error rates more than improvements to small code categories do
- Bias obtained through "over-training" is reduced (Appel and Scopp, 1987; Sebastiani, 2002), where over-training means too many of the specific characteristics of the cases in the training set are used to train the coder

Feedback is the process whereby information gained through testing is applied to upgrading the automated coding sofware and improving the codes assigned to cases in the training and test data sets. The amount of time taken to make improvments during each iteration is dependent on many factors, but is most dependent on the time allocated for the development project and the expected gain for each attempted improvement.

It is entirely possible that some expected improvements do not work or actually make things worse. By examining a subset of the training or test sets, a developer might see improvements in one area (e.g., a set of cases containing a specific word are all coded correctly), but after evaluating the automated coder against the entire training or test data sets, the developer might find that other cases are now coded worse. In this case, it may be necessary to remove the changes.

The feedback loop development process is time consuming and full of wrong turns or blind alleys. However, it is an effective way to build a good system.

5.0 Costs and Decision Criteria

Ultimately, the choice of using an automated coder boils down to costs. Is it more expensive to build and maintain the software for an automated coder than to code the cases manually? This comparison is based on the number of cases the automated coder assigns codes to (i.e., decides). So it is very important to have high quality training and test data sets to maximize the effectiveness of the automated coder.

Normally, one chooses to use an automated coder if it is less expensive to code cases that way. For example, if over a certain period of time it costs $100,000 to build and maintain an automated coder, and it costs $200,000 to code the same cases maunally that the automated coder can decide, then it makes sense to use the automated coder.

The first decision is how much error to let the automated coder produce. Usually, one sets the error rate to correspond with the estimated eror rate for the manual coding clerks. This may not be quite fair if the automated coder only decides the cases that are easy for the manual coding clerks to decide as well. However, the author knows of no studies to show that this is the case, either for a specific coding operation or for automated coding in general. Most importantly, it does not make sense for the automated coder to produce codes at an apprecialbly different error rate than the clerks. In order to make this choice properly, a good estimate of the error rate for the manual coding operation is required. This in turn requires the availability of high quality training and test data sets.

The acceptable error rate for the automated coder affects the productivty. The lower the acceptable error rate (i.e., higher accurracy), the lower the production rate for the automated coder, and the more cases the clerks have to decide. Not only does this raise overall processing costs, because each additional case the clerks need to decide raises costs, but it reduces the number of cases the automated coder decides. This makes it harder to prove that the automated coder is cost effective.

5.1 Cost Model

In determining costs, we make a basic assumption: every coding operation has a manual coding component. This means no automated coder will decide 100% of the cases. Given this basic assumption, costs are divided into 4 categories:

• **Indpependent** - Costs that are part of the coding operation, independently of whether an automated coder is used or not, e.g., development of computer-assisted manual coding systems
• **Both** - Costs associated with both manual and automated coding but allocated based on the application, e.g., the costs of developing and maintaining training and test data is really for the automated coder development, yet the data sets may be used as training for the manual coding clerks and for development of concordances for translating between classifications
• **Automated** - Costs associated with developing and maintaing the automated coding system only
• **Manual** - Costs associated with the manual coding operation only and dependent on the number of cases

The major cost components for each cost category are listed in the table below:

	Components
Independent	Classifications; Alphabetical indexes; Computer-assisted manual coding software; Training system; Computer hardware
Both	Training, test, and validation data sets; Selction criteria and weights; Quality control
Automated	Software development, maintenance, update, upgrades (especially from changes to classifications), validation
Manual	Labor; Recruitment; Supervision; Overhead (desks, etc.); Reference materials

---

5.2 Decision Criteria

The main decision criteria are as follows. Let
• C_A = Costs for using an automated coder
• C_M = Costs for using manual coding for the same cases the automated coder decides plus some fixed costs (see below)
• $C_D = C_A - C_M$ the cost difference between using automated versus manual coding

If $C_D < 0$, then the automated coder saves money, and otherwise it does not.

C_M is the difficult parameter to estimate. It is based on an estimate of the cost to manually code each case. The following values must be known or estimated:
• T = Total number of cases to be coded
• E = Error rate for the clerical coders
• P_E = Production rate for the automated coder, which depends on E (See section 3.4)
• C = Coding rate per hour, i.e., the number of cases decided per hour, for the clerks
• L = Labor costs per hour for the clerks (The labor costs per hour depend on overhead - fixed labor costs - plus the number and wages of the clerks hired.)
• F = Fixed costs, costs that do not depend on the number of cases to be decided

Another parameter that must be estimated is the allocation percentage for each of the costs in the **both** category. This is part of the fixed costs, since these costs do not depend on the number of cases to code. The **independent** costs are not part of the cost comparison, since they are borne whether automated coding is used or not.

Then, $C_M = T * P_E * L / C + F$.

6.0 Conclusion

The paper contains some decision criteria for determining whether it is cost effective to use an automated coder in survey coding operations. The criteria are based upon a comparison of the costs of developing and using an automated coder versus the cost of manually coding the same cases the automated coder can decide. If the cost of using an automated coder is less than the cost of manually

coding the same cases the automated coder can decide, then the use of an automated coder is cost effective.

The cost model provided is simply expressed, but the parameters are hard to calculate or estimate. Especially problematic are the manual coding costs and how to allocate the costs that belong to the **both** category.

A careful analysis is reqired to estimate the cost savings accurately.

7.0 References

• Appel, M. V. and Scopp, T. (1987), "Automated Industry and Occupation Coding", presented at Development of Statistical Expert Systems (DOSES), December 1987, Luxembourg.

• Biemer, P. and Caspar, R. (1994), "Continuous Quality Improvement for Survey Operations: Some General Principles and Applications", *Journal of Official Statistics*, Vol 10, No. 3, 1994, pp. 307-326.

• Chen, B., Creecy, R. H., and Appel, M. (1993), "On Error Control of Automated Industry and Occupation Coding", *Journal of Official Statistics*, Vol. 9, No. 4, pp. 729-745.

• 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, Vol. 35, No.8, 48-64.

• Gillman, D. W. (2000), "Developing an Industry and Occupation Autocoder for the 2000 Census", American Statistical Association – Proceedings of the Section on Government Statistics, Indianapolis, IN, August, 2000.

• Gillman, D. W., and Appel, M. V. (1993), "Analysis of the Census Bureau's Automated Industry and Occupation Coding System Algorithm", Proceedings of Government Statistics Section of the American Statistical Association, San Francisco, CA, August 1993.

• Gillman, D. W., and Appel, M. V. (1994a), "Automated Coding Research at the Census Bureau", SRD Research Report RR-94/04, 10/5/94.

• Gillman, D. W., Appel, M. V., and Jablin, C. (1993), "Certification of Decennial Automated I&O Coder for Current Surveys", Proceedings of Census Annual Research Conference, Arlington, VA, 21-24 March 1993.

• Lyberg, L. and Dean, P. (1990), "International Review of Approaches to Automated Coding", Conference on Advanced Computing in the Social Sciences, Williamsburg, VA, April 1990.

• Scopp, T. S., Tornell, S. W. (1991), "The 1990 Census Experience with Industry and Occupation Coding," presented at the Southern Demographic Association Annual Meeting, Jacksonville, FL, October 11, 1991.

• Sebastiani, F. (2002), "Machine Learning in Automated Text Categorization", *ACM Computing Surveys*, Vol. 34, No. 1, pp. 1-47.

• Speizer, H. and Buckley, P. (1998). "Automated Coding of Survey Data" in Couper, M. *et al* (eds.) *Computer Assisted Survey Information Collection*, Wiley Series in Probability and Statistics, 1998, pp. 223-243.