The Blaise Coding Facilities - a closer look

Brett Martin

Statistics New Zealand

1. Introduction

Coding verbal or descriptive responses is a significant activity in many surveys. Traditional methods of manual coding are slow, monotonous and labour intensive. It is also very difficult to maintain coding accuracy and consistency with manual coding. The introduction of Computer Assisted and Automated Coding systems have in general improved the productivity and reliability of coding. However many of these computer systems have taken a considerable amount of effort to develop.

The coding tools supplied with Blaise have made computer assisted coding more accessible. It is very easy to incorporate effective Computer Assisted Coding (CAC) into a Blaise questionnaire.

In Statistics New Zealand various types of Computer Assisted and Automatic coding have been employed over the past eight years. In this paper I briefly outline our experiences and then review the important things to consider when implementing CAC. The Blaise coding facilities are examined, including the new Trigram coding system available with Blaise 2.5. I then discuss the issues associated with constructing coding dictionaries and where CAC is best applied. To conclude a few ideas to extend the Blaise coding facilities are suggested.

2. Statistics New Zealand coding experiences

Computer Assisted Coding was a major innovation when it was introduced for processing the 1986 New Zealand Census of Population and Dwellings. It replaced the previous resource intensive clerical coding methods. About 85% of our Census data was numeric or 'self coded', where respondents ticked answer boxes on the questionnaire. The remaining 15% comprised written responses that required coding. Even using CAC, these accounted for 50% of the total resource employed.

CAC has a number of distinct advantages over manual coding.

- The computer searches the code list instead of the coder, making it significantly quicker.
- The codes are directly written to the data file avoiding the need to capture and edit them.
- Transcription and substitution errors are eliminated.
- Greater consistency is achieved through computer matching.

CAC operators working on the 1986 Census found that many descriptions they entered resulted in a single match. They felt the system would be improved if these matches could be accepted directly without confirmation. Experiments showed that as long as some conditions were met, accepting the match without operator confirmation was a useful enhancement.

Automatic coding was used along with CAC for processing the 1991 Census of Population and Dwellings. All the written descriptions were captured using PC's linked to a Local Area Network and transferred to our Mainframe for Automatic coding. The Automatic coding system combined the matching methods developed for CAC in 1986 with the acceptance criteria developed for accepting matches without operator

confirmation. Any descriptions not Automatically coded were passed to the CAC operators to code.

Automatic coding was adopted in order to balance the workload on the Mainframe. Management overheads and staffing problems were experienced with a two shift CAC operation in 1986 and a single shift for 1991 was seen as a priority. This combined with the need to improve system response times and support other departmental Mainframe users, would have required more computing resource during the day shift than was available. Automatic coding was run in Batch during non peak hours.

The computing resource bottle-neck during the day shift was reduced with Automatic coding. Around 70% of location, 50% of occupation and 90% of socio-demographic descriptions were Automatically coded. However, in hindsight, this approach was not as effective as these matching percentages figures might suggest. The majority of descriptions successfully coded were the easy to code ones. The effort required to code the remainder was not reduced proportionally. There was also the overhead of capturing the descriptions, although these have subsequently been extremely valuable in questionnaire evaluation and code file development.

Automatic coding would be more cost effective when there is a high proportion of descriptions coded, and where there are less overheads capturing the descriptions. For example administrative records or answers that are captured but not coded during field collection. Descriptions from paper questionnaires could also be captured for automatic coding using electronic imaging and Intelligent Character Recognition,

More recently the Blaise coding systems have been used in a number of coding applications in Statistics New Zealand. The prime advantage with Blaise CAC, is that it can be easily incorporated into the capture and edit system.

3. Important considerations when implementing CAC

At first glance the coding system and the matching methodology would appear to be the most significant factors in automating coding. However our experience has shown that training the CAC operators and supervisors in the subject matter to be coded and improving the contents of the coding files is more important that the technology employed. CAC systems improve coding rates but in order to ensure coding accuracy and consistency other factors must be considered.

Firstly the question asked and the classification used to code the answers, must reflect what the researcher wants to measure. For example, if a researcher wanted to measure skill levels rather than social status, a question on Occupation might be asked in a different way and certainly would be classified differently. It is important that the question clearly guides the respondent as to the answers required. Otherwise the question will be answered from different perspective's and the coding will be unreliable. The question should also elicit sufficient information to enable the responses to be classified to the level of detail required. All of this implies testing and evaluation of both the question and classification to ensure they are workable.

Computer based coding systems usually employ a special file for matching containing the classification titles and additional synonyms. This dictionary file provides a link between the respondents answer and the classification codes. The contents of the dictionary need to reflect the likely responses to the question. Ideally respondents would be canvassed for all their answers and these would be classified and used to construct the dictionary. In practice this is usually unrealistic. For simple classifications, such as Country of Birth or Relationship there is generally a finite number of answers and there is little problem in building a dictionary. However where the question is more general, such as Occupation or Industry, the range of possible

answers is much larger and the effort to construct a dictionary to cover them is correspondingly greater.

In the same way that question design has a significant influence on the responses obtained, the classification and dictionary used influence coding outcomes. If the dictionary contains the majority of the descriptions that are given as answers, the amount of interpretation is reduced along with the query rate. This results in greater coding consistency and improved throughput. While this should be obvious, the deficiencies and inaccuracies in our dictionaries have been a significant problem. Evaluations of our Census CAC coding have consistently recommended that more effort be spent on developing the dictionaries and classifications.

The 1986 Census CAC operators, entered the description from the questionnaire. They were permitted to enter whatever they thought was most likely to obtain an acceptable match. It did not take them long to discover that, when they entered the whole description, it took a longer time to obtain fewer responses. If they entered abbreviations, they obtained results more quickly, but usually containing many irrelevant descriptions. The pressure to improve throughput and the slow response times experienced led to more abbreviations being entered. This sometimes resulted in guess-work especially when a long list of potential matches was presented.

In 1991, with the introduction of Automatic coding, the whole description was captured. This was transferred to a CAC operator when it could not be Automatically coded. Generally they used this description, with maybe small modifications, to search. As the system performance was better, the speed of matching became less of an issue. Less matches were displayed and providing the description was in the dictionary, less interpretation was required. Entering the whole description reduces the need to scan a long list of potential matches and minimises the risk associated with human interpretation.

The training given to the CAC operators, their supervisors and staff involved in query resolution is very important in ensuring that consistent judgements are made. This training should include information on the basic structure of the classification and be backed up by documentation material on section. As there will be situations the operator cannot resolve using the system, specialist back-up services for query resolution are recommended. Concentrating this expertise helps to provide consistent and timely solutions to difficult referencing problems. These people can also be responsible for monitoring the types of query and providing feedback to enable new or amended descriptions to be incorporated into the coding files.

4. The Blaise coding facilities

There are now three types of computer assisted coding provided with the Blaise system: Stepwise, Alphabetic and Trigram coding. These methods can be combined or used separately, as required.

Stepwise coding only works with hierarchical classifications. This type of classification is useful in situations where sufficient information may not be available, to code to the lowest level of detail. In these cases a partial match to a higher level of the classification is sometimes possible. Blaise Stepwise coding supports this partial or 'incomplete' coding. Hierarchical classifications are also useful to distinguish between descriptions that may be similar at the lower levels but apply to different categories at the higher levels. For example an 'engineer' may belong to a 'professional' or a 'technical' or an 'trade' higher group. By working down from the higher level, the correct lower code can be obtained.

Trigram and Alphabetical coding are used to match a verbal or captured answer with a dictionary file containing classified descriptions.

In Alphabetic coding the answer is used to search an alphabetically sorted dictionary of descriptions. This list of descriptions is displayed starting at a point as close as possible to the entered description. This matching method works well where a single word or a fixed combination of words or abbreviations is provided. Country names such as 'Spain' or 'New Zealand' are a good example. The usefulness of an alphabetical dictionary can be increased by including reversed entries such as 'sheep farmer' and 'farmer sheep'. It can also be supplemented with common or standard abbreviations such as UK, USA, NZ etc. Very frequently occurring answers can be given a special code. The dictionary used for the 'country of origin' question, in our Migration card coding system includes a special code, 'A' for Australia. The operator has only to enter this 'quick key' letter to code some 50% of responses. Other 'quick keys' are also available for commonly encountered entries.

With Alphabetic coding there is also the ability to key in a sequence of letters and progressively refine the displayed descriptions. If say the first letter pressed is 'N' then the list displayed is positioned at the entries starting with N. If the next letter keyed is 'E' then the list is moved down to any entries starting with NE, and so this process of refinement can continue until the required description is displayed.

Trigram coding is similar to alphabetic coding in that the entered description is used to search for and display matched descriptions. However it is more flexible in that the order in which words are entered is not important and that spelling need not be 100% accurate. This means reversed entries are not needed and it provides a greater certainty that all the likely matched entries in the dictionary will be listed.

The overheads associated with the more sophisticated searching in Trigram coding can make it slower than Alphabetic coding. This may or may not be an issue depending on the size of the dictionaries, the size of the application and the hardware employed.

Stepwise coding can be combined with Alphabetic or Trigram coding. By coding one or more of the hierarchical levels of the classification using the Stepwise method, any subsequent Alphabetic or Trigram search is restricted to matches that contain the hierarchical levels already selected. Trigram coding executes much faster than Alphabetic coding when using this restricted search facility.

In my experience the performance of computerised coding systems rarely meets users expectations. In the learning phase the system performance is usually satisfactory. Subsequently, as skill levels increase, greater demands are placed on the system and it does not take long before the system is 'not fast enough'. To a certain extent this problem can be addressed by the operator tailoring their input to suit the system speed. For example entering in only the significant information from a response will result in a quicker match. This is because less searching and retrieval is required. However this approach usually leads to more choices being displayed and increases the likelihood of interpretation or incorrect selection. Entering in the whole description takes longer for the system to process but does result in less entries being displayed. If the entry of abbreviated answers is permitted the operators soon learn the appropriate amount to enter.

With Trigram coding there are several ways to optimise the system performance. During the dictionary preparation for Trigram coding any trigrams (three letter substrings) that occur more frequently than a predetermined threshold are discarded. This discarding threshold is maximum percentage of descriptions in which a Trigram can occur and still be included in the matching process. The default value is 10%. If the specified threshold is higher or lower than the maximum or minimum permitted, Blaise automatically adjusts it to the valid upper or lower limit. The value of the discarding threshold can only be changed by recreating the Trigram system file.

The discarding threshold affects the contents of the dictionary. A lower value reduces the size of the dictionary but may also reduce the chances of finding the match. Conversely higher values increase the size of the dictionary and the chances finding a match. However with a larger dictionary more irrelevant matches may be displayed. Determining an optimal value for any particular dictionary may require some experimentation. The number of similar words in the dictionary and the policy adopted with respect to the entry of abbreviated responses are factors in choosing an appropriate discarding threshold.

There are two other user adjustable factors that influence the contents of the screen display. These factors specify the fraction of trigrams that have to be common between the entered and the matched descriptions before they will appear in the search list or the extended search list. The developer can choose if items in the extended list are always displayed or only displayed on request. Using the default values a matched description will only appear in the search list if more than 50% of the trigrams are common with the entered description. Similarly if the extended list is activated those matched descriptions that have between 25% and 50% of trigrams in common will appear in the extended list display. These factors can be varied to suit the application. A low value will include more in the list and a high value less. The maximum size of the search list and the extended search list can be set to a value between 10 and 500 descriptions with a default value of 100. A lower value will use less memory and will run faster. However a higher value may cause more descriptions to be displayed and increase the chances of finding an appropriate match.

To improve performance of Blaise Trigram coding, the coding files can be loaded into any available memory. A function key is available to check this has occurred. Hardware optimisation can also yield valuable performance improvements. This subject is too complex cover here. However faster CPU's and more memory generally give better performance and this is especially so with Trigram coding. To effectively tune

CAC systems, performance needs to be monitored over a period of time to establish the degree of variability. The effects of any adjustments can then be measured in relation to these levels. Ad hoc fiddling may not result in any meaningful improvement.

5. Construction of coding files

There are many sources of data that can be used to construct the coding files. The classification itself will provide the basic material. This can then be supplemented with additional descriptions. The ideal source is the actual responses to the question if these are available. In Statistics New Zealand the results from question testing have been used along with samples of data from similar questions in past surveys. The descriptions captured for our 1991 Census Automatic coding process provided much useful material to improve our coding files. Where a description occurred more frequently than a minimum number of times it was included in the dictionary. Having a threshold level before a description is included is useful to prevent the dictionary from becoming overly large and slowing the search speed. Also answers with only minor syntactical differences are usually not worth including.

The matching methodology needs to be borne in mind when setting up the dictionaries. An obvious example of this is the need for reverse entries such as 'Accounts Clerk' or 'Clerk Accounts' in the dictionary used with Blaise alphabetic matching. With this method, the order of the words is significant and duplicating entries with the words reordered will improve the match rates. With matching methods such as the Trigram system words can be entered in any order and reversed entries in the dictionary are superfluous. More subtle effects of the matching methodologies need also to be considered especially when Automatic coding is contemplated. What is an obvious difference in meaning to a operator can sometimes meet the automatic acceptance criteria. For example 'service man' was mismatched as 'service manager' in our 1991 Census Automatic coding system.

Different words have different significance. Some add to the meaning of a description but are of little use by themselves in determining if a correct match has occurred. With Automatic coding it is undesirable to count this type of word as a match. The matching methods used in Statistics New Zealand have usually eliminated insignificant words such as 'a', 'of', 'and', 'the' from the dictionary. Certain words that usually occur frequently but are of little value in deciding if a match is correct were nominated as 'excluded' words. These were used in the matching process. However they could not by themselves constitute an acceptable match. This was an important factor in reducing spurious matches that might otherwise be incorrectly accepted or that might prevent the correct match from being accepted. Tuning the Blaise Trigram system to eliminate frequently occurring trigrams and adjusting the factors that determine the contents of the screen display improve the reliability of coding.

The management of the coding files while they are being used is an important issue. It is critical that the coding files contain only valid data and that they remain consistent throughout the coding process. When updates or changes are made it is possible to introduce errors. For example a description that what was being coded (or interpreted) to one code may be redistributed among others. The effects of any changes need to be carefully considered to avoid generating this type of inconsistency. The ideal strategy is to have the coding files correct to start with. As this seldom happens, the management of queries and subsequent additions or alterations is needed. Ensuring that the operators have a positive attitude to raising queries, and that conflicting expectations such as improving throughput do not get in the way of raising queries, help to reduce the likelihood of operators 'taking a guess'. The feedback obtained from queries enables deficiencies in the code file to be rectified. Over time this reduces the number of queries and the coding variability.

As answers to coded fields can only be entered with the CAC system the content of the coding file needs to be validated. Unfortunately, despite our best efforts in this regard, invalid codes have appeared in production. An automatic validation check of all code files has helped, and is recommended.

6. Where is coding best done?

The question about where coding is best done is not a simple one to answer. In theory if coding can be done while the respondent is available there is more likelihood of obtaining an accurate code. It is also more efficient to combine all data collection and coding into a single operation. However some of the issues discussed earlier make coding during an interview a less desirable option. For example the time taken to code a response may be unacceptable, especially if multiple interactions with the respondent are required. Also where an appropriate match cannot be located in the dictionary the interviewer may be tempted to interpret what the respondent means in order to avoid raising a query. In a field interview situation the consistency of support and level of training may not be as easily controlled.

These issues have been debated in our organisation. In principle it is accepted that coding and editing should be carried out during the interview where this is feasible in terms of the interview situation and data consistency. At the moment it is planned to only code straightforward questions such as Relationship during field interviewing. Responses for the more difficult to code questions will be recorded, and coded when the interview data is returned from the field. A study into the feasibility of interviewer coding will be undertaken when a base of in house coding is available for comparison. As our experience with CAPI increases it is likely that more complex coding will be undertaken in the field.

7. Possible extensions of the Blaise coding systems

There are a number of extensions that could be incorporated into the Blaise coding systems.

A useful option would be to would allow a match to be automatically accepted provided it meets some predetermined acceptance criteria. The major difficulty would be determining these criteria. With Trigram coding they could be based on the percentage of trigrams that are common between the entered and matched descriptions. For example, if descriptions were matched that had, say, 95% or more of the trigrams in common with the entered description, and only a single code, the match could be accepted without operator confirmation. The way discarded trigrams are counted would have to be considered. The threshold for acceptance would need to be adjustable to assist with tuning the system. Automatic coding might also be implemented in the form of a CADI batch process, using this approach.

The facility to return the text of the matched description has been requested in a number of our applications. As a number of descriptions can be associated with a particular classification code it is useful to return the actual description selected, to the Blaise program. This could overwrite the entered description or be available in a separate field. It would also provide users with a visual confirmation of what they have selected. This information would be helpful in studying coding patterns.

Imputation to the description entry field (.WORD) is now allowed. This is a useful enhancement which allows descriptions to be preloaded into the field. However in some circumstances it can also be necessary to impute to the code field (.CODE). For example where a code can be predetermined from previous answers. While this can be achieved by routing and computing HIDDEN fields it is a little cumbersome. Allowing imputation to the code field would be simpler.

The ability to accept incomplete codes is built into the Blaise coding module. While this feature is generally applicable, on some occasions it would be nice to be able to turn it off so that a full code has to be entered. Currently edit checks are used to determine that a full code has been entered, but this involves leaving and returning to the coding module.

8. Conclusion

The Blaise coding tools provide a straightforward means of implementing an effective computer assisted coding process in a Blaise questionnaire. The new Trigram system offers an especially powerful method of searching. Blaise provides a solution to most of the technical aspects of coding and leaves the user free to focus on the other factors associated with implementing CAC. With careful attention to these issues, Computer Assisted Coding improves the accuracy, consistency and efficiency of coding .