Method Series

Theme: Coding; interpreting short descriptions using a classification

12

Wim Hacking, Leon Willenborg

Statistics Methods (201204)

Statistics Netherlands

The Hague/Heerlen, 2012

Explanation of symbols

.	data not available
*****	provisional figure
******	revised provisional figure (but not definite)
x	publication prohibited (confidential figure)
–	nil
–	(between two figures) inclusive
0 (0.0)	less than half of unit concerned
empty cell	not applicable
2011–2012	2011 to 2012 inclusive
2011/2012	average for 2011 up to and including 2012
2011/'12	crop year, financial year, school year etc. beginning in 2011 and ending in 2012
2009/'10–2011/'12	crop year, financial year, etc. 2009/'10 to 2011/'12 inclusive

Due to rounding, some totals may not correspond with the sum of the separate figures.

Publisher
Statistics Netherlands
Henri Faasdreef 312
2492 JP The Hague

Prepress
Statistics Netherlands
Grafimedia

Cover
Teldesign, Rotterdam

Information
Telephone +31 88 570 70 70
Telefax +31 70 337 59 94
Via contact form:
www.cbs.nl/information

Where to order
E-mail: verkoop@cbs.nl
Telefax +31 45 570 62 68

Internet
www.cbs.nl

ISSN: 1876-0333

60165201204 X-37

Table of Contents

1. Introduction to the theme

Coding is an activity in the statistical process. It can be considered as a special type of derivation, and a rather difficult one. The purpose of coding is to match a code derived from a classification to textual information. The goal in this process is to reduce the large variety of answers to a convenient number, and to organise these answers (the classification used offers this option by means of its structure).

We view this matching as an interpretation of a description (the textual information) in the light of the classification concerned. An example is a description of an occupation (in a respondent's own words) that is interpreted taking into account the Standard Occupational Classification (SOC). Other examples concern descriptions of business activities, descriptions of education that people have, illnesses suffered by people, and causes of death. This summary is not exhaustive for the type of information coded at Statistics Netherlands, but it is illustrative for its coding activities.

Coding is also very similar to a doctor's diagnosis of patients who present him or her with various complaints and symptoms. The duty of the doctor is to diagnose an illness or abnormality based on a number of observations, answers from the patient and possibly additional tests (blood tests, for instance), and to select a treatment on this basis. See Hilden, Habbema, Bjerregaard (1978a,b,c).

The main reason why variables with open answers are used is that this is convenient for the respondent. Also, there is far less influence on the answer. For such variables, the person can answer with a personally formulated text. If the respondent were required to give an answer in the form ultimately needed by Statistics Netherlands to create statistics, then he or she would have to be knowledgeable about the classification that serves as the basis for such a variable, such as the Standard Occupational Classification (SOC), the Standard Industrial Classification (SIC), the Standard Educational Classification (SEC), etc. However, this is much too difficult, and from a practical point of view, impossible to expect from a non-specialist

For this reason, it was decided to have Statistics Netherlands deal with this problem, as this is where the answers provided will have to be interpreted in terms of such a classification. In the past, this invariably was done by human coders, specialists in coding of occupations, education, business activities, etc. The problem with this 'manual coding' is that it is time-consuming and expensive. Consequently, over time, increasingly computers have been used to assist in the coding. This ranges from computer-supported applications, where the computer is used to provide search facilities in a file with codes and their descriptions, to a fully automatic data processing application ('automatic coding'). To date, however, fully automatic data processing has not been feasible, and the question is whether it ever will be. But it is not a requirement that coding should be fully automated. Partially processing such information can already result in substantial efficiency gains. Another benefit is that automatic coding is bound to increase, for cases that are not too difficult, consistency of the answers (codes), without a loss of quality and possibly even with an improvement of quality. An additional advantage of automatic coding is that the coding process is well-defined, and reproducible; for computer-aided coding, an audit trail can render the process well-defined. Obviously, special efforts are required to make the coding software suitable for this purpose.

In automatic coding, there are two big problems that must be dealt with:

1. the interpretation of the written[1] text, and
2. the complex classification into which the descriptions must fit.

A computer program must choose which code best fits a description. The problem with written text is that many complications can arise, such as:

- Spelling problems
- Grammatical problems (relationships between words, syntax)
- Semantic problems (meaning of words, concepts, sentence fragments, a single sentence, several sentences)
- Interpretation problems (which code from the classification best fits a description).

A complication that can arise in conjunction with this last point is that, viewed from the classification, a description is incomplete, or that it relates to two or more issues having different codes. These complications may be due to the fact that a respondent is not likely to be familiar with the classification used, and therefore can provide ambiguous or irrelevant information, or information that lacks detail or is too detailed. Furthermore, it is possible that the classification has been set up purely from a theoretical perspective, without taking into account how it works as a tool for coding. For example, a classification expert's mental framework and/or vocabulary can be very different from that of the average respondent. The classification expert could also divide categories based purely on theoretical concepts, and not on empirical data, i.e. the frequencies with which codes appear in the population to which the classification relates.

The rest of this document addresses the problems and the solutions that are selected in practice, in particular at Statistics Netherlands.

The reader should be informed that Section 1.4 contains a list of definitions of the concepts used in document.

[1] What is meant here is primarily that the text is alphanumeric, not so much that the text is handwritten. In fact, it is better if the text is not handwritten, as this is an additional complicating factor.

1.1 General description and reading guide

1.1.1 Description of the theme

Coding an open-text question is a process of selection in which a decision is made to interpret an answer in terms of a predefined set of possible answers. This choice is sometimes made by respondents, during an interview or when filling in a questionnaire, possibly with an interviewer's assistance. However, this choice is also made afterwards by coders, at Statistics Netherlands, and usually with the respondent being absent and not available for answering further questions. Because this manual coding is a rather time (and money)-consuming process, automating the process is extremely worthwhile. This is known as automatic coding. In this process, descriptions (answers from respondents in their own words) are the input, and the output is a set of codes related to a certain classification.

Table 1. Possible places to code and by whom/what?

Who / what?	Where?	Advantages	Disadvantages
Respondent	Field	• direct feedback	• no knowledge of the classification
Interviewer	Field, CAPI of PAPI	• direct feedback	• superficial knowledge of the classification
Coder	Statistics Netherlands	• expert in the classification • can also use extra information that was included • in general, can interpret answers better than a computer program	• direct feedback not always possible (sometimes possible for businesses) • feedback is very time-consuming • coding may be inconsistent
Computer program	Statistics Netherlands	• fast, consistent coding • coding knowledge is specified in a system and is therefore transferrable • can operate day and night	• no direct feedback • only the relatively simple cases are coded (but that is often the bulk)

When respondents are permitted to give an answer in their own words, this gives them a lot of freedom. In addition, this prevents a situation where the respondents have to know the classification (which often requires specialist knowledge to be understood and used) or where respondents do not agree with the answer selection provided. A disadvantage, however, is that this must be followed by a rather expensive, time-consuming and error-prone coding process in order to code these answers. For that matter, it is highly questionable whether the answers provided always contain the precision and details that are desired or needed in order to code according to a given classification. To sidestep this problem, it is also possible to attempt to use a number of simple closed questions, and then to arrive at a desired code using a derivation scheme[2]. As a result, it is possible to exert influence on the desired type of information and the detail level of the answers. This type of approach is elaborated in Section 8.5.

Before answers to questions can be used to produce statistical results, coding is indispensible. As a matter of fact a kind of coding is also applied if a closed question is used, but in that case, it is the respondent who does the coding, and has to decide which answer is best among the possible ones. As a rule, coding can be done at different places in the data collection and throughput process steps, as indicated in Table 1.

In practice, combinations of the four options provided in Table 1 are generally always used. The selection of the options is often based on shifting the effort involved and the difficulties of the coding. The 'most convenient' approach depends on a large number of preconditions, such as:

- The *domain* or *area of application* of the question (including the 'hardness' / 'softness' of the question). 'Gender' concerns a harder piece of data than 'opinion about the government'. The first is more stable than the second and, furthermore, generally easier to indicate;
- The expertise of the respondent or the interviewer;
- The structure and complexity of the classification;
- The desired stability of the coding, i.e. how much or how often does the classification change over time?
- The number of respondents (or the net sample size);
- The input medium;
- The form of the source material: separate words, statements, short sentences, paragraphs;
- The desired balance between quality, output level and efficiency of the coding method;
- The desired speed ('throughput time') of the processing;
- The available budget;
- The desired detail of the coding results;
- The desire to make the coding process reproducible and transparent.

This document deviates somewhat from the normal usage of terms used at Statistics Netherlands. In this document, *coding*[3] refers to the activity with the goal of

---

[2] This route was chosen, for example, in the Clamour project, where a questionnaire (called DPQ) was developed to collect information on business activity and structure by using a number of closed questions. It is also the selected method for the coding of Education in PRAT.

converting descriptions (strings of symbols) to a code, originating from a classification. The way that this is done (manually, interactively / supported by a computer, or fully automatically) is not important here. At Statistics Netherlands, *coding* usually refers to coding by a specialist coder. In this document, this is called *interactive coding*. Coding with the help of a computer program is referred to as *automatic coding* (if the decisions about individual records are not taken by a person) and *computer-supported coding* (if, in a large part of the cases, the computer/an algorithm does not make any coding decisions but only presents suggestions to a human coder, or acts as an electronic reference file or index). *Coder* refers to a person that concentrates on coding according to one or several classifications. This could be a full-time coder at Statistics Netherlands, or *specially trained* interviewer in the field.

When coding descriptions, errors can be made, either by the coders or the coding program used. Insight into this can be gained through experiments (double blind coding), possibly depending on the detail level of the classification used.

In coding, both interactive and automatic, a consideration must continually be made between maximising the yield (the coding percentage) and maximising the quality (that is, minimising the number of errors). There is also a third maximisation to consider: the smallest possible effort (from the employer's perspective, to control costs, etc.), or the greatest possible human effort (from a coder's perspective, due to a coder's wish to retain his/her work). An important means of preventing incorrect coding is by establishing a *doubt category*. Traditionally, human coders were not permitted to have a doubt category (or only a very small one), but this was allowed for an automatic coding program. The records that are rejected by such a program because of difficulties encountered, are subsequently presented to human coders for coding.

Experience has shown that nearly every source and every coding contains a large fraction of easy records to code, and a smaller fraction of difficult records to code (this situation is often referred to by the 80% / 20% rule, but these percentages should not be taken too literally). Automatic coding focuses mainly on the easier fraction of records to code, which represents the bulk of the material to be coded.

The automatic classification techniques can generally be divided into two groups:

---

[3] At Statistics Netherlands, a distinction is sometimes made between 'coding' and 'typing'. Here, the term 'typing' is reserved for a more 'highbrow' activity, which requires specialist knowledge and experience and which utilises an extensive, complicated classification. In coding, there are often only a few dozen possible codes. Because this qualitative difference between typing and coding is not very important in this document, this distinction is not made here. 'Coding' and 'typing' are therefore considered synonyms in the present paper. We use the term 'coding' for both activities. In any event, 'coding' as referred to in this document does not have any relation to cryptography or communication theory. In that context 'coding' refers to an entirely different activity, related to secret or secure communication..

1. ***Language-based***: Here, we really look at the meaning of the words, and make use of language-specific attributes, such as grammar and the relationships between words and concepts (such as, synonyms, hyponyms, hypernyms, etc).

2. ***Statistical***: Here, descriptions are only viewed as a collection of words, which are often described by a sparse vector $Z = \{w_1, \cdots, w_n\}$, where n is equal to the number of words occurring in the vocabulary, and w_i the frequency of word i in the description. As a rule, the word order is not included as input for the classification. We could view this approach as classifying a house by first breaking it down and then looking at the stones in the pile of rubble. The assumption used here is known as the *bag-of-words* assumption.

These two approaches – interactive and automatic – are extremes. It is very well possible that, in practice, a mixed form will be selected. This could involve, for example, an approach with some ‘light grammatical pre-processing’, followed by automatic processing, and where the difficult cases are resolved using interactive coding.

1.1.2 Reading guide

Classifications form the basis for the interpretation of descriptions which form the inputs of the coding process. Before we focus on this, it is a good idea to take a moment to consider classifications. Their attributes can enable the coding process to be better understood, especially where to expect problems. Classifications are discussed in Chapter 2. A few known misconceptions with regard to coding are discussed in Chapter 3. There it is also explained why some common expectations often do not hold in practice.

Often a description must be pre-processed prior to the actual coding. This pre-processing enhances the subsequent coding. The pre-processing techniques used are described in Chapter 4.

The approach to a new coding problem is driven by the following aspects:

- *The material that is available*: Is there already coded material in electronic form? Methods based on already coded material are examined in Chapter 5. Methods that are applicable when there is no coded material are discussed in Chapter 6;

- *The coding method used*: interactive (Sections 5.2 and 6.2) or automatic / in batch (Sections. 5.1 and 6.1),

- *The intended quality of the coding* (Chapter 7).

Chapter 8 describes a few practical issues pertaining to coding. The report concludes with a list of references and two appendices.

1.2 Scope and relationship with other themes

The main theme of this document concerns the methods for automatic or semi-automatic (interactive) coding of answers to open questions. These are short descriptions (typically less than 10 words) in a respondent's own words formulated about the person's occupation, education followed, work performed, goods and services produced, etc. The code that is assigned to a description (if successful) originates from a classification. The classification itself is too complicated for respondents to directly search it for an answer. It is easier to let the respondent answer in his or her own words, and then to try to interpret this answer. Nowadays, this interpretation usually employs a computer if the material is delivered electronically, as we will assume. In the past, this coding was done completely 'manually'. That manual coding process was expensive, slow and untransparent. Nowadays, the goal is to have the bulk of the coding work done by computer running special coding software. The remaining 'difficult cases' are then resolved more or less 'manually' as in the past.

The main intention of the present document is to provide an overview of relevant aspects that play a role in this automatic coding process. Because classifications play an important role in coding, we also discuss this subject here. In addition, we examine alternatives for coding. Automatic coding has a number of problems, including the treatment of texts, searching for a suitable code value, and if that is not successful, starting possible follow-up actions to elicit extra information so that a code value can be found.

Most parts of coding are rather technical in nature, and are only touched upon in the present document. For a more in-depth examination of parts of the coding process the reader is referred to specialized references. Automatic/semi-automatic coding is a multidisciplinary specialist area that overlaps with linguistics, artificial intelligence, machine learning, statistics (classifications), and the cognitive and social sciences. These last two specialist areas are relevant in connection with the art of asking questions, the use of the right questionnaire, etc. This aspect of the subject is, however, not discussed in the present document.

As stated above, coding has ties with several other disciplines. We now briefly discuss four applications that have something in common with coding: that they process written natural language or because they can 'reason'.

- ***Matching***: In official statistics, matching is used to enrich a file with additional information from another file, or to confront files from different sources with each other, for example, to check and possibly improve the quality of the data. Various kinds of variables can be used as matching variables, including string variables, such as people's names, company names or addresses (street names). For string variables spelling variations must be taken into account to handle of alternative ways of representing names (Dorpsstraat / Dorpsstr.) or alternative spellings that are phonetically the same (for example, in Dutch, Jansen / Janssen / Janszen are pronounced the same, as are Hendriks / Hendrickx / Hendrikx). Similar problems are encountered in coding. For information about matching, see the Methods Series report on this subject (Willenborg and Heerschap, 2009).

- ***Spam removal***: This involves a collection of e-mail messages, of which the user has indicated whether or not a message is considered spam by him. Based on this collection – called a 'training set' – a classification model is derived for the assessment of future e-mails. Incidentally, this is an example of a classification that consists of two or three classes ('spam', 'not spam', 'possibly spam'). In practice, spam filters also use various other types of information, which does not come from the message text. The practical situation has demonstrated that, without this extra information (such as: Does the sender have an existing address and IP number? Are hundreds of copies of the message (or more) being sent at the same time? Etc.), the spam filters do not discriminate sufficiently.

- ***Expert systems***: These make use of inference engines (which can 'reason') and knowledge bases (where the basic knowledge about the application domain is held). In a limited number of cases, expert systems are also used as a coding tool (Chen et al., 1993). This approach proved to be labour-intensive and often not very robust. The use of expert systems in coding ia beyond the scope of the present document.

- ***OCR[4] and ICR***[5]: This involves the problem of enabling a computer to recognise an image: the digital image of printed text, with all the letters, figures, punctuation, etc., present. First of all, this involves the recognition of individual characters. Second, this concerns entire words, as a check. This transformation from image to characters takes place using pattern recognition software. ICR involves converting handwritten text, instead of printed text, to characters. To obtain good results, the software used for this purposed must be 'trained' using handwritten texts and their transcriptions.

1.3 Place in the statistical process

At Statistics Netherlands, coding is done in the input phase or the throughput phase (see Table 1). The actual place depends on the situation at hand. Sometimes, there is no choice to be made, especially when the source material is provided by a third party (such as for the Causes of Death Statistics, or for business activities originating from the VVK[6]). In the case of a survey, there is a choice, and a consideration must be made as to which method is selected.

At present, automatic/semi-automatic coding takes place for the following areas at Statistics Netherlands:

---

[4] OCR is Optical Character Recognition

[5] ICR is Intelligent Character Recognition

[6] This is an abbreviation for the Dutch Association of Chambers of Commerce (*Vereniging van Kamers van Koophandel).*

- Social statistics: education (SOI, ISCED[7]), occupations (BWC, ISCO), companies (SIC), articles, shops (Budget Study): this is done semi-automatically in the field for the electronic surveys, and also at Statistics Netherlands using the COBS system (fully automatic and interactive coding).
- Business statistics: the SBI (Dutch SIC) for companies. A large part of the semi-automatic coding takes place at the Chambers of Commerce. The responsibility for the content, however, is the responsibility of Statistics Netherlands.

The following subjects are not yet coded automatically or semi-automatically:

- Social statistics:
    - **Vacancy survey**: the input here is a short description of the education (usually only the level, such as MBO) and occupation; the SBI of the company is also available (see De Heij, 2002).
    - **The Dutch Parliamentary Elections Studies** (*Nationaal Kiezersonderzoek - NKO*) has an open question into the 'most important problems in the Netherlands' (Hacking, 2009). Automatic/semi-automatic coding is used here.
    - **Causes of death**: internationally, significant attention has been directed towards the automatic coding of causes of death. This was initially only done in English. However, at present, automatic coding also is done in at least the following languages: Swedish, French, German, Hungarian, Italian and Spanish. One of the problems in the Dutch situation in the Causes of Death Statistics is that the textual descriptions are not available in electronic form; however, the code is electronically recorded at Statistics Netherlands during the manual coding process (see Van den Meijdenberg and Kardaun, 1997.)
- Economic statistics: there was a pilot study (Smeets, 2007) that used rule-based coding in Manipula (part of Blaise) for a substantial part of the imported goods. The problem here is that a – selective – long list of light goods remains, which has not yet been coded.

[7] SOI, ISCED, BWC, ISCO and SBI are classifications; see the definitions and explanation in Section 1.4.

1.4 Definitions, acronyms and abbreviations

Table 2. Explanation of concepts and abbreviations relating to coding

Concept	Description
Adjacency matrix	0-1 matrix that indicates which nodes in a graph (or a digraph) are connected by an edge (or an arrow).
Automatic coding	Coding (in batch) using a program. The program takes all of the decisions.
Automatic coding yield	The percentage of descriptions that is automatic coded and, furthermore, correctly coded, as demonstrated after verification. Of course, the yield depends on the descriptions provided.
Bag-of-words assumption	The assumption that, for a description, only the separate words that occur play a role, and not the order and the combinations of these words in the description.
BWC	An abbreviation for BeWerker Codes ('processor codes') an interim code for the coding of occupations.
CAPI	Computer Assisted Personal Interviewing.
CATI	Computer Assisted Telephone Interviewing.
CAWI	Computer Assisted Web Interviewing.
Classification scheme	A hierarchical arrangement of kinds of things (classes) or groups of kinds of thing (definition from Wikipedia, English edition)
Clean string	See: cleansed string.
Cleansed string	The result of a series of grammatical pre-treatments on a raw (untreated, observed) string. The coding program can immediately use a cleansed string for coding.
COBS	The Dutch abbreviation for a computer-supported processing system (Computer Ondersteund BewerkingsSysteem). A system for the automatic and semi-automatic coding of the variables of education, occupation, company, shop type and article type.
Coder	A specialist trained to interpret and classify descriptions (in a certain area) in the light of a classification used for that purpose.
Coding	The activity in the statistical process in which it is determined whether a code from a classification can be assigned to a description, and, if so, which code this could be.
Computer-supported coding	A form of coding in which a coder makes all the coding decisions, possibly while using an electronic file or index.
Corpus	Coded set of descriptions.
DAG	Directed Acyclic Graph, a directed graph without cycles, i.e. without paths that have the same beginning and end node.
Doubt category	A category that can be used if a description cannot be classified with sufficient certainty. The same or other coders can review the descriptions designated as such at a later stage in the process.
Fuzzy string matching	The comparison of two texts, for which the outcome (usually) is a scalar that indicates the extent to which the texts are similar.
Good-Turing frequency estimation	A technique for predicting the probability of occurrence of objects belonging to an unknown number of species, given past observations of such objects and their species. (from: Wikipedia).
Hypernym	A generalisation of a term or a more general term. Opposite of 'hyponym'.
Hyponym	A specialisation of a term or a more specific term. Opposite of 'hypernym'.
ICD	International Classification of Diseases and Health Related Problems that is used for causes of death and the classification of dieases and illnesses.
ICR	Intelligent Character Recognition. This concerns the automated recognition of hand-written text.
Interactive coding	Coding using an interactive program, which presents the necessary background or other information to a coder, who makes all the coding decisions. The program also processes the answers (and the possible reason for the choices as indicated by the coder).
ISCED	International Standard Classification of Education.
ISCO	International Standard Classification of Occupation.
Key word	Word in a description that is usable for coding, in contrast to a stop word.
Levenshtein distance	Distance measure between two strings, defined as the minimum number of mutations needed to transform one string into the other. A mutation is one of three operations: insertion, deletion or substitution of a character/l into a stringl.
Manual coding	Coding performed by a coder, without substantial support from a program.

NKO	Nederlands Kiezers Onderzoek (Dutch Election Survey)
NLP	Natural Language Processing.
NSTR	
OCR	Optical Character Recognition. Automated recognition of printed text.
PAPI	Pencil And Paper Interviewing.
POS tagging	Part-of-speech tagging. Grammatical parsing in which text is broken down into types of words such as noun, verb, etc..
PRAT	The Dutch abbreviation for Programma Anders Typeren, a module in Blaise that allows a code to be determined interactively for an answer to an open question.
PRODCOM	A classification of industrial products
Raw description	Description recorded in an interview or specified by a respondent and that has not been (thoroughly) checked. This may contain various errors, along with insufficient or unnecessary (stop words) information. This is why descriptions are first subjected to several grammatical treatments. This creates a clean or cleansed string, which is used for automatic coding. This string is not intended to be readable, but is utilised as input for the coding program used.
Regular expression	A regular expression (abbreviated as “regexp”, “regex” or RE) is a method to describe patterns (in strings) to enable a computer to recognise text. Regular expressions have a formal syntax, which is largely standardised.
SBC	This is the abbreviation for the (Nationale) Standaard Beroepen Classificatie (in Dutch), or Standard Occupation Classification (see also ISCO).
SBI	see SIC
Semantic network	A network(or grph) consisting of words and concepts and semantic relationships between them. Examples of such relationships are synonyms, hypernyms and hyponyms.
Semi-automatic coding	Synonymous with computer-supported coding.
SIC	Standard Industrial Classification; in Dutch SBI (Standaard Bedrijven Indeling) is used.
SOI	This is the abbreviation for the (Nationale) Standaard Onderwijs Indeling, the Dutch version of the Standard Classification of Education (see also ISCED).
Soundex	Indexing technique based on the sound (or pronunciation) of words (and not how they are written), originally only for English, but later developed for Dutch as well. See also Appendix B.
Spreading activation	Method to search in a semantic network.
Stop word	Word in a description that does not contain any information or contains too little information, because it occurs too frequently. A stop word can therefore be deleted by an automatic coding system.
Subclass	See Hyponym.
Superclass	See Hypernym.
Synonym	Word or concept with the same meaning as another word, possibly in a special context.
Tokenising	Dividing sentences into words or n-grams (‘tokens’).
Topological sorting	The assigning of numbers to the nodes in a DAG so that the starting node of an arc (directed edge) always has a lowerassociated number than the one associated with ending node of that arc.
Training set	A corpus where the codes linked to the descriptions are verified. The codes originate from a classification. A training set is used in the coding methods that are based on supervised classification.
Trigram	String consisting of three consecutive characters. They are used in fuzzy string matching. The more trigrams two strings have in common, compared to the trigrams they have not in common, the more similar they are.
Typing	See Coding. This document discusses coding almost exclusively, for which ‘typing’ is considered as a synonym. In practice, at Statistics Netherlands, there is actually a difference between the two: ‘typing’ concerns the more difficult coding cases in manual coding.
VVK	The Dutch Association of Chambers of Commerce. VVK means Vereniging Van Kamers van Koophandel
Zipf distribution	An empirical distribution originating from linguistics, in which the frequency of a word in spoken language is roughly inversely proportional to its rank in the frequency table. So, for instance, the most common word appears roughly twice as often as the second most common one, The Zipf distribution is a discrete variant of the Pareto distribution.

2. Comments about classifications and coding

Before we discuss the main theme of this document, the automatic coding methods for descriptions, we want to take a moment to examine classifications in this section. A classification provides the codes that should be used to be associated with the descriptions provided by respondents, (if this is possible, which is not guaranteed).

2.1 Examples of classifications

Several classifications play a role at Statistics Netherlands, such as the following (the Dutch abbreviations are used below; they are explained in Table 2):

- SOI – Standard Classification of Education
- SBI - Standard Industrial Classification
- SBC - Standard Classification of Occupation
- ICD – illness and causes of death
- NSTR, PRODCOM – classifications of goods
- Problem classification in the NKO (Dutch Parliamentary Elections Studies).

This list is meant to be indicative, not to be exhaustive.

2.2 Classifying principles

In practice, classifications must generally be considered as given, only to be changed by special committees responsible for their maintenance, which are also often international in nature. This means that making any changes (if changes are possible at all) is a very slow process. This can be bothersome if a classification was not created with actual *usage* in mind, such as is the case in coding. Distinctions that are interesting purely from a more theoretical perspective can be difficult, or even impossible, to make in practice.

It is important that clear classifying principles form the basis for a classification. For example, in the classification of education, there are several different dimensions used to characterise educational programmes, such as subject, level and specialisation. We call these classifying principles. They can be used for a systematic derivation of the associated classification of education (see also Section 6.2).

In coding, use can be made of classifying principles that form the basis for a classification, such as the different dimensions that could play a role. It would also be a good idea to explicitly describe these classifying principles with a classification. Unfortunately, in practice, these kinds of principles are not always explicitly formulated, which means that one has to make guesses about them. It is also possible that a classification is set up based on clear principles, but that the practical situation forces compromises to be made, or even forces some principles to be violated. This can certainly be the case when there are multiple stakeholders involved in drawing up and maintaining a classification. These parties may each be trying to achieve their own objectives with the same classification. This may lead to solutions that are not favourable to all parties (and, in extreme cases, not favourable to any of the parties).

Sometimes, several of these classifying principles seem to be used simultaneously. That can occur when people are trying to use the classification to mirror the situation in practice. An example of this concerns the SBI, the Dutch Standard Industrial Classification. In the SBI, a distinction is made between the wholesale trade in outerwear (SBI 51421), work clothing (SBI 51422) and underwear (SBI 51423). Elsewhere in the SBI, a distinction is made between clothing for women, men, babies, toddlers and children (this distinction is important in the clothing trade, both wholesale and retail). Section 2.6 discusses this concept in more depth.

Initially, one may perhaps have expected a single division in clothing, which applies for both manufacturing and retail. However, the division according to clothing type is also different in practice. For example, the manufacture of underwear for women, men, children, etc., all takes place in a single factory. A similar situation applies for outerwear. However, for the retail trade, clothing is often grouped based on gender and age (clothing shops for adult men, adult women, sometimes combined) or primarily based on age (for children – possibly further divided into babies, toddlers and teenagers – and for adults, with another possible subdivision based on gender).

2.3 Classification structure

The structure that is usually defined for the set of categories in a classification is a tree.[8] This type of structure can also occur in multi-dimensional or multi-axis classifications, where each dimension consists of a separate tree structure. We will further examine these tree structures in this section.

A classification is a finite set C of categories for which a hierarchical structure is usually defined, i.e. a tree structure (see Figure 1).

---

[8] A tree has the characteristic that each node has 0 or 1 parents. (A node without a parent is a source node. In principle, there can be multiple source nodes, but in practice, there is a single source node in a classification.) In a tree, the path from a node to the associated source node is unique. In theory, it is possible that there are classifications in which a node has more than one parent. ‘Cars with two-stroke engines and automatic transmission’, are a subclass of ‘cars with two-stroke engines’ and also of ‘cars with automatic transmission’. A condition here is that the structure is not cyclical, so that classes have a clearly defined order (such a DAG represents a partial order). In that case, we have a directed acyclic graph, abbreviated as DAG. Because DAGs do not seem to be used in classifications (justifiably or not), we will not discuss this topic further here.

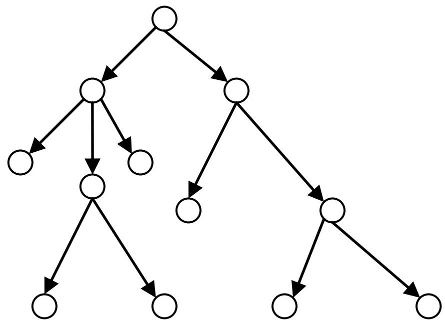


Figure 1. Example of a directed tree

We use a directed graph to represent such a classification. The arrows point from a higher abstraction level to a lower one. If **a** is a category in a classification from where an arrow points to a category **b**, then we call **a** the 'parent' of **b** and, conversely, **b** a 'child' of **a**. If the structure of the classification is in the form of a directed tree, each child has no more than one parent. The nodes without parents are called 'sources' or 'source nodes'. In a tree describing a classification, in practice, there is usually only one source node.[9] Without loss of generality, we can assume that the directed trees in this document all have exactly one source node. For a directed tree with a single source node, there is exactly one path from the source node to any particular node.

In the case of a classification tree, a splitting principle applies for each category: if this category occurs n_c times in the population (or sample), and each of the child categories i occurs $n_{c,i}$ times, then $n_c = \sum_i n_{c,i}$. Here, it is assumed that there are no missing values in all observations (records) for all of these categories.

In a classification based on a tree structure, it is possible to assign codes to the nodes (or: vertices) such that they reflect this structure. Figure 2 provides numbering for the tree in Figure 1.

[9] If there are multiple source nodes, it is always possible to add one node with references from this node to the original (or other) source nodes. In this case, this new node is the only source node in the new tree.

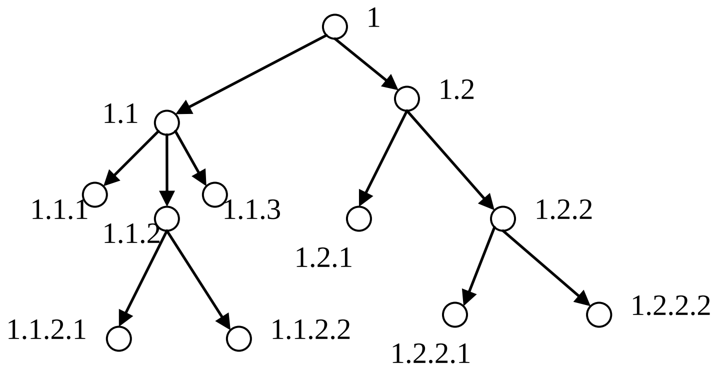


Figure 2. Example of a directed tree with labels for the nodes

The idea is that the source node is assigned the number 1, and for each subsequent generation, a decimal is placed after the label from the parent, and this is followed by a sequential number m representing the child in N. So, if the parent has a label *a*, then the children receive from a the labels *a.1, a.2,…,a.k*, where k is the number of children of *a*.[10] The total number of nodes in a label can be regarded as the shortest distance to the source node. Note that the numbering represents the – unique – path to the node in question from the source node.

A useful operation is to omit the information from a label from a certain decimal (including that decimal): it generates the labels of the ancestors of the node.

Classifications consist of a set of categories, which also have a relationship among themselves. This relationship moves from general to more specific (i.e. in the direction of the arrows). The numbers (frequencies) that pertain to the total (of the parent node) are divided among the 'children' nodes. In Figure 3, this is represented for the tree from Figures 1 and 2. For each node with 'children', the value associated with that node is equal to the sum of the values associated with the children of that node.

[10] The dots cannot usually be omitted, as the notation then may become ambiguous. After all, 127, can mean: 1.2.7 (the 7th child of the 2nd child of the 1st source node) or 1.27 (the 27th child of the 1st source node), or 12.7 (the 7th child of the 12th source node) or 127 (the 127th source node). If certain limitations apply, however, this situation is sometimes possible. For example, if a parent never has 10 or more children, a decimal notation is used for the numbers in the labels, and there is a single source node. In that case, 127 is the same as 1.2.7.

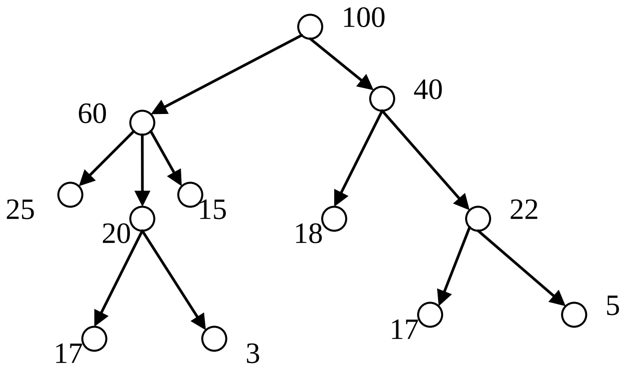


Figure 3. Example of frequencies per category in a tree structure

2.4 Special qualities of classifications

In this section, we describe several peculiarities that can occur in classifications. In the current Dutch standard industrial classification, we have a situation where the category of ‘clothing’ can be split in different ways, depending on the context. The ‘manufacture of clothing’ is split into the ‘manufacture of outerwear’ and ‘manufacture of underwear’. However, in the clothing retail trade, this category is split into: ‘retail trade of women’s clothing’, ‘retail trade of men’s clothing’, ‘retail trade of children’s clothing’ (and, perhaps, also the ‘retail trade of baby clothing’).

Another example, also from the Dutch standard industrial classification (SBI-93), concerns agricultural products. The activity associated with these products comprises the splitting level of these agricultural goods. This concerns the following:

- Cultivation of vegetables. (Additional detail about these vegetables is not necessary.)
- Wholesale trade in potatoes for seed and potatoes for the retail market. (A split into the type of potato is necessary in order to code the type of wholesale trade.)
- Processing of potatoes. In other cases, different splits can occur.

In the Dutch standard industrial classification (SBI-93), different splits of clothing are possible:

- According to age: clothing for babies, toddlers, children, teenagers and adults.
- According to gender: women’s and men’s clothing.
- According to how it is worn: underwear and outerwear.
- According to use: work clothing (including uniforms), leisure clothing, clothing for going out, clothing for formal events (for weddings, for

academic events, such as receiving a PhD, for a fancy ball, receiving a medal, etc.), and everyday clothing.

Depending on the industry sector, the above splits may or may not apply.

2.5 Classifications and coding

The nature of a classification has an influence on the coding process. This 'nature' partially relates to the structure of the classification, but primarily also to the categories themselves: how easy or difficult are they to characterise? How far apart, conceptually, are the categories? If there are two categories that are rather close together, then, in practice, it will be difficult to make a distinction between the two, based on descriptions. In this case, the descriptions must be quite precise. This can be difficult for a respondent who is not considered to be familiar with the classification, because this person will not be aware of a – probably quite subtle – difference between the two categories.

For a number of reasons, a classification can present difficulties when coding descriptions that relate to it. Possible reasons for this could be:

- The categories cannot clearly be distinguished;
- The categories are rare in the population;
- There is not very much empirical material available to describe the categories, or the empirical information is not sufficiently diverse;
- There are categories that are close together, and therefore it is difficult to distinguish between them;
- The categories are very clearly defined and also occur in practice, but they are not actually used in practice because nobody uses the associated distinction. Consequently, Statistics Netherlands never receives any information about such categories.[11]

These different causes also require different solutions that, however, are not always available. After all, there are more issues that play an important role in official classifications than these methodology-related matters. Usually, a classification was already officially established at an earlier date. This must be viewed as a given for the coding process. Changes to a classification generally are made with regards to the subject matter itself and not with observation/measurement in mind nor,the coding problems that the classification poses when used in practice. It would be preferable if a classification was set up also addressing these issues. Experiences could then be used to adapt a classification and make it useful and applicable. There

[11] For example: A distinction is usually made in the field between benign and malignant cancer (with continued growth as the criterion). However, this distinction is not made in the case of brain tumours, because, with this type of cancer, a 'benign' tumour is usually also fatal.

is little sense in retaining a theoretically ideal classification that cannot be used in practice due to observational or coding problems.

Sometimes, a single classification is used in different areas of application; as a result different demands from the statistical divisions have been crammed into a single classification. An example of this is the use of the same classification for both causes of death statistics and hospital admissions.

2.6 Estimation problems

For certain coding techniques (see, for example, Section 5.1.4), estimations of the frequency of occurrence in a population of certain words are required. This can be a difficult problem if these frequencies are small .This problem is not limited to coding. Small frequencies are usually dealt with by using specific statistical models. Small area estimators, for instance, have been developed to deal with a very similar problem, namely areas with relatively few people living there. Other areas where similar problems occur are linguistics, when estimating the size of a person's vocabulary or in ecology, when one is interested in the number of species of animals living in a certain areas, or the number of different plant species growing in a jungle, say..

Because these problems are mainly specialised and technically rather advanced, we only provide reference to part of the relevant literature for interested parties. The articles here are not the most recent, but the list includes a number of classics in this field: Bunge and Fitzpatrick (1993), Efron and Thisted (1976), Good (1953), Good (1965, Chapter 8), Hill (1974), Sichel (1975, 1986a, 1986b).

3. Misconceptions

Before we begin discussing the actual subject – coding – of this document, we want to point out several widespread misconceptions that seem to be rather persistent. These have to do with the quality of the input, certain attributes of the classification used, and the relationship between the input (short descriptions) and the output (codes from a classification).

3.1 Misconception 1: 'Low quality input versus high quality output'

This misconception conflicts with the generally known saying: 'garbage in, garbage out'. We must not turn a blind eye to this situation. If we do, and accept low-quality descriptions, then highly detailed coding will in some cases produce false accuracy: a vague description will indeed be assigned a very precise code, but without substantiation from the associated description.
In addition, it is also possible that a classification distinguishes between codes/subjects that seem the same to a 'naive' respondent. In this case as well, input data is obtained that does not offer adequate information for correct and sufficiently detailed coding.

The remedy for this could be as follows (while interviewing): in the event of vague / ambiguous texts, one can permit an appropriate code (a 'doubt category', or a less detailed code) or multiple (detailed) codes in instead of a single code, possibly with a probabilities assigned to the possible codes.

3.2 Misconception 2: 'Less detailed codes and therefore a higher yield'

In a number of cases, the codes used for coding are classified hierarchically (for example, the SBI), where a code XYZ details code XY. Inspired by the dilemma described in the previous section, the choice could be made to code to a less detailed level, with the hope of increasing the yield (the number of successful and correct coding instances). However, this is not necessarily the case, for two reasons.

First of all, the link between a description and a less detailed code is not necessarily less (or much less) ambiguous: if we would code all the occupations in the government a code 'occupation in the government', this would not necessarily simplify the coding. In addition, experiences with PRAT (see also Section 5.2) with regard to automatic coding have also led to this conclusion.

Second, the use of words used in a certain area of application (jargon) is usually distributed in a very skewed manner. In this case, a distribution that often applies is the Zipf distribution, in which the frequency of a word in spoken usage is roughly inversely proportional to the rank of the word in the frequency table. This distribution is fairly skewed.

Such a skewness , will, as a rule, also manifest itself in a corpus, a coded set of descriptions (according to a certain classification). In an occupation classification, certain occupations are much more frequent than others in a population.

The skewness in a corpus[12] is not necessarily limited to one level, but can manifest itself on multiple levels. For the practical situation, this may imply that we can obtain a reasonable coding result with relatively little effort (especially where this concerns the frequently occurring codes), while a relatively large amount of effort must be put into the remaining part. If the coded corpus has a skewed distribution, then, typically,, there are a few classes in the tail of the distribution for which too few examples are known to make a reliable and complete classification or classification model. This skewdness is often still present when less detailed coding is used, because it can occur on multiple levels. In Figure 4, for example, the skewedness occurs at the two lowest levels.

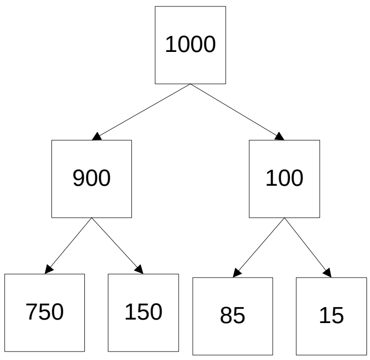


Figure 4. Example of an asymmetrically distributed corpus at all levels

This skewedness may be due to classifications that are designed in a rather unbalanced way. It is also possible that the they become more skewed due to changes in the population: certain occupations gradually disappear, while others start to emerge. If the classifications are insufficiently maintained, a situation may arise where categories are either very well or very poorly ‘filled’. In principle, it should be possible by means of combining or splitting categories, to create a more balanced classification. However, because classifications are usually standard objects that are not easily changed, this evolutionary process may take considerable time, effort and discussion and typically involves many stakeholders, who also may pursue other interests.

3.3 Misconception 3: ‘80% automatic coding is attainable’

The general opinion is that coding is an easy task: ‘It is no problem to see what code that is’. However, experience at Statistics Netherlands has shown that a yield of 40% of automatically coded records is a more realistic figure than the 80% claimed.

---

[12] In practice, we encounter skewedness in the corpus as well as in the codes. Both manifestations of skewedness may be different.

The yield strongly depends on the complexity of the coding problem, for which the two points discussed above (in Section 3.1 and 3.2) play a role.

This also involves a difference in definition: if the classification literature for example refers to, for example, a percentage of 70% coded records, then this means that, of the 1000 codable texts, some 700 were able to be correctly coded automatically. However, when the system must not only just code, but also ‘guarantee’ the quality of the coding, then the coding yield drops significantly. See also Chapter 7.

Another frequently occurring reason for overly optimistic estimations in the literature is that experiments have only been performed with codable descriptions, and that the non-codable descriptions have been ignored. This phenomenon is also discussed in Chapter 7. It also plays a role in the validation of a coding system.

4. Treatments of the raw description

Before the description provided by a respondent –the 'raw description'– can be coded, it must first be subjected to a number of grammatical treatments. In this process the description is treated as a string of symbols. These operations are intended to change the string into such a form that coding the pre-processed string is easier than coding the actual raw description.

We start with the raw descriptions, as recorded during a survey ('in the field'). They may contain a variety or errors or spelling deviations from the one used by Statistics Netherlands in the coding. Writing errors can occur;alternative spellings for words, , may have been used; unknown or unrecorded abbreviations or acronyms; words may have been concatenated as composite words, when they should actually have been written separately (a frequently occurring phenomenon in Dutch) or vice versa, etc. Detecting and eliminating these types of errors is typically the work of a spell checker which, in this case, must be geared towards the area to which the coding relates. In particular it should be able to handle specialised vocabulary and jargon in different areas in the field of application.

In the event of handwritten input, careless errors can occur such as the use of symbols[13], non-standard abbreviations and unreadable text phragments, for instance occurring at the ends of words. These problems cannot be resolved using a regular spell checker. In these cases the input may be considered to consist of incomplete strings or 'partially missing strings'. The non-standard abbreviations must be added as synonyms[14] (in the hope that they will occur again later, and can then be recognised), and the unreadable words or ends of words must be considered and treated as strings with missing values.

After the raw text is pre-processed and cleansed, the coding information in the description can be examined. Stop words could be removed because they do not contain any or sufficient information, or they could be left and will be ignored. Words and phrases must be recognised, which means that they must be related to words and phrases that occur in the descriptions used for the codes in the classification. This is done by making use of various semantic relationships that can exist between the words (synonyms, hyponyms, etc.), as will be explained later.

---

[13] Symbols are characters of a larger alphabet containing the target alphabet, and not belonging to the target alphabet. Text that uses at least one such symbol generates the same problems as partially missing text.

[14] Note: spelling errors should preferably not be solved this way: either use a spelling checker (in an interactive situation) or fuzzy matching (in a non-interactive situation).

4.1 General grammatical treatments

When we talk about ‘cleansing’ a description, we mean subjecting them to certain operations which reduce them to strings with characters in the target alphabet. We provide some examples of such operations

- non-standard characters are replaced by spaces (for instance such characters as @, or #);
- letters with accents, diacritical marks and other glyphs, etc. are replaced by the corresponding letters without these additions (for example á, è, ç are replaced by a, e, c, respectively);
- ligatures are replaced by their composite letters (for instance ‘œ’ by ‘oe’);
- capital letters are replaced by the corresponding lower-case letters (for example B, C, Z by b, c, z, respectively),
- non-informative words are replaced by spaces (for example, stop words like ‘a’ or ‘the’ in English),
- to reduce the variation in ways that words are written or spelled (for example, abbreviations of words can be replaced by their full equivalents (for instance ‘Ltd’ by ‘limited’).

These operations can sometimes be performed ‘mechanically’, without any necessary preparations (for example, the operations in Sections 4.1.1 and 4.1.2). In other cases, however, a great deal of preparation is required to do this, such as when replacing words by a suitable spelling variation.
To illustrate these operations a bit more, several specific examples are provided below.

4.1.1 Replacement of symbols by terms

If symbols occur in the description, then they are replaced in this step by terms that express the meaning in words. Examples of such substitutions are:

- ‘=>’ replaced by ‘is followed by’ or ‘implies’
- ‘+’ or ‘&’ replaced by ‘plus’ or ‘and’
- ‘/’ replaced by ‘or’
- ‘<‘ replaced by ‘less than’
- ‘ ≤ ’ replaced by ‘less than or equal to’
- ‘>‘ replaced by ‘greater than’
- ‘ ≥ ’ replaced by ‘greater than or equal to’
- ‘/=’ of ‘<>’ replaced by ‘not equal to’
- ‘€’ replaced by ‘euro’ (the same applies for other currencies).

The above list is not exhaustive, but should provide some typical cases.

4.1.2 Replacement of non-standard signs by spaces

Replacing the remaining ***non-standard characters*** by spaces, for example:

- ‘…’, ‘!’, ‘?’, ‘??’ , ‘???’ ‘@’, ‘#’, ‘%’, ‘^’, ‘*’ replaced by ‘ ’ (space).

This means that these non-standard signs will be eliminated from the description, not by just omitting them (because that could lead to words being inadvertently combined), but by replacing them with a space that does not contain any further information.

4.1.3 Replacement of abbreviations and acronyms

Expanding **abbreviations**, and homogenising acronyms. These can be very specific to the area of application. Examples of this are:

- ‘gen. manager’ replaced by ‘general manager’;
- ‘Prof.’ replaced by ‘professor’ (see also Section 4.1.4)
- ‘ltr’ replaced by ‘litre’.

Abbreviations that do not contain any information can be replaced by spaces. For example:

- ‘etc.’ replaced by ‘ ’ (space).

For that matter, it is not always easy to clearly ‘expand’ abbreviations. Sometimes, the context is needed to use the correct substitution. An example of this from the medical field is: *inf.* This can mean: ‘infection’, ‘infectious’, ‘infarct’ or ‘inferior’. In another context, the same abbreviation could also mean: ‘infinite’, ‘infinitum’, ‘informal’ or ‘information’.

4.1.4 Substitution of letters

Converting *capital letters or lower-case letters*, and *replacing diacritical characters by their non-diacritical equivalents*. Examples of this are:

- ‘ENT’ replaced by ‘ent’[15]
- ‘ä’, ‘à’ replaced by ‘a’
- ‘ç’ replaced by ‘c’.

4.1.5 Substitution of stop words

Removing frequently occurring words that contain little or no information. Examples of this are:

- ‘a’, ‘an’, ‘the’, ‘of’ replaced by ‘ ’ (space).

In addition, words occurring more than once in the text are also often removed because they do not usually provide any extra information.

---

[15] In ENT doctor = Ear, Nose and Throat doctor, also known as an otolaryngologist. Based on the substitutions referred to in section 4.1.3, writing out the acronym ‘ENT’ in full can also be considered. But given that this does not serve a clear purpose, it is better not to do this.

4.1.6 Splitting composite words

This applies to languages, such as Dutch and German, where new words are formed by concatenating two words and form a single word. The treatment involves ***splitting*** long words into their original components. Many composite words are not found in search files or dictionaries. In Dutch, an example of such a word is '*machinefabrieksopzichter'* (translated as 'machine factory supervisor'). To deal with these words, they must be split up. The example above would be split into the following parts: *machine-fabriek-s-opzichter'*.

In addition, there is also the problem of composite words that are incorrectly written as two words, which makes word recognition difficult. For example (using the same example as above), the use of *'machine fabriek'* ('machine factory') would create problems, because this should actually be written as one word: the term *'machinefabriek'* is used in the search file.

4.1.7 Spell checking

The text to be coded can contain ***spelling errors***. This can be resolved by using a spell checker (in the interactive situation) or fuzzy matching (trigrams, Levenshtein distance, Soundex for Dutch, etc.: see Navarro, 2001 and Hall, 1980) in order to match incorrectly spelled words in a search file. In the non-interactive coding situation, it is important that the substitution does not involve the wrong word; for example, the words 'motivate' and 'activate' only differ by two letters, yet they are very different in meaning.[16]

Spell checking is usually performed for each word separately (so without looking at the adjacent words), and assumes that the user approves the change.[17]

It should be clear that some of the above steps could potentially have an effect on other ones. The order in which these steps are performed is therefore important. Some of them could even have the opposite effect. The precise order of these steps depends on the attributes of the source material.

4.2 Treatments for the classification used

Now that the raw text has been pre-processed using general techniques as described in the previous section, subsequent steps can be applied that focus on the coding problem we are dealing with, or the classification used for this purpose. At this point, it is essential to know which words in the text are important for the classification used in the coding. To this end, several techniques are applied to the cleansed description.

---

[16] The fuzzy matching can be adapted, however, so that changes in the first characters has more impact than changes to characters at the end of a word.

[17] The spell and grammar checkers of modern versions of MS Word, however, go even beyond this.

4.2.1 Phrasing

In certain situations, the descriptions may consist of more than one element to be coded, even if everything must be entered ‘in boxes’. A phrase, or sentence, is the unit that must be coded separately. The phrasing method is very dependent on the source material.

4.2.2 Tokenizing

Each phrase, that is, a description S_i , must be ***split*** into tokens $T_{i,j}$. This is usually a simple operation. Generally, a description is split into words, but in some cases, *n-grams* are generated. In n-grams (n is usually 2, 3 or 4), the text is chopped into pieces the size of n characters. The advantage of this over splitting words is that this representation is more robust for spelling errors. To use the example from section 4.1.6 above: the Dutch word *‘machinefabriek’* produces the following trigrams: {‘ ’ma, mac, ach, chi, hin, ine, nef, efa, fab, abr, bri, rie, iek, ek‘ ’}, where ‘ ’ indicates a space. If someone searches for the phrase *‘machine fabriek’,* then 11 of the 14 trigrams still match, and that may be sufficient to recognise the word *machinefabriek.*

4.2.3 Reduction of word forms

In languages with word inflection, including Dutch and English, it is worth the effort to reduce the different inflected forms of a word to their base form (such as reducing various forms of a verb to its base form, or different forms of possessives to third person singular masculine). Examples: reducing ‘walks’ and ‘walked’ to ‘walk’, and ‘mine’, ‘her’, ‘our’, ‘their’ to ‘his’. This can produce unnatural or even incorrect language, but the meaning of the description is seldom affected. This reduction is known as **lemmatisation**. There is also the term **stemming,** which is related to lemmatisation. In stemming, an inflected word is reduced to its base form (the ‘stem’) by dropping some letters at the end. For example, the words ‘fishing’, ‘fished’, ‘fish’ and ‘fisher’ are all reduced to the stem ‘fish’.

4.2.4 Semantic relationships

When standardising descriptions, use can also be made of a synonym list, where a small group of words with a similar meaning is replaced by a single representative. This increases the ‘hitting probability’, the chance that the word will be found for all coding techniques. Whether or not a word is a synonym of another word depends on the context.

A special case of synonyms are loan words from another language, or words that have been adapted to a language from another. For example, in Dutch, the verbs ‘deleten’ and ‘chatten’ have been taken from the English verbs ‘delete’ and ‘chat’. Dutch has also incorporated words from English, such as ‘servicing,’, ‘marketing’, ‘accountmanager’, ‘call center’, and ‘acute myocardial infarction’ (= ‘heart attack’). The process of finding synonyms for a word or concept may seem easier than it actually is. The problem is that it is difficult to determine whether a word or concept is a full synonym, or a true translation of a similar word or concept from another language. In any case, compiling synonym lists requires an effort. The same applies for lists of other types of semantic relationships, which we will discuss below.

Besides synonym relationships, it is also possible to utilise other semantic relationships that can exist between words: hypernyms are an example of this. These are words that indicate a generalisation of a concept: for example, it is possible to replace terms such as ‘tomato’, ‘lettuce’, ‘courgette’, etc. by the hypernym ‘greenhouse vegetables’. If this is done for both the training set and the description to be coded, the ‘hitting probability’ is further increased. The opposite relationship of a hypernym is called a hyponym.

In addition to synonyms and hypernyms/hyponyms, there are also antonyms (words with an opposite meaning) and holonyms/meronyms (words that indicate the whole / a part). An example of this relationship is: an engine is part of a car, so ‘engine’ is a meronym of ‘car’, and ‘car’ is a holonym of ‘engine’. However, these semantic relationships are not actually used in automatic coding.

4.2.5 Part-of-speech tagging

Several pre-processing techniques exist specifically for the grammatical approach to coding, such as POS-tagging (Part-Of-Speech tagging). This is similar to text parsing. In principle, a word list can be used in which each word (or stem) is matched with the meaning of that word.

Natural language processing (NLP) techniques go even beyond the POS-tagging described above. Here, for example, we can rhetorically parse the sentence to better determine the meaning of words in this way. Given the sentence ‘I put my money in the bank’, the word ‘bank’ can have two different meanings[18]: a financial institution or the ground near a river. To obtain a good result based on this text, it is necessary to know whether ‘bank’ means a ‘banking institution’ or ‘the land sloping up around each side of a river or canal’. To clearly understand the meaning here, we have to use the context, in this case, the word ‘money’. These NLP techniques, however, do not always result in higher yields. Furthermore, they render the analysis even more complex.

[18] ‘Bank’ is a homonym.

5. Coding with training sets (supervised classification)

If descriptions are available in electronic form, then there are several techniques to try to assign codes to them. In this case, we distinguish between automatic and interactive (semi-automatic) coding. We will discuss these two techniques in separate sections below.

5.1 Supervised classification

5.1.1 Short description

The literature describes several techniques to classify text if a corpus is available, that is, previously coded (and verified) descriptions. These are techniques based on rule engines, or which utilise nearest neighbour methods, etc. That is what ‘supervised’ refers to in the title of this section. We will briefly address the existing techniques here.

In the international literature, much is known about supervised classification, but the applications concern more often numerical rather than textual data. Such applications are often referred to by the term ‘pattern recognition’, ‘data mining’ or ‘business intelligence’

5.1.2 Applicability

We assume the following situation:

- A training set of coded descriptions is available in electronic form, and a correct code (after verification) is assigned to each description. Incidentally, the descriptions used are first cleansed, in the same way as will be done for the descriptions recorded during a survey; in other words, using the methods as described in Section 4.1.
- New, uncoded descriptions are available that must be automatically coded.

The question is now how the training set can be used in the automatic coding of the new descriptions. We will discuss this in the remainder of the present section (Section 5.1).

5.1.3 Overview of existing techniques

Before we start to discuss the techniques as used at Statistics Netherlands, we first want to deal briefly with the most common methods for text and other classification in the literature, as described more extensively in Sebastiani (2001) and Joachims (2002).

The problem is as follows: there is a training set of descriptions w_{ij} for which the code c_j is known.. Using the training set, a coding model must be ‘trained’ to code new descriptions $w_{ij}^{'}$, which form the so-called test set. This type of techniques is generally used for classification problems for which the descriptions are documents (for example, web pages) that contain a lot of text (say 1000 words or more). This problem is text classification, and is close to but not the same as the coding problem considered in the present document. The descriptions used in coding usually do not contain more than 10 words. It was investigated by Rianne Kaptein (see Kaptein, 2005) to which extent so-called nearest neighbour and Naive Bayes classifiers (in the WEKA package) can be used for coding occupations and education, respectively. It turned out that percentages comparable to that of the COBS coding package (used at Statistics Netherlands) were obtained.

5.1.3.1 Variant used in the cause of death statement abroad

A method with virtually no algorithm for automatic coding is based on the repeat frequency of the textual statements. (This method was described in John, 1997) Stated simply, the method boils down to the following: code everything manually, but only once. If a statement repeats, then the computer remembers which code pertains to it. For virtually all classifications, some observations occur often, and others rarely. The reduction of N textual statements (per year, for example) to N_u *different* textual statements is what is saved on the coding effort. The trick is now to make the ratio N_u / N as small as possible, by properly pre-processing the material, so that unnecessary variations in statements are removed. That is done in steps as described in Chapter 4.

The advantage of this method is that everything is coded (once) by hand, and that therefore the results are very similar to those of manual coding. And, while perhaps not the highest conceivable yield from automatic coding is achieved, the number of *incorrectly* coded records by a computer program is reduced to virtually nil.

The ratio N_u / N is initially unfavourable, while the term lists are being compiled. However, as N increases, say to more than 100,000 terms, the ratio N_u / N becomes more favourable. It is mainly determined by

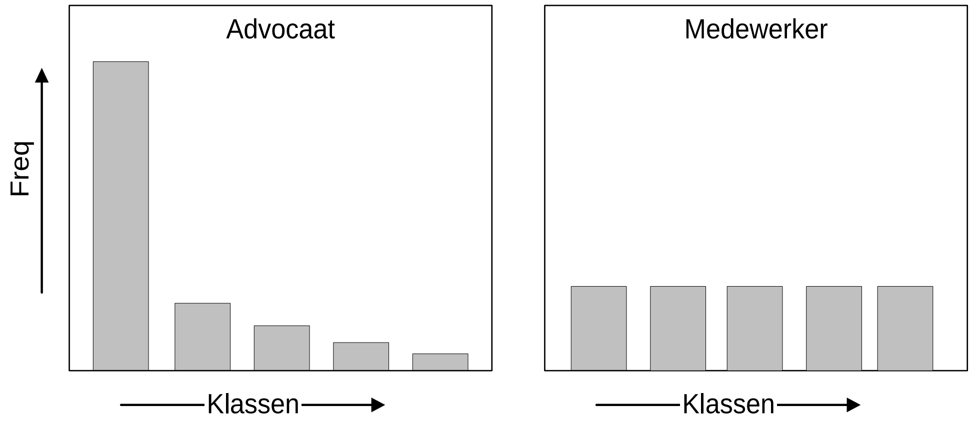
a) the type of input, and

b) the achieved compression of the variability due to the pre-processing.

A ratio N_u / N of approximately 0.1 - 0.25 has been achieved in other countries using the data from cause of death statements.

5.1.4 Methods implemented at Statistics Netherlands

An automatic classification has been implemented in COBS (see Roels and Hacking, 2003). The method used in COBS is, in principle, a nearest neighbour technique, for which a choice is made for a specific distance measure between two descriptions: first, each word (or combination of two words) is assigned a weight that indicates how specific that word is in the training set. This can be illustrated using Figure 5.



(a) **(b)**

Figure 5. Examples of conditional distributions over occupation classes given the key words 'lawyer' (a) and 'employee' (b)

Figure 5 shows histograms of $P(Code_i|Word)$ (the probability of $Code_i$, given that $Word$ is in the description). Subfigure (a) of Figure 5 shows the probability distribution (sorted by frequency) for *Word* = 'lawyer', and (b) depicts the histogram for *Word* = 'employee'. The asymmetry of the distribution indicates how specific a word is. Following Chen (1993), this specificity is quantified as[19]

$$F(W) = \frac{\sqrt{\sum_{i=1}^{n} P(C_i \mid W)^2}}{n}. \qquad (5.1)$$

where n is the number of codes C_i where W occurs in the description.

In this way, a word such as 'lawyer' is assigned a higher weight than a word such as 'employee', when comparing two descriptions. Note that, in this way, words with little meaning such as 'and', 'the', etc. (stop words) naturally have a minimal effect, because they would be given a low weight if they had not been filtered out earlier in the pre-treatments. The pre-processing step in which stop words are removed could, in principle, be omitted. However, because this step reduces the amount of descriptions to be checked, it is more efficient to include it in the pre-processing phase, and to filter out stop words at that point.

Based on formula (5.1), defined *per word (*or *word combination),* we can define a measure for the similarity of two *descriptions* D_1 and D_2 (after removing the words that occur multiple times in both descriptions):

[19] Other measures for this are: entropy and the skewness of the distribution.

$$Similarity\ (D_1, D_2) = \sum_{x_i \in D_1 \cap D_2} F(x_i)$$

where

$$D_1 = \{a_1,..., a_n\} \text{ and } D_2 = \{b_1, \ldots, b_m\}\ .$$

In other words, the similarity between two descriptions is determined by adding up the weights $F(x_i)$ of all shared words[20]. A new description D is compared with all the descriptions present in the training set, and the best N descriptions are retained. The code that occurs the most often among the codes associated with the N best fitting descriptions is either selected (provided that it occurs frequently enough) or presented to an expert. If the code does not occur frequently enough, it is not assigned. We now explain this in more detail. Let C be the most frequently occurring class among the N descriptions. C is selected unconditionally if $\#C \geq f_{GOOD} * N$, where $\#C$ is the frequency score of C among the scores associated with the N descriptions. If it is true for #C that ($f_{BAD} * N \leq \#C \leq f_{GOOD} * N$), then the selection of C is doubtful, and is presented to a specialist coder on this topic, who must then decide whether or not to assign C. If $\#C < f_{BAD} * N$, the selection of C is rejected: it simply does not occur frequently enough. The choices of f_{GOOD} and f_{BAD} are empirically determined using the data; for a higher quality, these can be lower than for a lesser quality.

5.1.5 Example: COBS

In the automatic coding of Occupations, StoreType and ArticleType, use is made of the techniques described above (see Hacking, 2006). This generically implemented automatic coding module is embedded in the coding system COBS (see Roels and Hacking, 2003). To illustrate, we include below a few screenshots of COBS in action. As the system is targeted to CBS work, it interacts only in Dutch with a user.

[20] The use of synonyms, hyponyms and hypernyms could further increase the returns of the matchings; this has not yet been studied.

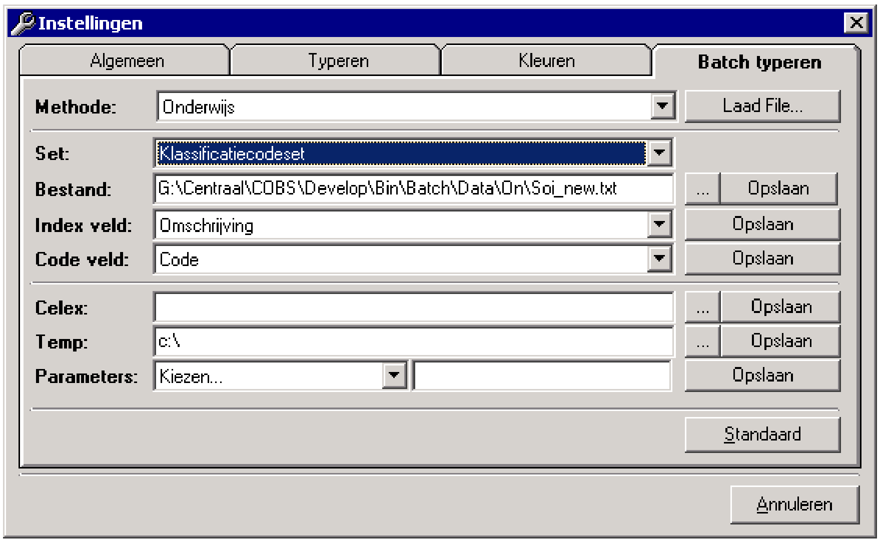


Figure 6. Screenshot of COBS in action

For each subject (education, business activities and occupations) where the files are found, along with the *index* field and the *code* field. Three files are needed, namely:

- A file with the classification (code + description);
- A file with all the previously coded records, from previous years;
- A file with the probabilities $P(C_j \mid W_i)$ and $P(C_j \mid W_i \,\&\, W_k)$.

The last file was calculated earlier based on previously coded data (second bullet), where the probability of a code was calculated per word (or combination of two words). If the frequencies of the words or the calculated conditional probability are too low, this information is not included in the file.

Records imported into the COBS system can be automatically coded. These records all have a certain processing status and are divided into portions that will usually be coded by one coder. After selecting one or more portions, the user must indicate which record status must be coded and what the new status of a record must be if a description is coded successful or partial (in the case of multiple possible codings in a single record).

Figure 7 shows a screenshot of the screen in which the choice can be made as to which ‘portion’ of the descriptions should be batch coded.

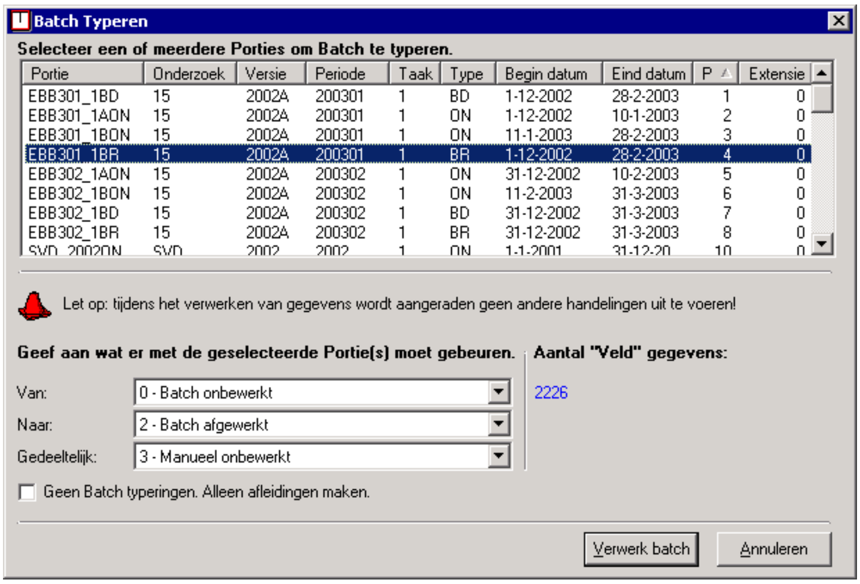


Figure 7. Screenshot of batch coding used in COBS

In the screenshot shown in Figure 7, a portion (‘EBB301_1BR’) is chosen in which all records are selected where study = ‘15’, version = ‘2002A’, period = ‘200301’, task = ‘1’, etc. Only the records that have the record status ‘Batch onbewerkt’ (‘Batch unprocessed’) will be selected (this equals 2226 records, *From (Van)*). For these selected records, the automatic classification will try to assign codes. If this is completely successful, a record will change to the status ‘Batch afgewerkt’ (‘Batch completed’) (*To (Naar)*). Records for which this is only partially successful (for example, not all the educational programmes from an education record can be automatically coded), will change to status ‘Manueel onbewerkt’ (‘Manually unprocessed’) (*Partial*), so that a coder can further process the record. Once the batch engine is finished, a complete summary is provided.

5.2 Interactive coding

5.2.1 Short description

We are assuming that coded material in electronic form is available for interactive coding of new descriptions. The main difference with Section 5.1 is the application. In Section 5.1, the computer must not just search for possible codes, but also make a decision. That is not necessary in the method described in the present section: it is sufficient if the computer gives the N most probable classifications. The user makes the final choice. To generate the most probable codes based on a description, in principle, (the same as in Section 5.1) the techniques from Sebastiani (2001) can be used here as well. This method is sufficient if the ‘real code’ is found among the N most probable. The descriptions of these N codes are then shown to the respondent, who will select the correct code from this list. In addition, the respondent is given the option of selecting the answer ‘none of the above’.

5.2.2 Detailed description

Let us assume that we are interested in coding the description ‘carpenter in the construction industry’ (as a possible answer to the open question ‘what is your occupation?’). This answer has already been given many times in previous surveys and has led to different occupation codes at Statistics Netherlands through manual coding. As a result, it is possible to calculate the conditional probabilities $P(Code_i|$ *‘carpenter’)*. This indicates the probability that a certain code ‘Code’ will be assigned if the search string contains the word ‘carpenter’. The probability $P(Code_i|Word_j)$ can be calculated in this way for all word-code combinations from the training set. If the open text contains more than one word, the vectors are added up to a vector of scores (not probabilities). For example, for the open text answer ‘caregiver in healthcare sector for the elderly’, the vectors ‘caregiver’, ‘healthcare sector’, and ‘elderly’ are added up.

Caregiver*:* $P(code_1|..)=0.60$ $P(code_2|..)=0.20$

Healthcare sector: $P(code_{14}|..)=0.02$ $P(code_2|..)=0.01$ $P(code_{11}|..)=0.01$

Elderly: $P(code_2|..)=0.5$ $P(code_4|..)=0.35$

\+ ---

caregiver + healthcare sector + elderly: $Score(code_2)=0.71$ $Score\ (code_1)=0.60$

As a formula:

$$Score\ (Code_i) = \sum_j P(Code_i\,|\,Word_j)$$

The current implementation (in COBS and PRAT programs used at Statistics Netherlands) also has word *combinations*, i.e. conditional probabilities such as $P(C_i|W_j\ \&\ W_k)$ *are calculated as well*. Suppose that the conditional probabilities of the individual words are as follows:

P(‘healthcare general’ | ‘elderly’) = 0.5730,

P(‘welfare state general’ | ‘elderly’) = 0. 2256,

…

and

P(‘healthcare general | ‘healthcare’) = 0.7538,

P(‘General economic or financial problems’| ‘healthcare’) = 0.2460,

Combining the words ‘healthcare’and ‘elderly’, the following actually applies:

P(‘healthcare general’ | ‘elderly’ , ‘healthcare’) = 1.0.

So, by using word combinations instead of the two separate words, we can derive a code in a much more precise manner. Note that, here, we extend the bag-of-words assumption, which does not model any interactions between the words in a single description.

5.2.3 Implementation at Statistics Netherlands

We now consider the example of the interactive coding of occupations (based on the method described above) as realised in PRAT (in addition, a comparable module was developed for the coding of business activities (see Hacking et al. (2009))). This coding is performed during the electronic interview process for CAPI and CATI where one of the answers must be coded. To this end, in the Blaise[21] interview, use is made of the option of using an external plugin, which makes it possible to integrate external programs during the interview process. This plugin reads information from the Blaise interview and, on this basis, starts a coding session in which one or more questions are asked to arrive at a coding. After the coding session, the selected classification code is written back to the Blaise form, and the interviewer or the respondent continues with the interview.

The method described in Section 5.2.2 is used to offer the respondent several options: sometimes one option in the case of a specific search string as in Figure 8, and sometimes several options in the case of a vague search string, as shown in Figure 9.

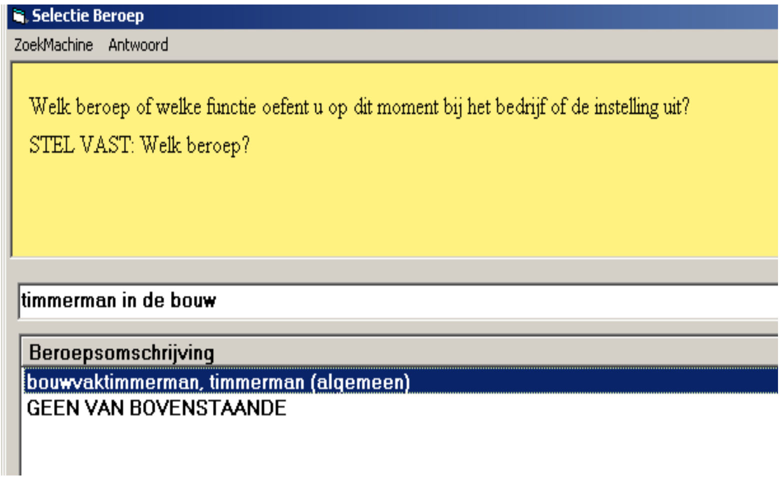


Figure 8. Screenshot of PRAT, used for the coding of an occupation

---

21 Blaise is a general package for designing and doing electronic interviews (see www.blaise.com).

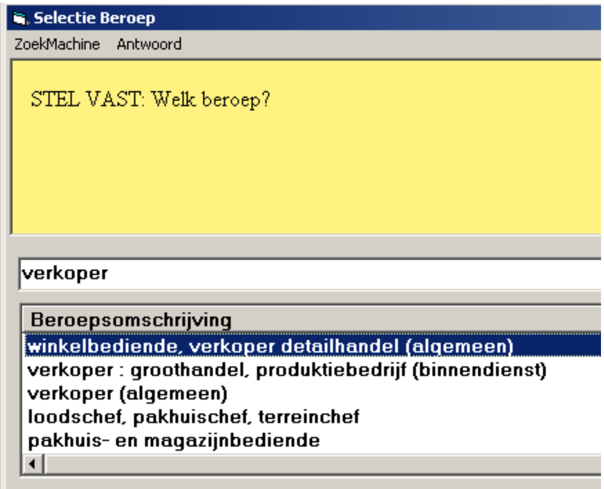


Figure 9. Screenshot of PRAT for the coding of an occupation

The interested user is referred to Michiels (2004) for more information.

6. Coding without a training set (unsupervised classification)

In some cases, there is no coded material available, at least not in electronic form. In this situation as well, we make a distinction between automatic (Section 6.1) and interactive coding (Section 6.2). This situation is usually more difficult than the one described in Chapter 5. For this reason, the best approach is sometimes to first ensure that the situation in Chapter 5 is achieved, by creating a standard file of carefully manually coded records.

6.1 Unsupervised classification

In some cases, no previously coded material is available in electronic form. The starting point then consists of the data to be coded and a classification with a textual description per code. In this situation, we can either ensure that electronically coded material is produced, by actually coding a reasonable part of the material and then working with the data based on techniques as described in Section 5.1. Or, if that is too expensive, an attempt should be made to code based on the texts themselves and the associated semantics.

6.1.1 Short description

'Spreading activation' is a search method in a semantic network in which, for the area of application of relevant concepts, links are made between their mutual relationships and codes. This method is described in more detail below because it is also used at Statistics Netherlands.

Before we address the techniques as used there, we want to first indicate what methods for *unsupervised* classification or text classification are described in the literature. Generally, there are two options: *clustering*, in which the descriptions are grouped, so that afterwards they can be more easily manually coded, and *manually constructing a model* (for example, a set of derivation rules or a search file).

6.1.2 Spreading activation

For the coding into SBI codes (the Dutch version of the SIC codes), a technique called 'spreading activation' is used, where coding is performed based on a semantic network (which may have been created manually). This is a directed graph, also called a digraph, where the nodes represent words, and where the edges or directed edges (or arcs) indicate relationships between words (the exact relationship is stated by listing that next to an arc), for example:

- greenhouse vegetables $\xrightarrow{hypernym}$ tomato, because greenhouse vegetables include tomatoes.
- tomato $\xrightarrow{hyponym}$ greenhouse vegetables, because totatoes are a kind of greenhouse vegetables.

- Agatha $\xrightarrow{synonym}$ potato, because, for the classification, the potato varieties like ‘Agata’, ‘Anya’, ‘Fingerling’,‘Jersey Royal’, ‘Kerr’s pink’, etc. are not important, and if they do occur in a description they can be considered synonyms to ‘potato’ which can be used instead.

- sale_of_childrens_clothing $\xrightarrow{Code}$ 12345, because the description ‘sale of children’s clothing’ unambiguously leads to the code ‘12345’.

The use of these relationships creates a semantic network, of which a small part is shown for illustrative purposes in Figure 10.

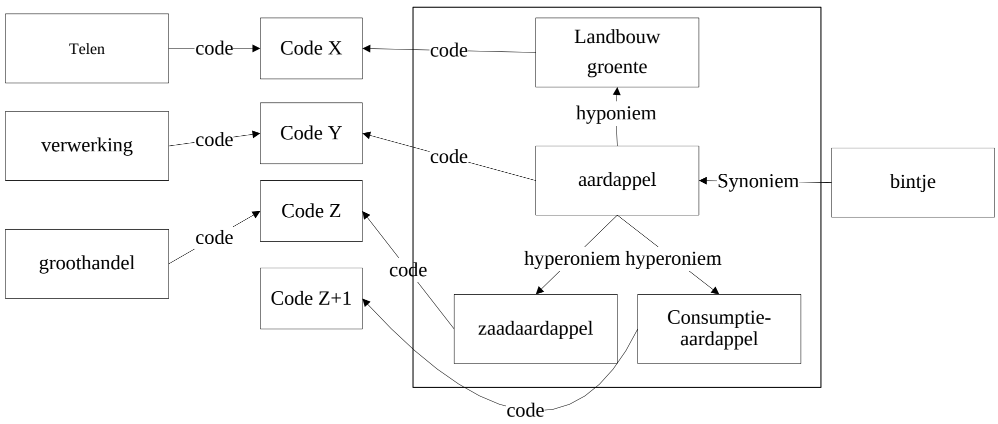


Figure 10. A fragment from a semantic network, as used in the coding of the SBI

In this network, we see interrelated words, due to certain semantic relationships. The words in a semantic network are also called nodes, for which an associated tag is ‘activation’; this serves to quantify the extent to which a word correlates with the terms from the search string. The binary or other relationships that exist between the nodes can (formally) be recorded in an adjacency matrix. [22]

In brief, the algorithm amounts to the following:

1. Let the set of nodes be denoted as $\{n_1,...n_m\}$. Call the activation values during iteration round k of the nodes $A_k = (a_{k1},..., a_{km})$. Call the adjacency matrix $P = (p_{ij})$ This means :
   $p_{ij} = 1$ if there is a link between two nodes n_i and n_j, and $p_{ij} = 0$ if that is not the case.
2. Next: for each word stated in the description:
   a. Set the activity of the node linked with word l from the description to 1: $a_{1l} := 1$.

[22] The actual implementation is likely to be different from the description given here, as this would be very efficient. It is only for the sake of the explanation of the algorithm that an adjacency matrix is used.

b. After this, all nodes that can be reached by an arrow from 'activated' nodes are also activated by means of the following relationship:

$$a_{k+1,i} = \sum_{i,j} a_{k,j} \cdot p_{i,j}$$

This must only be done for nodes not yet visited. In addition, there is a special restriction for the *hypernym* and *hyponym* relationships (the *parents* and *children*): if a path has already run along a *hypernym relationship*, then it may not run along any other *hyponym* relationships, and vice versa.[23]

3. The 'expansion' of the activity stops because all paths ultimately 'collide' on a code node, or because there are no more unvisited nodes near a node. The codes then contain an activity as described in 2a and 2b. All codes with an activity > 0, in order of activity, form the result of the search operation.

For more details, see Hacking (2009). For another application, see Berger et al. (2004). For a discussion of the use of semantic networks in coding, see Willenborg (2012).

6.1.3 Implementation at Statistics Netherlands

When coding according to the SBI code, a semantic network is made 'manually' (based on existing descriptions) (see Hacking, 2006), in which spreading activation is used. The provisional results are as follows: 80% correctly coded codes, and in 15% of the cases, multiple codes. For more details, see Hacking (2009). To illustrate: the total number of nodes is approximately 5200, the number of relationships is approximately 17200, and the search time in the implementation is around 0.23 sec on a 1.5 GHz machine.

[23] If this restriction were not present, then all the nodes in the classification tree would be visited, and this is not intended. We only want parents, grandparents, etc., and the 'subtree' of a classification node to be visited.

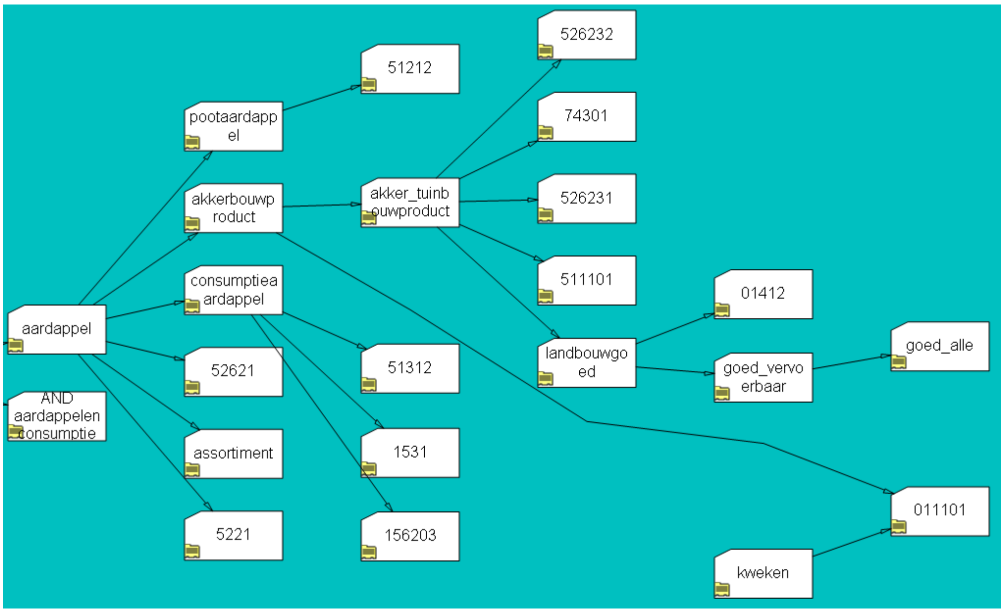


Figure 11. A screenshot that shows part of the semantic network that was visited after the search string 'cultivation of potatoes' was provided

In Figure 11, a screenshot is shown of the proof of concept that is used for the coding of the SBI. This shows a part of the semantic network that is 'visited' after providing the search string 'production of potatoes'. For reasons of clarity, the different relationships between the concepts in the network (*Code*, *Synonym*, …) are not included. Note that 'potatoes' (via the classification) leads to 'retail potato'; 'production' is defined as a synonym of 'preparation other', and this term leads to the same two codes as 'retail potato'.

6.2 Interactive coding

6.2.1 Short description

If no coded material is available, and the goal is interactive coding, then a file as described in Section 6.1 can also be used for this purpose.

6.2.2 Detailed description

The semantic networks described in the previous section can serve as the basis for an interactive search technique. By assigning a dimension with the associated question text to every word in the network, we can use the network for interactive questioning. To illustrate: for the coding of 'education', we can add the dimensions 'level', 'is a teacher training', 'subject', etc.

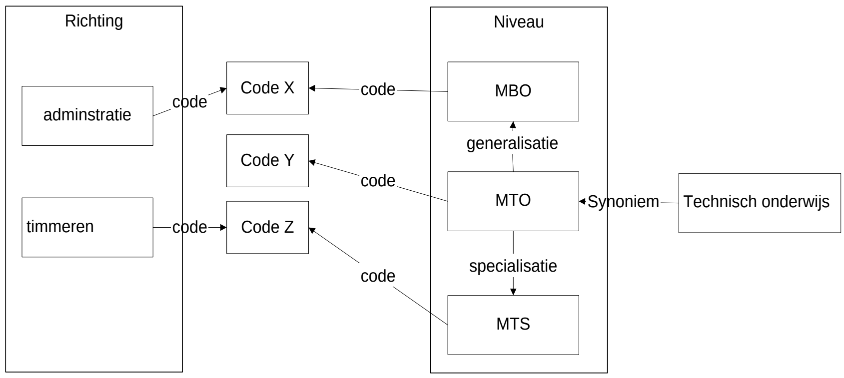


Figure 12. A fragment of the semantic network for the coding of 'education'

The query process starts with an open question about, for example, education. Based on the answer, which is used as a search string, a number, say N, of codes are selected in the semantic network (see Figure 12).

1. Open question $\rightarrow$ Codes $C = \{C_1, \ldots C_N\}$ with the associated scores $S = \{S_1, \ldots S_N\}$, sorted based on score $(S_{i-1} \geq S_i)$; N is the total number of hits. Call the number of codes with the same highest score M.
2. If $M = 0$ (in other words, no suitable codes have been found): either stop or ask for another description.
3. If $1 \leq M \leq_{MAX}$: show the codes found and let the user choose one.
4. If $M \geq M_{MAX}$: select the (next) dimension D_i and make a list of all words W_k , which are both linked with the dimension D_i and with the codes in C; now also add the synonyms. Show a question or additional question associated with D_i and let the user make a choice from the list W_k. Each word in this list leads to a subselection $S_k \subseteq S$ [24].
5. The user makes a selection: $S := S_k$; now continue with step 2.

To illustrate, Table 3 shows an extract from the semantic network for education, in the form of a search table, to emphasise the link between the words and their associated dimensions (columns) and codes (rows). The extract is based on the initial answer 'English'.

[24] This is therefore a 'hard' subselection. If this is not desired, we can, for example, also reduce the set of records by repeatedly expanding the search string with the selected string from list W_k.

Table 3. Table for the coding of education

C	*Level*	*Subject*	*IsTeacher-Training*	*University*	*TeacherType*
1	Senior secondary vocational education (MBO)	English	No		
2	Higher professional education (HBO)	English	No		
3	University	English literature	No	Master's	
4	Higher professional education (HBO)	Interpreter English	No		
5	Higher professional education (HBO)	Translator English	No		
6	University	English	Yes	Master's	First level teaching qualification
7	Higher professional education (HBO)	English	Yes		Second level teaching qualification

To further reduce the number of possible codes, we select the dimension 'IsTeacherTraining' with the associated question 'Is this a teacher training?' and answer set $W_k = \{yes, no\}$ and $S_{yes} = \{6,7\}$ and $S_{no} = \{1,2,3,4,5\}$. If, for example, we had chosen 'level', then the question would have been 'What is the level of the education?', $W_k = \{HBO, university, academic\}$ (*academic* is a synonym of *university* in the network) and $S_{hbo} = \{2,4,5,7\}$, $S_{university} = \{3,6\} = S_{academic}$, by definition).

6.2.3 Implementation at Statistics Netherlands

This continued questioning technique is currently used for both PRAT (for the coding of education) and the coding of the SBI[25]. Creating semantic networks in the form of synonym relationships, etc., is labour intensive and requires significant knowledge of the content.

25 For the coding of SBI, the subselection is slightly more subtle: instead of a hard subselection, there is a repeated search action based on an increasingly expanding search string.

7. Validation

In this chapter, we want to examine briefly the evaluation of the different codings. A coding is a coding method that is implemented for a practical application. This means that, in a coding, auxilliary data is used in addition to a method. This auxilliary data can, in a variety of ways, be present in a coding system; it can be used when developing a semantic network for such a system; it can be used to develop or maintain a thesaurus, or to build a training set for a coding system. The validation of a coding system involves testing the entirety of the method and the data and auxilliary data used. As a rule, this is done by providing the system with a set of descriptions for which the result is known for each description (for example, from an earlier 'manual' coding): whether a code was assigned, or not, and in case a code was assigned, which one.

Validation also plays a role in the diagnosis performed by doctors, because diagnosing is an activity akin to coding. See, for example, Hilden, Habbema, and Bjerregaard (1978a, b, c).

7.1 What exactly should be validated?

The result of each coding is a set of descriptions linked to a code from a classification, or an observation that the description cannot be coded at the desired level or not at all. Regardless of how this match was, (or was not) achieved, we can ask the question of whether this result is correct, and in case a match was obtained, what its quality is. Inspection of the results can be used to validate the method. However, this is less trivial than it might seem at first glance. Should only descriptions be used that unambiguously lead to a single code? Or should incomplete descriptions also be used, so that multiple codes fit? Or descriptions that do not lead to any code from the classification? Also in these cases, it is important to determine whether the correct decisions are made in the coding process. The desired use of the coding procedure is of importance here. This can be applied in such a way that, in case of sufficient doubt, the preference is to not code a description. This may be an erroneous unmatch. Or the exact opposite can be done, namely that the risk is accepted and the description is assigned a code. This may be an erroneous match. These 'Type I' and 'Type II' errors must be established using special parameters. Compiling a test set of descriptions that also provides good insight into how Type I and Type II errors are dealt with is not simple. In any case, sufficient descriptions (with or without codes) must be included in the test set in order to do this. This applies more generally to the validation of a coding system where, based on a sample of possible descriptions (with or without codes), a judgement must be made as to whether or not the system is functioning properly.

It is practically impossible to assess a coding *method* by itself, because a coding depends not only on the method but also on the quality (and the scope) of the data/auxilliary data used in the coding. In this context, it is important to consider previously made descriptions + codings, or a training set that was used in training a neural network, or the quality (including the scope) of a semantic network or a thesaurus. In practice, it is only possible to test a coding procedure (method + data). Creating other auxilliary data sets – if possible at all – involves so much work that, in practice, it is not possible to test a method by itself with a variety of auxilliary data.

We want to make a final comment concerning the descriptions (with or without codes[26]) that are used in the test set. These were created in an earlier coding round, probably a 'manual' (interactive) one. However, in principle, they can also contain errors. There are no descriptions with associated 'God given' codes. What can be done in practice is only to compare the results of two coding procedures. Because we can assume that the test data was carefully compiled, we tacitly assume that this data does not contain errors.

7.2 Descriptions to be used for the validation

When testing a coding procedure, we can fully abstract from the way in which it was done. Only the results matter. Generally, in validating a coding method, we want to know how well it works in practice. This is done by providing a set of descriptions with codes (one or several) or without (in the case that the description could not be coded).

Table 4 shows the result that can arise if different types of descriptions are provided to the coding system:

- $(\omega,-)$: a description that does not have an associated code;
- (ω,a) : a description that has exactly one associated code, here denoted as 'a';
- $(\omega,a \wedge b)$: a description that has more than one (not necessarily exactly two) associated codes (here denoted as 'a' and 'b'), because the description is ambiguous (or incomplete).

The second situation is the ideal one: the description is codable and associated with exactly one code. In any case, we want a coding system to correctly deal with this type of descriptions. Often, a coding system is tested with only this type of descriptions. While important, this is not always sufficient. The system should also be tested with non-ideal descriptions, such as presented in the first or third case above. The first case concerns descriptions to which no code can be assigned according to the classification used. In the third case, multiple codes can be assigned to a description. This last case actually encompasses several possibilities:

[26] We assume here that there are descriptions, some of which can be coded and assigned the correct code, and some of which cannot, and which therefore do not – justifiably – have an associated code.

1. The description is incomplete and can be finished in several ways, leading to a complete description with different associated codes.
2. The description consists of multiple sub-descriptions that each have their own unique code.

Table 4. Results of the coding of descriptions

		output				
		$(\omega,-)$	(ω,a)	(ω,c)	$(\omega,a \wedge b)$	$(\omega,c \wedge d)$
input	$(\omega,-)$	**correct**	[erroneously a code]	[erroneously a code]	[erroneously multiple codes]	[erroneously multiple codes]
	(ω,a)	[erroneously no code]	**correct**	[one code, but an incorrect one]	[erroneously multiple codes, but one of them is correct]	[erroneously multiple codes, but none of them are correct]
	$(\omega,a \wedge b)$	[erroneously no code]	[erroneously only one code, which, however, is correct]	[erroneously a code, which, moreover, is incorrect]	**correct**	[multiple codes, however, none of them are correct]

We give several examples to illustrate these two points, using fictitious examples from the business activities area (SBI).

- Sub 1. Suppose that the description is: ‘potatoes’. This can relate to the cultivation, trade or processing of potatoes, where, as we saw earlier, all of these have different codes.
- Sub 2. Suppose that description is: ‘repair and demolition of cars’ and that the descriptions ‘repair of cars’ and ‘demolition of cars’ each correspond to unique codes, but no single code corresponds to ‘repair and demolition of cars’.Apart from that, there can be descriptions that seemingly indicate two different codes, but that are actually represented by a single code. An example is: ‘sale and repair of cars’.Of course, it is not possible for a respondent to know when a combination of activities has its own code and when it does not.

In the case of multiple codes, there is little use in making a further distinction based on their nature, the system recognises such ambiguous cases with multiple codes. After identifying because the goal of the coding system is to find a unique code for a description – or no code if there isn’t any – but not multiple ones. This case deserves separate attention and can, in the case of automatic coding, best be handled interactively, in which case a coder ultimately also looks at the description. In the testing, it is only important that such a case, a coder should examine it and decide what to do.

Table 4 shows the possible output in addition to the possible input. This varies from the case where the system does not assign a code to the case that multiple codes are assigned to the description.

7.3 Restrictions

The above approach has its limitations. It is of limited use for codes that occur rarely. Few descriptions are available for such codes. For this reason, such codes are also difficult to test. If, in the classification, many codes occur that are rare, then it is only possible to validate the coding system for part of the codes: only the ones that occur frequently. (See also Chapter 2).

7.4 Alternative methods

A natural way to test a coding system is provided above. This checks, in fact, how the system performs if it is fed with descriptions that it could also expect in 'normal operation'. In addition, it is also possible to provide the system with selected descriptions, for example, those that are known to be uniquely codable . It is also possible to provide descriptions for which it is known that they are not ideal, for example, that they do not lead to unique codes because they contain either too much or too little information. In this way, it is possible to examine in a focused manner how the coding system responds to certain exceptions. If the response is not satisfactory, an attempt can be made to adjust the process. In the case of imperfect descriptions, the system should just recognize them. These descriptions could subsequently be dealt with by a coder. [27]

---

[27] Determining whether a coding is correct or not can be achieved by training the coding system based on $\{Code_{predicted}, Code_{true}\} \rightarrow \{correct, incorrect\}$. This is a meta coding problem (Kaptein, 2005)

8. Some practical aspects

In the previous chapters, we have focused mainly on the methods for automatic and semi-automatic coding. In this chapter, we want to address several other issues that play a role in the coding *process,* such as the management.

8.1 Multiple (simultaneous) classifications

Sometimes, there are multiple simultaneous valid classifications. This occurs, for example, in the classification of educational programmes. The reason for this is that there is both a national (SOI) and an international (ISCED) classification. To resolve this problem in the coding of education, an *intermediate code* is used for the coding (the so-called educational programme number, OPLNR); this intermediate code is defined such that a unique translation is possible from OPLNR to SOI and from OPLNR to ISCED. At present, efforts are being made to further adapt the SOI to the ISCED[28], leading to less OPLNR codes and thus improving the coding efficiency.

For the coding of economic activity, the difference between coding at a European and a national level is resolved by recording the first four digits at European level (NACE) and leaving the fifth digit open for national additions (SBI).

8.2 Version management

Classifications change from time to time. In this regard, for quality measurements and evaluations, it is important to keep track of the way that a coding was created: automatically, by a coder or corrected by a supervisor. In addition, the version or versions of the search files and/or code systems should also be recorded. In this way, sufficient information is available afterwards to make longer time series of variables where the classification plays a role.

In the transition from code system A to code system B, it is often necessary to convert variables coded in code system A to B. The easiest solution is based on a recoding table. However, for the cases in which there are multiple codes in B for a single code in A (the so-called *split cases*), another approach should be selected.

It is clear that, if code system A can be injectively mapped into code system B code system A could be seen as a part of code system B, and is therefore superfluous. (In terms of software versioning, we can say, in this case, that B is backwards compatible with A.)[29] In practice, this is often not the case, for example, due to the presence of split cases.

In addition, the search files must also be adapted:

[28] Due in part to all the changes in the Dutch educational system in recent years.

[29] It is also possible to define ‘forwards compatibility’. In this case, code system B could be seen as a part of code system A

- For the methods based on existing correctly coded material (Chapter 5), new models must be regenerated based on recoded material. For this purpose, it is necessary to recode the existing material; this is, of course, necessary only for the split cases.
- However, a new version must also be made for manually created search files (Chapter 6), where, in any case, the split cases must be examined manually.

We want to also refer here to the approach to the revision of the ICD-8, 9, 10 by the National Center of Health Statistics in the US. Terminology lists were created that were far more detailed than needed for the ICD. Therefore only these refined classes – the so-called Medical Entities – need to be recoded if a new ICD is published, and not the original terminological material. This approach was also used by Italy, for the same reason.

8.3 Management of the test sets/datasets/search files

In addition, the wording used by the respondent can change, or new classes can arise that fall within an existing code. An example: in the SBI93, there are only a few designations for the IT industry. A search file from 1993 will probably contain many new words, which would all lead to an SBI code in the IT sector in 2007. It is therefore important to:

- Continue monitoring the coverage and accuracy of the coding;
- If the quality is too low, adapt the search file by expanding it.

8.4 Tooling

Many of the methods described in the literature and their implementations are still in an academic phase, making their real-world application not (yet) really feasible. To illustrate, in many cases, a sentence to be coded is described using a 0-1 sparse vector. However, for example, once matrix multiplication is used (creating much less sparse vectors and matrices), the application needs a very large amount of memory and time to come to a solution. This was, certainly for interactive applications, not an option. In the past, a decision was made to use Statistics Netherlands' own generic implementation in C++ for PRAT, due to its speed (and flexibility). Most of the techniques described in this report were also implemented in the same software module; consequently, this can also be used for other coding problems. A brief description of a generic search module for a coding module can be found in Appendix A.

This document mainly describes methods as used and/or developed at Statistics Netherlands. There are also coding tools developed by other statistics bureaus; however, these tools are very small in number. A survey into the coding of occupations and companies at different statistical institutes revealed that, in most cases, either simple tooling is used (followed by primarily manual coding) or the institute has developed a specific tool to be used for a single application.

As far as the authors are aware of, the following generic[30] coding tools have been developed at other statistics bureaus or companies:

- SICORE from INSEE (see Rivière, 1994): this is based on decision trees.
- ACTR from Statistics Canada (see Wenzowski, 1988): this is based on a type of Nearest Neighbour technique.
- StafS from SPSS.

A further analysis of the methodology used by these tools is an option for a new version of this document.

8.5 Proxy variables instead of descriptions

The methods discussed in this document are based on descriptions as input for a coding system that, if possible, associates codes from a classification with these descriptions. This can be done in multiple steps: first fully automatically, then interactively or ‘manually’ by a coder. The drawback of this approach is that the information that should be included in a description is difficult to influence. At the most, the interviewer can give the respondent a few instructions, but the respondent is, in principle, free to answer as desired. Because the respondent is not familiar with the classification, this can easily lead to a description that creates problems when it is coded. This is especially the case if the respondent provides too little usable information.

An alternative for the coding problem as described in this document is that, instead of a freely chosen description – an answer to an open question – answers are collected to a number of relevant, closed questions (proxy questions), which are all easy to answer for most of the respondents. In this case, each closed question should have a limited number of answer categories, and be easy to answer. Using a derivation diagram, the answers to the proxy questions would then be used to unambiguously derive a code.

This approach is used at Statistics Netherlands for the coding of education and business activity. It may be advantageous to more closely guide the respondents, in comparison to the ‘open question’ approach. However, this depends on the classification in question.

[30] In addition, there are also specific coding tools, for example, for the coding of causes of death, such as SUPERMICAR, MICAR, ACME, STYX, MIKADO and IRIS.

References

Berger, H., Dittenbach, M. and Merkl, D. (2004), An accommodation recommender system based on associative networks. In: Frew, A. J., editor, *Proceedings of the 11th International Conference on Information Technologies in Tourism (ENTER 2004)*, pp. 216-227, Cairo, Egypt, January 26-28, 2004. Springer-Verlag.

Bunge, J. and Fitzpatrick, M. (1993), Estimating the number of species: A review. *Journal of the American Statistical Association* 88 (421), 364-373.

Chen, B., Creecy, R. and Appel, M. (1993), Error control of automated industry and occupation coding. *Journal of Official Statistics* 9, 729-745.

De Heij, V. and Langenberg, H. (2002), *Automatisch typeren van vacatures (tussen-rapportage)*, Statistics Netherlands, Voorburg.

Efron, B. and Thisted, R. (1976), Estimating the number of unseen species: how many words did Shakespeare know? *Biometrika* 63 (3), 435-447.

Good, I.J. (1953), The population frequencies of species and the estimation of population parameters. *Biometrika* 40, 237-264.

Good, I.J. (1965), *The estimation of probabilities: An essay on modern Bayesian methods*, MIT Press.

Good, I.J. and Toulmin, G.H. (1956), The number of new species, and the increase in population coverage, when a sample is increased, *Biometrika*, 43, 45-63.

Hacking, W.J.G and Janssen-Jansen, S. (2009), *The coding of economic activity based on spreading activation.* Statistics Netherlands, Heerlen.

Hacking, W.J.G., Michiels, J. and Janssen-Jansen, S. (2006), *Computer assisted coding by Interviewers.* IBUC2006.

Hall, P.V. and Dowling, G.R. (1980), Approximate string matching. *Computing Surveys* 12, 381-402.

Hilden, J., Habbema, J.D.F and Bjerregaard, B. (1978a), The measurement of performance in probabilistic diagnosis, I. The problem, descriptive tools, and measures based on classification matrices, *Methods of information in medicine*, 17, 217-226.

Hilden, J., Habbema, J.D.F and Bjerregaard, B. (1978b), The measurement of performance in probabilistic diagnosis, II. Trustworthiness of the exact values of the diagnostic probabilities, *Methods of information in medicine*, 17, 227-237.

Hilden, J., Habbema, J.D.F and Bjerregaard, B. (1978c), The measurement of performance in probabilistic diagnosis, III. Methods based on continuous functions of the diagnostic probabilities, *Methods of information in medicine*, 17, 238-246.

Hill, B.M. (1974), The rank-frequency form of Zipf's law, *Journal of the American Statistical Association*, 69 (348), 1017-1026.

Joachims T. (2002), *Learning to classify text using support vector machines*, Kluwer.

John, P. (1997), Automatisch coderen van afzonderlijke uitdrukkingen op het B-formulier, Internal report, Statistics Netherlands, Voorburg.

Kaptein, A.M. (2005), *Meta-Classifier Approaches to Reliable Text Classification.* Master's Thesis, Maastricht University.

Michiels, J. and Hacking, W. (2004), *Computer assisted coding by interviewers.* European Conference on Quality and Methodology in Official Statistics, Mainz, Germany.

Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. and Euler, T. (2006), YALE: Rapid Prototyping for Complex Data Mining Tasks. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06).*

Navarro, G. (2001), A guided tour to approximate string matching. *ACM Computing Surveys* 33, 31-88.

Rivière, P. (1994), The SICORE automatic coding system, Working Paper. In: *Conference of European Statisticians*, Cork.

Roels, J. and Hacking, W.J.G. (2003), *COBS Gebruikershandleiding en technische documentatie van de Userinterface.* Statistics Netherlands, Heerlen.

Sebastiani, F. (2001), Machine learning in automated text categorization. *ACM Computing Surveys* 34 (1), 1-47.

Sichel, H.S. (1975), On a distribution law for word frequencies. *Journal of the American Statistical Association* 70, 542-547.

Sichel, H.S. (1986a), Word frequency distributions and type-token characteristics. *Math. Scientist* 11, 45-72.

Sichel, H.S. (1986b), Parameter estimation for a word frequency distribution based on occupancy theory, *Comm. Statist.-Theor. Meth.* 15 (3), 935-949.

Smeets, P.S.G.M. (2007), *Aanbevelingen voor de verwerking van grote hoeveelheden goederenomschrijvingen.* Statistics Netherlands, Heerlen.

Van den Meijdenberg, M.A.C.C. and Kardaun, J.W.P.F. (1997), *Automatisch coderen doodsoorzaakverklaring.* Statistics Netherlands, Voorburg.

Wenzowski, M.J. (1988), ACTR – A Generalised Automated Coding System. *Survey Methodology* 14, 299-308.

Willenborg, L.C.R.J., Van der Plas, J., Van Hooff, H. and Chevreau, K. (2002), *Data providers questionnaire, Version 3 (DPQ3).* Statistics Netherlands, Voorburg.

Willenborg, L.C.R.J. (2012), S*emantic networks for automatic coding*. Statistics Netherlands, The Hague (in progress).

Willenborg, L. and Heerschap, N. (2009), *Koppelen*. Methods Series document, Statistics Netherlands, The Hague [English translation from Dutch in 2012].

Appendix A. Specification for a search module

The same search operations can be used for many of the methods referred to in this report. Such operations could be put in a single search module. This search module must then be managed using a search string composed of search operations and combinations thereof.

Below, we provide a format as it could be specified; the format of each search operation is described by:

```
<search type> '(' <search string>  ';' <parameters> ')' ,
```

First of all, there are a number of basic (fuzzy) search operations:

a. *word (baker)* search for records with the word 'baker';
b. *trigram (bakery; minscore=3)* search for records with the word 'bakery' that have at least 3 trigrams in common with 'bakery';
c. *regexp (^baker*)* search using the regular expression for the term 'baker', where ^ indicates the start of the text and * zero or more additional characters,
d. *levenshtein (bakery;minscore=998)* for levenshtein, the maximum score is 1000 (a word is found that is exactly equal to the search string); for each letter that is different or deleted / added, 1 point is subtracted from the score.

In addition, the model should also offer the following more complex search possibilities:

CondProb: implements the search behaviour as described in Section 5.2, where a score is calculated based on conditional probabilities P(Code|Word(s));

SpreadingActivation: implements the method as described in Section 6.2, where the search is performed in a semantic network;

These 'simple' search instructions should be able to be combined using and (**&**), or (|) and not (**!**) to generate more complex search instructions.

Examples:

e. *word (baker)* ***& !*** *word (pastry)* search for records with the word 'baker' but *not* with the word 'pastry';
f. *trigram (bakery; minscore=3)* ***&*** *levenshtein (bakery; minscore=998)* search for records with the word that have at least 3 trigrams in common

with ‘bakery’ and where, in terms of spelling, the word deviates in only two places from that of ‘bakery’.

In addition, it should also be possible to search in multiple columns at the same time:

Trigram (bakker; minscore=3) & regexp (^12.;**searchcolumn=sbi-code**).*

Here, we are looking for all the SBI descriptions that contain the word *bakker* (the Dutch word for ‘baker’) and for which the SBI code starts with 12.

The above operations can be partially found in existing open-source tooling, but the operations *CondProb* and *SpreadingActivation* cannot. For PRAT and COBS, a search module (C++/COM) was implemented at Statistics Netherlands, in which all the above operations can be used in a generic manner.

Appendix B. Soundex algorithm for Dutch

A version of the Soundex algorithm for the Dutch language (Dutch Russell Soundex) is as follows:

1. Keep the first letter of the string and delete the following letters that occur: a, aa, e, h, i, o, u, j, y.

2. Substitute groups of letters, as follows:

From	To
qu	kw
sch	see
ks , kx	xx
kc , ck	kk
dt , td	tt
ch	gg
sz	ss
ij	yy

3. Assign numbers to letters, as follows:

From	To
b , p	1
c , g, s, k, z, q	2
d , t	3
f , v, w	4
l	5
m , n	6
r	7
x	8

If two or more letters with the same numbers are next to each other in the original name (before step 1), then omit all of these, except for the first one.

Take the first four bytes supplemented with zeroes.

Version history

Version	Date	Description	Authors	Reviewers
Dutch version: Coderen				
1.0	05-11-2009	First Dutch version	Wim Hacking Leon Willenborg	Jan Kardaun Marly Odekerken Sander Scholtus
English version: Coding; Interpreting short descriptions using a classification				
1.0E	16-02-2012	First English version	Wim Hacking Leon Willenborg	