Automating Survey Coding by Multiclass Text Categorization Techniques

Daniela Giorgetti
Istituto di Linguistica Computazionale
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
E-mail: `daniela.giorgetti@ilc.cnr.it`

Fabrizio Sebastiani
Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
E-mail: `fabrizio.sebastiani@isti.cnr.it`

Abstract

Survey coding is the task of assigning a symbolic code from a predefined set of such codes to the answer given in response to an open-ended question in a questionnaire (aka *survey*). This task is usually carried out in order to group respondents according to a predefined scheme based on their answers. Survey coding has several applications, especially in the social sciences, ranging from the simple classification of respondents to the extraction of statistics on political opinions, health and lifestyle habits, customer satisfaction, brand fidelity, and patient satisfaction.

Survey coding is a difficult task, since the code that should be attributed to a respondent based on the answer she has given is a matter of subjective judgment, and thus requires expertise. It is thus unsurprising that this task has traditionally been performed manually, by trained coders. Some attempts have been made at automating this task, most of them based on detecting the similarity between the answer and textual descriptions of the meanings of the candidate codes.

We take a radically new stand, and formulate the problem of automated survey coding as a *text categorization* problem, i.e. as the problem of learning, by means of supervised machine learning techniques, a model of the association between answers and codes from a training set of pre-coded answers, and applying the resulting model to the classification of new answers. In this paper we experiment with two different learning techniques, one based on naïve Bayesian classification and the other one based on multiclass support vector machines, and test the resulting framework on a corpus of social surveys. The results we have obtained significantly outperform the results achieved by previous automated survey coding approaches.

1 Introduction

Survey coding is the task of assigning a symbolic code from a predefined set of such codes to a textual expression representing the answer given in response to an open-ended question in a questionnaire (aka *survey*). By *open-ended* we mean a question that requires or allows an answer consisting of free text; open-ended questions are the alternative to *multiple-choice* questions, which instead require the answer to be selected from a predefined set[1].

[1]In this paper we will only deal with text in *written* form. The implications of coding survey material in *audio* form are discussed in Section 6.

Survey coding is usually carried out in order to classify responses (and respondents) into a category scheme, thus superimposing a structure on what would otherwise be a totally unstructured corpus of different responses. Survey coding has several applications, especially in the social sciences, where the classification of respondents is functional to the extraction of statistics on political opinions, health and lifestyle habits, customer satisfaction, brand fidelity, and patient satisfaction.

As an example, in 1996 interviewers asked (among many others) the following question to a carefully chosen sample of 1370 subjects, in the framework of the General Social Survey [7] carried out by the US National Opinion Research Center (NORC)[2]:

```
Within the past month, think about the last time you felt really angry, irritated
or annoyed.  Could you describe in a couple of sentences what made you feel
that way - what the situation was?
```

Professional coders were then asked to classify the answers in exactly one among the following categories, each consisting of a code and a short explicatory caption:

ANGRYWRK:	Situation involved work
ANGRYFAM:	Situation involved family
ANGRYGVT:	Situation involved government or government officials
WRK&FAM:	Situation involved both work and family
WRK&GVT:	Situation involved both work and government
FAM&GVT:	Situation involved both family and government
OTHER:	Situation did not fit the above categories

Answers included for example[3]:

```
trying to teach my son something and he was being stubborn and wouldn't listen
to me i got angry at him
```

or

```
my wife went shopping & spent too much on a dress & it made me feel angry
```

which coders classified under the ANGRYFAM header.

Survey coding is a difficult task, since the code that should be attributed to an answer is a matter of subjective interpretation, and thus requires expertise. For instance, different coders, especially if little trained, might have different opinions as whether the answer

```
when people in authorities arent treating people right
```

should be classified under ANGRYGVT, or ANGRYWRK, or WRK&GVT, or even under OTHER: by "authorities", did the respondent only refer to authorities in government, or did she also include hierarchically superordinate colleagues at work? or did she also include authorities in public institutions other than government?

Given its difficulty, it is thus unsurprising that this task has traditionally been performed manually, by professional coders. Survey coding is thus also an expensive task, and this is the reason why social scientists (or other professionals in charge of designing and administering surveys) tend to avoid including too many open-ended questions in their surveys, and tend to rely more on the less expensive multiple-choice questions, which by definition do not require a coding phase, but on the other hand strictly limit the respondents' possible answers.

Some attempts have been made in the past at automating the survey coding task. Most of them have exploited simple techniques from the tradition of text retrieval, for matching (or detecting the

[2] http://www.norc.uchicago.edu/

[3] Actually, rather than from the real answers, NORC coders work from typewritten versions of the handwritten notes taken by the interviewers. As a consequence most "answers" are shorthands, and are thus, from a syntactic point of view, ill-formed sentences.

similarity between) the answer and textual descriptions of the meanings of the candidate codes [24]. In this paper we take a radically new stand, and formulate the problem of automated survey coding as a *(multiclass) text categorization* problem, i.e. as the problem of learning, by means of supervised learning techniques, a model of the association between answers and codes from a training set of manually pre-coded answers, and applying the resulting model to the classification of new answers into exactly one of the predefined codes. In this paper we experiment with two different supervised learning techniques (one based on naïve Bayesian classification [16, 19], and another based on multiclass support vector machines [11]) and test the resulting framework on a corpus of social surveys conducted by NORC. The results we obtain significantly outperform the results achieved by previous automated survey coding approaches.

This paper is structured as follows. Section 2 introduces survey coding, and reviews related work attempting to automate this task. Section 3 gives a brief introduction to text categorization, its methodology and its main techniques. In Section 4 we describe how the survey coding task can be framed as a multiclass text categorization problem, and describe how the tools outlined in Section 3 can be effectively used to this end. Section 5 illustrates the experiments we have performed in the application of naïve Bayesian classification and support vector machines to the problem of coding a corpus of social surveys collected by NORC. Section 6 concludes, commenting our results and discussing possible avenues for further research.

2 Survey coding and its automation

From a general point of view, survey analysis shares many aspects with the more general framework of *text analysis for the social sciences*, whereby quantitative methods (i.e. requiring some measurement of frequency) and/or qualitative methods (i.e. not requiring any such measurement) are applied to the study of text corpora representative of a sample of a given population, in order to infer properties of the population itself. In [1] Alexa describes the various issues involved in the computer-assisted analysis of text for the social sciences, while in [2] Alexa and Züll review commercial and research software packages designed for these tasks. In this paper we will not deal with the general problem of text analysis for the social sciences (which involves phases such as text import, export, management, and exploration, plus dictionary and classification scheme construction), but will just focus on the coding task of the survey analysis process.

Survey coding may be viewed as the task of identifying common meaningful concepts across different responses to the same set of questions. This task may come in two variants, depending whether the concepts sought are known in advance (in which case the task really consists in checking whether a given concept is present or not in the text under analysis) or not (in which case novel, unforeseen concepts may be "discovered" in the text being analyzed). In this paper we will concentrate on the former variant, which is the most common for the simple reason that surveys are usually run with a clear purpose, i.e. with a clear set of previously identified concepts whose presence in the text corpus must be assessed or measured in some way.

The process of mapping responses into codes is both slow and expensive, since a lot of manual effort by different professional figures is involved. For example, NORC interviewers take handwritten notes of the answers returned during the interview, and typists produce a typewritten text from these notes at a later stage; this text is then analyzed by professional coders, who perform the final coding task. Yet another drawback is that the process is likely to produce faulty encodings, as there are several potential sources of error: interviewers may misunderstand the answers or misrepresent them by their notes, typists may misread or misunderstand the handwritten notes or introduce further typing errors, and coders may misinterpret the meaning either of the answers or of the codes. Our automated approach currently deals only with the last phase, i.e. coding transcript data, thus ignoring the errors possibly introduced in the previous steps; an even better approach might be to code "first-hand data", i.e. data automatically transcribed directly from speech (see Section 6 for a discussion).

A further problem in survey coding is the so-called *inter-coder agreement* problem, i.e. the fact that different coders may classify the same data in different ways simply because they have different opinions as to the meaning of the answer and/or the code. Note that this problem does not only affect *manually performed* survey coding, since we may also expect different *automated* methods to differ in their decisions[4].

Given that text analysis for the social sciences is an important issue, several software packages that address it have been developed. However, they are not usually tailored for the specific task of survey analysis, and the solutions that they provide for the survey coding task are still unsatisfactory. Many of these software packages (see [2] for a review) mostly concentrate on helping coders in coding their data *manually*, and in visualizing them in several convenient ways. A few of these packages instead do perform automatic coding, by relying mainly on specialized *dictionaries* (or *rules*). This means that text fragments are automatically assigned to a specific category if and only if they contain words matching those in the dictionary relevant to the category. One of the disadvantages of this approach is that dictionaries have to be created *before* the coding process begins, i.e. when data is still totally unknown; this approach is thus extremely static. The second drawback is that specialized dictionaries need to be developed, one for each category of interest; this requires the intervention of expert personnel, who is then responsible for deciding which words, if present (either alone or in combination) or absent in an answer, should trigger the attribution of the code to the answer. The same expert personnel must re-intervene if a new category is added to the scheme, since a dictionary must be created for the new category, and the dictionaries for the old categories need to be updated in order to avoid "capturing" the answers that are instead to be filed under the newly introduced category.

The scientific literature on automating survey coding is scarce. Dillon [8] reports doing "automatic classification" of surveys, but what he actually does is grouping *questions* (and not responses, or respondents, as we do) into groups according to similarity; also, he does not start from a predefined set of groups, but generates the groups from scratch, which means that he uses unsupervised learning (i.e. clustering), and not supervised learning as we instead do. A few other researchers (e.g. [4, 15, 25]) have concentrated on the related issue of *grading* responses to open-ended questions, like those answered by students in their school essays. However, this task deals with concerns very different from those addressed in survey coding, since student essays have to be graded according to quality or merit, while surveys have to be coded according to topic-relatedness.

The approach that is closest in spirit to ours is probably the dictionary-based approach as it is described in Viechnicki's work [24]. In this paper responses to questions from NORC General Social Survey are classified by means of a set of codes predefined by NORC social scientists. Viechnicki proposes two alternative approaches. In the first one, words that characterize a given category are combined by means of Boolean operators, and the answer is classified under the category whose Boolean description it matches. The second method is instead based on computing the similarity between two weighted vectors of words extracted from the answer and from a textual explicatory caption of the code, and choosing the code with the highest similarity score. Similarly, Macchia and Murgia [18] present a dictionary-based automated approach in which the answer is assigned a unique code if there is an exact match with phrases belonging to a previously defined dictionary associated with the code, or is assigned a "best code" if the match is partial. These approaches have the typical drawback of dictionary-based methods, which need a dictionary to be manually developed before the actual coding step takes place, and to be manually updated as a result of changes in the structure and/or semantics of the coding scheme.

Our approach to survey coding has several advantages with respect to the dictionary-based approach. Firstly, in our learning-based approach the manual effort is directed towards the manual coding of a small training set of answers, and not towards the creation of specialized dictionaries. This is advantageous, as it is easier to manually classify a set of documents than to build and tune a dictionary of words that trigger the attribution of the code, for the simple fact that it is easier to characterize a concept extensionally (i.e. to select instances of it) than intensionally (i.e. to describe the concept in words, or to describe a procedure for recognizing its instances). Secondly,

[4] *Inter-indexer inconsistency* is a similar phenomenon well-known in information retrieval [5]: when two human experts decide whether to index document d_j with index term c_i, they may disagree, and this in fact happens with relatively high frequency.

our approach is solidly grounded in machine learning theory, and it can leverage on a wealth of results and techniques developed within text categorization, a discipline which has been bursting with activity in the last ten years (see e.g. [21]) and has produced systems whose accuracy rivals or exceeds that of a human (i.e. systems capable of generating codes that correlate with those attributed by a coder at least as well as the codes attributed by two human coders correlate with each other).

Of course, our approach is mostly useful for medium- to large-sized surveys, as in the learning phase we need a hand-coded set of answers to train the inductive learner. This means that if a survey is somewhat limited in the number of surveyed people, hand-coding the training set may coincide with hand-coding the entire set. NORC's surveys are examples of relatively large-sized surveys, since 40,933 interviews have been completed in the years from 1972 to 2000, and since the same survey (e.g. the General Social Survey mentioned in Section 1) is administered from year to year, with many questions being asked unmodified year after year.

3 A short introduction to text categorization

We now turn to a brief introduction to text categorization, since this will be the main tool we will adopt to tackle the survey coding task.

Text categorization (also known as *text classification*) is the task of approximating the unknown *target function* $\Phi : \mathcal{D} \times \mathcal{C} \rightarrow \{T, F\}$ (that describes how documents ought to be classified) by means of a function $\hat{\Phi} : \mathcal{D} \times \mathcal{C} \rightarrow \{T, F\}$ called the *classifier*, where $\mathcal{C} = \{c_1, \ldots, c_{|\mathcal{C}|}\}$ is a predefined set of thematic categories and $\mathcal{D}$ is a domain of documents. If $\Phi(d_j, c_i) = T$, then d_j is called a *positive example* (or a *member*) of c_i, while if $\Phi(d_j, c_i) = F$ it is called a *negative example* of c_i. The categories are just symbolic labels, and no additional knowledge (of a procedural or declarative nature) of their meaning is usually available. It is usually the case that no metadata (such as e.g. publication date, document type, publication source) are available either; therefore, classification must be accomplished only on the basis of knowledge extracted from the documents themselves.

Text categorization is a *subjective* task: when two experts (human or artificial) decide whether to classify document d_j under category c_i, they may disagree, and this in fact happens with relatively high frequency. A news article on George W. Bush attending a Texas Rangers game could be filed under Politics, or under Sport, or under both, or even under neither, depending on the subjective judgment of the expert.

Depending on the application, it might be the case that exactly one $c_i \in \mathcal{C}$ must be assigned to each $d_j \in \mathcal{D}$, or that any number $0 \leq n_j \leq |\mathcal{C}|$ of categories may be assigned to each $d_j \in \mathcal{D}$. The former case is usually dubbed the *binary* case or the *multiclass* case, depending on whether $|\mathcal{C}| = 2$ or $|\mathcal{C}| > 2$, respectively. Since the $|\mathcal{C}| > 2$ case will be the object of interest in this paper, from here on when speaking of TC we will actually mean *multiclass* TC[5].

We can roughly distinguish three different phases in the life cycle of a TC system: document indexing, classifier learning, and classifier evaluation. The three following paragraphs are devoted to these three phases, respectively; for a more detailed treatment see Sections 5, 6 and 7, respectively, of [21].

3.1 Document indexing

Document indexing denotes the activity of mapping a document d_j into a compact representation of its content that can be directly interpreted (i) by a classifier-building algorithm and (ii) by a classifier, once it has been built. The document indexing methods usually employed in TC are borrowed from information retrieval (IR), where a text d_j is typically represented as a vector of term *weights* $\vec{d_j} = \langle w_{1j}, \ldots, w_{|\mathcal{T}|j} \rangle$. Here, $\mathcal{T}$ is the *dictionary*, i.e. the set of *terms* (also known as *features*) that occur at least once in at least α documents, and $0 \leq w_{kj} \leq 1$ quantifies the importance of t_k in characterizing the semantics of d_j. Typical values of α are between 1 and 5.

[5]For strange reasons that we will not discuss here, multiclass TC is sometimes also referred to as *single-label* TC, which is admittedly confusing ...

An indexing method is characterized by (i) a definition of what a term is, and (ii) a method to compute term weights. Concerning (i), the most frequent choice is to identify terms either with the *words* occurring in the documents (with the exception of *stop words*, i.e. topic-neutral words such as articles and prepositions, which are eliminated in a pre-processing phase), or with their *stems* (i.e. their morphological roots, obtained by applying a stemming algorithm). Concerning (ii), either statistical or probabilistic techniques are used to compute terms weights, the former being the most common option. One popular class of statistical term weighting functions is $tf * idf$ (see e.g. [20]), where two intuitions are at play: (a) the more frequently t_k occurs in d_j, the more important for d_j it is (the *term frequency* intuition); (b) the more documents t_k occurs in, the less discriminating it is, i.e. the smaller its contribution is in characterizing the semantics of a document in which it occurs (the *inverse document frequency* intuition). Weights computed by $tf * idf$ techniques are often normalized so as to contrast the tendency of $tf * idf$ to emphasize long documents.

In TC, unlike in IR, a *dimensionality reduction* phase is often applied so as to reduce the size of the document representations from $\mathcal{T}$ to a much smaller, predefined number. This has both the effect of reducing *overfitting* (i.e. the tendency of the classifier to better classify the data it has been trained on than new unseen data), and to make the problem more manageable for the learning method, since many such methods are known not to scale well to high problem sizes. Dimensionality reduction often takes the form of *feature selection*: each term is scored by means of a scoring function that captures its degree of (positive, and sometimes also negative) correlation with c_i, and only the highest scoring terms are used for document representation.

3.2 Classifier learning

A text classifier for categories $\mathcal{C} = \{c_1, \ldots, c_{|\mathcal{C}|}\}$ is automatically generated by a general inductive process (the *learner*) which, by observing the characteristics of a set of documents preclassified under $\mathcal{C}$, gleans the characteristics that a new unseen document should have in order to belong to a generic category $c_i \in \mathcal{C}$. In order to build classifiers for $\mathcal{C}$, one thus needs a labelled corpus Ω of documents such that the value of $\Phi(d_j, c_i)$ is known for every $\langle d_j, c_i \rangle \in \Omega \times \mathcal{C}$. In experimental TC it is customary to partition Ω into three disjoint sets Tr (the *training set*), Va (the *validation set*), and Te (the *test set*). The training set consists of documents the learner "observes" to build up the classifier. The validation set is the set of documents the engineer uses to fine-tune the classifier, e.g. choosing for a parameter p on which the classifier depends, the value that has yielded the best effectiveness when evaluated on Va. The test set is used for the final evaluation of classifier effectiveness. In both the validation and test phase, "evaluating the effectiveness" means running the classifier on a set of preclassified documents (Va or Te) and checking the degree of correspondence between the output of the classifier and the preassigned labels. This is called a *supervised* learning activity, since learning is "supervised" by the information on the membership of training documents in categories.

Several methods have been proposed in the text categorization literature for learning a text classifier from training data (see [21] for a review), including probabilistic methods, regression methods, decision tree and decision rule learners, neural networks, batch and incremental learners of linear classifiers, example-based methods, support vector machines, genetic algorithms, hidden Markov models, and classifier committees. Some of these methods generate binary-valued classifiers of the required form $\hat{\Phi} : \mathcal{D} \times \mathcal{C} \rightarrow \{T, F\}$, but some others generate real-valued functions of the form $CSV : \mathcal{D} \times \mathcal{C} \rightarrow [0, 1]$ (CSV standing for *categorization status value*). For these latter, a set of thresholds τ_i needs to be determined (typically, by experimentation on a validation set) allowing to turn real-valued CSVs into the final binary decisions.

3.3 Classifier evaluation

Training efficiency (i.e. average time required to build a classifier $\hat{\Phi}$ from a given corpus Ω), as well as *classification efficiency* (i.e. average time required to classify a new document by means of $\hat{\Phi}$), and *effectiveness* (i.e. average correctness of $\hat{\Phi}$'s classification behaviour) are different measures of success for a learner. However, effectiveness is usually considered the most important criterion, since in most applications one is willing to trade training time and classification time for correct decisions. Also, it is the most reliable one when it comes to comparing different learners, since efficiency depends on too volatile parameters (e.g. different sw/hw platforms). As a result, we will only measure the success of our approach in terms of effectiveness.

In multiclass TC, effectiveness is usually equated to *accuracy*, which is defined as the percentage of classification decisions that are actually correct.

4 Automated survey coding by text categorization

In this section we describe our experiments with survey coding handled as a text categorization task, i.e. as the task of automatically generating a classifier that automatically selects, from a set of predefined codes, the correct code to attach to a given answer. In our survey coding context, the set of all answers to a given question q play the role of the domain $\mathcal{D}$, and the set of all possible codes that may be attributed to an answer to question q play the role of the set of categories $\mathcal{C}$ (coding the answers to different questions thus corresponds to different TC tasks).

The input to the learners (and to the classifiers, once they have been built), consists of an answer d_j represented as a vector of term weights $\vec{d_j} = \langle w_{1j}, \ldots, w_{|\mathcal{T}|j} \rangle$; we use a value of 1 for the α parameter (see section 3.1). Note that the fact that many of the answers in the corpus are syntactically ill-formed (see examples in Section 1) makes this "bag of words" approach to representation (i.e. the approach that just considers term occurrence and frequency of occurrence, disregarding deeper syntactic and semantic aspects) even more appropriate: given that current syntactic and semantic analysis techniques have not proven worthy (i.e. well-performing and robust at the same time) in standard TC, where we usually deal with syntactically well-formed text, it is easy to conjecture that they could hardly prove worthy here.

In this work we have run a series of experiments with two different classifier-learning methods. The first learner we use is a probabilistic naïve Bayesian learner, as implemented in the Rainbow package[6]. Probabilistic text classification methods assume that the data was generated by a parametric model, and use the training set to estimate the parameters of this model. Bayes' theorem allows to estimate from this model the probability that a given category has generated the document to be classified; classification thus consists in selecting the category with the highest probability. There are two well-known variants of this method, the multi-variate Bernoulli method and the multinomial method [19]. In this paper we chose the latter, since in comparative text classification experiments it performed better than the former [19].

The second learning method we use is a *multiclass support vector machine* (SVM) learner as embodied in the MCSVM package[7]. SVMs attempt to learn a hyperplane in $|\mathcal{T}|$-dimensional space that separates the positive training examples of category c_i from the negative ones with the maximum possible margin, i.e. such that the minimal distance between the hyperplane and a training example is maximum; results in computational learning theory indicate that this tends to minimize the generalization error, i.e. the error of the resulting classifier on yet unseen examples. SVMs were initially conceived for solving binary classification problems, and only recently they have been adapted to multiclass classification.

Crammer and Singer describe in [6] an algorithmic implementation of multiclass SVMs based on a notion of margin generalized to multiclass problems, which allows to train directly a multiclass classifier (while in most of previous work the multiclass problem is decomposed into multiple independent binary classification tasks [10]).

Regarding effectiveness, the text categorization literature has shown that naïve Bayesian approaches are, with respect to other learning methods, no more than average performers (see e.g. [9, 13, 17, 27]). On the contrary, support vector machines are currently (together with

[6]Rainbow was implemented by Andrew McCallum and can be downloaded from `http://www.cs.cmu.edu/~mccallum/rainbow`.

[7]MCSVM was implemented by Koby Crammer and Yoram Singer, and we were kindly provided with a pre-release version of it.

"boosting"-based classifier committees) the unsurpassed top performers in the TC field [9, 13]. The reason why we experiment with RAINBOW is that we want to show that a text categorization approach to survey coding is much more effective than the dictionary-based approach *regardless of the specific learning method adopted*, i.e. that even with an average-performing learning method our text categorization approach to survey coding can outperform the dictionary-based method. Instead, the reason why we experiment with MCSVM is that we want to show what level of effectiveness this approach can achieve, once instantiated with a top-performing learning algorithm [8].

We have used a binary representation as input to RAINBOW, and a non-binary one as input to MCSVM. This is due to the fact that the probabilistic models upon which RAINBOW is based require binary inputs, while this is not the case for SVMs. In the binary representation, w_{kj} represents just presence or absence of term t_k in answer d_j. Our non-binary representation is instead the *tfidf* function in its standard "ltc" variant [20], i.e.

$$tfidf(t_k, d_j) = tf(t_k, d_j) \cdot \log \frac{|Tr|}{\#_{Tr}(t_k)} \tag{1}$$

where $\#_{Tr}(t_k)$ denotes the number of answers in the training set Tr in which t_k occurs at least α times and

$$tf(t_k, d_j) = \begin{cases} 1 + \log \#(t_k, d_j) & \text{if } \#(t_k, d_j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

where $\#(t_k, d_j)$ denotes the number of times t_k occurs in answer d_j. Weights obtained by Equation 1 are normalized by cosine normalization, yielding

$$w_{kj} = \frac{tfidf(t_k, d_j)}{\sqrt{\sum_{s=1}^{|\mathcal{T}|} tfidf(t_s, d_j)^2}} \tag{2}$$

In all the experiments discussed in this paper, stop words, punctuation, and numbers, have been removed, and all letters have been converted to lowercase. No feature selection (see e.g. [21, Section 5.1]) has been performed. The reason is that, as shown in extensive experiments by Brank et al. [3], the effectiveness of SVMs is usually worsened by feature selection, irrespectively of the feature selection algorithm used and of the chosen reduction factor (this is also independently confirmed by the results of [23]), and the effectiveness of naïve Bayesian methods does not show systematic patterns of improvement either.

5 Experiments

As already pointed out, our experiments have been carried out on data from NORC's General Social Survey. This survey, which is ongoing since 1972, aims at investigating how people assess their physical and mental health, the balancing of security and civil liberties, external and internal security threats, intergroup relations and cultural pluralism, religious congregations, etc. We deal with three datasets (see Table 1) from the NORC General Social Survey administered in 1996. Each of these datasets (here nicknamed *angry_at*, *angry_why*, and *brkdhlp*) consists of a set of answers to a given question, plus their associated category codes manually chosen by NORC's professional coders from a predefined set of category codes[9]. The task consists in choosing exactly one code for each answer.

[8]Note that there are no published results yet concerning the application to TC of *multiclass* SVMs, because multiclass SVMs are a recent development, and because most TC applications are binary. The assumption that multiclass SVMs would be a top performer once used in a multiclass TC context is based on the top performance that multiclass SVMs have delivered in multiclass application contexts other than TC [6].

[9]The *angry_at* and *angry_why* datasets actually involve the same question, which deals with the description of a situation that caused anger to the respondent; each answer was classified according to *two* different sets of codes, one concerning the object of anger, the other concerning the cause of anger. Actually, *angry_why* contains only a subset of the answers contained in *angry_at*, in the sense that NORC coders classified some of the answers only according to the *angry_at* set of codes. The *brkdhlp* dataset (called *breakdown* in [24]) consists of answers to the question as to what source of help was used to deal with a nervous breakdown.

Dataset	**Category**	**# of instances**
angry_at	ANGRYFAM	275
	ANGRYWRK	345
	ANGRYGVT	74
	WRK&GVT	8
	WRK&FAM	27
	FAM&GVT	16
	OTHER	625
	total	1370
angry_why	SELF	29
	PREVENTED	36
	CRITICAL	88
	DEMANDING	60
	EXPECT	196
	OTHER	51
	total	460
brkdhlp	FAMILY	57
	FRIEND	33
	GROUP	2
	CLERGY	55
	PSYCHIATRIST	56
	AGENCY	16
	OTHER	148
	total	367

Table 1: Characteristics of the three datasets used in our experiments.

We have chosen these three datasets because they are the same datasets used in [24], which means that we will be able to obtain a direct comparison between the effectiveness of Viechnicki's method (which is an example of the dictionary-based approach to survey coding) and the effectiveness of our supervised learning approach.

Note that all three datasets include a class OTHER. This consideration alone indicates that these datasets are not "easy", since

- the classifier cannot simply "take a guess" by picking "the least inappropriate" category, since if all choices are sufficiently inappropriate, the category OTHER applies[10];
- the category OTHER will typically be a very hard category to work with, since it will not be characterized by a specific terminology, as is instead the case with categories that are strongly characterized in a topical sense. The presence of a category OTHER in the category set always tends to deteriorate the global performance of any text classifier.

For each dataset, the main steps we went through to run our experiments are the following:

1. preprocess the data in order to obtain a data format compatible with the learners (this had to be repeated once for Rainbow and once for MCSVM, since the two systems require different input formats);
2. partition the set of answers in each dataset in four random disjoint subsets of equal size;
3. run the learner to generate a classifier, using three of the four subsets as the training set and the fourth as the test set;
4. run the classifier to classify the data in each test set of each dataset and evaluate the results.

In order to achieve better statistical significance, in all experiments steps 3 and 4 were repeated four times, for all four possible choices of the test set. Each of the results we report is thus the

[10]This is confirmed by everybody's experience with multiple-choice tests. No student likes to take a multiple-choice test in which the last choice is always "None of the preceding answers apply"!

	Dictionary-Based [24]		Supervised Learning	
	Vector	Boolean	Rainbow	MCSVM
angry_at	0.451	0.465	0.714 (+54%)	**0.756 (+63%)**
angry_why	0.211	0.272	**0.389 (+43%)**	0.376 (+38%)
brkdhlp	0.646	**0.747**	0.653 (-13%)	0.746 (-0.13%)
Average	0.436	0.495	0.585 (+18%)	**0.626 (+26%)**
Std. Dev.	0.218	0.239	**0.173 (-28%)**	0.216 (-10%)

Table 2: Comparative accuracy results obtained on the *angry_at*, *angry_why* and *brkdhlp* datasets using a Boolean and a vector-based method and using a naïve Bayesian and a multiclass SVM TC method. The percentile improvements in accuracy and average accuracy, and the percentile reductions in standard deviation, are reported with respect to the Boolean method, the best dictionary-based method in [24]. **Boldface** indicates the best performance on the dataset.

result of averaging across four different experiments. We have computed the accuracy on the three datasets both with Rainbow and with MCSVM; the results are reported in Table 2, where they are compared with the accuracy obtained in [24] on the same datasets.

The first observation we can make is that the supervised learning approach to survey coding significantly outperforms the dictionary-based approach: the improvements with respect to the best-performing method reported in [24] are significant, a +18% on average for Rainbow and a +26% for MCSVM. The improvement is especially noteworthy on the "non-obvious" datasets: for instance, *angry_why* appears to be a hard to characterize dataset, as shown by the poor performance of the two dictionary-based methods, and on this dataset the supervised learning methods improve up to +43% with respect to the best one. The *angry_at* dataset looks somehow "easier" than *angry_why*, as witnessed by the fact that all four methods listed in Table 2 perform better on *angry_at* than on *angry_why*. This might also be explained by the fact that, as it can be seen from Table 1, it contains more data, since each category in *angry_at* has 195 positive examples on average, while this goes down to 76 for *angry_why*. On the contrary, the *brkdhlp* dataset seems easy to tackle by simple Boolean rules, as shown by the .747 accuracy figure of the Boolean method; in this case Rainbow underperforms the Boolean method by 13%, while MCSVM virtually delivers the same performance as the Boolean method.

Moreover, the supervised learning approach delivers a more stable performance across the three datasets, since the reductions in standard deviation with respect to the same best-performing method are very significant, a -28% for Rainbow and a -10% for MCSVM. These improvements are even more significant once we remember that they are obtained by a method that is much cheaper than the dictionary-based method in terms of expert human resources .

As anticipated in Section 4, the fact that improvements of this order of magnitude are obtained even with a method, such as the naïve Bayesian technique implemented in Rainbow, which is known as an average performer in the text categorization literature, bears witness to the superiority of the supervised learning approach to survey coding.

The fact that multiclass SVMs, known top-performers in the machine learning literature (see e.g. [6]), outperform Rainbow only by a very small margin, is more surprising, and might be dependent on the data we experimented with. In fact, a possible explanation might be that the vocabulary of the NORC corpora exhibits low internal stochastic dependence, hence approximating the conditions under which Bayesian approaches are theoretically optimal.

6 Conclusion

We have shown that automatic coding of responses to open-ended survey questions may be posed as a multiclass text categorization problem, and that text categorization techniques based on supervised learning may significantly outperform dictionary-based techniques, such as those used in [24] that have been up to now the dominant approach to automated survey coding. Another advantage of the supervised learning approach with respect to the dictionary-based approach, which requires that the text classifiers be handcrafted (by a knowledge engineer and a social scientist working together), is that the classifiers can be generated automatically from the training data, with substantive savings in terms of expert human resources.

The effectiveness levels that text categorization techniques have achieved in our experiments are far from being perfect, and also from being completely satisfactory. Although the results obtained in our research are promising, we think that more research is needed for the automatic approach to survey coding to clearly supersede the manual approach. There are several avenues for further research. One of these, which we are currently working at, is simply to run experiments on more survey data, in order to obtain results which are statistically more reliable. Another possible line of research is to experiment with more satisfactory multiclass TC learners, in order to improve upon the results of Rainbow and MCSVM.

In the future, we plan to combine automated survey coding by text categorization with speech recognition, in order to allow the survey coding task to proceed directly from the audio recording of the interview, since we believe that survey coding may be performed with much better effectiveness only by using better quality input, i.e. more faithful representations of the answers. Proceeding directly from the audio recording can eliminate the sources of noise mentioned in Section 2 (i.e. the noise possibly introduced by interviewers and typists), and also makes for greater savings in term of human resources, which means that the researchers who design the survey could afford having more open-ended questions and less multiple-choice ones.

Although such a process requires to apply text categorization to noisy text (due to the current performance of speech recognition software), we think that there are reasons for optimism, since previous research in the optical character recognition field [12, 14] shows that effectiveness levels comparable to those obtainable in the case of standard text may be achieved. Thus it is possible that similar effectiveness patterns might result also in the case of noise introduced by speech recognition.

Acknowledgements

We wish to thank Irina Prodanof for providing the initial impetus behind this research. The open-ended text used in this work was collected in the General Social Surveys of the National Opinion Research Center (NORC), University of Chicago, and supplied by NORC to the authors. We are grateful to Tom Smith and Jennifer Berktold for providing these texts and for assisting us in their interpretation. We are also grateful to Koby Crammer, Yoram Singer and Andrew McCallum for making the MCSVM and Rainbow packages available, to Peter Viechnicki for clarifying several points of his experiments, to Henri Avancini for helping with several preprocessing issues, and to David Lewis for suggesting pointers to the automated survey coding literature.

References

[1] Melina Alexa. Computer-assisted text analysis methodology in the social sciences. Technical Report 97/07, Zentrum für Umfragen, Methoden und Analysen, Mannheim, DE, 1997.

[2] Melina Alexa and Cornelia Züll. Text analysis software: Commonalities, differences and limitations. The results of a review. *Quality and Quantity*, 34:299–321, 2000.

[3] Janez Brank, Marko Grobelnik, Natasa Milić-Frayling, and Dunja Mladenić. Interaction of feature selection methods and linear classification models. In *Proceedings of the ICML-02 Workshop on Text Learning*, Sydney, AU, 2002.

[4] Jill Burstein, Karen Kukich, Susanne Wolff, Chi Lu, Martin Chodorow, Lisa Braden-Harder, and Mary Dee Harris. Automated scoring using a hybrid feature identification technique.

In *Proceedings of the ACL-98, 36th Annual Meeting of the Association for Computational Linguistics*, pages 206–210, Montreal, CA, 1998.

[5] Cyril Cleverdon. Optimizing convenient online access to bibliographic databases. *Information Services and Use*, 4(1):37–47, 1984. Also reprinted in [26], pp. 32–41.

[6] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

[7] James A. Davis and Tom Smith. *General Social Surveys, 1972-1996: Cumulative Codebook.* National Opinion Research Center, Chicago, US, 1996.

[8] Martin Dillon. Automatic classification of Harris survey questions: An experiment in the organization of information. *Journal of the American Society for Information Science*, pages 294–301, 1982.

[9] Susan T. Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In Georges Gardarin, James C. French, Niki Pissinou, Kia Makki, and Luc Bouganim, editors, *Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management*, pages 148–155, Bethesda, US, 1998. ACM Press, New York, US.

[10] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

[11] Chih-Wei Hsu and Chih-Jen Lin. A simple decomposition method for support vector machines. *Machine Learning*, 46:291–314, 2002.

[12] David J. Ittner, David D. Lewis, and David D. Ahn. Text categorization of low quality images. In *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 301–315, Las Vegas, US, 1995.

[13] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE. Published in the “Lecture Notes in Computer Science” series, number 1398.

[14] Markus Junker and Rainer Hoch. An experimental evaluation of OCR text representations for learning document classifiers. *International Journal on Document Analysis and Recognition*, 1(2):116–122, 1998.

[15] Leah S. Larkey. Automatic essay grading using text categorization techniques. In W. Bruce Croft, Alistair Moffat, Cornelis J. van Rijsbergen, Ross Wilkinson, and Justin Zobel, editors, *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 90–95, Melbourne, AU, 1998. ACM Press, New York, US.

[16] David D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE. Published in the “Lecture Notes in Computer Science” series, number 1398.

[17] Hang Li and Kenji Yamanishi. Text classification using ESC-based stochastic decision lists. *Information Processing and Management*, 38(3):343–361, 2002.

[18] Stefania Macchia and Manuela Murgia. Coding of textual responses: Various issues on automated coding and computer assisted coding. In *Proceedings of JADT-02, 6th International Conference on the Statistical Analysis of Textual Data*, pages 471–482, St-Malo, FR, 2002.

[19] Andrew K. McCallum and Kamal Nigam. A comparison of event models for naive Bayes text classification. In *Proceedings of the 1st AAAI Workshop on Learning for Text Categorization*, pages 41–48, Madison, US, 1998.

[20] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988. Also reprinted in [22], pp. 323–328.

[21] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[22] Karen Sparck Jones and Peter Willett, editors. *Readings in information retrieval.* Morgan Kaufmann, San Mateo, US, 1997.

[23] Hirotoshi Taira and Masahiko Haruno. Feature selection in SVM text categorization. In *Proceedings of AAAI-99, 16th Conference of the American Association for Artificial Intelligence*, pages 480–486, Orlando, US, 1999. AAAI Press, Menlo Park, US.

[24] Peter Viechnicki. A performance evaluation of automatic survey classifiers. In Vasant Honavar and Giora Slutzki, editors, *Proceedings of ICGI-98, 4th International Colloquium on Grammatical Inference*, pages 244–256, Ames, US, 1998. Springer Verlag, Heidelberg, DE. Published in the “Lecture Notes in Computer Science” series, number 1433.

[25] Dave Whittington and Helen Hunt. Approaches to the computerized assessment of free text responses. In *Proceedings of the 3rd Annual Computer-Assisted Assessment Conference*, pages 207–219, Loughborough, UK, 1999.

[26] Peter Willett, editor. *Document retrieval systems.* Taylor Graham, London, UK, 1988.

[27] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In Marti A. Hearst, Fredric Gey, and Richard Tong, editors, *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pages 42–49, Berkeley, US, 1999. ACM Press, New York, US.