Mapping an Automated Survey Coding Task into a Probabilistic Text Categorization Framework ⋆

Daniela Giorgetti[1], Irina Prodanof[1], Fabrizio Sebastiani[2]

[1] Istituto di Linguistica Computazionale del CNR di Pisa, Italia.
[2] Istituto di Elaborazione dell'Informazione del CNR di Pisa, Italia.
{daniela.giorgetti, irina.prodanof}@ilc.cnr.it, fabrizio@iei.pi.cnr.it

Abstract. This paper describes how to apply a probabilistic Text Categorization method to a different and new domain where documents are answers to open end questionnaires and codes viewed as categories consist of a hierarchical model. A reduced size training set may be used taking advantage of the hierarchical organization of categories. The system developed in this framework aims at helping psychologists in the evaluation of open end surveys inquiring about job candidates' competencies.

1 Introduction

Text Categorization (TC for short) aims at the classification of a text document under one or more predefined categories deciding where the document belongs to, relying just on its content. TC is usually applied to automatic indexing for boolean Information Retrieval (IR for short), document organization, document filtering, word sense disambiguation, categorization of Web pages into hierarchical catalogues, but to our knowledge there are very few attempts at using it for more unusual tasks such as automatic essay grading [4]. Our TC approach to survey coding is part of the JobNet project, which is developing a temporary job agency on the Web, where job applications and job offers have to be matched with efficient and effective criteria. Job candidates connect to the Web site, and after the registration they have to fill in some forms with their personal data, education titles, skills and so on. Besides these standard forms there is a final survey form with open end questions which inquire the candidate's competencies as defined in a psychological behaviorist model. As the psychologists' coding of these surveys is both costly and time consuming we are exploring ways to reduce their amount of work through an automatic evaluation of free form texts.

Synopsis. In Section 2 we describe more in detail the automatic survey coding task, stressing its originality with respect to "traditional" TC and IR tasks. In Section 3 we overview the approaches previously used for text coding, focussing on their drawbacks w.r.t. our task. In Section 4 we give an idea of the underlying model of the hierarchical code categories in JobNet, in Section 5 we introduce a probabilistic approach to TC augmented with a statistical technique which takes advantage of the hierarchical structure of categories, in Section 6 we show how to adapt and instantiate the improved probabilistic TC approach to our case, and in Section 7 we draw our conclusions.

⋆ Research supported by the Sintesi company (Perugia, Italia), which funds the JobNet project.

2 Automated Survey Coding

Hand coding of open end surveys is a classical problem from the social sciences, but up to now there are very few attempts at automating the task, though the Internet world wide diffusion has made it possible to collect huge quantities of such text data from e.g. market surveys and exploratory interviews. In the literature we have found just a few papers which deal with the automatic coding issue, by Perrin [8] by Pratt and Mays [9], by Raud and Fallig [10], and by Viechnicki [14], and none of them applies a TC solution to the problem. Supervised coding of an answer means to find its meaningful part and attach to it a code label from a predefined set of code labels. Reformulating the aim of the process in terms of TC we can say that supervised coding for each answer (or part of it) looks for the category it belongs to (in our case we may also have answers that do not belong to any of the predefined categories) as defined by a predetermined set of categories.

2.1 Automated Survey Coding vs. Information Retrieval

Automatic Survey Coding (ASC for short) shares some aspects of IR since they both operate on unstructured text documents, but there are differences in the data as well as in the nature of the task [14], the main being:

- the length of documents, which is usually much shorter in ASC (6.86 terms on average as stated in [14] versus 50 terms on average as stated in [2]).
- the style of language, which is usually more informal and close to spoken language in ASC, while typical IR documents have a more formal and fixed structure, as they usually come from the academics, news, and technical fields.
- homogeneity of answers depends heavily on the formulation of the questions (queries in IR terms).
- the nature of the task is different as in IR we look for documents relevant to a certain query, while in ASC we have to systematically evaluate all the answers and assign them, if appropriate, to one or more of the predetermined categories.

2.2 Automated Survey Coding vs. Text Categorization

Although we redefine our coding problem as a TC problem we have to point out that ASC has its own peculiarity both in the data (e.g. style and homogeneity of language) and the nature of the task. Moreover, the examples, i.e. the training set, are usually already available in TC (e.g. classified ads), while in ASC building a training set may mean to code the whole set of questionnaires, thus eliminating the necessity of an automatic coder (classifier in TC terms).

2.3 Automated Survey Coding vs. Text Clustering

ASC differs from Text Clustering as in the latter the clustering is a bottom up process, where we start from data and group them according to some similarity criteria without any predefined categories (*qualitative content analysis* in social science terms), whilst the former applies a top down class-driven process as data is assigned to a predefined category set (*quantitative content analysis* in social science terms).

3 Other Approaches to Automated Survey Coding

Text coding, and more in general text content analysis in social sciences, is a long-standing issue, and several software systems have been developed to aid solving this task [1]. Most of these software systems either facilitate the users to hand-code their data and view them in various ways or perform automatic coding relying mostly on word based dictionaries. In the automatic coding case, text fragments are assigned to a specific category 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, and thus it does not rely on the data being analyzed. As this approach is word based it also needs some explicit mechanism to disambiguate word meanings. There exist different variations of the dictionary approach; for instance in one of the two methods described in [14] words defining categories are mutually related through Boolean operators, thus allowing a better characterization of categories. Rule based coding systems are another kind of approach to automatic text coding. Rules can be derived automatically or by hand trying to capture and organize the coders' knowledge of the domain. One such system is described in [8], and relies on the ability of inferring coding rules from pre-coded samples of text. The rule based approach relies on pattern matching, and can be more sophisticated than the dictionary based approach, which relies just on word matching in its basic version, but it still doesn't make use of any form of learning to predict the likelihood of patterns in text. A third approach, which uses neural networks to code answers to open end questions is proposed in [10], but it is stressed that the method works well in the case of supervised learning, only if the distribution of the various input patterns is stable and predictable.

4 Hierarchical Code Categories in JobNet

JobNet domain consists of a set of questionnaires filled in via Web pages by temporary job seekers. Answers are open end but their length is limited to 2 or 3 sentences each. The aim of the questionnaire is to discover the best profiles for a given job offer, through the interpretation and coding of its answers in terms of a competence model. Competencies in JobNet derive from a theoretical model developed in the 70s by the Harvard psychologist David McClelland[3], but such model has become operative and widely used only in the 90s (e.g. adopted by Alcatel). The basic idea is to identify what differentiate outstanding job performers from the average ones. The notion of competence is orthogonal to the notion of technical skill and it is defined by L. Spencer and S. Spencer [12,13], as "an underlying characteristic of an individual that is casually related to criterion-referenced effective and/or superior performance in a job or situation". One of the most important tools to identify competencies is the Behavioral Event Interview (BEI), where the job candidate has to give an example of a situation or task which led her to take a certain course of action. For the selection in JobNet, a new focused BEI interview protocol has been developed, represented by a written survey, where specific competencies are being investigated. For example, to inquiry about achievement

[3] McClelland (1917-1998) also founded the McBer company for human resources management that today is part of the Hay Group.

competencies, questions are of the kind: "What did you do to reach your goal", in a context where the candidate is describing a difficult situation she faced. Competence models developed by the Hay Group [3] and adapted by the JobNet psychologists are scaled, i.e., not only is it important to identify the presence of a competence but also the grade of its presence, because that is what makes the difference between outstanding candidates and the others. The developed model is described by a scaled competence dictionary organized in hierarchical competence groups, where the four main clusters are:

- Realization competencies
- Relational competencies
- Cognitive competencies
- Crisis management competencies

Each of these groups includes from 4 to 7 competencies, for example, the cognitive competencies group includes Analytical Reasoning (AR), Conceptual Thinking (CONC), Information Seeking (INF), Synthetical Thinking (ST). Each of these competencies includes from 3 to 8 behavior indicators, i.e., typical behaviors that reveal different grades of competence in a certain task.

5 Bayesian TC Improved by Shrinkage Applied to Hierarchical Categories

The problem of TC is usually dealt with using not only Machine Learning tehniques, but also techniques from fields such as Information Retrieval, Data Mining applied to unstructured text (also called Text Mining), and Natural Language Processing. Here we briefly recall which are the fundamental steps for the automatic categorization of a document (see [11] for a thorough introduction to the field):

- the generation of a *text representation* automatically interpretable by a classifier, which may have three substeps: pre-processing, indexing, and term reduction
- the application of a method for building a classifier by induction from a set of pre-categorized documents, called *training set*
- the evaluation of the quality of the classifier obtained experimentally from a set of pre-categorized documents disjoint from the set used for training, called *test set*.

5.1 Bayesian TC

We describe more in detail the probabilistic bayesian approach [7] to TC (adopting the notation in [6]), as it has been shown to be very effective in many cases, and we have decided to start our experimentation in JobNet in such a framework. A probabilistic classifier given a document d and a category c returns a normalized real number in $[0,1]$ representing its grade of certainty that document d belongs to category c. In probabilistic approaches textual data is assumed to be generated by an unknown parametric mixture model (parameterized by θ), whose parameters are estimated by means of a

collection of labeled training examples. Categorization of a previously unseen document is then achieved using Bayes rule to estimate the category that is most likely to have generated the new document.

$$P(d_i|\theta) = \sum_{j=1}^{|C|} P(c_j|\theta)P(d_i|c_j;\theta) \tag{1}$$

In a multinomial model a document d_i is seen as an ordered sequence of word events (as many as the length of d_i) drawn from the same vocabulary V [5]. Under the Naive Bayes assumption that each word event in a document is independent of the word's context and position in it given its class, the document may be represented by the usual *bag of words* vector and the classification task then coincides with the choice of the category that maximizes the probability of generating the words appearing in the document. Moreover we assume that the length of the document $|d_i|$ is independent of the category. Though these hypotheses are violated by real data (e.g. the probability of seeing in this paper the word *categorization* after the word *text* is greater than with other preceding words) Naive Bayes classifiers have been shown to behave very well in many TC applications. The probability of generating a document d_i given its category c_j and the model θ is:

$$P(d_i|c_j;\theta) = P(|d_i|)\prod_{k=1}^{|d_i|} P(w_{d_{ik}}|c_j;\theta) \tag{2}$$

where $w_{d_{ik}}$ denotes the word in position k of document d_i, the subscript d_{ik} indicates an index into the vocabulary, and $|d_i|$ denotes document d_i length. The category prior probability $P(c_j|\theta)$ may be estimated by:

$$P(c_j|\theta) = \frac{\sum_{i=1}^{|D|} P(c_j|d_i)}{|D|} \tag{3}$$

Given these estimates of the parameters computed from the training documents, classification can be performed by calculating the posterior probability of each category given the words observed in the test document, and selecting the category with the highest probability.

$$P(c_j|d_i;\hat{\theta}) = \frac{P(c_j|\hat{\theta})P(d_i|c_j;\hat{\theta})}{P(d_i|\hat{\theta})} = \frac{P(c_j|\hat{\theta})\prod_{k=1}^{|d_i|} P(w_{d_{ik}}|c_j;\hat{\theta})}{\sum_{r=1}^{|C|} P(c_r|\hat{\theta})\prod_{k=1}^{|d_i|} P(w_{d_{ik}}|c_r;\hat{\theta})} \tag{4}$$

The first equality is obtained by Bayes rule, while in the second we substitute Equations 1 and 2 for $P(d_i|\hat{\theta})$ and $P(d_i|c_j;\hat{\theta})$. The probability of word w_t given class c_j is expressed by the *maximum likelihood* (ML) estimate:

$$P(w_t|c_j;\hat{\theta}) = \frac{1+\sum_{i=1}^{|D|} N(w_t,d_i)P(c_j|d_i)}{|V|+\sum_{s=1}^{|V|}\sum_{i=1}^{|D|} N(w_s,d_i)P(c_j|d_i)} \tag{5}$$

where $N(w_t,d_i)$ denotes the frequency of word w_t in document d_i. To avoid having probability 0 for previously unseen words, Laplace smoothing is used (1 at the numerator and $|V|$ at the denominator).

5.2 Shrinkage for Hierarchical Categories

Shrinkage is a statistical technique first adopted in a TC context by McCallum et al. [6] to improve the estimation of the probabilistic model parameters when classes are organized hierarchically. Although they use this technique with a Naive Bayes classifier, shrinkage can be used with all parametric classifiers, i.e., classifiers whose model parameters are estimated from a training set. The leaves of the hierarchy often have few scattered training data, while their ancestors comprise more data but less specific than data in the leaves (as their data come from all of their descendants). The idea is to trade specifity for robustness estimating the probability in a node through a linear interpolation of all the ML probabilities along the path from the root to the node in exam. The improved estimate of the probability of word w_t given category c_j is expressed by:

$$P(w_t|c_j;\breve{\theta}_j) = \lambda_j^1\hat{\theta}_{jt}^1 + \lambda_j^2\hat{\theta}_{jt}^2 + \ldots + \lambda_j^k\hat{\theta}_{jt}^k = \sum_{r=0}^{k+1}\lambda_j^r\hat{\theta}_{jt}^r \tag{6}$$

where k is the depth in the tree of category c_j, $\hat{\theta}_{jt}^0 = \hat{\theta}_{jt}$, $\hat{\theta}_{jt}^r$ estimates are obtained according to (5), and λ_j^r are the interpolation weights, with $\sum_{r=0}^{k+1}\lambda_j^r = 1$. Note that (6) assumes the existence of a "virtual" parent $\pi^{k+1}(c_j)$ of the root characterized by the uniform estimate, i.e. such that $\hat{\theta}_{jt}^{k+1} = \frac{1}{|\mathcal{V}|}$ for all $w_t \in \mathcal{V}$; this is done in order to smooth the parameters for those terms that are rare also in the root category (i.e. in the entire training set), and eliminates the need for Laplace smoothing. The λ_j^r weights are determined by applying a variant of the *expectation maximization* (EM) algorithm on a validation set.

6 The JobNet Approach

The key aspect of the automatic coding method in JobNet is the capability of coding properly linguistic patterns which may be very different in the terms they use, but entailing the same competence. Sentences which do not share any keyword may be assigned to the very same competence category, and it is not clear a priori whether the statistic distribution of words may be related to the semantics of coding.

The JobNet questionnaire requires the description of a real or imaginary situation in first person singular and with sentences describing precisely dialogues, thoughts, and emotions (not generic phrases like e.g. "Yes, that's right") the candidate had *during* the situation she's describing. We believe that a hierarchical approach to the automatic coding is better than a flat one in our case, as there are not many training data available, and manual coding is a costly activity. In the JobNet hierarchy internal nodes are trivial, i.e., they just contain training data from all of their descendants, and therefore they may be viewed as a generalization, a gross grain view of the categories below. In Fig. 1 we sketch how shrinkage may be applied to our hierarchy of competencies.

6.1 Scenario

In our competence hierarchical model we have 22 leaf categories plus 4 internal nodes, and hence we have to build a probabilistic classifier for each of them (i.e., a unigram

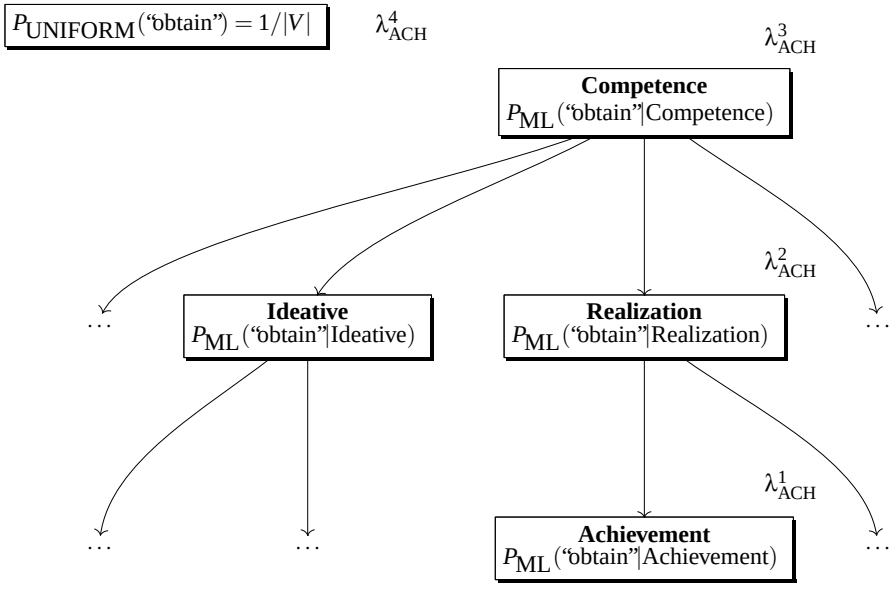

Fig. 1. Shrinkage for a better estimate of the probability of word *"obtain"* given the *"Achievement"* category consists of the linear interpolation of the ML estimates from the leaf to the root, with the addition of a uniform probability

model for each node in terms of statistical language modelling). Our training instances are (fragments of) answers, and the target concepts classifiers have to learn are of the kind "answer denoting an orientation to leadership" or "answer denoting an orientation to teamwork" (we leave out for the moment the internal grades). In order to apply a Bayesian method we have to define how to represent the text, and how to calculate the required prior estimates. In our first setting, text is represented by the bag of words paradigm, where each distinct word in the text represents a feature whose value is given by the number of times the word is observed in the text. No stemming is applied but we can use a stop word list to eliminate the function words. Let us assume a 400 manually coded answers uniformly distributed over the 4 main clusters. For example, consider the 100 documents in the Realization cluster having the distribution in Fig. 2.

Let us instantiate the Naive Bayes Equation to estimate the category of a given answer by simplifying Eq. 4 to:

$$C_{\mathcal{NB}} = argmax_{c_j \in C} P(c_j|\hat{\theta}) \prod_{k=1}^{|d_i|} P(w_{d_{ik}}|c_j;\hat{\theta}) \tag{7}$$

$C_{\mathcal{NB}}$ is the category which maximizes the probability of observing the words occurring in d_i under the hypothesis of mutual independence of the attributes. To estimate the prior probability $P(c_j|\hat{\theta})$ we use the percentages obtained by the training data, dividing

	positive samples	negative samples	percentage
Achievement	40	360	10
Initiative	10	390	2.5
Accuracy	20	380	5
Directivity	30	370	7.5

Fig. 2. Distribution of samples for the *"Realization"* cluster.

the number of instances belonging to category c_j by the total number of instances. For instance, $P(\text{"}Accuracy\text{"}) = .05$.

$P(w_{d_{ik}}|c_j;\hat{\theta})$ in a leaf node is given by Eq. 6, where the $\hat{\theta}^r_{jt}$ estimates are obtained by the combination of the ML probabilities along the path to the node. For each ancestor, data from its child in the path under exam are subtracted in order to avoid counting them twice. Note that we don't need Laplace smoothing as we use a pseudo-root with uniform probability (Eq. 5), i.e., $P(w_{d_{ik}} = 1/|V|)$. To estimate the ML probability $P(w_{d_{ik}}|c_j;\hat{\theta})$ we have $|d_i| * |V| * |C|$ combinations, where $|d_i|$ is the document length, $|V|$ is the dictionary size (about 100.000 entries in the Italian dictionary, much reduced in the dictionary obtained from the training data), and $|C|$ is the number of categories. The number of combinations is reduced to $|V| * |C|$ by assuming that $P(a_l = w_{d_{ik}}|c_j;\hat{\theta}) = P(a_m = w_{d_{ik}}|c_j;\hat{\theta}) \forall j,k,l,m$, i.e., the probability of a word occurrence given its category is independent of its position in the observed text.

7 Conclusion and Future Work

The main contribution of this paper is the description of a new approach to an Automated Survey Coding task by means of Text Categorization probabilistic techniques. Preliminary experiments indicate that this approach performs better than previous approaches, though we need to compare more results on homogeneous datasets, as the nature of input data may affect the final results. An open issue regards the feature selection; we start as a baseline from a bag of words text representation, but we believe that alternative approaches to feature selection (e.g. n-grams or patterns) might be much more effective in our case. Though we describe here a probabilistic TC approach, other TC methods should be investigated as well.

References

1. Melina Alexa and Cornelia Zuell. Text analysis software: Commonalities, differences and limitations: The results of a review. *Quality & Quantity*, (34):299–321, 2000.
2. 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.
3. Hay Group. Web site: http://www.haygroup.com. Last visited on April 8, 2002.

4. 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.
5. Andrew K. McCallum and Kamal Nigam. A comparison of event models for Naive Bayes text classification. In *Proceedings of AAAI/ICML-98 Workshop on Learning for Text Categorization*, pages 41–48, Madison, US, 1998. AAAI Press.
6. Andrew K. McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In Jude W. Shavlik, editor, *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pages 359–367, Madison, US, 1998. Morgan Kaufmann Publishers, San Francisco, US.
7. Tom M. Mitchell. *Machine Learning*. McGraw Hill, New York, US, 1997.
8. Andrew J. Perrin. The CodeRead system: Using natural language processing to automate coding of qualitative data. *Social Science Computer Review*, 19(2):213–220, 2001.
9. Daniel J. Pratt and William Mays. Automatic coding of transcript data for a survey of recent college graduates. In *Proceedings of the section on Survey Methods of the American Statistical Association Annual Meeting*, pages 796–801, 1989.
10. Raymond Raud and Michael Fallig. Automating the coding process with neural networks, 1995.
11. Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
12. Lyle M. Spencer and Signe M. Spencer. *Competence at Work: models for Superior Performance*. John Wiley & Sons, New York, US, 1993.
13. Lyle M. Spencer and Signe M. Spencer. *Competenza nel Lavoro - Modelli per una Performance Superiore*. Franco Angeli, 1995.
14. 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.