An agent that understands job descriptions

Andraž Bežek and Matjaž Gams
Jožef Stefan Institue, Jamova 39, 1000 Ljubljana, Slovenia
Phone: +386 61 1773 900, Fax: +386 61 1251 038
andraz.bezek@ijs.si, matjaz.gams@ijs.si

Keywords: intelligent agents, understanding, intelligent services

Received: July 6, 2000

Abstract: *An important property of intelligent agents is semi-understanding of desired tasks. We have designed an agent module referred to as Professional Description Assistant that is able to identify the corresponding job definition from a job description in textual form. Thus it plays a significant role in the advanced services of our EMA employment agent. By means of some standard and modified learning methods the findings proved to be quite encouraging. The new function aids the user performing some standard tasks, which normally demand human decision and experience, to a great extent.*

1 Introduction

"If software agents are such a promising technology, why can't they do something simple for me, like finding a movie? This comment is proffered as a sort of litmus test for agent capability."[10]

The above question refers to the task in which a certain amount of information is available, e.g. a movie description. The needed inference is the relation between a text description and one of possible choices.

We are basically dealing with a similar problem, only that it refers to the domain of employment. The aim of the task is accordingly to find the appropriate job definition from the description of a job. The problem, for example, might be as follows:

Job description: Construction worker
Job definition: auxiliary manual labour, constructing, heavy machinery.

The major problem we need to overcome is how to create and as well incorporate knowledge into an employment agent to enable such kind of understanding. These kinds of services are too difficult for classical systems to perform, and are rather new in the field of agents. These types of services are in general related to advanced Internet services, which apply at least some kind of knowledge or intelligence.

While the Internet on one hand is becoming more and more sophisticated, simple services on the other hand are only of rare occurence. They evolved and acquired the characteristics of intelligent systems and agents. With the latter we refer to software programs that, with some degree of autonomy, perform operations on behalf of the human user or another program. They help to automate a variety of activities, mostly the time-consuming ones. Software agents differ from "traditional" software in their ability to get personalized and social [3]. Agents are also semi-autonomous and can run continuously. We can distinguish four categories of agent functionality [1]:

- Problem-solving (typical research agents; intelligent agents, expert systems)
- User-centric (interaction with user; "intelligent" filtering, user guidance)
- Control (exclusive in multiagent systems, control services for other agents)
- Translation agents (bridge between systems with different data standards).

Here we shall limit ourselves to user-centric agents. They help users to achieve their goals in a more natural and usually in a more flexible way. Agents are in principle more intelligent, flexible and robust than the classical systems. Another important advantage of agents is their potential to struggle against the information overload[2]. Internet agents filter huge amounts of data, discover data relevant to user and present it in a structured way (e.g. a group with profession, location, salary etc.).

We have implemented a major national employment Internet system six years ago[8]. Recently we developed as well as implemented a couple of agent-related modules including the one presented here that is capable to find the appropriate job profession from a context-free job description.

2 Employment tasks

Online employment databases can be regarded as typical Internet services. They help users to accomplish job-related tasks. Instead of walking to the employment agency users simply surf WWW at home thus saving time, money and human resources. Currently, several hundreds of employment-related sites are available on the Internet [5]. They have attention-catching user interfaces combined with big databases of job seekers and job providers. However, current employment sites are limited to browsing and simple searching through a database. When a large amount of data is presented to user – this especially holds true for extensive employment databases – people find themselves lost in heap of information.
Employment databases have two major types of users: job seekers and job providers.
Job-seekers-related tasks:

- Predisposition: frequently provide job offers
- Browse/search through available jobs
- Enter job-searcher's data and match it with those of job providers
- Alert job searcher about new job offers (by means of an intelligent filter).

Job seekers are looking for an interesting job. As users they need to enter job descriptive words into the search query. The system offers the user corresponding, entry-related data in order that he may browse through. However, this kind of approach is appropriate only for small databases or special jobs. Many pieces of information in the system's reply render searching for the best job too time-consuming. Advanced search is therefore required. An agent can retrieve more descriptive data directly from the user, from his/her user profile, from the history of user actions. User profiles consist of job description, location of work, amount of salary, working conditions, working experience and schooling. With all this information an agent can provide more relevant data. Any additional facts of similar content also help to create information-rich output. Items can either be sorted by similarity score and/or grouped by several categories.

The tasks for job providers are similar to those of job seekers. Instead of searching for job providers the system searches for corresponding persons looking for a job. Thus also the problems regarding searching and the huge amount of available data are the same. The main quality for employment-related agents are therefore better understanding of users intention which help to extract relevant information from a heap of potential data.

The following employment-related sites can be found on WWW:

- **http://careers.altavista.com/**
  Provides simple searching with keywords combined with city- and state-limit criteria.
- **http://www.ajb.dni.us/**
  America's Job Bank contains more than one million available jobs. Job searching is limited with only one job title per search. It allows users to limit the distance of wanted job location by requesting zip code and radius.
- **http://www.hotjobs.com/**
  Hotjobs offers advanced search options. One can limit the search with a keyword, location, or the type of job. Results can be browsed, sorted out alphabetically by industry, location or company name.
- **http://www.occ.com/**
  Apart from the standard keyword search, location and job category selections, @Monster Jobs provides sub-search within the extracted data. One can as well limit the output by the date of job offer.
- **http://www.espan.com/esp/plsql/espan_enter.espan_home**
  Job Options uses typical job searching strategies and also provides the search for the following categories as employers, posting of résumés, notification of new jobs and a set of helpful career-dependant information (articles, surveys, links).

Slovenian job-related sites:

- **http://www.ef.uni-lj.si/jobprovider/**
  A job provider can search among available jobs and possible applicants. Job seekers must specify their profession and the area of activity. Simple and detailed view is available. Job providers can search for résumés and can give more search-narrowing conditions including schooling, the type and location of work as well as other skills.
- **http://www-ai.ijs.si/~ema/**
  EMA, an employment agent, is described in the next section.

3 EMA

EMA is an Employment Agent (http://www-ai.ijs.si/~ema/) developed by Intelligent Systems Department, Jozef Stefan Institute, Slovenia[8]. EMA's basic task is to help those who are looking for an employment or those providing it, and also to provide information on scholarships. It consist of several modules:

- Automatically updated database of available jobs
- Text matching search for available jobs
- Database of job seekers
- Automatic notification of job seekers for new jobs
- English and Slovenian speaking speech agent for browsing results.

The input
The input database of available jobs is automatically updated from the Web pages of the Employment Service of Slovenia (ESS, http://www.ess.gov.si/). EMA's communication agent daily transfers all employment data available on ESS's Web pages and incorporates it into EMA's database. This results in a daily updated database without user's interaction. The communicating agent can in principle parse other employment sites. The process demands adding a new Web address and parsing rules. In this way EMA has become a social agent since it can get data from any employment database on the Internet. Another major source of input data are Internet users who directly enter data.

3.1 Search

The search method matches text over the name of a job or the whole job description. In addition, users can limit their search by geographical areas of available jobs. Results are grouped by job names so that users can search the document for a wanted job.

3.1.1 Job seekers

EMA also handles job-seekers' database. Each job seeker must fill in a form with corresponding data (personal data, education, working experience, wanted job, expected salary). Job providers can search the job-seekers' database and browse it in a simple or more detailed way.

3.1.2 Notification

For registered users EMA can track the available employment information and notify users interested of any relevant new offers. When changes in any database occur, an agent, using interesting strings provided by the user, searches for appropriate jobs and mails them to registered users. In this way users do not have to visit employment web pages on daily basis; instead they are informed when relevant new information appears. In our opinion this feature significantly helps users to save time and computer resources.

3.1.3 Speech agents

EMA incorporates an English and Slovene speech agent. Results are "read" when a user clicks on a specific job on the screen. Microsoft Speech Agent was used for English language while the Slovenian one was entirely developed by the Intelligent Systems Department at Jozef Stefan Institute [6].

4 Profession description assistant

In order to improve EMA we have developed a profession-description auxiliary module. The idea is to help users searching for a job without having to know the exact definition of that job. In everyday life this is quite of frequent occurrence. When users register as job seekers they have to provide a lot of job-related data. Human employment agents find relevant information based on their experience. The whole process is called job matching since the systems tries to match user's data with an appropriate job.

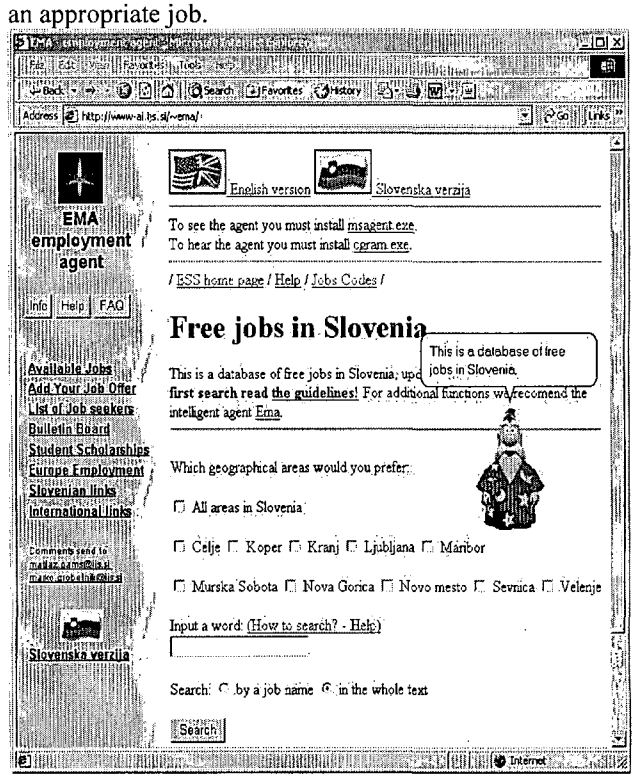


Figure 1: EMA's agent interface in English.

For EMA the main difficulty is how to recognize which job is appropriate for a specific text description. Matching can be done in the following way. All available data are transferred into text for a specific description. This text is matched with a database of available jobs. Jobs that have the best matching scores are displayed to the user. A similar option is also to recognize job descriptions on the basis of previous examples. We applied this approach to develop a system, which is able to suggest appropriate jobs from a job descriptive text.

4.1 The algorithm

Text classification process can be presented in the following steps[7]:
A preprocessing step for determining the values of the features or attributes that will be used for representing the individual documents within a collection. This is essentially the dictionary creation process.
A representation step for mapping each individual document into a training sample using the above

dictionary and associating it with a label that identifies its category.
An induction step for finding patterns that distinguish categories from each other.
An evaluation step for choosing the best solution based on minimizing the classification error or cost.

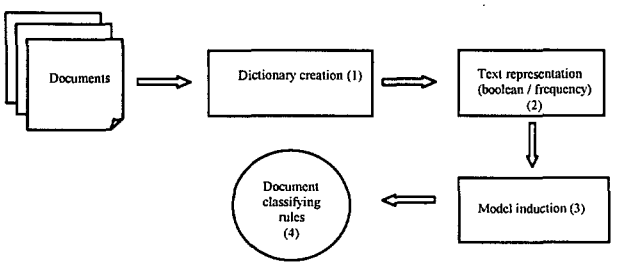


Figure 2: A model for automatic classification based on previous examples.

In Figure 2, the test classification process is represented. Numbers in parentheses in Figure 2 correspond to the above four items. The initial task is to produce a list of attributes from samples of text of labeled documents, i.e. the dictionary. The attributes are single words or word phrases. Given an attribute list, sample cases can be described in terms of the words or phrases found in the documents. Each case consists of the values of the attributes for a single article, where the values could be either Boolean, i.e. indicating whether the attribute appears in the text or not, or numerical, i.e. frequency of occurrence in the text being processed. In addition, each case is labeled to indicate the classification or topic of the article it represents.

4.2 Learning dataset

We collected employment data for 23 weeks. The whole dataset has 15125 records categorized into 590 classes - job names. For further processing all records were cleared of irrelevant attributes. Only a job description and a job name remained in the record. As irrelevant attributes, the location of an employee and working conditions are regarded. Each word in a job description was considered an attribute. There were 5390 different attributes. An average record had 3.5 attributes; the maximum being 18 and the minimum 1. A class, on average, had 25,64 records, the smallest having one and the biggest 651 of them.

A dictionary of all words was created. Each word in the dictionary has its unique identification number. Each record was converted from the text into numeric presentation using word id-s combined with frequency statistics. This conversion was a necessary step in order to achieve reasonable short learning times.

An example of a learning dataset:

```
CAR MECHANIC
CAR MECHANIC - HEAD OF WORKROOM
CONSTRUCTION WORKER
ASSISTANT CONSTRUCTION WORKER
MACHINE MECHANIC
MACHINE LOCKSMITH
```

4.3 Classifying methods

We tested several classifying methods, namely:

- k-nearest neighbors (kNN)
- Naive Bayes method
- the method described here
- Linear combination of methods

kNN

kNN stands for k-nearest neighbor classification, a well-known statistical approach that has been intensively studied[11]. kNN has been applied to text categorization since the early stages of the research [12][17][15]. It is one of the top-performing methods on the benchmark Reuters corpus (the 21450 version, Apte set).

The kNN algorithm is quite simple: given a test document, the system finds k nearest neighbors among the training documents, and uses the categories of the k neighbors to weight the category candidates. The similarity score between test and all neighbor documents is used as a weight for the categories. If several of the k nearest neighbors share a category, the per-neighbor weights of that category are added together, and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. With the process of sorting the scores of candidate categories we get a ranked list for the test document.

The algorithm is as follows:

```
∀ e ∈ E
  similar[e] := similar(e, W_test)
sort(similar[])
score[] := 0
∀ i ∈ 1..K
  c := class[example[i]]
  score[c] := score[c] + similar[i]
sort(score[])
```

Naive Bayes' method

This is one of the most successful algorithms for learning how to classify text documents[8]. Naive Bayes' classifier computes conditional probabilities of the classes given the instance and picks the class with the highest posterior probability. Attributes are assumed to be independent, an assumption that is unlikely to be true, but the algorithm is nonetheless very robust to violations of this assumption. Naive Bayes' formula is

$$P(c \mid V) = P(c) * \prod_{i=1}^{|V|} \frac{P(v_i \mid c)}{P(v_i)}$$

where P(c | V) represents the conditional probability of class c (in our case specific job) in vector of attributes (job description).
Because of

$$P(v_i \mid c) = P(v_i) * \frac{P(c \mid v_i)}{P(c)}$$

we can show that:

$$P(c \mid V) = P(c) * \prod_{i=1}^{|V|} \frac{P(c \mid v_i)}{P(c)}$$

The classifying algorithm therefore was as follows:

```
∀ c ∈ C
  score[c] := P(c)
  ∀ w ∈ Wtest
    score[c]:= score[c]*P(c|w)/P(c)
sort(score[])
```

For each class we compute P(c | V). A class with the biggest conditional probability is the most probable one. If we want to have a list of most probable classes, we can propose a couple of most probable classes. This is useful since users are often looking for several relevant jobs. Another reason is that the learning process is never 100 % accurate and the suggestion of only one post might be misleading. Therefore it is up to user to choose the corresponding job name from a couple of good candidates. In this way we give the final decision to the user. The agent is thus a true auxiliary – it only helps users to achieve their task by sensibly proposing a reasonable amount of relevant information.

Our method

We developed a statistical method based on conditional probability of attributes. The idea is that a conditional probability of attribute correctly evaluates the class given the attribute. P(w | C)=0, meaning that the word w does not belong to class C which is therefore not probable. A similar case holds true for P(w | C)=1. The attribute is contained only in class C which makes it the most probable. The conditional probabilities can be computed by using study examples.

Because of:

$$\forall w \notin c : P(w \mid c) = 0$$

we can take into account only classes which contain compared attributes.

$\forall w \in W_{test}$:

$$\sum_{c \in C} \sum_{w \in c} P(w \mid c) = 1$$

Therefore:

$$\sum_{w \in Wtest} \sum_{c \in C} \sum_{w \in c} P(w \mid c) = |W_{test}|$$

Because of the above rule we have to normalize class score by dividing it with the number of attributes in the test example. The result is probability distribution of classes.

$$P(c \mid V) = \frac{\sum_{i=1}^{|V|} P(v_i \mid c)}{|V|}$$

The algorithm being:

```
Score[] := 0
∀ w ∈ Wtest
  ∀ c : (w ∈ c)
    Score[c] := Score[c]+P(w|c)
∀ c ∈ C
  Score[c] := Score[c]/|Wtest|
sort(score[])
```

4.4 Linear combination of methods

The last tested method was linear combination of two classifying methods. The concept is based on the assumption that linear combination would have a superposition effect. Combination of two different methods can result in better classifying accuracy due to elimination of each method's bias.

score[c] := α*scoreX[c] + (1-α)*scoreY[c]

Our assumption can be shown with the following example:

method X score	method Y score	50%X + 50%Y
A 30%	D 35%	B 27,5%
B 25%	B 30%	A 25%
C 10 %	A 20%	D 17%
...	...	C 5%

Table 1: An example of the effect of linear combination.

By means of both methods class B scored high but not the best; with linear combination, however, it ranked the highest, which is correct.

Time complexity is a sum of time complexity for each method.

$$O_{lin.combination} = O_{methodX} + O_{methodY}$$

In this case we chose the Naive Bayes' and our own method. The application of both methods proved to be good, and consequently we made an assumption that two good methods can produce even better results.

$$\text{score[c]} := \alpha * \text{scoreNaiveBayes[c]} + (1-\alpha) * \text{scoreOurMethod [c]}$$

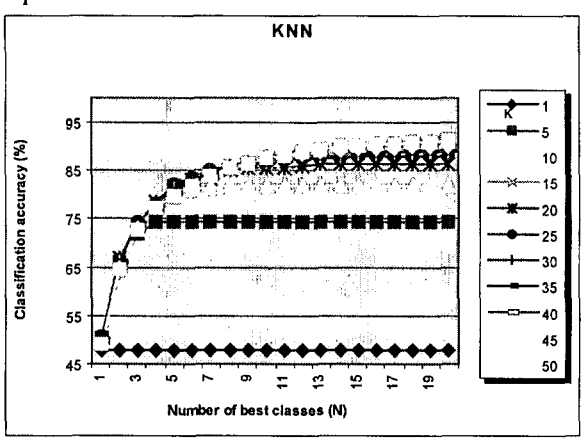
The parameter α adjusts the combination of methods. If α=0% only our method is applicable, whereas the value α=100% indicates that only Naive Bayes' method can be used.

4.5 Results

The ten fold cross-validation was used to test classifiers. Test examples represented 5% out of 15,125 in this case 756. These results show the average value of all tests. From that a conclusion can be drawn that accuracy for the single best class was 63.53%. This is obviously not accurate enough to be used for wider practice. However, when adding more possibilities one obtains significantly better results: 87.54% for 5, 92.54% for 10 and 95.05% for 20 best classes. Therefore, with a reasonable amount of suggested choices, the desired job definition was practically always included in the proposed list. This is presented in Figure 2.

Tests of the linear combination method are presented in Figure 3. In general, better accuracy is obtained in all cases. Although the accuracy gain might sometimes be small it cannot be regarded as harmful. To achieve significant improvements the tuning of the parameter α is necessary.

The results in Figures 2 and 3 are quite encouraging. One must note that study examples contain many errors such as noise, syntax errors, different abbreviations and inconsistent classifications. From a single text description even the best human experts shall give you several job definitions. All these properties of learning dataset influence the accuracy of classification. On this basis we can conclude that Profession Description Assistant has achieved the sufficient quality to be used in practice.



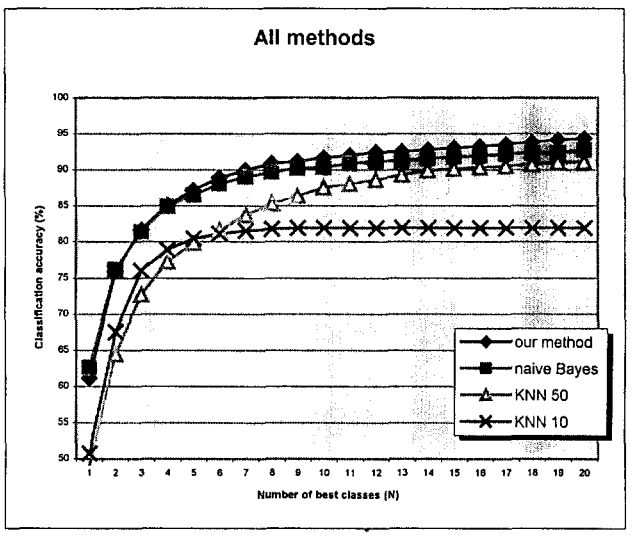


Figure 2: Testing methods on 15,125 records show that useful results are obtained with already 5-10 suggestions for the best job description.

5 Discussion

Profession Description Agent is a new agent module in the EMA employment agent. It shows that agents can help users with advanced functions based on domain-dependant knowledge. Introduction of new concepts enhances agent's functionality and usability. In this way the human-computer interaction is improved.

While modern systems are becoming more and more complex, thus overloading users with enormous amount of information, software designers must find ways to simplify interaction. Our implementation shows a possible solution to this problem.

Although intelligent agents are far from perfect they give additional functionality to existent systems. We think that future intelligent agents will incorporate the domain dependant knowledge. They will be able to execute advanced tasks and therefore be of greater importance to users.

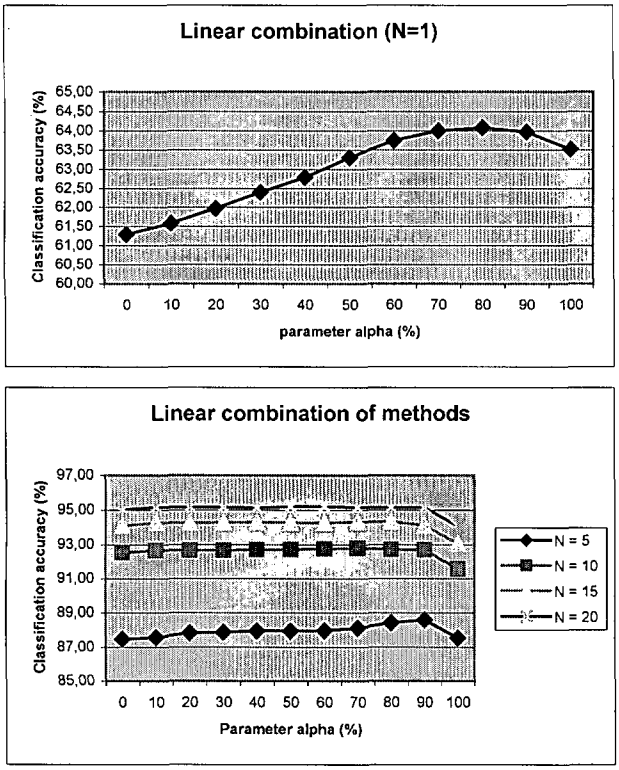


Figure 3: Linear combination practically always achieves better classification accuracy than any single method.

6 References

[1] James Hendler, Making Sense out of Agents, IEEE Intelligent Systems, pp. 32-37, March/April 1999.

[2] Matjaž Gams, Information Society and the Intelligent Systems Generation, Informatica 23, 4, pp. 449-454, 1999.

[3] Robert H. Gutmann, Alexandros G. Moukas, Pattie Maes, Agents as Mediators in Electronic Commerce, Electronic Markets, VOl. 8, No. 1, pp. 22-27, January 1998.

[4] Andraž Bežek, Automatic classification of job descriptions, Graduation thesis, University of Ljubljana, Slovenia, September 1999.

[5] List of job huntings sites, http://www-ai.ijs.si/~ema/EMA_Links-e.html

[6] Tomaž Šef, Aleš Dobnikar, Matjaž. Gams, Improvements in Slovene Text-to-Speech Synthesis, Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP '98), pp. 2027-2030, 1998, Sydney

[7] Chidanad Apté, Fred Damerau and Sholom M. Weiss, Towards language independent automated learning of text categorization models. Proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval, pp. 23 – 30, 1994, Dublin, Ireland,

[8] Tom Mitchell, Machine Learning, McGraw Hill, 1997

[9] Matjaž Gams, Pavle Golob, Aram Karalič, Matija Drobnič, Marko Grobelnik, Jože Glazer, J., Pirher, Tone Furlan, Erik Vrenko, Radovan Križman, EMA - zaposlovalni agent, 1998, http://www-ai.ijs.si/~ema/EMA_Info-e.html

[10] Dana Moore, Ed Greengrass, Design Considerations for Agent Systems That Glean the Internet, IEEE Intelligent Systems, pp. 76-81, March/April 1999.

[11] Yiming Yang, Xin Liu, A re-examination of text categorization methods, Proceedings on the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 42 – 49, 1999

[12] Masand, B., Linoff, G. in Waltz, D. (1992) Classifying news stories using memory based reasoning. Proceedings of the Fifteenth Annual International ACM SIGIR conference on Research and development in information retrieval, pp. 59 – 65

[13] Apté, C., Damerau, F. Weiss in Sholom M. (1994) Towards language independent automated learning of text categorization models. Proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval, pp. 23 – 30

[14] Cohen, William W. in Singer, Y. (1999) Context-sensitive learning methods for text categorization. ACM Trans. Inf. Syst. 17, 2, pp. 141 – 173

[15] Tokunaga, T. in Iwayama, M. (1995) Cluster-based text categorization a comparison of category search strategies. Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, strani 273 – 278

[16] J. van Rijsbergen, C. (1979) Information retrieval. Butterworths, http://sherlock.berkeley.edu/IS205/IR_CJVR/Preface.html

[17] Yang, Y. (1994) Expert network: effective and efficient learning from human decisions in text categorization and retrieval. Proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval, pp. 13-22