Filtering of Large Numbers of Unstructured Text Documents by the Developed Tool TEA

Jan Žižka[1] and Aleš Bourek[2]

[1] Department of Information Technologies, Faculty of Informatics,
Masaryk University in Brno, Botanická 68a, 602 00 Brno, Czech Republic
E-mail: zizka@informatics.muni.cz
[2] Department of Biophysics, Faculty of Medicine,
Masaryk University in Brno, Joštova 10, 662 43 Brno, Czech Republic
E-mail: bourek@med.muni.cz

Abstract. This paper describes a text-document-filtering software tool TEA (TExt Analyzer), which was originally developed for physicians to support selections of large numbers of unstructured medical text documents obtained from available Internet services. TEA learns interesting and relevant documents for *individual* users basically by the naïve Bayes algorithm. Moreover, TEA provides a number of additional functions that improve its classification accuracy. The learning process of TEA is based on a set of labeled positive and negative examples of text documents, which obtain their labels from users interested in documents of certain, usually very specific topics. Experiments and real uses of TEA by physicians have demonstrated that a classification accuracy—separating the documents between two classes (interesting and uninteresting)—can be expected from 70% up to 97%, typically 85% and better.

1 Introduction

Users, like physicians, of modern data-processing technologies mostly expect obtaining very specific information and knowledge from extensive resources provided, for example, by the Internet. Unfortunately, in too many cases the resources provide a lot of very raw data retrieved using only a set of key-words. Such a situation is still unsatisfactory because it is often impossible to manually separate hundreds or thousands of documents within a reasonable time. On the other hand, the users can employ computers and special software for the processing of rather raw data to obtain the requested information. One possibility is to use computers for learning which text documents are relevant for a specific user if this user can provide his or her parameters describing the area of interest. The text-filtering tool TEA (TExt Analyzer), described in the following sections, supports its users in separating unstructured text documents into two classes, *interesting* and *uninteresting*. TEA disposes of the basic functions *learning* and *classification*; moreover, it contains many additional functions which improve the final result—text documents that are mostly relevant and interesting for the users' specific needs. This approach is very important also in medicine because

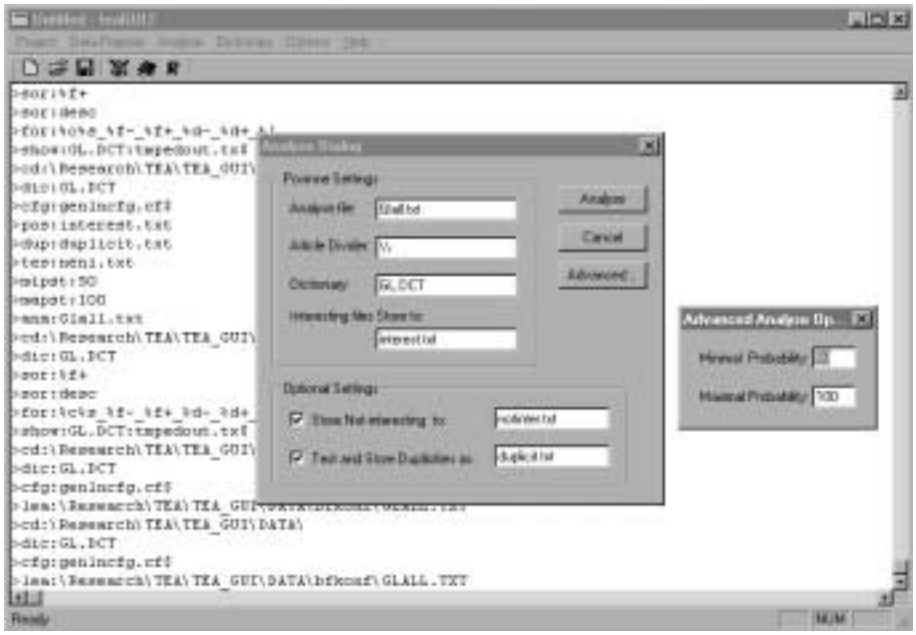

Fig. 1. The GUI (graphical user interface) of the text analyzer TEA—an example of a user's new text-project establishment.

there is very often a huge number of documents, however, physicians usually need only a fragment covering their actual requirements. In addition, physicians as well as other users also deeply need to manage a lot of unstructured text materials which include great numbers of mutual links. Preprocessing of huge numbers of text documents, which are accessible via different electronic archives distributed in the Internet, is generally inescapable, e.g., because of more than 10,000 medical documents published per day. On the other hand, text-document filtering can reveal that within a certain time interval, publication numbers of certain topics rapidly decrease, or specific new terms start to arise. Potential users of this kind of filtering software would also require *individual* adaptation of such tools (unlike the common WWW searching tools and browsers) for their specific needs. The described tool TEA has been developed and implemented for the operating system Windows, using the graphical user interface as illustrated in Fig 1.

2 The essential functions—learning and classification

The text-document classification tool TEA learns to recognize a user's interesting and uninteresting documents by the *naïve Bayes algorithm*. For its learning, TEA needs user-labeled sets of training examples, i.e., training text documents. Each example has a label—provided by a user before starting the learning process—as

a mark of belonging either in a group of interesting or uninteresting documents. In other words, the learning process uses positive (interesting, relevant) and negative (uninteresting, irrelevant) examples of text documents, according to an individual user's requirements. Different users can naturally mark the same text documents differently if they need it.

TEA manipulates with the text documents as with sequences of words (or character strings), where these words are separated by the standard white space and punctuation marks and signs. In addition, all upper-case letters are converted into their lower-case equivalents. The separated individual words are then used as *distinct words* and are stored—together with their frequencies in all the training documents—into the dictionary.

To avoid a very long time of computation, the naïve Bayes algorithm assumes that positions of words are independent. Despite the fact that this assumption is not quite correct, results with (and not only) text documents are practically acceptable, which is described, e.g., in Lewis (1998). Thus, the frequencies of the distinct words are used for computing degrees to which a text document belongs to the interesting (+) or uninteresting (−) class. Let $Degree(+/-)$ stands for a degree of belonging to the (+) or (−) class, N_{all} stands for the total number of documents in both classes, $N_{(+/-)}$ stands for the number of documents in the (+) or (−) class, and $P(w_i|+/-)$ stands for the relative frequency of a word w_i in the (+) or (−) class (actually, *a posteriori probability*). Then, for n being the number of words in a classified text document, the belonging degree is given by the following equation (the degrees for (+) and (−) are computed separately):

$$Degree(+/-) = \frac{N_{(+/-)}}{N_{all}} \prod_{i=1}^{n} P(w_i|+/-)$$

After computing the degree, a classified document obtains the class (+) or (−) that corresponds to a greater degree value.

3 Classification results of TEA

The implemented system TEA was tested using the cross-validation method and—at the present time—is used by physicians in the area of medical text documents obtained in huge numbers by various Internet tools and browsers. Among resources of text documents were, for example, on-line medical databases provided by the National Library of Medicine, many fulltext databases accessible to academics and professional health-care providers (Biological Abstracts, Zoological Record—database of BIOSIS comp., DL ACM—digital library of ACM, EIFL Direct—most important fulltext databases of EBSCO, LINK—scientific journals of Springer-Verlag, Web of Science—journals bibliography/citation DB, MEDLINE, etc.). During the processing of text documents, users of the initial TEA tool suggested and needed additional functions, which gradually extended possibilities of TEA and contributed to higher accuracies of classifications. The main goal of the users has always been to eliminate many uninteresting and irrelevant

documents obtained from the Internet using mainly browsers and key-words because in too many cases there were so many documents that users did not have time enough to read them and to select only what was necessary.

The original tests, which used real data from the MEDLINE source, are published in Žižka et al. (2000) and (2002). The document set contained 701 interesting and 1,109 uninteresting documents with 12,631 different word forms. Classification accuracies were between 73% – 94%, depending on different approaches: lower accuracy could be expected for sets of very similar documents (like, e.g., a very narrow medical branch), while higher accuracies were obtained for sets with more different document contains (e.g., documents from a medical branch with more aspects). Later, after the initial tests of the TEA's core, the experiments used much more text documents from different resources, and users could apply new additional functions as excluding certain words, setting up minimum and maximum occurrence of words, and so like. With this extended additional support, the TEA analyzer is now able to filter the retrieved material consistently with the average accuracy usually better than 85%. The experiments with large and different data sets provided accuracies from 70% up to 97%.

The other group of extensive testing was performed using publicly accessible Internet newsgroups from 20 topics (*alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.windows.x, misc.forsale, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.misc,* and *talk.religion.misc*). The number of unstructured text documents was around 20,000 articles with 1,000 articles per class. The task of the TEA analyzer was based on the classification of text documents using 20 classes (20 more or less different newsgroup topics). The number of distinct words in all the original documents was 119,717. In this case, the classification accuracy was between 86% and 89%. The TEA's functionality for these 20 classes was almost the same as for the purely medical set of text documents mentioned above.

According to the extensive experiments, the classification accuracy depends on several points. Higher numbers of training documents typically provide better accuracy. In addition, the number of training positive and negative documents should be as close as possible, e.g., 40%:60%, or even better, approximately 50%:50%. It is naturally not possible to obtain always the ideal data in practice, however, users very often prefer the help of the tool even in less advantageous situations, mainly if the number of the classified documents is very high (e.g., hundreds or thousands). Another important point is the possibility to exclude common words (typically the first 100 most common words for English) as well as certain words defined by a user. Such an exclusion improved the classification accuracies by 0.5% to 5%. On the other hand, the experiments also have revealed that the naïve Bayes algorithm becomes less effective for sets of data where positive and negative examples are very similar—however, in this case it is also often rather difficult for humans to unambiguously decide whether a text document belongs to the positive or negative class. Therefore, users can ex-

pect higher classification accuracies for large data sets containing more different documents within a certain area of interest or within different areas.

4 Additional functions supported by TEA

The current version of the TEA tool includes the naïve Bayes algorithm as its basic function for the learning and classification purposes. As the main result of learning, there is a dictionary containing words from documents together with computed probabilities of each word occurrence both in positive and negative text-document classes. In addition of learning, TEA enables its users to print the dictionary, to modify its contents in various ways, and to use the potentially modified dictionary subsequently as a basis for analyses of new documents. Another TEA's important function is the possibility to set up different restrictions for the learning and analyzing data to improve the classification results.

The additional functions supported by the TEA program can be divided into several groups:

- the system configuration,
- the program control functions,
- the users' projects (scripts as plans of actions) for running and controlling,
- the support of works with files and folders (directories),
- the setting up input and output files,
- the learning and analysis functions, and
- the work with words and dictionaries.

A brief, not complete description of the functions is provided in the following subsections. All the mentioned functions have more details and parameters as well as error messages, if necessary. The following description covers only the basic properties.

4.1 The system configuration

The program TEA is designed and implemented to enable the work in an arbitrary language (provided that a certain language support was created). To select a communication language, for example, for error messages, there is a command *language:file*, where *file* defines a file containing messages. Moreover, the command *charset:file* can be used for changing a character set for a specified language (without this definition, the English alphabet characters are only used). The command *settings* displays the set up configuration. Finally, *free:configuration* cancels one of the configuration parameters; *free* without a specification simply sets up the program initial, default configuration.

4.2 The program control functions

To finish the program run, the *exit* command is used; *help* supports displaying *Help* of the program. The command *print:message* displays a requisite *message* while *nprint:message* displays *message* with a new line.

4.3 The users' projects for running and controlling

The users' projects are actually scripts supporting repeated types of works, i.e., repeated sequences of commands. TEA uses *interpret:file* for running a project *file*. Without *file*, an interactive command line is started, so a user can combine both scripts and interactive commands. If a user wants to display script commands, he or she can turn on/off the activation by *say:on-off*.

4.4 The support of works with files and folders

The system TEA allows displaying of a current folder by *ls:mask*, for example, *ls:*.txt* displays all files having the *txt* extension. The command *cd:folder* changes a current *folder*, and *more:file* displays a requested *file*.

4.5 The setting up input and output files

A dictionary is set up by *dictionary:file*; if the file does not exist, it is created. The functions of learning or analysis use a file defined by *input:file*. TEA also needs a configuration system containing string definitions of beginnings of interesting texts (the positive marking string starts with the character +, e.g., $+abc$) as well as uninteresting texts (the negative marking string starts with the character −, e.g., $-abc$). In addition, if interesting texts within a file have as their marks numbers, digits can be replaced by the character #, e.g., interesting text units having eight-digit numbers are represented in the configuration file as *+########*. If a user wants to add his or her comments into the text units, it is possible to define a beginning string of comments by *estring*, e.g., if a comment should start with *XXX*, then *eXXX*. The command *posfile:file* sets up an output file for interesting texts. To avoid a relatively frequent problem with duplicate or multiple text units (typically obtained from the Internet), a user can employ the command *test-file:file*; in addition, *dupfile:file* defines an output file for duplicates or multiplicities.

4.6 The learning and analysis functions

The system TEA starts learning by *input:file* and *learn* or simply *learn:file*. After its learning, TEA can classify new texts by *analyse:file* or just *analyse* provided that a file was defined earlier. Interesting documents obtain probabilities higher than 50%, up to 100%. However, if a user wants to change the default value 50%, the command *mipst:number* is available, e.g., from 75% up to 100%: *mipst:75*. On the other hand, if there are text units having suspicious, too high values, e.g., 98%, a user can exclude such units by *mapst:number*, e.g., *mapst:98* in our case.

4.7 The work with words and dictionaries

Users very often need to influence importance of certain words defined by the human approach because machines are still not so intelligent. During a normal work, TEA stores words—found in the testing data—with certain information. Each word is classified as, e.g., interesting, uninteresting, etc., using a scale *0, +, −, =, 1, 2, 3, 4, 5, 6, 7, 8, 9, n*. Generally, a word has the implicit classification =; if TEA during its analysis of a new text document (not included in the training set) finds an unknown word, it assigns *n* (as *unknown*) to it. In addition, a user can assign any item from the scale to a word; *0* means a word is not interesting at all, + means a positive key-word, − a negative key-word, = a word without any restriction. The other scale items can be used according to the user's meaning, if necessary. For example, in this way, the user can exclude common words (like *a, an, the, this, of, at, in, ...*, etc.) to improve TEA's classification. In addition, there is also an often used function that dynamically eliminates uninteresting words. Users can activate this function by *minpos:number*, *minsum:number*, and/or *masum:number*, which influences a needed ratio of words in positive and negative documents; therefore, depending of defined ratios, some words can be ignored. For example, *minsum:5* means that all words occurring less than five times in the testing data are ignored. Similarly, *masum:100* ignores words occurring more than 100 times (to avoid too frequent words). On the other hand, *minpos:0.4* says that a word occurring, for example, 45 times in interesting documents and 55 times in uninteresting documents will be ignored because $45/(45 + 55) = 0.45 > 0.4$.

The command *info* provides information about a current dictionary, and *show:dictionary:file* displays (or stores, if the *file* parameter is used) a dictionary as a text file. To retrieve a text file as a dictionary, the command *compile:file* is used. Sorting a dictionary is a task of *sort:asc* or *sort:desc* in the ascending or descending order, respectively. The command *cp:file* creates a copy of a dictionary; *file* is a name of the copy. Adding explicitly a new word is supported by *add:word*, where the classification is supposed to be =, otherwise the command *current:classification* can change the standard classification. If two dictionaries should be joined, the command *join:file* enables this function, where the *file* dictionary joins a current one.

5 Conclusions

As the popularity of the World Wide Web and other Internet services continues to increase, there is a growing need to develop tools and techniques that would help improve their overall usefulness. The all the time growing taking advantage of the Internet services among physicians indicates that this (and surely not only this) kind of users need efficient personal tools to enable PCs to boost user-creative thinking in areas requiring manipulation of vast amounts of textual information. Such a tendency has clearly been demonstrated in practice: the case of Czech Standards of Efficient Medical Care, Bourek, Suchý et al. (2000), and in tracking infertility treatment trends, Bourek, Žižka et al. (2000).

Experiments with a lot of real medical text-data verified the application of the naïve Bayes algorithm to be a useful method supporting results obtained from the Internet, especially when it was inevitable to process large numbers of more or less different documents selected only by key-words. The additional functions, which enable users to modify the document classification, e.g., excluding certain words or documents, increase the classification accuracy and decrease substantially the number of irrelevant, uninteresting text documents. However, this modifications and their results mostly depend on specific users' needs and on particular types of text documents—what is advantageous for a certain user could be disadvantageous for another one, even if they would work with the same set of documents. Therefore, the additional functions support individual settings of searching parameters while the naïve Bayes algorithm is generally responsible for the filtering itself. This also means that the next advantage of using a tool as TEA is the possibility to exploit the same set of documents for different purposes, even for more users, depending on a specific parameterization of the TEA system during its run.

References

1. Bourek, A., Suchý, M., and Svoboda, P. (2000): Standards of Efficient Medical Care (SEMC). In: *Proceedings of the* 7^{th} *International Conference on System Science in Health Care, "Sustainable structure for better health."* Lyon, ISSHC, 436-439.
2. Bourek, A., Žižka, J., Ventruba, P., and Frey, L. (2000): The Use of the Internet for Monitoring Trends in Assisted Reproduction and Reproductive Medicine. *Gynekolog, 5*, 220-223 (in Czech).
3. Lewis, D. D. (1998): Naïve (Bayes) at Forty: The Independence Assumption in Information Retrieval. In: *Proceedings of the* 10^{th} *European Conference on Machine Learning ECML'98*. Springer Verlag, Berlin Heidelberg New York, 4-15.
4. Žižka, J. and Bourek, A. (1998): Learning and Classifying Medical Text Documents Using the Naïve Bayes algorithm. In: *Proceedings of the First Workshop TSD'98 on Text, Speech, and Dialogue*. Masaryk University Press, Brno, Czech Republic, 147-150.
5. Žižka, J., Bourek, A., and Frey, L. (2000): TEA: A Text Analysis Tool for the Intelligent Text Document Filtering. In: *Text, Speech, and Dialogue*. Springer Verlag, Berlin Heidelberg New York, LNCS 1902, 151-156.
6. Žižka, J., Bourek, A. (2002): Automated Selection of Interesting Medical Text Documents. In: *Computational Linguistics and Intelligent Text Processing*. Springer Verlag, Berlin Heidelberg New York, LNCS 2276, 402-404.