Text Categorization

Fabrizio Sebastiani
Dipartimento di Matematica Pura e Applicata
Università di Padova
35131 Padova, Italy

Abstract

Text categorization (also known as text classification, or topic spotting) is the task of automatically sorting a set of documents into categories from a predefined set. This task has several applications, including automated indexing of scientific articles according to predefined thesauri of technical terms, filing patents into patent directories, selective dissemination of information to information consumers, automated population of hierarchical catalogues of Web resources, spam filtering, identification of document genre, authorship attribution, survey coding, and even automated essay grading. Automated text classification is attractive because it frees organizations from the need of manually organizing document bases, which can be too expensive, or simply not feasible given the time constraints of the application or the number of documents involved. The accuracy of modern text classification systems rivals that of trained human professionals, thanks to a combination of information retrieval (IR) technology and machine learning (ML) technology. This chapter will outline the fundamental traits of the technologies involved, of the applications that can feasibly be tackled through text classification, and of the tools and resources that are available to the researcher and developer wishing to take up these technologies for deploying real-world applications.

1 Introduction

Text categorization (TC – also known as *text classification*, or *topic spotting*) is the task of automatically sorting a set of documents into *categories* (or *classes*, or *topics*) from a predefined set. This task, that falls at the crossroads of information retrieval (IR) and machine learning (ML), has witnessed a booming interest in the last ten years from researchers and developers alike.

For IR researchers, this interest is one particular aspect of a general movement towards leveraging user data for taming the inherent subjectivity of the IR task[1], i.e. taming the fact that it is the user, and only the user, who can say whether a given item of information is relevant to a query issued to a Web search engine, or to a private folder in which documents should be filed according to content. Wherever there are predefined classes, documents manually classified by the user are often available; as a consequence, this latter data can be exploited for automatically learning the (extensional) meaning that the user attributes to the classes, thereby reaching levels of classification accuracy that would be unthinkable if this data were unavailable.

For ML researchers, this interest is due to the fact that IR applications prove an excellent and challenging benchmark for their own techniques and methodologies, since IR applications usually feature extremely high-dimensional feature spaces (see Section 2.1) and provide data by the truckload. In the last five years, this has resulted in more and more ML researchers adopting TC as one of their benchmark applications of choice, which means that cutting-edge ML techniques are being imported into TC with minimal delay from their original invention.

For application developers, this interest is mainly due to the enormously increased need to handle larger and larger quantities of documents, a need emphasized by increased connectivity and availability of document bases of all types at all levels in the information chain. But this interest is also due to the fact that TC techniques have reached accuracy levels that rival the performance of trained professionals, and these accuracy levels can be achieved with high levels of efficiency on standard hw/sw resources. This means that more and more organizations are automating all their activities that can be cast as TC tasks.

This chapter thus purports to take a closer look at TC, by describing the standard methodology through which a TC system (henceforth: *classifier*) is built, and by reviewing techniques, applications, tools, and resources, pertaining to research and development in TC.

The structure of this chapter is as follows. In Section 2 we will give a basic picture of how an automated TC system is built and tested. This will involve a discussion of the technology (mostly borrowed from IR) needed for building the internal representations of the documents (Section 2.1), of the technology (borrowed from ML) for automatically building a classifier from a "training set" of preclassified documents (Section 2.2), and of the methodologies for evaluating the quality of the classifiers one has built (Section 2.3). Section 3 will discuss some actual technologies for performing all of this, concentrating on representative, state-of-the-art examples. In Section 4 we will instead discuss the main domains to which TC is applied nowadays.

Section 5 concludes, discussing possible avenues of further research and development.

[1]This movement spans several IR tasks including *text mining*, *document filtering and routing*, *text clustering*, *text summarization*, *information extraction*, plus other tasks in which the basic technologies from these latter are used, including *question answering* and *topic detection and tracking*. See e.g. the recent editions of the ACM SIGIR conference for representative examples of research in these fields.

2 The basic picture

TC may be formalized as 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, according to a supposedly authoritative expert) 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 categories and $\mathcal{D}$ is a (possibly infinite) set 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: no additional knowledge (of a procedural or declarative nature) of their meaning is usually available, and it is often the case that no metadata (such as e.g. publication date, document type, publication source) is available either. In these cases, classification must be accomplished only on the basis of knowledge extracted from the documents themselves. Since this case is the most general, this is the usual focus of TC research, and will also constitute the focus of this chapter[2]. However, when in a given application either external knowledge or metadata is available, heuristic techniques of any nature may be adopted in order to leverage on these data, either in combination or in isolation from the IR and ML techniques we will discuss here.

TC is a *subjective* task: when two experts (human or artificial) decide whether or not 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 selling his shares in the Texas Bulls baseball team could be filed under Politics, or under Finance, or under Sport, or under any combination of the three, or even under neither, depending on the subjective judgment of the expert. Because of this, the meaning of a category is subjective, and the ML techniques described in Section 2.2, rather than attempting to produce a "gold standard" of dubious existence, aim to reproduce this very subjectivity by examining its manifestations, i.e. the documents that the expert has manually classified under $\mathcal{C}$. The kind of learning that these ML techniques engage in is usually called *supervised* learning, as it is supervised, or facilitated, by the knowledge of the preclassified data.

Depending on the application, TC may be either a *single-label* task (i.e. exactly one $c_i \in \mathcal{C}$ must be assigned to each $d_j \in \mathcal{D}$), or a *multi-label* task (i.e. any number $0 \leq n_j \leq |\mathcal{C}|$ of categories may be assigned to a document $d_j \in \mathcal{D}$)[3]. A special case of single-label TC is *binary* TC, in which, given a category c_i, each $d_j \in \mathcal{D}$ must be assigned either to c_i or to its complement $\overline{c}_i$. A *binary classifier* for c_i is then a function $\hat{\Phi}_i : \mathcal{D} \rightarrow \{T, F\}$ that approximates the unknown target function $\Phi_i : \mathcal{D} \rightarrow \{T, F\}$.

A problem of multi-label TC under $\mathcal{C} = \{c_1, \ldots, c_{|\mathcal{C}|}\}$ is usually tackled as $|\mathcal{C}|$ independent binary classification problems under $\{c_i, \overline{c}_i\}$, for $i = 1, \ldots, |\mathcal{C}|$. In

[2] A further reason why TC research rarely tackles the case of additionally available external knowledge is that these sources of knowledge may vary widely in type and format, thereby making each such application a case in its own from which any lesson learned can hardly be exported to different application contexts.

[3] Somewhat confusingly, in the ML field the single-label case is dubbed the *multiclass* case.

this case, a classifier for $\mathcal{C}$ is thus actually composed of $|\mathcal{C}|$ binary classifiers.

From the ML standpoint, learning a binary classifier (and hence a multi-label classifier) is usually a simpler problem than learning a single-label classifier. As a consequence, while all classes of supervised ML techniques (among which the ones discussed in Section 2.2) deal with the binary classification problem since their very invention, for some classes of techniques (e.g. support vector machines - see Section 2.2) a satisfactory solution of the single-class problem is still the object of active investigation [1]. In this chapter, unless otherwise noted, we will always implicitly refer to the binary case.

Aside from actual operational use, which we will not discuss, we can roughly distinguish three different phases in the life cycle of a TC system, which have traditionally been tackled in isolation of each other (i.e. a solution to one problem not being influenced by the solutions given to the other two): 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 [2].

2.1 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 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 k documents (in TC: in at least k *training* documents), and $0 \leq w_{kj} \leq 1$ quantifies the importance of t_k in characterizing the semantics of d_j. Typical values of k are between 1 and 5.

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 document (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 [3]). A popular choice is to add to the set of words or stems a set of *phrases*, i.e. longer (and semantically more significant) language units extracted from the text by shallow parsing and/or statistical techniques [4]. Concerning (ii), term weights may be binary-valued (i.e. $w_{kj} \in \{0, 1\}$) or real-valued (i.e. $0 \leq w_{kj} \leq 1$), depending on whether the classifier-building algorithm and the classifiers, once they have been built, require binary input or not. When weights are binary, these simply indicate presence/absence of the term in the document. When weights are non-binary, they are computed by either statistical or probabilistic techniques (see e.g. [5]), the former being the most common option. One popular class of statistical term weighting functions is $tf * idf$ (see e.g. [6]), 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 fre-*

quency 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. Alternatively, dimensionality reduction may take the form of *feature extraction*: a set of "artificial" terms is generated from the original term set in such a way that the newly generated terms are both fewer and stochastically more independent from each other than the original ones used to be.

2.2 Classifier learning

A text classifier for c_i is automatically generated by a general inductive process (the *learner*) which, by observing the characteristics of a set of documents preclassified under c_i or $\overline{c}_i$, gleans the characteristics that a new unseen document should have in order to belong to c_i. In order to build classifiers for $\mathcal{C}$, one thus needs a set 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 is the set of documents observing which the learner builds the classifier. The validation set is the set of documents on which the engineer fine-tunes 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 the set on which the effectiveness of the classifier is finally evaluated. 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 classes.

Different learners have been applied in the TC literature. 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 [7].

It is worthwhile to notice that in several applications, the fact that a method implements a real-valued function can be profitably used, in which case determining thresholds is not needed. For instance, in applications in which the quality of

the classification is of critical importance (e.g. in filing patents into patent directories), post-editing of the classifier output by a human professional is often necessary. In this case, having the documents ranked in terms of their estimated relevance to the category may be useful, since the human editor can scan the ranked list starting from the documents deemed most appropriate for the category, and stop when desired.

2.3 Classifier evaluation

Training efficiency (i.e. average time required to build a classifier $\hat{\Phi}_i$ from a given corpus Ω), as well as *classification efficiency* (i.e. average time required to classify a document by means of $\hat{\Phi}_i$), and *effectiveness* (i.e. average correctness of $\hat{\Phi}_i$'s classification behaviour) are all legitimate measures of success for a learner.

In TC *research*, effectiveness is usually considered the most important criterion, since it is the most reliable one when it comes to experimentally comparing different learners or different TC methodologies, given that efficiency depends on too volatile parameters (e.g. different sw/hw platforms). In TC *applications*, however, all three parameters are important, and one must carefully look for a tradeoff among them, depending on the application constraints. For instance, in applications involving interaction with the user, a classifier with low classification efficiency is unsuitable. On the contrary, in multi-label TC applications involving thousands of categories, a classifier with low training efficiency also might be inappropriate (since many thousands of classifiers need to be learnt). Anyway, effectiveness tends to be the primary criterion in operational contexts too, since in most applications an ineffective although efficient classifier will be hardly useful, or will involve too much post-editing work on the part of human professionals, which might defy the purpose of using an automated system.

In single-label TC, effectiveness is usually measured by *accuracy*, i.e. the percentage of correct classification decisions (*error* is the converse of accuracy, i.e. $E = 1 - A$). However, in binary (henceforth: in multi-label) TC, accuracy is not an adequate measure. The reason for this is that in binary TC applications the two categories c_i and $\overline{c}_i$ are usually *unbalanced*, i.e. one contains far more members than the other[4]. In this case, building a classifier that has high accuracy is trivial, since the *trivial rejector*, i.e. the classifier that trivially assigns all documents to the most heavily populated category (i.e. c_i), has indeed very high accuracy; and there are no applications in which one is interested in such a classifier[5].

As a result, in binary TC it is often the case that effectiveness wrt category c_i is measured by a combination of *precision wrt* c_i (π_i), the percentage of documents deemed to belong to c_i that in fact belong to it, and *recall wrt* c_i (ρ_i), the percentage of documents belonging to c_i that are in fact deemed to belong to it.

[4] For example, the number of Web pages that should be filed under the category NuclearWaste-Disposal is orders of magnitude smaller than the number of pages that should not.

[5] One further consequence of adopting accuracy as the effectiveness measure when classes are unbalanced is that in the phase of parameter tuning on a validation set (see Section 2.2), there will be a tendency to choose parameter values that make the classifier behave very much like the trivial rejector.

Table 1: Averaging precision and recall across different categories; TP_i, TN_i, FP_i and FN_i refer to the sets of true positives, true negatives, false positives, and false negatives wrt c_i, respectively.

	Microaveraging	**Macroaveraging**
Precision (π)	$\pi = \frac{\sum_{i=1}^{\|\mathcal{C}\|} TP_i}{\sum_{i=1}^{\|\mathcal{C}\|} TP_i + FP_i}$	$\pi = \frac{\sum_{i=1}^{\|\mathcal{C}\|} \pi_i}{\|\mathcal{C}\|} = \frac{\sum_{i=1}^{\|\mathcal{C}\|} \frac{TP_i}{TP_i + FP_i}}{\|\mathcal{C}\|}$
Recall (ρ)	$\rho = \frac{\sum_{i=1}^{\|\mathcal{C}\|} TP_i}{\sum_{i=1}^{\|\mathcal{C}\|} TP_i + FN_i}$	$\rho = \frac{\sum_{i=1}^{\|\mathcal{C}\|} \rho_i}{\|\mathcal{C}\|} = \frac{\sum_{i=1}^{\|\mathcal{C}\|} \frac{TP_i}{TP_i + FN_i}}{\|\mathcal{C}\|}$

In multi-label TC, when effectiveness is computed for several categories the precision and recall results for individual categories must be averaged in some way; here, one may opt for *microaveraging* ("categories count proportionally to the number of their positive training examples") or for *macroaveraging* ("all categories count the same"), depending on the application desiderata (see Table 1). The former rewards classifiers that behave well on heavily populated ("frequent") categories, while classifiers that perform well also on infrequent categories are emphasized by the latter. It is often the case that in TC research macroaveraging is the method of choice, since producing classifiers that perform well also on infrequent categories is the most challenging problem of TC.

Since most classifiers can be arbitrarily tuned to emphasize recall at the expense of precision (and viceversa), only combinations of the two are significant. The most popular way to combine the two is the function $F_\beta = \frac{(\beta^2+1)\pi\rho}{\beta^2\pi+\rho}$, for some value $0 \leq \beta \leq \infty$; usually, β is taken to be equal to 1, which means that the F_β function becomes $F_1 = \frac{2\pi\rho}{\pi+\rho}$, i.e. the harmonic mean of precision and recall. Note that for the trivial rejector, $\pi = 1$ and $\rho = 0$, so $F_\beta = 0$ for any value of β (symmetrically, for the *trivial acceptor* it is true that $\pi = 0$, $\rho = 1$, and $F_\beta = 0$ for any value of β).

Finally, it should be noted that some applications of TC require cost-based issues to be brought to bear on how effectiveness is computed, thus inducing a *utility-theoretic* notion of effectiveness. For instance, in spam filtering (i.e. a binary TC task in which e-mail messages must be classified in the category Spam or its complement NonSpam), precision is more important than recall, since filing a legitimate message under Spam is a more serious error (i.e. it bears more cost) than filing a junk message under NonSpam. One possible way of taking this into account is using the F_β measure with $\beta \neq 1$; using values of $0 \leq \beta < 1$ corresponds to paying more attention to precision than to recall, while by using values of $0 < \beta < \infty$ one emphasizes recall at the expense of precision.

3 Techniques

We now discuss some of the actual techniques for dealing with the problems of document indexing and classifier learning, discussed in the previous section. Presenting a complete review of them is outside the scope of this chapter; as a consequence, we will only hint at the various choices that are available to the designer, and will enter into some detail only for a few representative cases.

3.1 Document indexing techniques

The TC community has not displayed much creativity in devising document weighting techniques specific to TC. In fact, most of the works reported in the TC literature so far use the standard document weighting techniques, either of a statistical or of a probabilistic nature, which are used in all other subfields of IR, including text search (e.g. $tfidf$ or BM25 – see [5]). The only exception to this we know is [8], where the idf component in $tfidf$ is replaced by a function learnt from training data, and aimed at assessing how good a term is at discriminating categories from each other.

Also in TC, as in other subfields of IR, the use of larger indexing units, such as frequently adjacent pairs (aka "bigrams") or syntactically determined phrases, has not shown systematic patterns of improvement [4, 9], which means that terms are usually made to coincide with single words, stemmed or not.

Dimensionality reduction is tackled either by feature *selection* techniques, such as mutual information (aka information gain) [10], chi square [11], or gain ratio [8], or by feature *extraction* techniques, such as latent semantic indexing [12, 13] or term clustering [9]. Recent work on term extraction methods has focused on methods specific to TC (or rather: specific to problems in which training data exist), i.e. on supervised term clustering techniques [14, 15, 16], which have shown better performance than the previously mentioned unsupervised techniques.

3.2 Classifier learning techniques

The number of classes of classifier learning techniques that have been used in TC is bewildering. These include at the very least 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 (which include boosting methods). Rather than attempting to say even a few words about each of them, we will introduce in some detail two of them, namely support vector machines and boosting. The reasons for this choice are twofold. First, these are the two methods that have unquestionably shown the best performance in comparative TC experiments so far. Second, these are the newest methods in the classifier learning arena, and the ones with the strongest justifications from computational learning theory.

3.2.1 Support vector machines

The *support vector machine* (SVM) method has been introduced in TC by Joachims [17, 18] and subsequently used in several other TC works [19, 20, 21]. In geometrical terms, it may be seen as the attempt to find, among all the surfaces $\sigma_1, \sigma_2, \ldots$ in $|\mathcal{T}|$-dimensional space that separate the positive from the negative training examples (*decision surfaces*), the σ_i that separates the positives from the negatives by the widest 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 usually conceived for binary classification problems [22], and only recently have they been adapted to multiclass classification [1].

As argued by Joachims [17], one advantage that SVMs offer for TC is that dimensionality reduction is usually not needed, as SVMs tend to be fairly robust to overfitting and can scale up to considerable dimensionalities. Recent extensive experiments by Brank and colleagues [23] also indicate that feature selection tends to be detrimental to the performance of SVMs.

Recently, efficient algorithms for SVM learning have also been discovered; as a consequence, the use of SVMs for high-dimensional problems such as TC is no more prohibitive from the point of view of computational cost.

There are currently several freely available packages for SVM learning. The best known in the binary TC camp is the SVMLIGHT package[6], while one that has been extended to also deal with the general single-label classification problem is BSVM[7].

3.2.2 Boosting

Classifier *committees* (aka *ensembles*) are based on the idea that k different classifiers $\Phi_1, \ldots, \Phi_k$ may be better than one if their individual judgments are appropriately combined. In the *boosting* method [24, 25, 26, 27] the k classifiers $\Phi_1, \ldots, \Phi_k$ are obtained by the same learning method (here called the *weak learner*), and are trained not in a conceptually parallel and independent way, but sequentially. In this way, in training classifier Φ_t one may take into account how classifiers $\Phi_1, \ldots, \Phi_{t-1}$ perform on the training examples, and concentrate on getting right those examples on which $\Phi_1, \ldots, \Phi_{t-1}$ have performed worst.

Specifically, for learning classifier Φ_t each $\langle d_j, c_i \rangle$ pair is given an "importance weight" h_{ij}^t (where h_{ij}^1 is set to be equal for all $\langle d_j, c_i \rangle$ pairs), which represents how hard to get a correct decision for this pair was for classifiers $\Phi_1, \ldots, \Phi_{t-1}$. These weights are exploited in learning Φ_t, which will be specially tuned to correctly solve the pairs with higher weight. Classifier Φ_t is then applied to the training documents, and as a result weights h_{ij}^t are updated to h_{ij}^{t+1}; in this update operation, pairs correctly classified by Φ_t will have their weight decreased, while

[6] SVMLIGHT is available from `http://svmlight.joachims.org/`

[7] BSVM is available from `http://www.csie.ntu.edu.tw/~cjlin/bsvm/`

pairs misclassified by Φ_t will have their weight increased. After all the k classifiers have been built, a weighted linear combination rule is applied to yield the final committee.

Boosting has proven a powerful intuition, and the BOOSTEXTER system[8] has reached one of the highest levels of effectiveness so far reported in the literature.

4 Applications

As mentioned in Section 1, the applications of TC are manifold. Common traits among all of them are:

- The need to handle and organize documents in which the textual component is either the unique, or dominant, or simplest to interpret, component.
- The need to handle and organize *large* quantities of such documents, i.e. large enough that their manual organization into classes is either too expensive or not feasible within the time constraints imposed by the application.
- The fact that the set of categories is known in advance, and its variation over time is small[9].

Applications may instead vary along several dimensions:

- The nature of the documents; i.e. documents may be structured texts (such as e.g. scientific articles), newswire stories, classified ads, image captions, e-mail messages, transcripts of spoken texts, hypertexts, or other.
  If the documents are hypertextual, rather than textual, very different techniques may be used, since links provide a rich source of information on which classifier learning activity can leverage. Techniques exploiting this intuition in a TC context have been presented in [28, 29, 30, 31] and experimentally compared in [32].
- The structure of the classification scheme, i.e. whether this is flat or hierarchical. Hierarchical classification schemes may in turn be tree-shaped, or allow for multiple inheritance (i.e. be DAG-shaped). Again, the hierarchical structure of the classification scheme may allow radically more efficient, and also more effective, classification algorithms, which can take advantage of early subtree pruning [33, 21, 34], improved selection of negative examples [35], or improved estimation of word occurrence statistics in leaf nodes [36, 37, 38, 39].
- The nature of the task, i.e. whether the task is single-label or multi-label.

Hereafter, we briefly review some important applications of TC. Note that the borders between the different classes of applications listed here are fuzzy, and some of these may be considered special cases of others.

[8] BOOSTEXTER is available from `http://www.cs.princeton.edu/~schapire/boostexter.html`

[9] In practical applications, the set of categories does change from time to time. For instance, in indexing computer science scientific articles under the ACM classification scheme, one needs to consider that this scheme is revised every five to ten years, to reflect changes in the CS discipline. This means that training documents need to be created for newly introduced categories, and that training documents may have to be removed for categories whose meaning has evolved.

4.1 Automatic indexing for Boolean information retrieval systems

The application that has stimulated the research in TC from its very beginning, back in the '60s, up until the '80s, is that of automatic indexing of scientific articles by means of a controlled dictionary, such as the ACM Classification Scheme, where the categories are the entries of the controlled dictionary. This is typically a multi-label task, since several index terms are usually assigned to each document.

Automatic indexing with controlled dictionaries is closely related to the *automated metadata generation* task. In digital libraries one is usually interested in tagging documents by metadata that describe them under a variety of aspects (e.g. creation date, document type or format, availability, etc.). Some of these metadata are *thematic*, i.e. their role is to describe the semantics of the document by means of bibliographic codes, keywords or keyphrases. The generation of these metadata may thus be viewed as a problem of document indexing with controlled dictionary, and thus tackled by means of TC techniques. In the case of Web documents, metadata describing them will be needed for the Semantic Web to become a reality, and TC techniques applied to Web data may be envisaged as contributing part of the solution to the huge problem of generating the metadata needed by Semantic Web resources.

4.2 Document organization

Indexing with a controlled vocabulary is an instance of the general problem of document base organization. In general, many other issues pertaining to document organization and filing, be it for purposes of personal organization or structuring of a corporate document base, may be addressed by TC techniques. For instance, at the offices of a newspaper, it might be necessary to classify all past articles in order to ease future retrieval in the case of new events related to the ones described by the past articles. Possible categories might be HomeNews, International, Money, Lifestyles, Fashion, but also finer-grained ones such as ThePittAnistonMarriage.

Another possible application in the same range is the organization of patents into categories for making later access easier, and of patent applications for allowing patent officers to discover possible prior work on the same topic [40]. This application, as all applications having to do with patent data, introduces specific problems, since the description of the allegedly novel technique, which is written by the patent applicant, may intentionally use non standard vocabulary in order to create the impression that the technique is indeed novel. This use of non standard vocabulary may depress the performance of a text classifier, since the assumption that underlies practically all TC work is that training documents and test documents are drawn from the same word distribution.

4.3 Text filtering

Text filtering is the activity of classifying a stream of incoming documents dispatched in an asynchronous way by an information producer to an information consumer. Typical cases of filtering systems are e-mail filters [41] (in which case the producer is actually a multiplicity of producers), newsfeed filters [42], or filters of unsuitable content [43]. A filtering system should block the delivery of the documents the consumer is likely not interested in. Filtering is a case of binary TC, since it involves the classification of incoming documents in two disjoint categories, the relevant and the irrelevant. Additionally, a filtering system may also further classify the documents deemed relevant to the consumer into thematic categories of interest to the user. A filtering system may be installed at the producer end, in which case it must route the documents to the interested consumers only, or at the consumer end, in which case it must block the delivery of documents deemed uninteresting to the consumer.

In information science document filtering has a tradition dating back to the '60s, when, addressed by systems of various degrees of automation and dealing with the multi-consumer case discussed above, it was called *selective dissemination of information* or *current awareness*. The explosion in the availability of digital information has boosted the importance of such systems, which are nowadays being used in diverse contexts such as the creation of personalized Web newspapers, junk e-mail blocking, and Usenet news selection.

4.4 Hierarchical categorization of Web pages

TC has recently aroused a lot of interest also for its possible application to automatically classifying Web pages, or sites, under the hierarchical catalogues hosted by popular Internet portals. When Web documents are catalogued in this way, rather than issuing a query to a general-purpose Web search engine a searcher may find it easier to first navigate in the hierarchy of categories and then restrict a search to a particular category of interest. Classifying Web pages automatically has obvious advantages, since the manual categorization of a large enough subset of the Web is unfeasible. With respect to previously discussed TC applications, automatic Web page categorization has two essential peculiarities (both discussed in Section 4), namely the hypertextual nature of the documents, and the typically hierarchical structure of the category set.

4.5 Word sense disambiguation

Word sense disambiguation (WSD) is the activity of finding, given the occurrence in a text of an ambiguous (i.e. polysemous or homonymous) word, the sense of this particular word occurrence. For instance, `bank` may have (at least) two different senses in English, as in `the Bank of England` (a financial institution) or `the bank of river Thames` (a hydraulic engineering artifact). It is thus a WSD task to decide which of the above senses the occurrence of `bank`

in `Last week I borrowed some money from the bank` has. WSD may be seen as a (single-label) TC task (see e.g. [44]) once, given a word w, we view the contexts of occurrence of w as documents and the senses of w as categories.

4.6 Automated survey coding

Survey coding is the task of assigning a symbolic code from a predefined set of such codes to the answer that a person has 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, 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.

Survey coding is a difficult task, since the code that should be attributed to a respondent based on the answer given is a matter of subjective judgment, and thus requires expertise. The problem can be formulated as a single-label TC problem [45], where the answers play the role of the documents, and the codes that are applicable to the answers returned to a given question play the role of the categories (different questions thus correspond to different TC problems).

4.7 Automated authorship attribution and genre classification

Authorship attribution is the task of determining the author of a text of disputed or unknown paternity, choosing from a predefined set of candidate authors [46, 47, 48]. Authorship attribution has several applications, ranging from the literary (e.g. discovering who is the author of a recently discovered sonnet) to the forensic (e.g. identifying the sender of an anonymous letter, or checking the authenticity of a letter allegedly authored by a given person). Authorship attribution can also be seen as a single-label TC task, with possible authors playing the role of the categories. This is an application in which a TC system typically cannot be taken at face value; usually, its result contributes an "opinion" on who the possible author might be, but the final decision has to be taken by a human professional. As a result, a TC system that ranks the candidate authors in terms of their probability of being the true author, would be useful (see Section 2.2).

The intuitions that must be brought to bear in these applications are orthogonal to those that are at play in topic-based classification, since an author normally writes about multiple topics. Because of this, it is unlikely that topic-based features can be good at discriminating among authors. Rather, stylistic features are the most appropriate choice; for instance, vocabulary richness (i.e. ratio between number of distinct words and total number of words), average word length, average sentence length, are important, in the sense that it is these features that tend "to give an author away".

Genre classification is also an applicative context which bears remarkable similarities to authorship attribution. There are applicative contexts in which it is desirable to classify documents by genre, rather than by topic [49, 50, 51, 52]. For instance, it might be desirable to classify articles about scientific subjects into one of the two categories PopularScience and HardScience, in order to decide whether they are suitable for publication into popular science magazines or not; likewise, distinguishing between ProductReviews and Advertisements might be useful for several applications. In genre classification too, topic-dependent words are not good separating features, and specialized features need to be devised, which are often similar to the ones used for authorship attribution applications.

4.8 Spam filtering

Filtering *spam* (i.e. unsolicited bulk e-mail) is a task of increased applicative interest that lies at the crossroads between filtering and genre classification. In fact, it has the dynamical character of other filtering applications, such as e-mail filtering, and it cuts across different topics, as genre classification. Several attempts, some of them quite successful, have been made at applying standard text classification techniques to spam filtering, for applications involving either personal mail [53, 19, 54] or mailing lists [55]. However, operational spam filters must rely not only on standard ML techniques, but also on manually selected features. In fact, similarly to the case of genre classification or authorship attribution, it is the stylistic (i.e. topic-neutral) features that are important, rather than the topic-based ones. In fact, spam deals with a multiplicity of topics (from miraculous money making schemes to Viagra pills), and cues indicative of topics can hardly be effective unless they are supplemented with other topic-neutral ones. On the contrary, a human eye may immediately recognize a spam message from visual cues, such as e.g. the amount of all-caps words in the subject line or in the text of the message, the number of exclamation marks in the subject line, an unknown sender with an unknown Web e-mail address (e.g. a `yourfriend@yahoo.com`), or even the peculiar formatting of the message body. Representing these visual cues (as well as taking into accout other standard phrases such as "Make money fast!") as features is important to the effectiveness of an operational spam filter.

One further problem that makes spam filtering challenging is the frequent unavailability of negative training messages. A software maker wishing to customize its spam filter for a particular client needs training examples; while positive ones (i.e. spam messages) are not hard to collect in large quantities, negative ones (i.e. legitimate messages) are difficult to find, because of privacy issues, since a company dealing with industrially sensitive data will not disclose samples of their own incoming legitimate messages even to someone who is going to use these messages for improving a service to them. In this case, ML methods that can do without negative examples need to be used.

4.9 Other applications

The applications described above are just the major among the ones TC has been used for. Here, we only briefly hint at a few other ones.

Myers and colleagues [56], and Schapire and Singer [25] have attacked speech categorization by means of a combination of speech recognition and TC, in the context of a phone call routing application. Sable and Hatzivassiloglou classify instead images through the classification of their textual captions [57]. Larkey [58] instead uses TC to tackle automated essay grading, where the different grades that can be attributed to an essay play the role of categories. In a question answering application, Li and Roth [59] classify questions according to question type; this allows a question answering system to focus on locating the right type of information for the right type of question, thus improving the effectiveness of the overall system.

5 Conclusion

Text categorization has evolved, from the neglected research niche it used to be until the late '80s, into a fully blossomed research field which has delivered efficient, effective, and overall workable solutions that have been used in tackling a wide variety of real-world application domains. Key to this success have been (i) the ever-increasing involvement of the machine learning community in text categorization, which has lately resulted in the use of the very latest machine learning technology within text categorization applications, and (ii) the availability of standard benchmarks (such as Reuters-21578 and OHSUMED), which has encouraged research by providing a setting in which different research efforts could be compared to each other, and in which the best methods and algorithms could stand out.

Currently, text categorization research is pointing in several interesting directions. One of them is the attempt at finding better representations for text; while the bag of words model is still the unsurpassed text representation model, researchers have not abandoned the belief that a text must be something more than a mere collection of tokens, and that the quest for models more sophisticated than the bag of words model is still worth pursuing [60].

A further direction is investigating the scalability properties of text classification systems, i.e. understanding whether the systems that have proven the best in terms of effectiveness alone stand up to the challenge of dealing with very large numbers of categories (e.g. in the tens of thousands) [61].

Last but not least are the attempts at solving the labeling bottleneck, i.e. at coming to terms with the fact that labeling examples for training a text classifier when labeled examples do not previously exist, is expensive. As a result, there is increasing attention in text categorization by semi-supervised machine learning methods, i.e. by methods that can bootstrap off a small set of labeled examples and also leverage on unlabeled examples [62].

References

[1] Crammer, K. & Singer, Y., On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, **2**, pp. 265–292, 2001.

[2] Sebastiani, F., Machine learning in automated text categorization. *ACM Computing Surveys*, **34(1)**, pp. 1–47, 2002.

[3] Frakes, W.B., Stemming algorithms. *Information Retrieval: Data Structures and Algorithms*, eds. W.B. Frakes & R. Baeza-Yates, Prentice Hall: Englewood Cliffs, US, pp. 131–160, 1992.

[4] Caropreso, M.F., Matwin, S. & Sebastiani, F., A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text Databases and Document Management: Theory and Practice*, ed. A.G. Chin, Idea Group Publishing: Hershey, US, pp. 78–102, 2001.

[5] Zobel, J. & Moffat, A., Exploring the similarity space. *SIGIR Forum*, **32(1)**, pp. 18–34, 1998.

[6] Salton, G. & Buckley, C., Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, **24(5)**, pp. 513–523, 1988. Also reprinted in [63], pp. 323–328.

[7] Yang, Y., A study on thresholding strategies for text categorization. *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, eds. W.B. Croft, D.J. Harper, D.H. Kraft & J. Zobel, ACM Press, New York, US: New Orleans, US, pp. 137–145, 2001.

[8] Debole, F. & Sebastiani, F., Supervised term weighting for automated text categorization. *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, ACM Press, New York, US: Melbourne, US, pp. 784–788, 2003. An extended version appears as [64].

[9] Lewis, D.D., An evaluation of phrasal and clustered representations on a text categorization task. *Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval*, eds. N.J. Belkin, P. Ingwersen & A.M. Pejtersen, ACM Press, New York, US: Kobenhavn, DK, pp. 37–50, 1992.

[10] Lewis, D.D. & Ringuette, M., A comparison of two learning algorithms for text categorization. *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, US, pp. 81–93, 1994.

[11] Yang, Y. & Pedersen, J.O., A comparative study on feature selection in text categorization. *Proceedings of ICML-97, 14th International Conference on Machine Learning*, ed. D.H. Fisher, Morgan Kaufmann Publishers, San Francisco, US: Nashville, US, pp. 412–420, 1997.

[12] Wiener, E.D., Pedersen, J.O. & Weigend, A.S., A neural network approach to topic spotting. *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, US, pp. 317–332, 1995.

[13] Schütze, H., Hull, D.A. & Pedersen, J.O., A comparison of classifiers and

document representations for the routing problem. *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*, eds. E.A. Fox, P. Ingwersen & R. Fidel, ACM Press, New York, US: Seattle, US, pp. 229–237, 1995.

[14] Baker, L.D. & McCallum, A.K., Distributional clustering of words for text classification. *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, eds. W.B. Croft, A. Moffat, C.J. van Rijsbergen, R. Wilkinson & J. Zobel, ACM Press, New York, US: Melbourne, AU, pp. 96–103, 1998.

[15] Bekkerman, R., El-Yaniv, R., Tishby, N. & Winter, Y., On feature distributional clustering for text categorization. *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, eds. W.B. Croft, D.J. Harper, D.H. Kraft & J. Zobel, ACM Press, New York, US: New Orleans, US, pp. 146–153, 2001.

[16] Slonim, N. & Tishby, N., The power of word clusters for text classification. *Proceedings of ECIR-01, 23rd European Colloquium on Information Retrieval Research*, Darmstadt, DE, 2001.

[17] Joachims, T., Text categorization with support vector machines: learning with many relevant features. *Proceedings of ECML-98, 10th European Conference on Machine Learning*, eds. C. Nédellec & C. Rouveirol, Springer Verlag, Heidelberg, DE: Chemnitz, DE, pp. 137–142, 1998. Published in the "Lecture Notes in Computer Science" series, number 1398.

[18] Joachims, T., Transductive inference for text classification using support vector machines. *Proceedings of ICML-99, 16th International Conference on Machine Learning*, eds. I. Bratko & S. Dzeroski, Morgan Kaufmann Publishers, San Francisco, US: Bled, SL, pp. 200–209, 1999.

[19] Drucker, H., Vapnik, V. & Wu, D., Support vector machines for spam categorization. *IEEE Transactions on Neural Networks*, **10(5)**, pp. 1048–1054, 1999.

[20] Dumais, S.T., Platt, J., Heckerman, D. & Sahami, M., Inductive learning algorithms and representations for text categorization. *Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management*, eds. G. Gardarin, J.C. French, N. Pissinou, K. Makki & L. Bouganim, ACM Press, New York, US: Bethesda, US, pp. 148–155, 1998.

[21] Dumais, S.T. & Chen, H., Hierarchical classification of Web content. *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, eds. N.J. Belkin, P. Ingwersen & M.K. Leong, ACM Press, New York, US: Athens, GR, pp. 256–263, 2000.

[22] Vapnik, V.N., *The nature of statistical learning theory*. Springer Verlag: Heidelberg, DE, 1995.

[23] Brank, J., Grobelnik, M., Milić-Frayling, N. & Mladenić, D., Interaction of feature selection methods and linear classification models. *Proceedings of the ICML-02 Workshop on Text Learning*, Sydney, AU, 2002.

[24] Schapire, R.E., Singer, Y. & Singhal, A., Boosting and Rocchio applied to text filtering. *Proceedings of SIGIR-98, 21st ACM International Conference*

on Research and Development in Information Retrieval, eds. W.B. Croft, A. Moffat, C.J. van Rijsbergen, R. Wilkinson & J. Zobel, ACM Press, New York, US: Melbourne, AU, pp. 215–223, 1998.

[25] Schapire, R.E. & Singer, Y., BOOSTEXTER: a boosting-based system for text categorization. *Machine Learning*, **39(2/3)**, pp. 135–168, 2000.

[26] Sebastiani, F., Sperduti, A. & Valdambrini, N., An improved boosting algorithm and its application to automated text categorization. *Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management*, eds. A. Agah, J. Callan & E. Rundensteiner, ACM Press, New York, US: McLean, US, pp. 78–85, 2000.

[27] Nardiello, P., Sebastiani, F. & Sperduti, A., Discretizing continuous attributes in AdaBoost for text categorization. *Proceedings of ECIR-03, 25th European Conference on Information Retrieval*, ed. F. Sebastiani, Springer Verlag: Pisa, IT, pp. 320–334, 2003.

[28] Chakrabarti, S., Dom, B.E. & Indyk, P., Enhanced hypertext categorization using hyperlinks. *Proceedings of SIGMOD-98, ACM International Conference on Management of Data*, eds. L.M. Haas & A. Tiwary, ACM Press, New York, US: Seattle, US, pp. 307–318, 1998.

[29] Oh, H.J., Myaeng, S.H. & Lee, M.H., A practical hypertext categorization method using links and incrementally available class information. *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, eds. N.J. Belkin, P. Ingwersen & M.K. Leong, ACM Press, New York, US: Athens, GR, pp. 264–271, 2000.

[30] Slattery, S. & Craven, M., Discovering test set regularities in relational domains. *Proceedings of ICML-00, 17th International Conference on Machine Learning*, ed. P. Langley, Morgan Kaufmann Publishers, San Francisco, US: Stanford, US, pp. 895–902, 2000.

[31] Getoor, L., Segal, E., Taskar, B. & Koller, D., Probabilistic models of text and link structure for hypertext classification. *Proceedings of the IJCAI-01 Workshop on Text Learning: Beyond Supervision*, Seattle, US, pp. 24–29, 2001.

[32] Yang, Y., Slattery, S. & Ghani, R., A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, **18(2/3)**, pp. 219–241, 2002. Special Issue on Automated Text Categorization.

[33] Chakrabarti, S., Dom, B.E., Agrawal, R. & Raghavan, P., Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. *Journal of Very Large Data Bases*, **7(3)**, pp. 163–178, 1998.

[34] Koller, D. & Sahami, M., Hierarchically classifying documents using very few words. *Proceedings of ICML-97, 14th International Conference on Machine Learning*, ed. D.H. Fisher, Morgan Kaufmann Publishers, San Francisco, US: Nashville, US, pp. 170–178, 1997.

[35] Ng, H.T., Goh, W.B. & Low, K.L., Feature selection, perceptron learning, and a usability case study for text categorization. *Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information*

Retrieval, eds. N.J. Belkin, A.D. Narasimhalu & P. Willett, ACM Press, New York, US: Philadelphia, US, pp. 67–73, 1997.

[36] Gaussier, É., Goutte, C., Popat, K. & Chen, F., A hierarchical model for clustering and categorising documents. *Proceedings of ECIR-02, 24th European Colloquium on Information Retrieval Research*, eds. F. Crestani, M. Girolami & C.J. van Rijsbergen, Springer Verlag, Heidelberg, DE: Glasgow, UK, pp. 229–247, 2002. Published in the "Lecture Notes in Computer Science" series, number 2291.

[37] McCallum, A.K., Rosenfeld, R., Mitchell, T.M. & Ng, A.Y., Improving text classification by shrinkage in a hierarchy of classes. *Proceedings of ICML-98, 15th International Conference on Machine Learning*, ed. J.W. Shavlik, Morgan Kaufmann Publishers, San Francisco, US: Madison, US, pp. 359–367, 1998.

[38] Toutanova, K., Chen, F., Popat, K. & Hofmann, T., Text classification in a hierarchical mixture model for small training sets. *Proceedings of CIKM-01, 10th ACM International Conference on Information and Knowledge Management*, eds. H. Paques, L. Liu & D. Grossman, ACM Press, New York, US: Atlanta, US, pp. 105–113, 2001.

[39] Vinokourov, A. & Girolami, M., A probabilistic framework for the hierarchic organisation and classification of document collections. *Journal of Intelligent Information Systems*, **18(2/3)**, pp. 153–172, 2002. Special Issue on Automated Text Categorization.

[40] Larkey, L.S., A patent search and classification system. *Proceedings of DL-99, 4th ACM Conference on Digital Libraries*, eds. E.A. Fox & N. Rowe, ACM Press, New York, US: Berkeley, US, pp. 179–187, 1999.

[41] Weiss, S.M., Apté, C., Damerau, F.J., Johnson, D.E., Oles, F.J., Goetz, T. & Hampp, T., Maximizing text-mining performance. *IEEE Intelligent Systems*, **14(4)**, pp. 63–69, 1999.

[42] Amati, G., D'Aloisi, D., Giannini, V. & Ubaldini, F., A framework for filtering news and managing distributed data. *Journal of Universal Computer Science*, **3(8)**, pp. 1007–1021, 1997.

[43] Chandrinos, K.V., Androutsopoulos, I., Paliouras, G. & Spyropoulos, C.D., Automatic Web rating: Filtering obscene content on the Web. *Proceedings of ECDL-00, 4th European Conference on Research and Advanced Technology for Digital Libraries*, eds. J.L. Borbinha & T. Baker, Springer Verlag, Heidelberg, DE: Lisbon, PT, pp. 403–406, 2000. Published in the "Lecture Notes in Computer Science" series, number 1923.

[44] Escudero, G., Màrquez, L. & Rigau, G., Boosting applied to word sense disambiguation. *Proceedings of ECML-00, 11th European Conference on Machine Learning*, eds. R.L.D. Mántaras & E. Plaza, Springer Verlag, Heidelberg, DE: Barcelona, ES, pp. 129–141, 2000. Published in the "Lecture Notes in Computer Science" series, number 1810.

[45] Giorgetti, D. & Sebastiani, F., Automating survey coding by multiclass text categorization techniques. *Journal of the American Society for Information Science and Technology*, **54(12)**, pp. 1269–1277, 2003.

[46] Vel, O.Y.D., Anderson, A., Corney, M. & Mohay, G.M., Mining email content for author identification forensics. *SIGMOD Record*, **30(4)**, pp. 55–64, 2001.

[47] Forsyth, R.S., New directions in text categorization. *Causal models and intelligent data management*, ed. A. Gammerman, Springer Verlag: Heidelberg, DE, pp. 151–185, 1999.

[48] Diederich, J., Kindermann, J., Leopold, E. & Paass, G., Authorship attribution with support vector machines. *Applied Intelligence*, **19(1/2)**, pp. 109–123, 2003.

[49] Finn, A., Kushmerick, N. & Smyth, B., Genre classification and domain transfer for information filtering. *Proceedings of ECIR-02, 24th European Colloquium on Information Retrieval Research*, eds. F. Crestani, M. Girolami & C.J. van Rijsbergen, Springer Verlag, Heidelberg, DE: Glasgow, UK, pp. 353–362, 2002. Published in the "Lecture Notes in Computer Science" series, number 2291.

[50] Kessler, B., Nunberg, G. & Schütze, H., Automatic detection of text genre. *Proceedings of ACL-97, 35th Annual Meeting of the Association for Computational Linguistics*, eds. P.R. Cohen & W. Wahlster, Morgan Kaufmann Publishers, San Francisco, US: Madrid, ES, pp. 32–38, 1997.

[51] Lee, Y.B. & Myaeng, S.H., Text genre classification with genre-revealing and subject-revealing features. *Proceedings of SIGIR-02, 25th ACM International Conference on Research and Development in Information Retrieval*, eds. M. Beaulieu, R. Baeza-Yates, S.H. Myaeng & K. Järvelin, ACM Press, New York, US: Tampere, FI, pp. 145–150, 2002.

[52] Stamatatos, E., Fakotakis, N. & Kokkinakis, G., Automatic text categorization in terms of genre and author. *Computational Linguistics*, **26(4)**, pp. 471–495, 2000.

[53] Androutsopoulos, I., Koutsias, J., Chandrinos, K.V. & Spyropoulos, C.D., An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, eds. N.J. Belkin, P. Ingwersen & M.K. Leong, ACM Press, New York, US: Athens, GR, pp. 160–167, 2000.

[54] Gómez-Hidalgo, J.M., Evaluating cost-sensitive unsolicited bulk email categorization. *Proceedings of SAC-02, 17th ACM Symposium on Applied Computing*, Madrid, ES, pp. 615–620, 2002.

[55] Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C.D. & Stamatopoulos, P., A memory-based approach to anti-spam filtering for mailing lists. *Information Retrieval*, **6(1)**, pp. 49–73, 2003.

[56] Myers, K., Kearns, M., Singh, S. & Walker, M.A., A boosting approach to topic spotting on subdialogues. *Proceedings of ICML-00, 17th International Conference on Machine Learning*, ed. P. Langley, Morgan Kaufmann Publishers, San Francisco, US: Stanford, US, pp. 655–662, 2000.

[57] Sable, C.L. & Hatzivassiloglou, V., Text-based approaches for non-topical image categorization. *International Journal of Digital Libraries*, **3(3)**, pp.

261–275, 2000.

[58] Larkey, L.S., Automatic essay grading using text categorization techniques. *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, eds. W.B. Croft, A. Moffat, C.J. van Rijsbergen, R. Wilkinson & J. Zobel, ACM Press, New York, US: Melbourne, AU, pp. 90–95, 1998.

[59] Li, X. & Roth, D., Learning question classifiers. *Proceedings of COLING-02, 19th International Conference on Computational Linguistics*, Taipei, TW, 2002.

[60] Koster, C.H. & Seutter, M., Taming wild phrases. *Proceedings of ECIR-03, 25th European Conference on Information Retrieval*, ed. F. Sebastiani, Springer Verlag, Heidelberg, DE: Pisa, IT, pp. 161–176, 2003.

[61] Yang, Y., Zhang, J. & Kisiel, B., A scalability analysis of classifiers in text categorization. *Proceedings of SIGIR-03, 26th ACM International Conference on Research and Development in Information Retrieval*, eds. J. Callan, G. Cormack, C. Clarke, D. Hawking & A. Smeaton, ACM Press, New York, US: Toronto, CA, pp. 96–103, 2003.

[62] Nigam, K., McCallum, A.K., Thrun, S. & Mitchell, T.M., Text classification from labeled and unlabeled documents using EM. *Machine Learning*, **39(2/3)**, pp. 103–134, 2000.

[63] Spärck Jones, K. & Willett, P., (eds.) *Readings in information retrieval*. Morgan Kaufmann: San Francisco, US, 1997.

[64] Debole, F. & Sebastiani, F., Supervised term weighting for automated text categorization. *Text Mining and its Applications*, ed. S. Sirmakessis, Physica-Verlag, Heidelberg, DE, Number 138 in the "Studies in Fuzziness and Soft Computing" series, pp. 81–98, 2004.