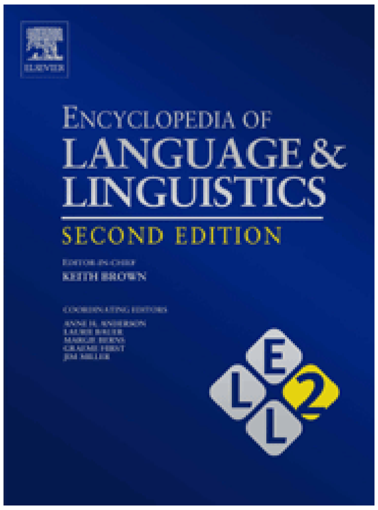

Sebastiani F (2006), Classification of Text, Automatic. In: Keith Brown, (Editor-in-Chief) *Encyclopedia of Language & Linguistics, Second Edition*, volume 2, pp. 457-462. Oxford: Elsevier.

Language Classification (cont.)

Tai Languages
- Lao
- Thai

Uralic Languages
- Estonian
- Finnish
- Hungarian
- Nenets
- Saami

Language isolates and Languages of disputed affiliation
- Ainu
- Basque
- Burushaski
- Elamite
- Japanese
- Ryukyuan
- Ket
- Korean
- Nivkh
- Sumerian
- Yukaghir

Artifical Languages
Esperanto

Language Classification
- The Languages of the World
- Austric hypothesis
- Austro-Tai hypothesis
- Ethnologue
- SIL

Bibliography

Comrie B (ed.) (1987). *The world's major languages.* London: Croom Helm.
Comrie B (1981). *The languages of the Soviet Union.* Cambridge: Cambridge University Press.
Dixon R M W (2002). *The languages of Australia.* Cambridge: Cambridge University Press.
Dixon R M W & Aikhenvald A Y (eds.) (1999). *The Amazonian languages.* Cambridge: Cambridge University Press.
Dolgopolsky A (1998). *The Nostratic macrofamily and linguistic paleontology.* Cambridge: McDonald Institute for Archaeological Research.
Greenberg J H (1987). *Language in the Americas.* Stanford: Stanford University Press.
Grimes B F (2000). *Ethnologue: languages of the world* (14th edn.). Dallas: Summer Institute of Linguistics (http://www.ethnologue.com).
Mithun M (1999). *The languages of native North America.* Cambridge: Cambridge University Press.
Ruhlen M (1991). *A guide to the world's languages. Volume 1: classification* (rev. edn.). Stanford: Stanford University Press.
Suárez J A (1983). *The Mesoamerican Indian languages.* Cambridge: Cambridge University Press.

Classification of Text, Automatic

F Sebastiani, Università di Padova, Padova, Italy

Introduction

In the last two decades, the production of textual documents in digital form has increased exponentially, due to the increased availability of inexpensive hardware and software for generating digital text (e.g., personal computers, word processors) and for digitizing textual data not in digital form (e.g., scanners, optical character recognition software). As a consequence, there is an ever-increasing need for mechanized solutions for organizing the vast quantity of digital texts that are being produced, with an eye toward their future use. The design of such solutions has traditionally been the object of study of information retrieval (IR), the discipline that, broadly speaking, is concerned with the computer-mediated access to data with poorly specified semantics.

There are two main directions for providing convenient access to a large, unstructured repository of text:

- Providing powerful tools for searching relevant documents within this repository. This is the aim of text search (*see* **Document Retrieval, Automatic**), a subdiscipline of IR concerned with building systems that take as input a natural language query and return, as a result, a list of documents ranked according to their estimated degree of relevance to the user's information need. Nowadays the tip of the iceberg of text search is represented by Web search engines (*see* **Web Searching**), but commercial solutions for the text search problem were being delivered decades before the very birth of the Web.
- Providing powerful tools for turning this unstructured repository into a structured one, thereby easing storage, search, and browsing. This is the aim of text classification (TC), a discipline at the crossroads of IR, machine learning (ML), and (statistical)

natural language processing, concerned with building systems that partition an unstructured collection of documents into meaningful groups (Sebastiani, 2002).

Text Clustering and Text Categorization

There are two main variants of TC. The first is **text clustering**, which is characterized by the fact that only the desired number of groups (or clusters) is known in advance: no indication as to the semantics of these groups is instead given as input. The second variant is **text categorization**, whereby the input to the system consists not only of the number of categories (or classes), but also of some specification of their semantics. In the most frequent case, this specification consists in a set of labels, one for each category and usually consisting of a noun or other short natural language expression, and in a set of example labeled texts, i.e., texts whose membership or nonmembership in each of the categories is known. Clustering may thus be seen as the task of finding a latent but as yet undetected group structure in the repository, while categorization can be seen as the task of structuring the repository according to a group structure known in advance. In logical-philosophical terms, we can see clustering as the task of determining both the extensional and intensional level (*see* **Extensionality and Intensionality**) of a previously unknown group structure, and categorization as determining the extensional level only of a group structure whose intensional level is known.

It is the latter task that will be the focus of this article (text clustering is covered elsewhere in this volume – *see* **Text Mining**). From now on we will thus use the expressions ‘text classification’ and ‘text categorization’ interchangeably (abbreviated as TC), and the expression ‘(text) classifier’ to denote a system capable of performing automatic TC.

Note that the central notion of TC, that of membership of a document d_j in a class c_i based on the semantics of d_j and c_i, is an inherently subjective notion, since the semantics of d_j and c_i cannot be formally specified. Different classifiers (be they humans or machines) might thus disagree on whether d_j belongs to c_i. This means that membership cannot be determined with certainty, which in turn means that any classifier (be it human or machine) will be prone to misclassification errors. As a consequence, it is customary to evaluate automatic text classifiers by applying them to a set of labeled (i.e., preclassified) documents (a set that here plays the role of a gold standard), so that the accuracy (or effectiveness) of the classifier can be measured by the degree of coincidence between its classification decisions and the labels originally attached to the preclassified documents.

Single-Label and Multi-Label Text Categorization

TC itself admits of two important variants: single-label TC and multi-label TC. Given as input the set of categorories $C = \{c_1, \ldots, c_m\}$, single-label TC is the task of attributing, to each document d_j in the repository, **the** category to which it belongs. Multi-label TC, instead, deals with the case in which each document d_j may in principle belong to zero, one, or more than one category; it thus comes down to deciding, for each category c_i in C, whether a given document d_j belongs or does not belong to c_i.

The technologies for coping with either single-label or multi-label TCs are slightly different (the former problem often being somehow more challenging), especially concerning the phases of feature selection, classifier learning, and classifier evaluation (see below). In a real application, it is thus of fundamental importance to identify whether the application requires single-label or multi-label TC from the beginning.

Hard or Soft Text Categorization

Taking a binary decision, yes or no, as to whether a document d_j belongs to a category c_i, is sometimes referred to as a ‘hard’ categorization decision. This is the kind of decisions that are taken by autonomous text classifiers, i.e., software systems that need to decide and act accordingly without human supervision. A different type of decision, sometimes referred to as a ‘soft’ categorization decision, is one which consists of attributing a numeric score (e.g., between 0 and 1) to the pair (d_j,c_i), reflecting the degree of confidence of the classifier in the fact that d_j belongs to c_i. This allows, for instance, ranking a set of documents in terms of their estimated appropriateness for category c_i, or ranking a set of categories in terms of their estimated appropriateness for d_j. Such rankings are often useful for nonautonomous, interactive classifiers, i.e., systems whose goal is to recommend a categorization decision to a human expert, who is responsible for making the final decision. For instance, in a single-label TC task a human expert in charge of the final classification decision may take advantage of a system that preranks the categories in terms of their estimated appropriateness to a given document d_j.

Again, the technologies for coping with either soft or hard categorization decisions are slightly different, especially concerning the phases of classifier learning and classifier evaluation (see below). In any real-world application, it is thus important to establish whether the task is one requiring soft or hard decisions from the beginning.

Applications

Maron's seminal paper (Maron, 1961) is usually taken to mark the official birth date of TC, which at the time was called automatic indexing; this name reflected that the main (or only) application that was then envisaged for TC was automatically indexing (i.e., generating internal representations for) scientific articles for Boolean IR systems (*see* **Indexing, Automatic**). In fact, since index terms for these representations were drawn from a fixed, predefined set of such terms, we can regard this type of indexing as an instance of TC (where index terms play the role of categories). The importance of TC increased in the late 1980s and early 1990s with the need to organize the increasingly larger quantities of digital text being handled in organizations at all levels. Since then, frequently pursued applications of TC technology have been newswire filtering, i.e., the grouping, according to thematic classes of interest, of news stories produced by news agencies, thus allowing personalized delivery of information to customers according to their profiles of interest (Hayes and Weinstein, 1990); patent classification, i.e., the organization of patents and patent applications into specialized taxonomies, so as to ease the detection of existing patents related to a new patent application (Fall *et al.*, 2003); and Web page classification, i.e., the grouping of Web pages (or sites) according to the taxonomic classification schemes typical of Web portals (Dumais and Chen, 2000).

The applications above all have a certain thematic flavor, in the sense that categories tend to coincide with topics, or themes. However, TC technology has been applied to real-world problems that are not thematic in nature, among which spam filtering, i.e., the grouping of personal e-mail messages into the two classes LEGITIMATE and SPAM, so as to provide effective user shields against unsolicited bulk mailings (Drucker *et al.*, 1999); authorship attribution, i.e., the automatic identification of the author of a text among a predefined set of candidates (Diederich *et al.*, 2003) (*see* **Authorship Attribution: Statistical and Computational Methods**); author gender detection, i.e., a special case of the previous task in which the issue is deciding whether the author of the text is a MALE or a FEMALE (Koppel *et al.*, 2002); genre classification, i.e., the identification of the nontopical communicative goal of the text (such as determining if a product description is a PRODUCTREVIEW or an ADVERTISEMENT) (Stamatatos *et al.*, 2000); survey coding, i.e., the classification of respondents to a survey based on the textual answers they have returned to an open-ended question (Giorgetti and Sebastiani, 2003); or even sentiment classification, as in deciding if a product review is a THUMBSUP or a THUMBSDOWN (Turney and Littman, 2003).

Techniques

Approaches

In the 1980s, the most popular approach to TC was one based on knowledge engineering, whereby a knowledge engineer and a domain expert working together would build an expert system capable of automatically classifying text. Typically, such an expert system would consist of a set of 'if ... then ...' rules, to the effect that a document was assigned to the class specified in the 'then' clause only if the linguistic expressions (typically: words) specified in the 'if' part occurred in the document. The drawback of this approach was the high cost in terms of human-power required for (i) defining the rule set, and (ii) for maintaining it, i.e., for updating the rule set as a result of possible subsequent additions or deletions of classes or as a result of shifts in the meaning of the existing classes.

In the 1990s, this approach was superseded by the machine-learning approach, whereby a general inductive process (the learner) is fed with a set of example (training) documents preclassified according to the categories of interest. By observing the characteristics of the training documents, the learner may generate a model (the classifier) of the conditions that are satisfied by the documents belonging to the categories considered. This model can subsequently be applied to new, unlabeled documents for classifying them according to these categories.

This approach has several advantages over the knowledge engineering approach. First of all, a higher degree of automation is introduced: the engineer needs to build not a text classifier, but an automatic builder of text classifiers (the learner). Once built, the learner can then be applied to generating many different classifiers, for many different domains and applications: one only needs to feed it with the appropriate sets of training documents. By the same token, the above-mentioned problem of maintaining a classifier is solved by feeding new training documents appropriate for the revised set of classes. Many inductive learners are available off the shelf; if one of these is used, the only human power needed in setting up a TC system is that for manually classifying the documents to be used for training. For performing this latter task, less skilled human power than for building an expert system is needed, which is also advantageous. It should also be noted that if an organization has previously relied on manual work for classifying documents, then many preclassified documents are already available to be used as training documents when the organization decides to automate the process.

Most importantly, one of the advantages of the ML approach is that the accuracy of classifiers built by these techniques now often rivals that of human professionals, and usually exceeds that of classifiers built by knowledge engineering methods. This has brought about a wider and wider acceptance of learning methods even outside academia. While for certain applications such as spam filtering a combination of ML and knowledge engineering still lies at the basis of several commercial systems, it is fair to say that in most other TC applications (especially of the thematic type), the adoption of ML technology has been widespread.

Note that the ML approach is especially suited to the case in which no additional knowledge (of a procedural or declarative nature) of the meaning of the categories is available, since in this case the classification rules can be determined only on the basis of knowledge extracted from the training documents. This case is the most frequent one, and is thus the usual focus of TC research. Solutions devised for the case in which no additional knowledge is available are extremely general, since they do not presuppose the existence of e.g., additional lexicosemantic resources that, in real-life situations, might be either unavailable or expensive to create (*see* **Computational Lexicons and Dictionaries**). 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 instance of their application to TC a case in its own, from which any lesson learned can hardly be exported to different application contexts. When in a given application external knowledge of some kind 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. However, it should be noted that past research has not been able to show any substantial benefit from the use of external resources (such as lexicons, thesauri, or ontologies) in TC.

As previously noted, the meaning of categories is subjective. The ML techniques used for TC, rather than trying to learn a supposedly perfect classifier (a gold standard of dubious existence), strive to reproduce the subjective judgment of the expert who has labeled the training documents, and do this by examining the manifestations of this judgment, i.e., the documents that the expert has manually classified. The kind of learning that these ML techniques engage in is usually called supervised learning, since it is supervised, or facilitated, by the knowledge of the preclassified data.

Learning Text Classifiers

Many different types of supervised learners have been used in TC (Sebastiani, 2002), including probabilistic 'naive Bayesian' methods, Bayesian networks, regression methods, decision trees, Boolean decision rules, neural networks, incremental or batch methods for learning linear classifiers, example-based methods, classifier ensembles (including boosting methods), and support vector machines. While all of these techniques still retain their popularity, it is fair to say that in recent years support vector machines (Joachims, 1998) and boosting (Schapire and Singer, 2000) have been the two dominant learning methods in TC. This seems attributable to a combination of two factors: (i) these two methods have strong justifications in terms of computational learning theory, and (ii) in comparative experiments on widely accepted benchmarks, they have outperformed all other competing approaches. An additional factor that has determined their success is the free availability, at least for research purposes, of well-known software packages based on these methods, such as SVMlight and BoosTexter.

Building Internal Representations for Documents

The learners discussed above cannot operate on the documents as they are, but require the documents to be given internal representations that the learners can make sense of. The same is true of the classifiers, once they have been built by learners. It is thus customary to transform all the documents (i.e., those that are used in the training phase, the testing phase, or the operational phase of the classifier) into internal representations by means of methods used in text search, where the same need is also present (*see* **Indexing, Automatic**). Accordingly, a document is usually represented by a vector lying in a vector space whose dimensions correspond to the terms that occur in the training set, and the value of each individual entry corresponds to the weight that the term in question has for the document.

In TC applications of the thematic kind, the set of terms is usually made to coincide with the set of content-bearing words (which means all words but topic-neutral ones such as articles, prepositions, etc.), possibly reduced to their morphological roots (stems – *see* **Stemming**) so as to avoid excessive stochastic dependence among different dimensions of the vector. Weights for these words are meant to reflect the importance that the word has in determining the semantics of the document it occurs in, and are automatically computed by weighting functions. These functions usually rely on intuitions of a statistical kind, such as (i) the more often a term occurs in a document, the more important it is for that document; and (ii) the more documents a term appears in, the less important it is in characterizing the semantics of a document it occurs in.

In TC applications of a nonthematic nature, the opposite is often true. For instance, it is the frequency of use of articles, prepositions, and punctuation (together with many other stylistic features) that may be a helpful clue in authorship attribution, while it is more unlikely that the frequencies of use of content-bearing words can be of help (*see* **Computational Stylistics**). This shows that choosing the right dimensions of the vector space for the right classification task requires a deep understanding, on the part of the engineer, of the nature of the task.

It is fairly evident from the above discussion that internal representations used in TC applications are, from the standpoint of linguistic analysis, extremely primitive: with the possible exception of applications in sentiment classification (Turney and Littman, 2003), hardly any sophisticated linguistic analysis is usually attempted in order to provide a more faithful rendition of the semantics of the text. This is because previous attempts at applying state-of-the-art natural language processing techniques (including techniques for parsing text robustly (Moschitti and Basili, 2004), extracting collocations (Koster and Seutter, 2003), performing word sense disambiguation (Kehagias *et al.*, 2003), etc.) have not shown any substantial benefit with respect to the basic representations outlined above.

Reducing the Dimensionality of the Vectors

The techniques described in the previous section tend to generate very large vectors, with sizes in the tens of thousands. This situation is problematic in TC, since the efficiency of many learning devices (e.g., neural networks) tends to degrade rapidly with the size of the vectors. In TC applications, it is thus customary to run a dimensionality reduction pass before starting to build the internal representations of the documents. Basically, this means identifying a new vector space in which to represent the documents, with a much smaller number of dimensions than the original one. Several techniques for dimensionality reduction have been devised within TC (or, more often, borrowed from the fields of ML and pattern recognition).

An important class of such techniques is that of feature extraction methods (examples of which are term clustering methods and latent semantic indexing – *see* **Latent Semantic Analysis**). Feature extraction methods define a new vector space in which each dimension is a combination of some (or all) of the original dimensions; their effect is usually a reduction of both the dimensionality of the vectors and the overall stochastic dependence among dimensions.

An even more important class of dimensionality reduction techniques is that of feature selection methods, which do not attempt to generate new terms, but try to select the best ones from the original set. The measure of quality for a term is its expected impact on the accuracy of the resulting classifier. To measure this, feature selection functions are employed for scoring each term according to this expected impact, so that the highest scoring ones can be retained for the new vector space. These functions mostly come from statistics (e.g., Chi-square) or information theory (e.g., mutual information, also known as information gain), and tend to encode (each one in their own way) the intuition that the best terms for classification purposes are the ones that are distributed most differently across the different categories.

Challenges

TC, especially in its ML incarnation, is today a fairly mature technology that has delivered working solutions in a number of applicative contexts. Interest in TC has grown exponentially in the last 10 years, from researchers and developers alike.

For IR researchers, this interest is one particular aspect of a general movement toward leveraging user data for taming the inherent subjectivity of the IR task, 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 she has issued to a Web search engine, or relevant to a private folder of hers in which documents should be filed according to content. Wherever there are predefined classes, documents previously (and manually) classified by the user are often available; as a consequence, these latter data can be exploited for automatically learning the (extensional) meaning that the user attributes to the categories, thereby reaching accuracy levels that would be unthinkable if these data were unavailable.

For ML researchers, this interest is because TC applications prove a challenging benchmark for their newly developed techniques, since these applications usually feature extremely high-dimensional vector spaces and provide large quantities of test data. In the last 5 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 applied to TC with minimal delay since 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 also results from TC techniques having reached accuracy levels that often rival the performance of trained professionals, levels that can be achieved with high efficiency on standard hardware and software resources. This means that more and more organizations are automating all their activities that can be cast as TC tasks. Still, a number of challenges remain for TC research.

The first and foremost challenge is to deliver high accuracy in **all** applicative contexts. While highly effective classifiers have been produced for applicative domains such as the thematic classification of professionally authored text (such as newswire stories), in other domains reported accuracies are far from satisfying. Such applicative contexts include the classification of Web pages (where the use of text is more varied and obeys rules different from the ones of linear verbal communication), spam filtering (a task which has an adversarial nature, in that spammers adapt their spamming strategies so as to circumvent the latest spam filtering technologies), authorship attribution (where current technology is not yet able to tackle the inherent stylistic variability among texts written by the same author), and sentiment classification (which requires much more sophisticated linguistic analysis than classification by topic).

A second important challenge is to bypass the document labeling bottleneck, i.e., tackling the facts that labeled documents for use in the training phase are not always available, and that labeling (i.e., manually classifying) documents is costly. To this end, semisupervised methods have been proposed that allow building classifiers from a small sample of labeled documents and a (usually larger) sample of unlabeled documents (Nigam and Ghani, 2000; Nigam *et al.*, 2000). However, the problem of learning text classifiers mainly from unlabeled data unfortunately is still open.

See also: Authorship Attribution: Statistical and Computational Methods; Computational Lexicons and Dictionaries; Computational Stylistics; Document Retrieval, Automatic; Extensionality and Intensionality; Indexing, Automatic; Latent Semantic Analysis; Stemming; Text Mining; Web Searching.

Bibliography

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

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

Dumais ST & Chen H (2000). 'Hierarchical classification of Web content.' In Belkin NJ, Ingwersen P & Leong M-K (eds.) *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval.* Athens: ACM Press. 256–263.

Fall CJ, Törcsvári A, Benzineb K & Karetka G (2003). 'Automated categorization in the International Patent Classification.' *SIGIR Forum 37(1),* 10–25.

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

Hayes PJ & Weinstein SP (1990). 'CONSTRUE/TIS: a system for content-based indexing of a database of news stories.' In Rappaport A & Smith R (eds.) *Proceedings of IAAI-90, 2nd Conference on Innovative Applications of Artificial Intelligence.* Menlo Park: AAAI Press. 49–66.

Joachims T (1998). 'Text categorization with support vector machines: learning with many relevant features.' In Nédellec C & Rouveirol C (eds.) *Proceedings of ECML-98, 10th European Conference on Machine Learning.* Lecture Notes in Computer Science series, no. 1398 Heidelberg: Springer Verlag. 137–142.

Kehagias A, Petridis V, Kaburlasos VG & Fragkou P (2003). 'A comparison of word- and sense-based text categorization using several classification algorithms.' *Journal of Intelligent Information Systems 21(3),* 227–247.

Koppel M, Argamon S & Shimoni AR (2002). 'Automatically categorizing written texts by author gender.' *Literary and Linguistic Computing 17(4),* 401–412.

Koster CH & Seutter M (2003). 'Taming wild phrases.' In Sebastiani F (ed.) *Proceedings of ECIR-03, 25th European Conference on Information Retrieval.* Pisa: Springer Verlag. 161–176.

Maron M (1961). 'Automatic indexing: an experimental inquiry.' *Journal of the Association for Computing Machinery 8(3),* 404–417.

Moschitti A & Basili R (2004). 'Complex linguistic features for text classification: a comprehensive study.' In McDonald S & Tait J (eds.) *Proceedings of ECIR-04, 26th European Conference on Information Retrieval Research.* Lecture Notes in Computer Science series, no. 2997. Heidelberg: Springer Verlag. 181–196.

Nigam K & Ghani R (2000). 'Analyzing the applicability and effectiveness of co-training.' In Agah A, Callan J & Rundensteiner E (eds.) *Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management.* McLean: ACM Press. 86–93.

Nigam K, McCallum AK, Thrun S & Mitchell TM (2000). 'Text classification from labeled and unlabeled documents using EM.' *Machine Learning 39(2/3),* 103–134.

Schapire RE & Singer Y (2000). 'BoosTexter: a boosting-based system for text categorization.' *Machine Learning 39(2/3),* 135–168.

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

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

Turney PD & Littman ML (2003). 'Measuring praise and criticism: inference of semantic orientation from association.' *ACM Transactions on Information Systems 21(4),* 315–346.

Relevant Websites

http://svmlight.joachims.org – SVMlight web site.

http://www.research.att.com/%schapire/BoosTexter/ – BoosTexter website.

http://www.math.unipd.it/~fabseb60 – F. Sebastiani's website.