Adaptive Bayesian Information Filtering

by

Brian D. Chambers

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Abstract

Adaptive Bayesian Information Filtering

Brian D. Chambers

Master of Science

Graduate Department of Computer Science

University of Toronto

1999

A new approach to *interactive information filtering* is presented: *incremental* Bayesian inference is applied to a multinomial model of text-document relevance as a means of *learning user information needs* over extended periods of time, through interactive data sampling. An information filtering agent acts autonomously on a user's behalf by filtering on-line document streams for text documents that are relevant to the user's information need and forwarding such documents to the user. For each forwarded document, the user is prompted to *confirm* or *deny* its relevance; a filtering agent that is able to incorporate such *user feedback* into its decision process can significantly improve its future document selection accuracy.

In contrast to nonprobabilistic information filtering models, which are based on heuristics and ad hoc techniques, the proposed probabilistic model provides a theoretical foundation for interactive information filtering. During empirical trials, the proposed probabilistic model has shown improved performance for certain measures, relative to nonprobabilistic models.

Acknowledgements

I would first like to thank my supervisors, John Mylopoulos and Grigoris Karakoulas, for their guidance, encouragement, and continued interest in my work. Their enthusiasm, generous donation of time, and the professionalism with which they conduct their own work has been a source of inspiration during the completion of the current work.

In addition, I would like to thank several other individuals for their assistance during my graduate work: Brian Nixon provided invaluable academic and administrative advice, and Christina Christara and Kathy Yen provided ongoing administrative assistance, answering my numerous questions in a friendly and helpful manner.

Financial support was gratefully received from the Department of Computer Science and the University of Toronto.

Finally, I would like to thank my family for their ongoing support; I would especially like to thank my sister and her husband for the initial encouragement that prompted me to pursue graduate studies.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

The recent explosive growth of networked, disseminated on-line information sources has led to a significant need for *automated* methods of retrieval of such information. For example, every user of the *World Wide Web* is familiar with commercial search engines or topical directories, such as *Lycos* [Lyc94] or *Yahoo!* [Yah94]. The immense volume of source information, however, often leads to query results which are too long and unwieldy for human users to manage effectively. The need therefore arises for more "intelligent" aids for information access tasks. **Information filtering** is an example of such an information access process and is the subject of the current work. Information filtering processes are applicable to information access situations in which user needs are relatively stable and information sources have dynamically varying content (a characteristic of much of today's on-line information environment).

Information filtering, ad hoc retrieval, and *database management systems* represent three important and contrasting automated information access methodologies, and are illustrated in Table 1.1. The table illustrates differences between these information access methodologies according to the following subjective measures, suggested by Oard and Marchionini [Oar96]:

- **Rate of change of user information needs**
  Users may have relatively stable long-term needs, or their needs may vary for each data access.

- **Rate of change of the source information content**
  The information content of the information source may be stable, or may change dynamically and independently of user needs.

- **Structure of the stored information**
  The stored information may be rigidly formatted according to some convention, or it may be "free form".

- **Nature of the output of the access process**
  The *output* of an information access process either (1) **is** the information desired (direct access), or (2) **contains** the information desired (indirect access). An example of the former is the value of the salary field of an employee payroll record returned by a relational-database query; an example of the latter is a document returned by an information filtering process.

Process	*User Need*	*Source*	*Information Structure*	*Output*
Information filtering	**static**	**dynamic**	**unstructured**	**indirect**
Ad hoc retrieval	dynamic	static	unstructured	indirect
Database access	dynamic	static	structured	direct

Table 1.1: Important Information Access processes

Information filtering and ad hoc retrieval have complementary usage patterns (static vs. dynamic) for *both* user need and source information; i.e., information filtering processes are characterized by relatively stable long-term user information requests of dynamically changing information sources. Ad hoc retrieval processes are characterized by frequently changing user information requests of relatively static information sources.

It is most informative to compare information filtering with database access: these two information access processes are complementary for *each* of the four measures. Perhaps the most important difference between the two processes is illustrated by the dynamic

nature of information filtering's information **source**, versus the relatively stable information sources characteristic of database systems. For database systems, an information need can always be precisely mapped into a query for which there will be a precise definition of which database items form the answer to the query. For information filtering, neither the query (user profile) nor the answer to the query can be precisely formulated, i.e., there is no precise definition of *which* source documents will match a given user's query. Therefore uncertainty is implicit in the information filtering process: information filtering's highly **nondeterministic** nature makes probability theory a natural tool for formulating the task. This is the central theme of the current work.

1.2 Machine Learning and Information Filtering

Machine learning is defined to be any automated process that *improves its performance* at some task through experience, where performance is measured based on some prespecified measure. The *classification* problem within machine learning is the task of classifying observed phenomena into two or more discrete sets of possible categories. *Binary* classification into two categories such as yes/no or relevant/nonrelevant is often used. Classification involves partitioning a set of previously **unseen** input items of some domain into these two categories, based *only* on observations of the features of *previously classified "training examples"* of items from the domain.

Given a data domain, *inductive learning* may be defined as a form of inference where a person or system generalizes beyond training examples to infer the classification of *new data instances*. It is characterized by an inductive inference *assumption* or inductive bias; e.g., to ensure inductive learning in the case of document information filtering, some assumption must be made about the *manner* in which document features may be used to classify documents. Without such an assumption, the best that can be achieved is rote-learning. Mitchell [Mit97] provides an excellent exposition of inductive bias and of

machine learning in general.

Users typically initialize document information filtering systems in one of two ways: by providing a set of labeled *training documents* partitioned into two disjoint subsets of relevant and nonrelevant documents, **or** by providing a *natural language description* of some specific interest. In the former case, information filtering becomes a batch-oriented classification supervised learning problem, as discussed above. If the user profile is in the form of a natural-language statement, the task is more difficult because it is widely recognized that users have great difficulty accurately describing or verbalizing their interests in a concise manner. In this case, users are required to guide the process *during* its operation. This is called *relevance feedback* supervised learning: during the operation, the user provides judgement feedback to the process about the *true* relevance of documents which the process has estimated as relevant to the user's information need. Many such user judgements allow the process to iteratively improve its stored user-profile. The user-profile therefore "evolves" into a much more accurate representation of the user's information need than is represented by the initially provided natural language description.

The second of the above two methods (the initial profile as a natural language description of a user's interest, with relevance feedback) is used in the subsequent work because it is most consistent with the *interactive* nature of today's distributed computing environment. Today's highly interactive computer environment has led to a shift in focus from batch-oriented information retrieval learning algorithms towards **on-line interactive** informative processing algorithms. Interactive information filtering systems are expected to learn user information needs on-line, based upon user feedback, rather than from prespecified training examples. Such systems are expected to respond immediately to user input, based only on information processed up to that point in time. Interactive information filtering systems rely on user relevance feedback to improve performance with experience. However, research in the area of relevance feedback has largely

been in the context of nonprobabilistic filtering models. This work investigates relevance feedback within an **adaptive Bayesian probabilistic model** and demonstrates that such an approach is an effective means of achieving on-line learning.

The primary learning performance measures used for information filtering are *recall* and *precision*. Recall is the percentage of all possible relevant documents that the process has successfully retrieved. Precision is the percentage of retrieved documents that are, in fact, truly relevant. Recall is usable only in experimental environments because in "real-world" environments, the number of truly relevant documents within a given document source is usually unknown.

1.3 Information Filtering Models

Information filtering models may be interpreted as decision functions whose domain is the set of all possible document features and whose range is the set {relevant, nonrelevant}.

Filtering models fall into two broad categories: nonparametric and parametric. *Non-parametric models*, such as the the Vector Space Model (VSM) [Sal83], do not assume the nature of the distribution of the source data. The Vector Space Model represents queries and documents as vectors in a vector space, with component terms weighted in some manner, as discussed below. The relevance of an unseen document to a given query is judged by calculating the distance between the query vector and the document vector and comparing this distance to a threshold value (vector distance is established using some prespecified distance-metric). *Parametric* models [vanR79], [Fuh93], make assumptions about *how* the data is generated and postulate a probabilistic model that embodies these assumptions. A collection of labeled training examples and relevance feedback is used to estimate the parameters of the generative model, and classification of new examples is made by selecting the class that is most likely to have generated each

example. For example, in the two-class relevant/nonrelevant case, the most likely class is selected based on the *maximum* of

$$P(relevant \mid d; t) \quad \text{and} \quad P(nonrelevant \mid d; t),$$

where d is a new (previously unseen) document, and t represents a user topic of interest.

The "bag of words" model of representing documents is typically used for both the vector space and probabilistic models, i.e., the features (tokens) used to represent a document possess no internal structure, distinguishing semantics, or relationships with each other. Given an enumerated vocabulary $V = \{t_1, t_2, ..., t_{|V|}\}$ representing an arbitrary set of tokens such as English language words, or any other prespecified set of tokens, then any document d may be represented as a *weight vector*

$$w^d = (w_1, w_2, ..., w_{|V|}),$$

where, for $i = 1, \ldots, |V|$, w_i is either (1) zero or one, indicating the *absence or presence* of t_i in d or (2) a non-negative integer, indicating the *frequency* of occurrence of t_i in d. In the probabilistic model, the former leads to the multi-variate Bernoulli document model and the latter leads to the Poisson or multinomial document models.

The current work investigates the *probabilistic* model because (1) it fits naturally with the uncertainty inherent in the information filtering process, as discussed in section 1.2, and (2) its firm theoretical foundation allows rigorous analytic treatment of the problem; in contrast, the vector space model uses heuristics and ad hoc techniques which are not easily amenable to analysis.

The *multinomial* probabilistic model was chosen because (1) in contrast to the multi-variate Bernoulli model, it is intuitively appealing that the relevance decision function should depend on document word *frequencies* (i.e., the more frequently a user-specified "important" word occurs in a document, the more likely that the document will be relevant to the user's interest), (2) the multinomial model outperforms the multi-variate Bernoulli model for text classification, especially at large vocabulary sizes, as shown by

McCallum and Nigam [McC98], and (3) to the best of our knowledge, multinomial models have not been applied to the information filtering task.

1.4 Conceptual Modeling and Telos

Research in the areas of information retrieval and filtering has historically addressed only the process of *accessing* information; i.e., it has tended to ignore the important follow-up task of *storing* retrieved information in an *organized persistent form* that can be easily used by users[1]. Filtering an on-line news feed for a specific topic will yield documents that will need to be stored and somehow related to the topic, its subtopics, and its parent subject-domain. Subject-domains, topics, and documents are complex objects with complex inter-relationships which are not easily representable in, for example, a relational database. In addition, documents are aggregations of other complex objects such as document ID, title, header, and body. The ideal solution is to store not only a document's low-level information tokens but **also** its internal structure and relationships with its topic classes, in a manner that is consistent with the way *humans* view such structure and relationships. This ideal implementation-independent modeling approach is called *Conceptual Modeling* [Myl92a].

To provide the information filtering user with the ability to store filtering results persistently in a conceptual manner, the conceptual modeling language *Telos* [Kou89], [Myl90], [Myl92a] has been adopted: a Telos information base is implemented in the form of a query-capable repository that is able to to represent subject-domains, topics, subtopics, retrieved documents, *and* relationships between these objects. Telos offers many powerful and novel features including (1) a sophisticated object-oriented framework, which allows complex real-world structures to be represented without being arbi-

[1]It is only recently that this trend has begun to change: see, for example, [Cra98] and [Hah98]; the former discusses the use of machine learning to build computer-understandable knowledge bases from information extracted from the World Wide Web; the latter introduces a methodology for automating the maintenance of domain-specific taxonomies based on new concepts retrieved from real-world texts.

trarily decomposed into tables and records, as is required for relational database representation, (2) a single *proposition* building-block with which all Telos objects are constructed, (3) first-class nature of an attribute (an attribute is simply a proposition), (4) three structuring mechanisms (classification, aggregation, and generalization), (5) classification along an *infinite* dimension of metaclasses (rather than the single object-class dimension characteristic of object-oriented programming languages), and (6) multiple instantiation.

1.5 Thesis Contribution

The following summarizes the contribution of the current work to *interactive information filtering*:

1. The application of the multinomial distribution as a model of text-document relevance;
2. The application of *incremental* Bayesian methods, using the Dirichlet conjugate family and *sequential data sampling*, as a means of estimating the parameters of the multinomial document model;
3. Conceptual modeling of the information domain of a specific document filtering application.

Items 1 and 2 are novel approaches to information filtering and have led to favorable performance results, relative to other information filtering techniques.

1.6 Thesis Organization

Chapter 2 presents architectural overviews of the information filtering and document repository subsystems and discusses the manner in which these subsystems are integrated into an overall system. Chapter 3 discusses term-weighting document representation schemes and provides a brief literature review of information retrieval methods. Chapter 4 introduces probabilistic machine learning: Bayesian learning and the naive

Bayes methodology. Chapter 5 presents the theoretical framework for achieving relevance feedback in an *interactive* probabilistic information filtering environment, through an adaptive Bayesian approach (the Dirichlet conjugate family) that is able to incrementally estimate the parameters of multinomial models. Chapter 6 presents empirical results of an implementation of the approach outlined in Chapter 5 and compares performance with that of two *nonprobabilistic* information filtering processes: a vector-space machine learning system, and a static "benchmark" system. Chapter 7 discusses conceptual modeling and the background and motivation behind Telos. Chapter 8 discusses the rationale behind the Telos implementation of the repository and outlines in detail several repository features. Chapter 9 concludes with a summary of results and suggests possible directions for future research.

Chapter 2

System Architecture

2.1 Introduction

Interactive information filtering systems are characterized by the asynchronous arrival of textual information from external on-line sources such as news-wire feeds. Such documents must be processed immediately in order of arrival. For each arriving document, the system must make an immediate relevance judgement whether the document is relevant to a user's prespecified information need; if the system judges that the document is relevant, then it is forwarded to the user for confirmation of the system's judgement. The user's confirmation or denial of the system's judgement is provided as feedback to the system so that it may update its user-profile accordingly.

This chapter provides a detailed description of the architecture and functionality of the document filtering and repository system. The system consists of a **source-document subsystem**, an **autonomous filtering agent**, a **user-interface agent**[1], a **repository agent**, and a **Telos repository**. To highlight the nature of inter-agent communication, the following exercitive, interrogative, and assertive *speech act performatives* are employed, as suggested in [Fer99]:

[1]In the remainder of the chapter, "user agent" will be used in place of "user-interface agent".

- **Request**: Request that another agent perform a specific *task*
- **Question**: Request that another agent provide specific *information*
- **Reply**: Response to another agent's Question
- **Assert**: Provide another agent with *information* (the target agent adds the received information to its set of beliefs)
- **Inform**: Provide another agent with *information* (the target agent may optionally add or not add the received information to its set of beliefs).

Figure 2.1 provides a graphical overview of the system; only those system messages and information flows that are central to a general understanding of the system are shown.

2.2 Source-Document Subsystem

The purpose of the **source-document subsystem** is to transform randomly arriving raw text news-documents into a form usable by the remainder of the automated system. The news source is assumed to be a standard on-line "live" news-feed such as *Reuters, AP, CNN*, etc.. Arriving news documents are in *text* form with minimal internal structure; they consist of a title, a brief header description, and a body. The source-document subsystem parses each arriving document and creates a standard *structured vector representation* of the document for use by the other subsystems. This vector representation of the document, called *term-weighting* or *VSM*-representation, was mentioned briefly in Chapter 1 and will be discussed in more detail in Chapter 3. For now, it is sufficient to consider this subsystem as a *transformation process* that creates structured *machine-readable vector* representations of unstructured human-readable document text.

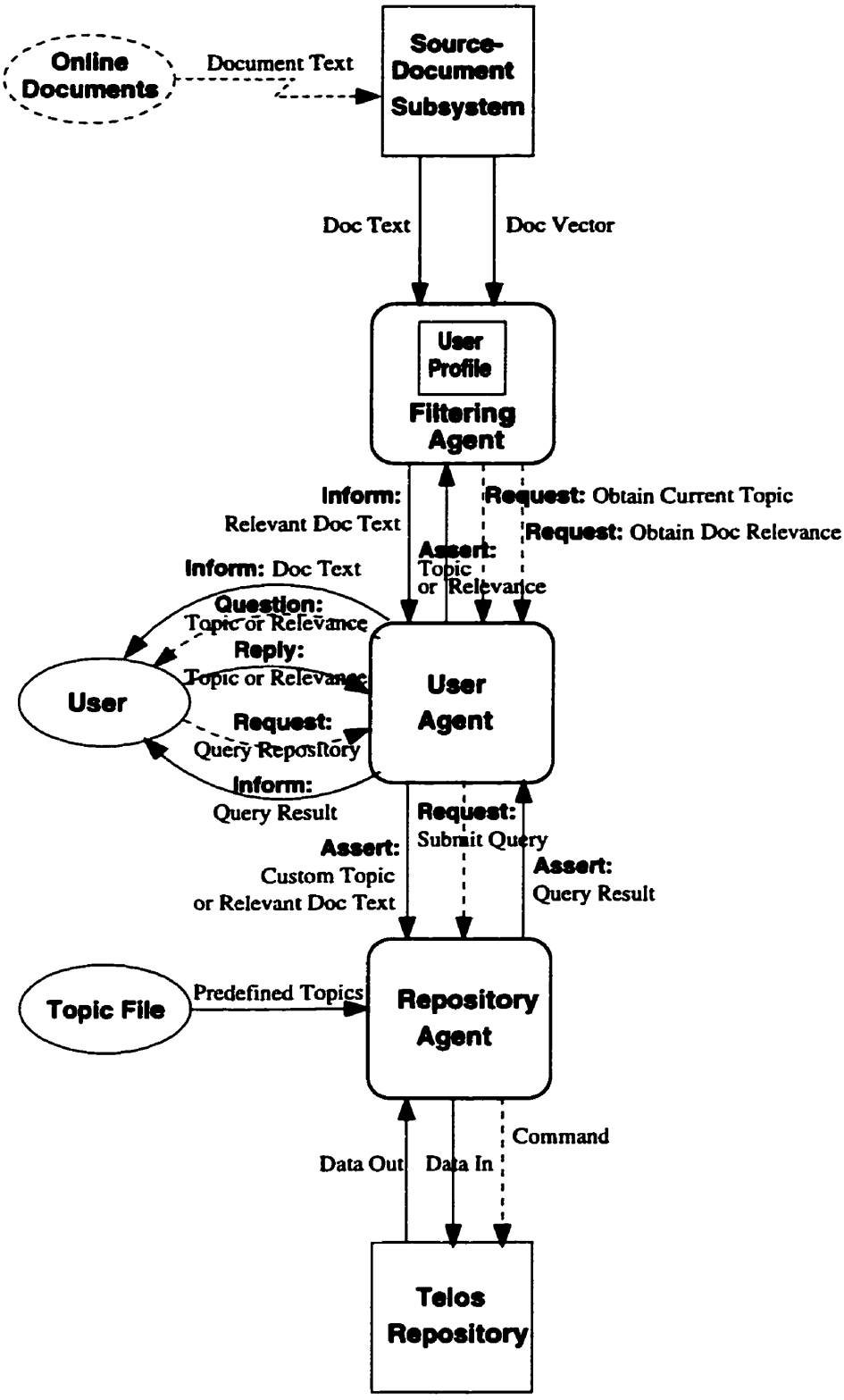


Figure 2.1: System architecture.

2.3 Filtering Agent

The **filtering agent** occupies the *position* of a **personal assistant** to a human user (acting through a user agent); it possesses *specialized knowledge* of news-domains and collaborates with the user, acting *autonomously* on the user's *behalf*, with the goal of assisting the user in detecting news-documents that are *relevant* to a given news-topic. It accomplishes this goal by

- *minimizing* the user's time and effort in locating relevant documents, and
- *providing an abstraction to the user* of the specialized news-domain knowledge required to perform the news-document filtering task.

The filtering agent filters out incoming irrelevant news-documents and presents to the user agent only those documents that it believes that the user will find interesting. Over time, it becomes more effective as it gradually learns the user's news preference and becomes more accurate in performing the filtering task. As a personal assistant to the user, it covers several *roles* within the system:

- *Interfacing* with the source-document subsystem,
- *Managing* the user-profile,
- *Calculating* the "closeness"[2] or relevance of a document-vector to the user-profile,
- *Communicating* with the user agent.

The following is a sequential outline of the *tasks* associated with the filtering agent's roles:

1. **Request**(user agent; obtain from user, the *topic* for the current document filtering session)
2. Initialize user-profile vector t using the current topic's title and text

[2]Chapter 3 will discuss in detail the notion of "closeness" of a document-vector to a user-profile.

3. Obtain next news-document vector d from source-document subsystem
4. Retrieve current user-profile vector t
5. Calculate $P_R = P(Relevant|d;t)$ and $P_N = P(NonRelevant|d;t)$
6. If $P_R > P_N$, then for the *hypothesized* relevant document d:
   (a) Obtain text of d from source-document subsystem
   (b) **Inform**(user agent; text of d)
   (c) **Request**(user agent; obtain from user, the *actual* relevance-judgement of d)
   (d) Update user-profile vector t with the *actual* relevance-judgement of d
7. Repeat from Step 3.

2.4 User Agent

The user **agent's** *role* is that of an **interface** between the *human user* and (1) the filtering agent, and (2) the repository agent. It acts on *behalf* of the human user with the *goal* of providing an *abstraction* of the entire system to the human user by hiding all implementation details of the system from the user. It is responsible for the following *tasks*:

- **Question**(user; choice between reinitializing the repository with a set of *predefined* topics or adding a new *custom* topic definition to the existing set of topics)
- **Assert**(repository agent; user's choice of initializing with predefined topics or adding a new custom topic)
- **Question**(user; description of the new custom topic, if any: subject-area, title, and narrative text)
- **Assert**(repository agent; description the custom topic, if any)
- **Question**(user; topic for the *current* news-document filtering session)

- **Assert**(filtering agent; topic for the *current* news-document filtering session)
- **Inform**(user; text of each document *hypothesized* by the filtering agent as *relevant* to the current topic)
- **Question**(user;
  1. *actual* relevance-judgement of the hypothesized relevant document, and
  2. optional *semantic-categories* of selected document keywords)
- **Assert**(filtering agent; *actual* relevance-judgement of the hypothesized relevant document)
- If user confirms that the proposed document is actually relevant, then:

  Assert(repository agent; text of relevant document and optional keyword semantic-categorization for classification of document token under topic simple class)
- Transform user queries into repository agent queries
- **Request**(repository agent; submit repository queries to repository)
- Transform query-results into user understandable form
- **Inform**(user; query-results).

2.5 Repository Agent

The **repository agent's** *role* is that of a **Telos repository manager**. It acts on *behalf* of the user agent with the *goal* of providing an *abstraction* of the Telos repository to the user agent by hiding all implementation details of the repository from the user agent. It is responsible for the following *tasks*:

- Accept from user agent: directive to initialize repository with predefined topics *or* add a custom topic
- Input from file system: predefined topics and subject-domains, if required
- Accept from user agent: custom topic description, if required

- Repository TELL: initial instantiation of predefined or custom topics (simple classes as instances of subject-domain meta classes)
- Accept from user agent: text of *relevant* news-documents for classification under the current topic, and optional keyword semantic categorization
- Repository TELL: instantiate *relevant* news-documents (tokens as instances of topic simple classes)
- Accept from user agent: repository queries
- Transform user agent queries into Telos queries
- Repository QUERY: Telos queries
- Transform Telos query-results into user-agent understandable form
- **Assert**(user agent; query-results).

2.6 Telos Repository

The **Telos repository** represents an *information base* of a Telos conceptual model of the *information domain* or *subject world* of the news-document filtering system. Conceptual modeling and Telos are discussed in Chapter 7, and the *specific* Telos conceptual model implemented in the repository is outlined in Chapter 8.

Chapter 3

Introduction to Information Access

Information filtering and ad hoc retrieval were briefly introduced and compared to database access in Section 1.1 and related to machine learning in Section 1.2. Section 1.3 discussed the two classes of term-weighting information access models: nonparametric and parametric models. This chapter provides brief overviews of these models, together with their related document representation schemes. The goal of information access is to *learn a user's information need*, in the form of a user-profile, through the use of training documents or user feedback; the learned user-profile is then used to judge how well *new previously unseen documents* satisfy the user's information need. The following definitions illustrate the broad nature of information access:

Information Classification[1] *Batch-oriented* training using a *static collection* of user-provided relevant/nonrelevant-*labeled* training documents, where the training documents are processed as a group. The goal is to *classify* each document of a set of *new* documents as *relevant or nonrelevant.*

Information Filtering[2] *Interactive* training using an *asynchronous dynamic stream*

[1]See Chapter 4 for a discussion of *batch-oriented* Bayesian probabilistic methods and binary classification.

[2]See Section 4.5 and Chapter 5 for a discussion of *incremental* Bayesian probabilistic methods and information filtering.

of *unlabeled* documents, where each document is processed individually, as it arrives, and a *binary decision* made whether or not it is relevant to the user's information need (only documents previously processed may influence the decision); proposed relevant documents are forwarded to the user who provides relevance feedback which is used as training information. The goal is to *classify* each newly arriving document as *relevant or nonrelevant.*

Ad hoc Retrieval *Batch-oriented* training using a *static collection* of user-provided relevant/nonrelevant-*labeled* training documents, where the training documents are processed as a group. The goal is to numerically *rank* all documents of a set of *new* documents according to their relevance.

Information Routing *Multiple independent* information filtering processes operating in parallel on one stream of unlabeled documents, where each process is assigned a *unique* user information need; each arriving document is simultaneously tested for relevance against several independent user needs and is routed to the respective user, if it is judged as relevant to that user.

3.1 Term-Weighting

Term-weighting models are based on the association of weights with the terms (noncommon words or word-stems[3]) occurring in a topic or document, for the purpose of quantifying the "importance" of each term in the topic or document. As discussed in Section 1.3, given an enumerated vocabulary $V = \{t_1, t_2, ..., t_{|V|}\}$, the "bag of words" assumption allows a document d to be represented by a *weight vector* $w^d = (w_1, w_2, \ldots, w_{|V|})$, where each w_i is either (1) zero or one, indicating the *absence or presence* in d of term $t_i \in V$, or (2) a non-negative integer, indicating the *frequency* in d of term $t_i \in V$, $i = 1, \ldots, |V|$. Depending on the type of information access model, the document weight vector w^d may

[3]See Section 3.4.

be transformed into an *adjusted weight vector* consisting of *real-valued* term-weights[4]. Similarly, a topic t will have an associated topic weight-vector w^t. The ultimate purpose of such term-weighting is to determine the "similarity" between a document and a given topic, through (1) *nonparametric* approaches: computation of a *similarity metric* between the document and the topic, or (2) *parametric* approaches: construction of *probabilistic models* of relevant and nonrelevant documents (relative to the topic). Such similarities are used either for *ranking* of documents (as in ad hoc retrieval) or for relevance *classification* of documents relative to some threshold value (as in information filtering).

3.2 Nonparametric Information Access Models

Nonparametric information access models are established using *heuristics* that are based on commonsense observations of text documents and document collections. Salton and Buckley [Sal87] provide a comprehensive overview of common heuristic approaches to information access. The widely-used nonparametric Vector Space Model (VSM) [Sal83] adopts as a heuristic the assumption that a high degree of importance should be assigned to those terms that occur *frequently* in only a *few* documents of a collection. That is, the importance of each term t_i in a document d is (1) proportional to its frequency tf_i in d, and (2) inversely proportional to the total number of documents df_i in the collection that contain t_i. In the literature, this is referred to as *tf.idf* term-weighting:

$$w_i^{tf.idf} = tf_i \log \frac{N}{df_i}, \tag{3.1}$$

where $i = 1, \ldots, |V|$ and N is the total number of documents in the collection.

Given a document d with weight vector w^d and a topic t with weight vector w^t, the *similarity* measure $sim(d, t)$ between d and t is calculated as either the linear *inner*

[4]See Section 3.2.

product between w^d and w^t

$$sim(d,t) = \sum_{i=1}^{N} w_i^d w_i^t, \tag{3.2}$$

or the nonlinear *cosine correlation*

$$sim(d,t) = \frac{\sum_{i=1}^{N} w_i^d w_i^t}{||w^d||\,||w^t||}, \tag{3.3}$$

where $||w^d|| = \sqrt{\sum_{i=1}^{N}(w_i^d)^2}$ and $||w^t|| = \sqrt{\sum_{i=1}^{N}(w_i^t)^2}$.

Given a user topic of interest, nonparametric information-access models incorporate these topic/document similarity calculations in a manner that depends on the nature of the specific information-access task: (1) batch-oriented classification and interactive filtering tasks establish individual document *relevance* based on whether or not the document's similarity score exceeds some arbitrarily chosen *threshold value*; (2) the ad hoc retrieval task *ranks* documents according to their similarity scores.

3.2.1 Relevance Feedback

The above similarity measures are static calculations, given a topic t (also called user-profile) and its weight vector w^t. Provided with a set of labeled training documents, it is possible to improve the accuracy of nonparametric information access by adjusting the topic (user-profile) weight vector w^t in an incremental and cumulative manner: as each training document d is encountered that is sufficiently similar to topic (user-profile) t, t's weight vector w^t is *updated* to reflect information contained in the "similar" document d. In this manner, called *relevance feedback*, the topic (user-profile) weight vector w^t is incrementally improved and becomes more representative of the user's information need than the user's initial topic description. Negative information may also be incorporated into relevance feedback (i.e., information from documents dissimilar to t may be used to negatively weight w^t). In addition, *topic* or *query expansion* may be achieved by adding to t those terms that exist in relevant training documents but do not currently exist in t.

The current state-of-the-art for *batch-oriented* nonparametric relevance feedback is the *Rocchio* algorithm [Roc71], augmented with Dynamic Feedback Optimization (DFO) [Buc95]. The Rocchio algorithm, applied to the binary classification task, *adjusts* the weight of each individual topic term by adding to or subtracting, from its original value, the weight of the term in each relevant or nonrelevant training document, respectively (subject to weighting parameters that control the *relative* impact of the original user-profile weight vector w^t, the relevant training documents, and the nonrelevant training documents).

Interactive nonparametric relevance feedback algorithms employ *learning rate* mechanisms that allow gradual learning of user-profiles *over time*; for example, (1) *Rocchio* implemented as an incremental algorithm [All96], (2) the *LMS* or *Widrow-Hoff* algorithm [Wid85], and (3) the *exponentiated-gradient* (EG) algorithm [Kiv94]. Empirical evaluations of the batch-Rocchio, Widrow-Hoff, and EG algorithms are presented in [Lew96].

3.3 Parametric Information Access Models

Parametric information access models treat documents and terms as data sample instances that follow some underlying theoretical *probability distribution.* The goals of parametric modeling include (1) establishing the *parameters* of the underlying probability distribution through statistical data sampling, and (2) calculating the *probability* that the random variable modeled by the distribution will assume a specified value.

3.3.1 Ad Hoc Retrieval

Parametric ad hoc retrieval requires that the probability $P(relevant \mid d\,; t)$ be determined for every document d in a given *static* document collection, for a given topic t; this value represents each document's absolute probability of relevance with respect to t and may be used to rank the documents in order of relevance.

The batch-oriented Binary Independence Model (BIM) proposed by Robertson and Sparck Jones [Rob76] has been the most influential parametric model in the area of information access. Given vocabulary V, the BIM vector-representation of individual documents consists of *boolean-valued* term-weights $x_i \in \{0,1\}, i=1,\ldots,|V|$, that indicate the absence or presence of each vocabulary term $t_i \in V$ in a document. Given a static collection of labeled training documents and a document class c_j, each vocabulary term t_i is assigned a probability $p_{ij} = P(x_i = 1 \mid c_j) = n_{x_i=1}/n_{c_j}$, where n_{c_j} is the number of documents of class c_j in the collection, and $n_{x_i=1}$ is the number of these n_{c_j} documents with $x_i = 1$ (i.e., that contain term t_i); p_{ij} represents the probability that term $t_i \in V$ exists in an arbitrary chosen document of class c_j. If there are only two classes: $c_1 = relevant$ and $c_2 = nonrelevant$, then [Rob76] and [Lew98] provide the following formula for calculating $P(relevant \mid d\,; t)$:

$$P(relevant \mid d\,; t) \;=\; \sum_{i=1}^{|V|} x_i \, log \frac{p_{i1}(1-p_{i2})}{(1-p_{i1})p_{i2}}. \tag{3.4}$$

Despite its historical influence, the BIM has at least two shortcomings: (1) it ignores the *frequencies* of terms in documents (the more frequently an "important" term occurs in a document, the greater its predictive value), and (2) it ignores document length (an "important" term that occurs in a short document should possess more predictive value than that of the same term occurring in a long document).

3.3.2 Information Filtering

Parametric information filtering requires that *both* of the probabilities

$$P(relevant \mid d\,; t) \quad \text{and} \quad P(nonrelevant \mid d\,; t)$$

be calculated for each arriving document d in an on-line data stream, for a given topic t; the maximum value of these two probabilities determines the *most probable classification* of d. Only those documents classified as "relevant" are removed from the document stream and presented to the user.

There has been much research on *batch-oriented* parametric document classification; see, for example, [McC98] and [Lew98]; however, we are unaware of any studies on *sequential* parametric document classification (i.e., parametric information filtering). The latter area is the focus of the current work. The research closest in spirit to the current work, although addressing ad hoc retrieval, is outlined in [Kei97]: it combines a Bernoulli (binomial) parametric document model with incremental Bayesian methods (the Beta conjugate family) as a means of achieving *incremental* relevance feedback.

3.3.3 Relevance Feedback

Relevance feedback and query expansion are commonly applied to *batch-oriented* parametric information access environments; for example, Harman [Har92] provides a brief survey of such research in the area of parametric ad hoc retrieval, citing work by Robertson, Croft and Harper, Harper and van Rijsbergen, and Wu and Salton. An interesting result of this work is that the performance of query expansion under parametric models tends to be heavily dependent on the specific document collection used for empirical testing.

Bookstein [Boo83] provides perhaps the first example of *sequential* relevance feedback within a statistical decision theoretic framework, where retrieval judgements of documents are made individually for each document, and feedback is part of the model *itself*[5]. Aalbersberg [Aal92] provides further evidence (although in the context of non-parametric models) of the value of *incremental* relevance feedback, where each feedback iteration occurs after the retrieval of a *single* document and before the retrieval of the subsequent document: such incremental feedback yielded better performance than the widely used batch-Rocchio and Ide relevance feedback algorithms.

[5]Earlier parametric feedback methods used various *ad hoc* techniques based on word frequencies to estimate the parameters of the probabilistic model.

3.4 Controlling Dimensionality

The size of an English vocabulary that includes scientific, technical, and business terms, and acronyms may easily exceed 50,000 terms. It is therefore clear that a document-vector feature space that is based on such a vocabulary will have an enormous dimensionality; some control over this dimensionality is therefore required. The following are three methods that have been adopted in the current work to control document-feature dimensionality.

3.4.1 Elimination of Stop Words

Many words in the English language possess no inherent topical information. For example, articles, pronouns, conjunctions, and prepositions provide structure to the language but provide no content value. In addition, very common words and contractions provide little information value, e.g., "million", "company", "didn't", "couldn't", "Co", "Corp", etc.. Therefore, a *stop word* list has been created that contains approximately 600 articles, pronouns, conjunctions, prepositions, contractions, and various common words. Documents are preprocessed against this list and and document terms appearing in this list (and punctuation and numbers) are removed from each document before it is placed into lower-case form and converted into its vector representation.

3.4.2 Word Stemming

Under the "bag of words" document representation scheme discussed in Section 1.3, suffixes applied to a given base English word generally add no significant extra topical information. For example, the words "abduct", "abducts", "abducted", "abduction", "abducting", and "abductor", although different parts of speech, possess little differentiation of content under this representation scheme. Therefore, after elimination of stop words, punctuation, and numbers, the remaining words are stemmed using the well-

known stemming algorithm by Porter [Por80].

3.4.3 Zipf's Law

English words have varying frequencies of occurrence in document collections. For example, the word "hostage" would presumably occur very frequently in contemporary news-document collections; however, there are many obscure words in the English language that will occur very infrequently in such document collections. Sahami [Sah98] cites work by Zipf and van Rijsbergen that provide empirical evidence that words occurring only once or twice in an entire document *collection* account for approximately one-half of the total unique terms in the collection, but have little resolving power between documents. Therefore, words that have very low frequencies of occurrence in a collection have been omitted from document-vector representations in the current work. This step occurs immediately after the elimination of stop words in the case where documents are provided as a static collection. When documents are provided incrementally, one at a time (as in information filtering), a cumulative frequency count is maintained for each unique word that has appeared. At periodic intervals, say every fifty days or every 10,000 documents, words that have occurred only once or twice are added to a *low-frequency* stop list; words appearing in this list are then excluded from the document-vectors of future documents.

Chapter 4

Bayesian Learning and Document Classification

4.1 Introduction

Bayesian learning methods[1] take a **probabilistic approach** to the task of learning which of several alternative hypotheses best explains observed phenomena; probabilistic methods provide a strictly quantitative means of weighing all the available evidence that supports alternative hypotheses. Under this approach, for a given set of observed data, each possible hypothesis is assigned a probability that *it* is the "best" hypothesis that explains the data. If the **best hypothesis** is assumed to be the **most probable hypothesis** (the hypothesis with the highest probability, called the *maximum a posteriori* or MAP hypothesis) then probability theory can be used to determine such a hypothesis. Given a set of observed data, then the probability that a given hypothesis is the best hypothesis may be calculated using the following information:

[1] See [Mit97] Chapter 6 for a general introduction to Bayesian learning, [McC98] for a comparison of event models for batch-oriented Bayesian text classification, and [Mar89], [Rob94], and [Gel95] for theoretical coverage of empirical Bayesian analysis.

- The *prior* probability that the hypothesis is the best hypothesis, before making data observations (determined using background knowledge of the problem domain or from assumptions of the underlying probability distributions of the prior).

- The probability that the *observed data* will actually occur in a world where such a hypothesis holds.

Bayesian methods have the distinct advantage that each observed data element can *incrementally* increase or decrease the estimated probability that a hypothesis is consistent with the hypothesis. This offers more flexibility over certain other machine learning methods that eliminate a hypothesis if it is found to be inconsistent with any single training example[2]. Bayesian methods are therefore more tolerant of "noisy" data (e.g., outlying data) than other common machine learning methods. However, Bayesian methods have disadvantages: they require large amounts of initial probabilistic information (which may be very difficult to obtain), and they require significant amounts of computational resources (which may make their use impractical for *large* hypothesis spaces).

4.2 Bayes Theorem

Bayes theorem provides a direct method for calculating the most *probable hypothesis* h_{MAP}, given a set of observed data D and background knowledge of the *prior probability* $P(h)$ of each hypothesis h in a hypothesis space H. Let $P(D)$ represent the probability that the data would be observed without any knowledge of which hypothesis is correct. Let $P(D|h)$ represent the probability that the data would be observed, given that hypothesis h is the correct hypothesis. The Bayesian learning task described in Section 4.1 involves determining, for *each* hypothesis h the *posterior probability* $P(h|D)$, i.e., the probability that hypothesis h is the correct hypothesis, given the observed data D. The

[2]For example, the *Find-S* and *Candidate-Elimination* concept learning algorithms. See [Mit97] for details of these, and many other machine learning methodologies.

following formula, called *Bayes theorem*, provides the means of calculating $P(h|D)$:

$$P(h|D) = \frac{P(h)P(D|h)}{P(D)}.$$

Any hypothesis which has a *maximum* such value, over all hypotheses $h \in H$ given the observed data D, is a *most probable* or *maximum a posteriori* (MAP) hypothesis (i.e., a hypothesis that *best* explains the observed data). A hypothesis $h_{MAP} \in H$ is a MAP hypothesis provided

$$\begin{aligned} h_{MAP} &= \underset{h \in H}{\text{argmax}}\ P(h|D) \\ &= \underset{h \in H}{\text{argmax}}\ \frac{P(h)P(D|h)}{P(D)} \\ &= \underset{h \in H}{\text{argmax}}\ P(h)P(D|h) \quad \text{as } P(D) \text{ is a constant, independent of } h. \end{aligned}$$

4.3 Binary Classification

Binary classification is defined to be the learning problem where the hypothesis space H is defined such that each hypothesis $h \in H$ is a **boolean-valued function** of features of data items of a data domain D. Each data item $d \in D$ is described by a conjunction of the values of n predefined-features (attributes). The boolean-value generated by a hypothesis is with respect to some *concept* pertaining to an individual data item and can take on any value from a set C, where $|C| = 2$. For example, {*yes, no*}, {*rainy, sunny*}, {*relevant, nonrelevant*}, etc.. Provided with a set $D_T \subset D$ of positive and negative training examples and an actual concept value $c(d)$ for each $d \in D_T$, the classifier attempts to determine a hypothesis function h such that $h(d) = c(d)$ for all training examples $d \in D_T$. Once the hypothesis has been determined, new previously unseen data instances may be classified. In Bayesian learning, this translates to determining the *most probable* or *maximum a posteriori* (MAP) hypothesis, as discussed in Section 4.2.

4.4 Naive Bayes Classifier

Once the most probable hypothesis h_{MAP} has been found, given the training data set $D_T \in D$ and a **new** data instance $d \in D$ is presented, h_{MAP} can be used to find the most probable *classification* c_{MAP} of d as follows: $c_{MAP} = h_{MAP}(d) \in C$. A less computationally expensive method of probabilistic classification, the **naive Bayes** "frequency counting" classifier, is discussed next. This classifier avoids searching through a space of possible hypotheses by counting the frequencies of various data combinations within the training data.

Assume that data instances are represented by n features or attributes where each attribute i, $i = 1 \ldots n$, may take values from an attribute set X_i, and the new data instance d is represented by the n-tuple $(x_1, x_2, \ldots, x_n)$ of *attribute values* $x_i \in X_i$. Then the **most probable classification** $c_{MAP} \in C$ of d may be calculated as follows:

$$
\begin{aligned}
c_{MAP} &= \underset{c_j \in C}{\text{argmax}}\ P(c_j|d) && (4.1)\\
&= \underset{c_j \in C}{\text{argmax}}\ \frac{P(c_j)P(d|c_j)}{P(d)} \text{ by Bayes theorem} && (4.2)\\
&= \underset{c_j \in C}{\text{argmax}}\ \frac{P(c_j)P(x_1, x_2, \ldots, x_n|c_j)}{P(x_1, x_2, \ldots, x_n)} && (4.3)\\
&= \underset{c_j \in C}{\text{argmax}}\ P(c_j)P(x_1, x_2, \ldots, x_n|c_j) \text{ as } P(x_1, x_2, \ldots, x_n) \text{ is constant.} && (4.4)
\end{aligned}
$$

In this formulation, the $P(c_j)$ may be calculated by counting the frequencies of occurrence of each possible classification $c_j \in C$ within the training data. However, the values of *all* the $P(x_1, x_2, \ldots, x_n|c_j)$ terms must be calculated. But there are usually an astronomical number of possible values for $d = (x_1, x_2, \ldots, x_n)$, so the number of such terms is immense. Let $D_{Unique} \subset D_T$ be set of unique instances of training data; then $|D_{Unique}| \ll |D_T|$, i.e., huge data training sets would be required to ensure that each unique training instance occurs a sufficient amount of times so that frequency counts would be reliable.

If the simplifying assumption is made that each of the n attributes is *conditionally independent*, given the value of c_j, then $P(x_1, x_2, \ldots, x_n|c_j) = P(x_1|c_j)P(x_2|c_j) \cdots P(x_n|c_j) =$

$\prod_i^n P(x_i|c_j)$. The number of distinct $P(x_i|c_j)$ terms that must be estimated from the training data is just $n \cdot |C|$ (much less than the number of $P(x_1, x_2, \ldots, x_n|c_j)$ terms which would have to be estimated without an attribute independence assumption). If $c_{NB} \in C$ denotes the *target concept* output by the **naive Bayes classifier** for a new data instance $d = (x_1, x_2, \ldots, x_n)$, then substituting $\prod_i^n P(x_i|c_j)$ for $P(x_1, x_2, \ldots, x_n|c_j)$ in equation 4.4 yields

$$c_{NB} = \underset{c_j \in C}{\text{argmax}} \; P(c_j) \prod_i^n P(x_i|c_j). \tag{4.5}$$

Both $P(c_j)$ and $P(x_i|c_j)$ are estimated using frequency counts of training data instances. Given a training data set D_T and a specific concept value $c_j \in C$, let n_{c_j} be the number of data instances in D_T that have c_j as their concept value, and let n_{x_i} be the number of these n_{c_j} data instances that have value x_i for attribute i. Then

$$P(c_j) = \frac{n_{c_j}}{|D_T|}, \quad \text{where } |D_T| = \sum_{j=1}^{|C|} n_{c_j} \tag{4.6}$$

$$P(x_i|c_j) = \frac{n_{x_i}}{n_{c_j}}. \tag{4.7}$$

Equations 4.6 and 4.7, however, are appropriate for use only in *batch-oriented* supervised learning environments where

- *all* training and test data instances are available at one time;
- each training data instance possesses a specific known concept target value c_j.

New methods are required to adapt naive Bayes classification to *interactive* learning environments where

- data instances arrive over a period of time;
- concept target values c_j are *not* known for arriving data instances.

The next section and Chapter 5 will introduce a new approach for calculating the $n \cdot |C|$ terms $P(x_i|c_j)$ in such interactive learning situations.

4.5 Adaptive Naive Bayes Document Classification

The naive Bayes formulation may be applied to document classification in a number of ways. We will be concerned here with binary classification where a document may be **Relevant** or **Nonrelevant** to some particular **topic** t of interest, i.e., $C = \{R, N\}$ with respect to t. The document representation scheme discussed in Chapter 3 will be adopted where each document has an associated weight vector that is composed of the frequencies of occurrences of the (non-common) words that it contains. That is, given an enumerated vocabulary $V = \{t_1, t_2, ..., t_{|V|}\}$, a document d may be represented as a weight vector of term frequencies $w^d = (w_1, w_2, \ldots, w_{|V|})$, where w_i is a non-negative integer indicating the frequency in d of term $t_i \in V$, $i = 1, \ldots, |V|$.

To obtain the naive Bayes classification of a given document d, the value $R^j \in \{R, N\}$ must be determined that maximizes $P(R^j|d)$, i.e., the most probable classification of d. From Equation 4.5, this requires calculation of the $n \cdot |C| = |V| \cdot |C| = |V| \cdot 2$ terms $P(x_i|R^j) = P(t_i|R^j)$. Thus Equations 4.6 and 4.7 must be applied to training data to derive point estimates of $P(t_i|R)$ and $P(t_i|N)$ for all terms $t_i \in V$; these are estimates of the probability of each term t_i occurring in a document d that is relevant or nonrelevant to the current topic, respectively.

However, as discussed above, interactive learning environments do not allow the $P(t_i|R^j)$ parameter values to be calculated using the batch-oriented frequency-count method of Equations 4.6 and 4.7 (because document arrival is spread out over time, and document relevance $R^j \in \{R, N\}$ is available only for those arriving documents that the system selects and presents to the user for a relevance judgement). The absence of point estimates for these probabilities leads to the need for an **incremental** Bayesian approach: prior **distributions** are assigned to the parameters, and *sequential data sampling* is used to dynamically update the prior distributions, yielding posterior distributions for the parameters. This may be achieved through the use of the model's *conjugate prior* distribution, which will be discussed in Chapter 5. The prior distributions embody any

prior knowledge that may be available about the relevance and nonrelevance of documents to a given topic. Prior relevance knowledge may be obtained by observing the occurrences of terms in the text of the topic description. The intuition behind this assumption is the fact that documents relevant to a given topic are likely to possess many of the same terms as the topic description, and in roughly the same proportions as in the topic description. Prior nonrelevance knowledge is generally unavailable, therefore the use of uniform (equal) nonrelevance priors is justified.

The next consideration is the choice of the specific probability distribution for document representation. As discussed in Chapter 1, the *frequency of occurrence* of a topic term in a document is indicative of the possible relevance of the document to the topic. A natural probabilistic model that takes into account frequencies is the *multinomial model*: a document d of length N may be considered as resulting from N word events or draws from a vocabulary V. If the naive Bayes assumption is again made that the probability of each word event is independent of the position (draw) of the word, and of other word events of the document, then each document d can be said to result from N independent draws on a $|V|$-valued multinomial variable.

Chapter 5 discusses in more detail the application of the multinomial model to adaptive Bayesian document classification or filtering, and provides experimental verification of the validity and usefulness of this approach.

Chapter 5

Adaptive Bayesian Information Filtering: Theory

5.1 Introduction

A central problem faced by **interactive** information filtering systems is the difficulty of creating accurate representations of a user's information need: users have difficulty verbalizing their needs in a concise manner but are easily able to verify which documents satisfy their needs and which do not, when presented with such documents. A user typically initializes such systems with a concise query, topic description, or set of keywords. Such initial information is normally only an approximate description of the users *actual* information need. This chapter outlines how Bayesian learning may be applied in an interactive information filtering environment in an **adaptive** manner that allows a user's information need (or user-profile) to be *learned* over a period of time by incorporating user feedback into the learning process[1]. This may be contrasted with, for example, batch-oriented learning where users typically provide large amounts of labeled training

[1] See [Mar89], [Rob94], and [Gel95] for theoretical coverage of empirical Bayesian analysis; the current chapter illustrates how this theory may be adapted to an interactive information filtering domain.

examples at a single point in time and learning occurs all at once.

After initialization, the system begins filtering arriving documents using the initial user-profile as a guide, and presents to the user only those documents that "match" the user-profile. The user reviews each presented document in turn and provides the system with feedback of the *actual* relevance of each presented document. The system adjusts its user-profile based on this feedback, and is able to learn from both its successes *and* its failures. The initial profile gradually evolves into a more accurate user-profile over time.

By constantly *interacting* with the user, the process is potentially able to achieve a more accurate representation of user information needs than is possible through batch-oriented supervised learning; at any time the user may change her information requirements in subtle ways, and the interactive system will adapt to such changes.

However, such an interactive environment imposes the constraint that the system must generalize a user's information need based only on user feedback from documents that the system has seen up to a point in time. That is, the system must pass a judgement on a document *before* it sees all future documents. Thus, such a system does not possess the learning efficiency of batch-oriented learning systems where all positive and negative training documents are available at one point in time, and model parameters are optimized *before* the system is asked to classify new documents.

5.2 Multinomial Document Representation

Section 4.5 provided the intuition behind the usefulness of the multinomial distribution for modeling document word frequency information in an *interactive* document classification or filtering environment. Given a vocabulary V and the naive Bayes assumption that the occurrence of a term in a document is *independent* of (1) the term's position in the document, and (2) all other term occurrences in the document (including multiple occurrences of the specified term), then each document d may be considered drawn from

a multinomial distribution of terms $t_i \in V$, $i=1,\ldots,|V|$, through $|d|$ independent trials. Let $f_i \geq 0$ denote the count of the number of times term $t_i \in V$ occurs in document d; then the *probability of occurrence* of d, given its *relevance value* $R^j \in \{R, N\}$ to some user-topic of interest, is provided by the multinomial distribution:

$$P(d \mid \vec{\theta}; R^j) = P(f_1, f_2, \ldots, f_{|V|} \mid \vec{\theta}; R^j) = \frac{|d|!}{\prod_{i=1}^{|V|} f_i!} (\theta_{R_1^j})^{f_1} (\theta_{R_2^j})^{f_2} \cdots (\theta_{R_{|V|}^j})^{f_{|V|}} \quad (5.1)$$

where $|d| = \sum_{i=1}^{|V|} f_i$, and $\theta_{R_i^j} = P(t_i|R^j)$ is the probability of term $t_i \in V$ being selected in a trial, given the relevance value R^j of d to the current topic, $i = 1, \ldots, |V|$.

The *most probable classification* (relevant or nonrelevant) of d, with respect to a given topic, is determined by the maximum of $P(R \mid d)$ and $P(N \mid d)$, where

$$\begin{aligned} P(R \mid d) &= \frac{P(R)\, P(d|R)}{P(d)} \text{ by Bayes theorem} && (5.2)\\ &= \frac{P(R)\, \frac{|d|!}{\prod_{i=1}^{|V|} f_i!} \prod_{i=1}^{|V|} \theta_{R_i}^{f_i}}{P(d)} \text{ by (5.1) where } R^j = R && (5.3)\\ &= c \cdot P(R) \prod_{i=1}^{|V|} \theta_{R_i}^{f_i} && (5.4) \end{aligned}$$

and

$$\begin{aligned} P(N \mid d) &= \frac{P(N)\, P(d|N)}{P(d)} \quad \text{by Bayes theorem} && (5.5)\\ &= \frac{P(N)\, \frac{|d|!}{\prod_{i=1}^{|V|} f_i!} \prod_{i=1}^{|V|} \theta_{N_i}^{f_i}}{P(d)} \text{ by (5.1) where } R^j = N && (5.6)\\ &= c \cdot P(N) \prod_{i=1}^{|V|} \theta_{N_i}^{f_i}, && (5.7) \end{aligned}$$

where $c = \frac{|d|!}{P(d) \prod_{i=1}^{|V|} f_i!}$ is a scaling constant that ensures that the two conditional probabilities sum to one, for a given document d.

Point estimates cannot be made of the $2|V|$ multinomial parameters $\theta_{R_i} = P(t_i|R)$ and $\theta_{N_i} = P(t_i|N)$ in an interactive on-line document filtering environment[2]. However, these

[2]The reason for this is the same as that discussed in Section 4.5: such point estimates cannot be derived when data sampling occurs over an extended period of time, as occurs in interactive environments.

parameters may be represented through **probability distributions** $p(\theta_{R_i})$ and $p(\theta_{N_i})$, respectively, $i = 1, \ldots, |V|$. Such distributions can be established through a combination of (1) prior knowledge of their form, and (2) sequential document data sampling. The next section will discuss in detail the manner in which these distributions are determined.

5.3 Conjugate Priors and Learning Model Parameters

The method of *conjugate priors*[3] may be used to provide *incremental Bayesian learning* of the parameters of a given probabilistic model in the case where

- data sampling is *sequential* in nature (e.g., from on-line data sources);
- *prior* information of the model's parameters is available.

A family $\mathcal{C}$ of probability distributions on parameter θ is said to be *conjugate* (or closed under sampling) if, for every (prior) distribution $\pi(\theta)$ in $\mathcal{C}$, the *posterior* distribution $\pi(\theta|x)$ also belongs to $\mathcal{C}$ [Rob94]. That is, if a data sample x is obtained from data distributed according to probability distribution $p(x|\theta)$, written $x \sim p(x|\theta)$, and π represents the *conjugate prior* of p, then

$$x \sim p(x|\theta) \text{ and } \pi(\theta) \in \mathcal{C} \implies \pi(\theta|x) \in \mathcal{C}.$$

If the family of priors $\mathcal{C}$ is parameterized, then for any $\pi \in \mathcal{C}$, π has has the desirable property that switching from prior to posterior distribution is reduced to simple arithmetic updating of the parameters of π. Thus conjugate priors are a mathematically convenient means of obtaining posterior distributions from prior distributions in an *incremental manner*, as data samples arrive over time.

Once a probability distribution $\mathcal{D}$ has been postulated as a model of the manner in which specified data is generated, the *general procedure* outlined in Figure 5.1 illustrates how the above *incremental* Bayesian approach may be used to learn the parameters of $\mathcal{D}$.[4]

[3] See [Mar89], [Rob94], or [Gel95].

[4] The procedure of Figure 5.1 is an expanded version of that presented in [Cas97], p. 48.

1. Determine $\mathcal{D}$'s conjugate family of priors $\mathcal{C}$.
2. Approximate a prior $\mathcal{C}$ *distribution* on each of $\mathcal{D}$'s n parameters and initialize the means and variance of each of these distributions by setting $\mathcal{C}$'s n hyperparameters, using
   - frequency counts from prior data, or
   - uniform (noninformative) priors.
3. Obtain a data sample d from data distributed according to $\mathcal{D}$.
4. Increment $\mathcal{C}$'s n hyperparameters with d, yielding n posterior $\mathcal{C}$ distributions.
5. Repeat from Step 3 until data sampling has been completed.
6. Calculate the mean of each of $\mathcal{C}$'s n posterior distributions.
7. Assign the n posterior means as point estimates of $\mathcal{D}$'s n parameters.

Figure 5.1: General procedure for *incremental* Bayesian learning.

The following subsections will elaborate on this general procedure for the specific document filtering application; that is, through incremental document sampling, *probability distributions* and hence point estimates of the parameters of the relevant- and nonrelevant-document multinomial models may be established, with respect to a given topic. A detailed algorithm for this procedure is presented in Figure 5.2 of Section 5.6.

5.3.1 Dirichlet Conjugate Family

A *multinomial* distribution of n parameters has as its conjugate the *Dirichlet* distribution $\mathcal{DIR}_n^{R^j}(\alpha_1, \alpha_2, \ldots, \alpha_n)$ with n hyperparameters $\alpha_1, \alpha_2 \ldots, \alpha_n$ which are normally initialized with frequency counts obtained from *prior* information of the specific multinomial distribution being modeled, relative to *given* information R^j of the concept being modeled. If the multinomial distribution has parameters $\vec{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$, then the Dirichlet density function for $\vec{\theta}$ is

$$p(\vec{\theta} \mid \alpha_1, \ldots, \alpha_n) = \frac{\Gamma(\alpha_1 + \alpha_2 + \cdots + \alpha_n)}{\Gamma(\alpha_1)\Gamma(\alpha_2)\cdots\Gamma(\alpha_n)} \theta_1^{\alpha_1 - 1} \theta_2^{\alpha_2 - 1} \ldots \theta_n^{\alpha_n - 1}, \tag{5.8}$$

where $\Gamma(\alpha_i) = \int_0^\infty t^{\alpha_i - 1} e^{-t} dt$ is the *gamma* function, and $\sum_{i=1}^{n} \theta_i = 1$.

For a given user-topic, Equations (5.4) and (5.7) of Section 5.2 indicate that *two* multinomial document models are required: a model of document *relevance* and a model of document *nonrelevance*. Therefore, letting $n = |V|$ and $R_j \in \{R, N\}$ in Equation (5.8) leads to the following Dirichlet distributions over the multinomial parameters $\vec{\theta}_R = P(\vec{t} \mid R)$ and $\vec{\theta}_N = P(\vec{t} \mid N)$, respectively:

$$p(\vec{\theta}_R) = \mathcal{DIR}^R_{|V|}(\alpha_{R_1}, \ldots, \alpha_{R_{|V|}}) \propto \prod_{i=1}^{|V|} \theta_{R_i}^{\alpha_{R_i} - 1} \tag{5.9}$$

$$p(\vec{\theta}_N) = \mathcal{DIR}^N_{|V|}(\alpha_{N_1}, \ldots, \alpha_{N_{|V|}}) \propto \prod_{i=1}^{|V|} \theta_{N_i}^{\alpha_{N_i} - 1}. \tag{5.10}$$

5.3.2 Initialization of Prior Distributions

For **relevance** priors, the information implicitly provided by topic description word frequencies may be used as prior information to provide objective priors with which to initialize the hyperparameters α_{R_i} of $p(\vec{\theta}_R)$. In a given topic description T, the frequency of occurrence f_i of an individual term $t_i \in V$ in T, as a proportion of the length $|T|$ of T, may be considered as a rough approximation of the proportion of times that t_i will occur in "typical" documents that are relevant to T. Given vocabulary V, then for each term $t_i \in V$, $i = 1, \ldots, |V|$, the *relevance prior ess frequency-estimate* of t_i is

$$\alpha_{R_i} = f_i + \frac{ess_R}{|V|}, \tag{5.11}$$

where $f_i \geq 0$ is the frequency of occurrence of term t_i in topic T, and ess_R is a constant called the *relevance equivalent sample size* that represents ess_R *virtual samples* of data before actual data sampling begins. The relevance prior ess frequency-estimate α_{R_i} serves two important purposes:

1. A means of initializing *every* term $t_i \in V$ (even those with $f_i = 0$) with a *non-zero prior* probability of appearing in a relevant document, and

2. The *total* $\sum_{j=1}^{|V|} \alpha_{R_j}$ provides a *weighting* of our confidence of how well the *"prior"* proportions $\alpha_{R_i} / \sum_{j=1}^{|V|} \alpha_{R_j}$ estimate the relevance-model's parameters θ_{R_i}, *relative to*

the proportions from term-frequencies obtained during document *data sampling*[5].

The first item is justified because the number of *unique* terms n_T in T is such that $n_T \ll |V|$; setting α_{R_i} to zero for all $|V| - n_T$ terms that do *not* appear in T would understate the probabilities of these terms appearing in relevant documents (i.e., there will exist *many* vocabulary terms that may potentially occur in relevant documents in addition to the n_T unique terms in T). Also, setting α_{R_i} to zero for some $t_i \in V$ would result in $\theta_{R_i} = P(t_i|R) = 0$ in (5.4), if t_i does not occur in subsequent training documents. If such a t_i then occurs in a document d that is to be classified, the right-hand side of (5.4) would become zero, making a comparison between (5.4) and (5.7) meaningless.
The second item is justified because *both* the prior frequencies of (5.11) and the actual training data frequencies contribute to the determination of the final values of the hyperparameters. In the case of relevance priors, the actual frequencies obtained from subsequent (unlimited) document data sampling will have potentially *much* more influence in providing term relevance information than the prior frequencies. Hence, it is expected that ess_R will be a "small" positive integer; this intuition will be confirmed with empirical analysis in Chapter 6.

For **nonrelevance** priors, there is *no* prior information regarding which terms will be indicative of a nonrelevant document, with respect to a given topic; therefore, noninformative (uniform) priors are appropriate. For each term $t_i \in V$, $i = 1, \ldots, |V|$, the *nonrelevance prior ess frequency-estimate* of t_i is

$$\alpha_{N_i} = \frac{ess_N}{|V|}, \tag{5.12}$$

where ess_N is a constant called the *nonrelevance equivalent sample size.* The nonrelevance prior frequencies given by (5.12) intuitively should have much *more* influence in providing term nonrelevance information than *actual* nonrelevant document sampling (because the latter sampling represents only a very small example of *all possible* unseen nonrelevant

[5]See Section 5.3.4 for a discussion of parameter estimation.

documents). Hence, it is expected that ess_N will be a "large" positive integer, i.e., $ess_N \gg ess_R$; this intuition will also be confirmed with empirical analysis in Chapter 6.

5.3.3 Updating Distributions with Observed Data

As each document data observation d arrives, with $w^d = (f_1, f_2, \ldots, f_{|V|})$, where $f_i \geq 0$ represents the frequency of term $t_i \in V$ in d, interactive document filtering systems require that the system make an *immediate* judgement of d's relevance $R^j \in \{R, N\}$, with respect to the given topic T. Only if the system judges d as *relevant to* T, is d presented to the user for a final relevance judgement. The *user's* relevance judgement of d is then used to update *either* $p(\vec{\theta}_R)$ *or* $p(\vec{\theta}_N)$, depending on whether his judgement is "relevant" or "nonrelevant".

The initial system relevance judgment for each d is made by obtaining point estimates for $\theta_{R_i} = P(t_i|R)$ and $\theta_{N_i} = P(t_i|N)$, $i = 1, \ldots, |V|$, and then evaluating and comparing (5.4) and (5.7). The calculation of point estimates for θ_{R_i} and θ_{N_i} will be discussed below, in subsection 5.3.4.

Assume (1) the system has judged document $d = (f_1, f_2, \ldots, f_{|V|})$ as relevant to T and has presented d to the user, and (2) the user has provided the system with d's *true* relevance value $R^j \in \{R, N\}$. Then the frequencies $f_i \geq 0$ of occurrences of terms $t_i \in V$ in d, $1, \ldots, |V|$, are used to update the hyperparameters of *either* $p(\vec{\theta}_R)$ *or* $p(\vec{\theta}_N)$, depending on the value of R^j:

$$p(\vec{\theta}_R \,|\, d) = \mathcal{DIR}^R_{|V|}(\alpha_{R_1} + f_1, \ldots, \alpha_{R_{|V|}} + f_{|V|}) \propto \prod_{i=1}^{|V|} \theta_{R_i}^{\alpha_{R_i} + f_i - 1} \tag{5.13}$$

$$p(\vec{\theta}_N \,|\, d) = \mathcal{DIR}^N_{|V|}(\alpha_{N_1} + f_1, \ldots, \alpha_{N_{|V|}} + f_{|V|}) \propto \prod_{i=1}^{|V|} \theta_{N_i}^{\alpha_{N_i} + f_i - 1}. \tag{5.14}$$

Thus (5.13) and (5.14) represent the *posterior* distributions $p(\vec{\theta}_R \,|\, d)$ and $p(\vec{\theta}_N \,|\, d)$, given observed document d. These become the *prior* distributions $p(\vec{\theta}_R)$ and $p(\vec{\theta}_N)$ for the next iteration. In this manner, *incremental* Bayesian updating of prior to posterior distributions is possible.

5.3.4 Parameter Estimation

As data sampling progresses, the Dirichlet distributions $p(\vec{\theta}_R \mid d)$ and $p(\vec{\theta}_N \mid d)$ gradually evolve into accurate posterior distributions that describe the model's *individual* parameters $\theta_{R_i} = P(t_i|R)$ and $\theta_{N_i} = P(t_i|N)$, $i = 1, \ldots, |V|$. At any time, *point estimates* $\hat{\theta}_{R_i}$ and $\hat{\theta}_{N_i}$ for these parameters may be obtained by calculating the $2|V|$ *means* of the Dirichlet posterior distributions as follows:

$$\hat{\theta}_{R_i} = \hat{P}(t_i|R) = E[p(\theta_{R_i}|d)] = \frac{\alpha_{R_i}}{\sum_{j=1}^{|V|} \alpha_{R_j}} \tag{5.15}$$

$$\hat{\theta}_{N_i} = \hat{P}(t_i|N) = E[p(\theta_{N_i}|d)] = \frac{\alpha_{N_i}}{\sum_{j=1}^{|V|} \alpha_{N_j}}, \tag{5.16}$$

for each term $t_i \in V$, $i = 1, \ldots, |V|$.

5.4 Most Probable Document Classification

Substituting the point estimates (5.15) and (5.16) of the multinomial parameters $\theta_{R_i} = P(t_i|R)$ and $\theta_{N_i} = P(t_i|N)$ in (5.4) and (5.7) respectively, yields the following:

$$P(R \mid d) = c \cdot P(R) \prod_{i=1}^{|V|} \hat{\theta}_{R_i}^{f_i} \tag{5.17}$$

$$P(N \mid d) = c \cdot P(N) \prod_{i=1}^{|V|} \hat{\theta}_{N_i}^{f_i}. \tag{5.18}$$

For convenience, it is useful to express (5.17) and (5.18) in "log odds" form:

$$\log \frac{P(R \mid d)}{P(N \mid d)} = \log \frac{P(R) \prod_{i=1}^{|V|} \hat{\theta}_{R_i}^{f_i}}{P(N) \prod_{i=1}^{|V|} \hat{\theta}_{N_i}^{f_i}} \tag{5.19}$$

$$= \log \left(\frac{P(R)}{P(N)} \cdot \prod_{i=1}^{|V|} \left(\frac{\hat{\theta}_{R_i}}{\hat{\theta}_{N_i}} \right)^{f_i} \right) \tag{5.20}$$

$$= \log \frac{P(R)}{P(N)} + \sum_{i=1}^{|V|} \log \left(\frac{\hat{\theta}_{R_i}}{\hat{\theta}_{N_i}} \right)^{f_i} \tag{5.21}$$

$$= \log \frac{P(R)}{P(N)} + \sum_{i=1}^{|V|} f_i \cdot \log \frac{\hat{\theta}_{R_i}}{\hat{\theta}_{N_i}} \tag{5.22}$$

$$= \log \frac{P(R)}{P(N)} + \sum_{i=1}^{|V|} f_i \cdot \log \frac{\hat{P}(t_i|R)}{\hat{P}(t_i|N)}. \tag{5.23}$$

The maximum of $P(R \mid d)$ and $P(N \mid d)$, and hence d's most probable classification, may now be established by noting the sign of $\log \frac{P(R|d)}{P(N|d)}$. If $\log \frac{P(R|d)}{P(N|d)} > 0$ then $P(R \mid d) > P(N \mid d)$, and d is most probably a *relevant* document; otherwise d is most probably a *nonrelevant* document, relative to the current topic.

The term $\log \frac{P(R)}{P(N)}$ in (5.23) is an unknown constant for a given topic. As a general rule, $P(R) \ll P(N)$, i.e., an arbitrarily chosen document is most likely nonrelevant to a given topic. Omitting $\log \frac{P(R)}{P(N)}$ from (5.23) results in the decision point (threshold) moving a constant amount in the positive direction on the x-axis, away from the origin. The exact threshold is unimportant if the goal is simply to rank documents according to their log odds. But if the goal is to determine the most probable classification of documents, then the threshold should be learned, as a prerequisite to document classification. However, the empirical analysis of Chapter 6 shows that the use of zero as an approximate threshold is sufficient (as the exact threshold is generally "close" to zero, relative to the log odds being compared, e.g., a typical threshold of $O(10^1)$ versus log odds of $\pm O(10^2)$). Therefore, (5.23) can be reduced to

$$\log \frac{P(R \mid d)}{P(N \mid d)} = \sum_{i=1}^{|V|} f_i \cdot \log \frac{\hat{P}(t_i|R)}{\hat{P}(t_i|N)}. \tag{5.24}$$

Equation (5.24) is used by the filtering system to judge documents as either relevant or nonrelevant to a given topic. If the system judges a document as relevant to the current topic, it "retrieves" the document on the user's behalf and presents it to the user, for confirmation of actual relevance.

The next two sections present implementation details of the user-profile and an algorithm for adaptive Bayesian information filtering. Chapter 6 provides empirical verification of the usefulness of this information filtering approach and provides comparisons of results with two non-probabilistic information filtering methodologies.

5.5 User-Profile Implementation

Given a vocabulary V and a user-topic of interest T, the *user-profile* U is represented as a dynamically growing vector of size $|U|$, consisting of tuples of six elements each; each tuple $i = 1, \ldots, |U|-1$, represents a *unique term* of V that has been encountered in either T or the document-stream, up to a point in time; tuple $|U|$ has label $t_{|U|} =$ **other** and represents all $|V|-|U|+1$ vocabulary terms *not yet encountered* up to that point in time. Each *tuple* i of U consists of *six* components:

- term label t_i
- (a) relevance and (b) nonrelevance cumulative term-frequency counts, represented as *cumulative* hyperparameters α_{R_i} and α_{N_i}, Equations (5.13) and (5.14), respectively
- (a) relevance and (b) nonrelevance point-estimates $\hat{P}(t_i|R)$ and $\hat{P}(t_i|N)$ of the multinomial parameters θ_{R_i} and θ_{N_i}, respectively, Equations (5.15) and (5.16)
- estimated log odds of t_i appearing in a document, $\log \frac{\hat{P}(t_i|R)}{\hat{P}(t_i|N)}$.

Totals of the second and third components of all of U's tuples are maintained, to facilitate calculation of the fourth, fifth, and sixth components of each tuple. The fourth and fifth components of tuple $|U|$ are calculated as averages over the $|V|-|U|+1$ unseen terms; therefore, the sixth component of tuple $|U|$, $\log \frac{\hat{P}(t_{|U|}|R)}{\hat{P}(t_{|U|}|N)}$, represents the *average* log odds of an unseen term.

At initialization, the relevance and nonrelevance cumulative frequency counts (second and third components) of tuple $|U|$ are assigned the priors *equivalent sample size* values defined in Section 5.3.2, ess_R and ess_N, respectively; as each *new* vocabulary term is encountered in T and during document sampling, these cumulative frequency counts are decremented by $\frac{ess_R}{|V|}$ and $\frac{ess_N}{|V|}$, respectively. The latter values are then used to initialize the second and third components of the newly added term's tuple (e.g., topic-term initialization, Equations (5.11) and (5.12)).

5.6 Algorithm

Figure 5.2 presents an *algorithm* for adaptive Bayesian information filtering of on-line news-documents, given a vocabulary V and the multinomial distribution as document model.

1. Let the conjugate family $\mathcal{C}$ be the *Dirichlet* conjugate family of Section 5.3.1, with density function (5.8) and members $\mathcal{C}_n^{R^j} \in \mathcal{C}$. Given $R^j \in \{R, N\}$ and $n = |V|$, then $\mathcal{C}_{|V|}^R = \mathcal{DIR}_{|V|}^R$ and $\mathcal{C}_{|V|}^N = \mathcal{DIR}_{|V|}^N$, Equations (5.9) and (5.10), respectively.

2. Let: $T = (t_1, t_2, \ldots, t_{|V|})$ be the user-topic term vector,
   $w^t = (f_1, f_2, \ldots, f_{|V|})$ be T's term-frequency vector,
   U be the user-profile, as defined in Section 5.5,
   ess_R be the relevance equivalent sample size,
   ess_N be the nonrelevance equivalent sample size.

3. Let the second and third components of tuple $|U|$ of U be ess_R and ess_N, respectively.

4. For each t_i in T:
   If $f_i > 0$ then:
   $\alpha_{R_i} = f_i + ess_R/|V|$,
   $\alpha_{N_i} = ess_N/|V|$,
   Add a new tuple to U to represent t_i,
   Initialize second and third components of new tuple with α_{R_i} and α_{N_i}, respectively,
   Decrement second and third components of tuple $|U|$ with $ess_R/|V|$ and $ess_N/|V|$, respectively
   EndIf
   EndFor,
   Update U's second and third component *totals*, and recalculate the fourth, fifth, and sixth components of *every* tuple of U.

5. Sample the news-wire feed for the next document $d = (t_1, t_2, \ldots, t_{|V|})$,
   Let $w^d = (f_1, f_2, \ldots, f_{|V|})$ be d's term frequency vector,
   Let $l^d = (l_1, l_2, \ldots, l_{|V|})$ be d's term log odds vector, initialized with $l_i = 0$, $i = 1, \ldots, |V|$.

6. For each t_i in d with $f_i > 0$:
   If $(t_i \in U)$
   then Set l_i to t_i's log odds in U: $\log \frac{\hat{P}(t_i|R)}{\hat{P}(t_i|N)}$
   else Set l_i to *unseen-term* log odds: $\log \frac{\hat{P}(t_{|U|}|R)}{\hat{P}(t_{|U|}|N)}$ (from tuple $|U|$ of U)
   EndIf
   EndFor.

7. Calculate $d_{LogOdds} = w^d \cdot l^d = \sum_{i=1}^{|V|} f_i \cdot \log \frac{\hat{P}(t_i|R)}{\hat{P}(t_i|N)}$, Equation (5.24),
   If $(d_{LogOdds} > 0)$ then: *(If true, then the system has judged d as relevant to T: update U)*
   If any of d's terms are *not* explicitly contained in U, then:
   Add tuples to U to represent these new terms,
   Initialize the second and third components of each of these new tuples with $ess_R/|V|$ and $ess_N/|V|$, respectively,
   For each new term added to U, decrement the second and third components of tuple $|U|$ with $ess_R/|V|$ and $ess_N/|V|$, respectively
   EndIf,
   Forward d to the user,
   Let j be the **user's** relevance-judgement for d,
   If $(j ==$ **Relevant**$)$ then
   Update the *second* component of U's tuples with w^d frequencies (Equation (5.13))
   else
   Update the *third* component of U's tuples with w^d frequencies (Equation (5.14))
   EndIf,
   Update U's second and third component *totals*, and recalculate the fourth, fifth, and sixth components of *every* tuple of U
   EndIf.

8. Repeat from Step 5 until news-document sampling has been completed.

Figure 5.2: Algorithm for adaptive Bayesian information filtering.

Chapter 6

Adaptive Bayesian Information Filtering: Empirical Tests

6.1 Introduction

This chapter describes empirical tests of the Dirichlet conjugate prior approach for estimating the parameters of a multinomial model of text documents in a specific interactive news-document filtering environment. The experimental results are compared with the performance of two other interactive information filtering *non*probabilistic methods: a VSM/Rocchio learning approach, and a static "benchmark" system that does not employ learning techniques. The probabilistic Dirichlet model performed significantly better than the other models for the performance measure F3; however, its performance for the measure "geometric mean of precision and recall" was slightly less than that of the other models. These performance measures are defined in Section 6.2.1.

6.2 Experimental Evaluation

6.2.1 Experimental Method

Performance Measures

The following are the performance measures adopted for the experiments (defined relative to a given user-topic of interest):

Precision. The *percentage* of documents retrieved by the system and presented to the user that are *actually* relevant,

Recall. The *percentage* of *actually* relevant documents in the data set that are retrieved by the system and presented to the user.

Precision and recall are inversely proportional to each other; e.g., a system with high precision will usually have low recall. Different systems may exhibit dramatically different performances, depending on the adopted measure. Therefore a weighted mean of the two measures is sometimes used as a measure. The weighting depends on the relative importance of each component, which varies from user to user. Typically, the *geometric mean* of the two measures is used, with equal weighting given to each.

TREC F-Utility. Of the documents *retrieved by the system as relevant*, let R and N represent the number that are *actually* relevant and nonrelevant, respectively; then the following F-Utility measure represents the weighted *net absolute number* of documents correctly retrieved, as defined by the *Text REtrieval Conference-7* (TREC-7) [Hul98]:

$$\mathsf{F3} \;=\; 4R - N. \tag{6.1}$$

Data

The document set used for the experiments consists of 164,597 Associated Press news-documents provided by TREC-7 for their filtering track competition. The topic set consists of 50 predefined topics, also provided by TREC-7. A subset of the provided documents is labeled with relevance and nonrelevance judgements with respect to the

topics in the topic set. Any document that has not been labeled with such a judgement is assumed to be *nonrelevant* to all 50 topics. The documents are partitioned into 686 days: February 12, 1988 to December 31, 1989. Each day consists of a few hundred documents.

Method

Documents are presented to the filtering system one at a time, in chronological order. Documents occurring in the first 441 days are *training* documents, and subsequent documents, up to day 686, are *test* documents. Cumulative results are tabulated in a 2×2 misclassification matrix (also called confusion matrix) that forms the basis for the calculation of the three performance measures: precision, recall, and F3. The first row of the matrix contains counts of all documents that the system has retrieved and presented to the user as relevant; the second row contains counts of all documents that the system classes as nonrelevant and does not retrieve; the first column contains counts of all documents that are labeled as relevant (retrieved or not); and the second column contains counts of all documents that are labeled as nonrelevant *or* have no relevance label (implicitly nonrelevant).

User-provided Parameters

Vocabulary. The model's vocabulary V is the usual English language vocabulary, augmented with proper nouns and acronyms. All text is converted to lower-case and punctuation and numbers are ignored[1]. Use of the formulas in Chapter 5 requires that a vocabulary size $|V|$ be assigned: the value of 50,000 has been chosen as an approximation to $|V|$ as it represents a reasonable choice for the number of distinct terms and acronyms that may be encountered in a general news-document environment.

[1]Digits are allowed to form portions of of acronyms; future work uses punctuation to disqualify certain groups of words from forming phrases

Equivalent Sample Size. Formulas (5.11) and (5.12) require that values be chosen for the priors *equivalent sample sizes* ess_R and ess_N. Figures 6.2 and 6.3 provide graphs of performance results at the *end* of the 686 day training and testing period, for ess_R between 1 and 700 and ess_N between 1,000 and 50,000; the specific performance measure chosen for this experiment is the geometric mean of precision and recall, averaged over the TREC-7 topic numbers 1 to 10. Figure 6.2 is a 3-D view, with the z-axis representing the geometric mean of precision and recall; Figure 6.3 includes 2-D perspectives of the first graph with respect to each independent variable.

These graphs indicate that *average* performance is rather insensitive to changes over fairly wide ranges of *ess* values; however, it was noticed that performance varies considerably for *individual* topics as *ess* values change. Therefore, the following *arbitrarily ess* values were chosen for the experiments: $ess_R = 20$ and $ess_N = 20,000$; this choice guards against the introduction of "tuning bias" into the experiments from choosing *ess* values that optimize performance for certain topics[2]. These selected *ess* values were used for two sets of tests: (1) *comparative tests* of the performance of the current parametric information filtering model against two nonparametric models, and (2) tests of the parametric model's *performance variation over time* (training and testing periods).

Log Odds Threshold. As explained in Section 5.4 and illustrated by Equation (5.24), a log odds relevance threshold of zero has been chosen.

Other Models for Comparative Purposes

For comparative purposes, the performance of the current probabilistic model is compared with two *nonprobabilistic interactive* filtering models: an incremental VSM/Rocchio learning method, denoted "Single PG Agent", and a benchmark method that does not employ any learning, denoted "Benchmark". All three methods begin with a user-profile

[2]Such "tuning" is strictly prohibited during testing of incremental (adaptive) filtering algorithms because tuning is implicitly a batch process.

initialized with term frequencies from the current topic of interest, as discussed in Section 3.1. The "Single PG Agent" nonprobabilistic method employs a form of the *tf.idf* term-weighting heuristic discussed in Section 3.2, with user-profile updating based on user relevance feedback using an incremental-Rocchio approach, as discussed in Section 3.2.1. The "Benchmark" nonprobabilistic method also uses a *tf.idf* term-weighting heuristic; however, for each given topic, the user-profile is static over all documents and is not updated by user relevance feedback. See [Kar96] for more detailed descriptions of these nonprobabilistic document filtering methods.

6.2.2 Results

Figures 6.4(a) and 6.4(b) provide detailed *comparative results by topic* for the three evaluated systems at the *end* of the 686 day training and testing period, for the performance measures geometric mean of precision and recall, and F3, respectively. The following table summarizes the performance *averages* of Figure 6.4, over all topics:

	Performance Averages	
Model	*G. Mean*	F3
Benchmark	0.2208	−6.22
PG Agent	0.2186	−27.14
Probabilistic	**0.1678**	**34.38**

Figure 6.1: Performance averages (topics 1–50 of the TREC-7 adaptive filtering track).

Figures 6.5 and 6.6 show the manner in which the probabilistic model's performance *changed over time*. The model's parameters $\theta_{R_i} = P(t_i|R)$ and $\theta_{N_i} = P(t_i|N)$, $i = 1, \ldots, |V|$, are learned only during the training period from Day 1 to Day 441. During the testing period from Day 442 to Day 686, the parameter values in effect at the end of the training period are used.

See Section 6.2.3 for a discussion of these results.

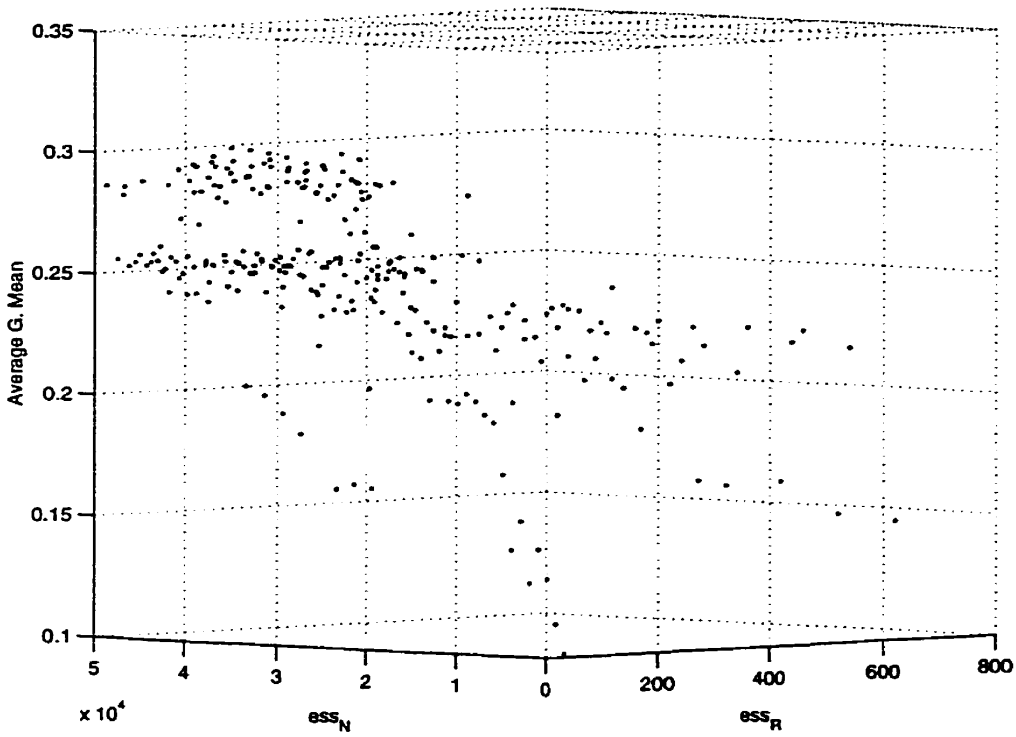
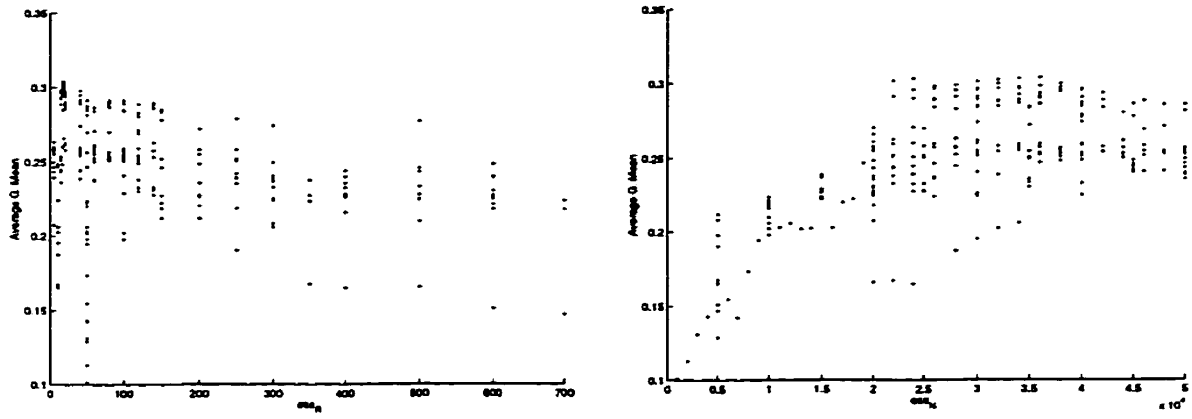


Figure 6.2: Performance as a function of *equivalent sample sizes* ess_R and ess_N.



(a) ess_R (b) ess_N

Figure 6.3: Performance as a function of *equivalent sample sizes* (single variable view).

Topic	Bench	PG Agent	Prob
1	0.12	0.21	0.25
2	0.15	0.18	0.15
3	0.19	0.23	0.10
4	0.33	0.31	0.38
5	0.38	0.51	0.50
6	0.47	0.43	0.45
7	0.37	0.34	0.28
8	0.07	0.07	–
9	0.47	0.46	0.03
10	0.41	0.35	0.43
11	0.46	0.44	0.54
12	0.29	0.30	0.13
13	0.35	0.69	0.03
14	0.13	0.10	0.09
15	0.40	0.39	–
16	–	–	–
17	–	–	0.44
18	0.14	0.15	–
19	0.10	0.09	0.17
20	0.58	0.37	0.34
21	0.62	0.70	0.62
22	0.19	0.41	0.35
23	0.44	0.57	0.55
24	0.04	–	0.18
25	0.37	0.26	0.12
26	–	–	–
27	0.41	0.41	0.10
28	0.27	–	–
29	0.05	–	–
30	–	–	–
31	–	–	–
32	–	–	–
33	0.08	0.08	0.13
34	–	–	–
35	0.41	0.45	0.38
36	0.21	0.13	0.10
37	–	–	–
38	0.12	0.14	0.03
39	–	–	–
40	0.48	0.44	0.43
41	0.07	0.09	–
42	0.02	–	–
43	0.18	0.16	0.09
44	0.27	0.36	0.02
45	–	–	0.31
46	0.62	0.40	0.51
47	0.25	0.24	0.03
48	–	–	–
49	0.52	0.46	0.11
50	–	–	–
Average	**0.2208**	**0.2186**	**0.1678**

(a) Geometric Mean of Prec. & Recall

Topic	Bench	PG Agent	Prob
1	-3	-125	30
2	-14	-108	22
3	2	-34	-1
4	-92	-153	32
5	36	55	62
6	154	49	162
7	-26	-239	52
8	-88	-41	-14
9	72	41	-12
10	90	-306	174
11	224	146	411
12	125	158	21
13	50	190	-17
14	-36	-24	-6
15	46	-27	-16
16	-28	-26	-18
17	0	0	175
18	-1257	-1493	-39
19	-14	-12	8
20	109	35	39
21	48	66	55
22	109	539	385
23	108	224	243
24	-26	-11	26
25	28	-178	-18
26	-1	-1	-23
27	4	4	-13
28	-87	-6	-17
29	-58	-2	-9
30	-14	-14	-16
31	-7	-4	-8
32	-1	-1	-21
33	-8	-8	-8
34	0	0	-18
35	-1	0	-2
36	-5	-81	-12
37	0	0	-5
38	18	26	-10
39	0	0	-11
40	140	2	165
41	-19	-10	-10
42	-43	-8	-15
43	-3	3	-10
44	26	45	-28
45	-1	-1	20
46	91	-29	62
47	11	-7	-18
48	-11	-11	-7
49	41	20	-13
50	0	0	-10
Average	**−6.22**	**−27.14**	**34.38**

(b) F3 Utility

Figure 6.4: Comparative tables of performance for the three document filtering models (by topic, where performance is *cumulative* over the entire training and testing period).

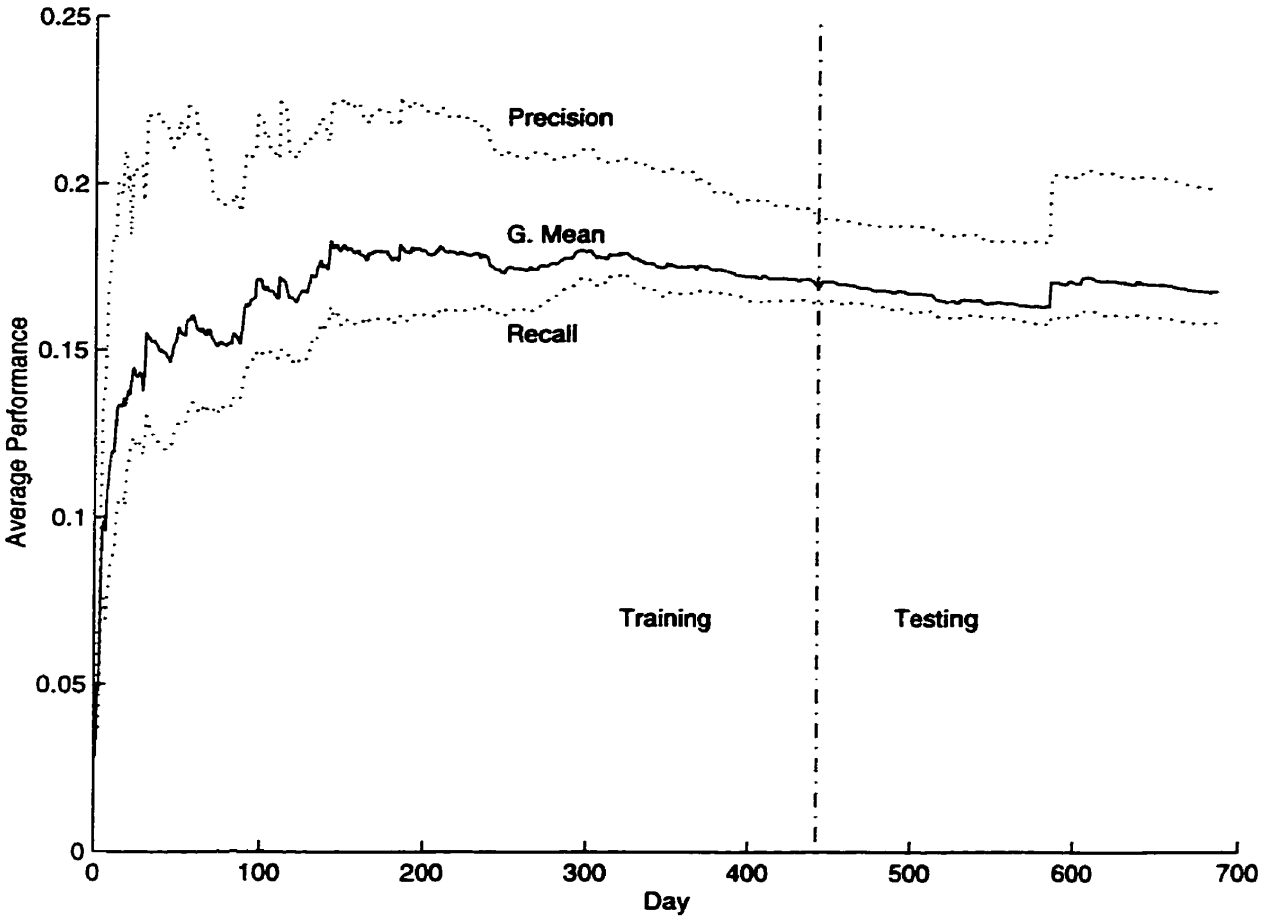


Figure 6.5: Probabilistic model performance over time (average of topics 1–50).

6.2.3 Discussion

Figure 6.1 shows that, over the 50 topics, the current probabilistic model obtained an impressive *average* F3 value of 34.38, compared to −6.22 and −27.14 for the Benchmark and PG Agent, respectively[3]. The measure F3 reflects filtering precision as a weighted net *absolute number* of relevant documents correctly retrieved (net of nonrelevant documents incorrectly retrieved, or *fallout*). The geometric mean of precision and recall, however, averaged 0.1678 for the the probabilistic model, which is slightly less than the PG Agent's average of 0.2186 and the Benchmark's average of 0.2208. The inverse relationship between F3 and the geometric mean can be explained by the fact that an

[3]A higher score represents better performance.

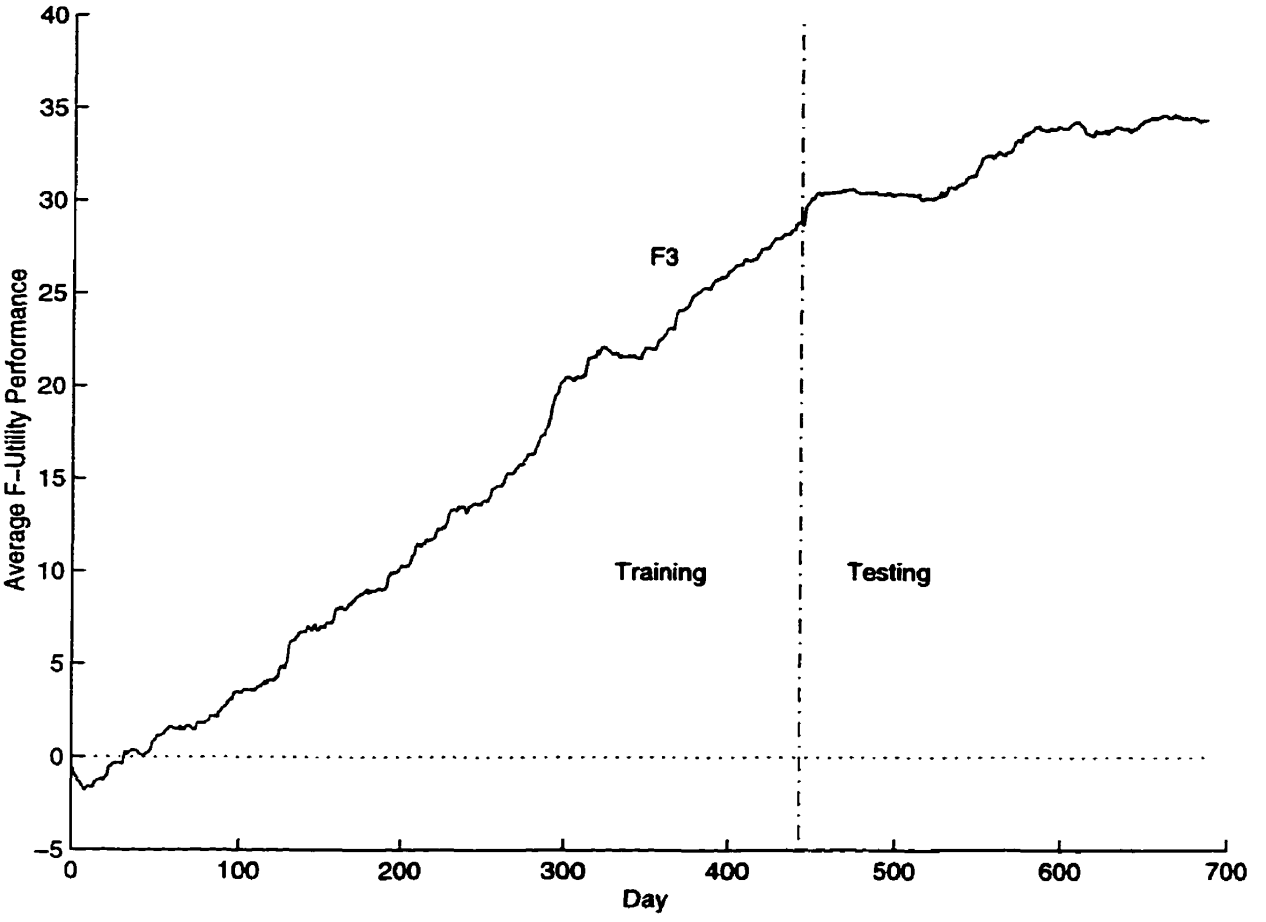


Figure 6.6: Probabilistic model F-Utility performance over time (average of topics 1–50).

increase in geometric mean is often associated with an increase in recall. But an increase in recall (the percentage of relevant documents retrieved) generally implies an increase in the number of *nonrelevant* documents that are *also* being retrieved; such fallout is represented by the N value in Formula (6.1), where it can be seen that an increase in N may cause a decrease in F3.

The probabilistic model clearly performs very well for the TREC F-Utility measure F3. This verifies the theoretical validity of the current probabilistic model for interactive document filtering, and indicates that such a model is at least as powerful as the interactive VSM/Rocchio nonprobabilistic model.

Figure 6.5 shows that average precision improved during training from Day 1 to Day

200, and average recall improved from Day 1 to Day 320; it is apparent, however, that very slight overfitting has occurred during training: the model's user-profile has grown to such an extent that performance degrades very slightly, halfway through the 441 day training period. Two *independent* experiments were performed in an attempt to alleviate this degradation: (1) training was restricted to the period from Day 1 to Day 200, and (2) the profile-size was limited to a maximum of 500 terms (rather than an unlimited size). Both experiments, however, yielded approximately 15% reduction in geometric mean at Day 686, from the value indicated in Figure 6.5. Overfitting may be occurring because only a *single* profile is being used to model *all* relevance regions of the very large news-document space. In effect, the model may have reached its "limit of effectiveness" (for a single model). Hence, the use of multiple models may be justified, where each individual model approximates a local area of the document space[4].

Overfitting in the late stages of training may also be occurring due to the nature of frequency-updating in the Dirichlet model: the impact of individual frequency updates on the model's parameters *decreases* with training time because frequency counting is cumulative in nature; the earliest updates have the greatest impact in changing the model's parameters, and the latest updates have the least impact. This becomes a serious problem when, for a given topic, the probability of a term t_i occurring in a relevant document, $P(t_i|\,R)$, *changes over time* (called term nonstationarity). The cumulative nature of the frequency-counts for t_i (both relevance and nonrelevance) result in the model being unable to adjust for such nonstationarity.

Figure 6.6 shows that average F3 performance increased almost linearly with training and testing time. This indicates that, cumulatively over the training and testing periods, the weighted number of correctly retrieved (relevant) documents *consistently exceeded* the number of incorrectly retrieved (nonrelevant) documents or fallout; i.e., the model

[4]See [Kar96] for a discussion of the SIGMA information filtering approach that employs *multiple* models for exactly this reason.

performed very well at minimizing fallout at *all* stages of the learning process, not just after training had completed. This observation strongly confirms the model's theoretical suitability for *interactive* environments: it is able to make "good quality" predictions midway through training, *before* seeing all training data.

Chapter 7

Conceptual Modeling and Telos

7.1 Introduction

Research in the field of information filtering has tended to address only the process of *accessing* information and has generally ignored the important requirement that retrieved information must be *stored* in an *organized* persistent form that is easily understandable by users. While filtering on-line news feeds for documents relevant to specific topics, users generally wish to store *both* the relevant documents located *and* the *relationships* between the documents, topics, subtopics, and subject-domains. However, subject-domains, topics, and documents have complex internal structure and possess complex inter-relationships. A relational database representation of this information is, in some sense, "artificial" because the real-world entities and relationships must be decomposed into tables and records in order that they fit the logical model; i.e., the correspondence between the model and the real-world becomes unclear to users: users are not easily able to *interpret* the semantic content of such a logical data model. Therefore, it is clear that information models are required that are able to capture the *semantics* of applications.

Automated information modeling has experienced significant advances since the early days of Computer Science when *implementation issues*, such as efficiency, were a primary

concern to computer users. The introduction of *logical data models* in the early 1970s freed users from such implementation concerns; hardware advances since that time have made possible the development of a family of more compute-intensive *people-oriented* information modeling techniques, called **conceptual modeling.**

This chapter briefly discusses information modeling and introduces conceptual modeling; several directions of information modeling research closely related to conceptual modeling are also discussed. The **Telos** conceptual modeling language is introduced, and an overview of its representational framework is provided. Telos will be used in Chapter 8 to model the information domain of the news-document filtering system presented in the preceding chapters.

7.2 Information Modeling

Information modeling is the use of computer-based *symbol structures* to *formally describe* and capture the *meaning* of some aspect of the real world, for use by automated systems or by people. An **information model**[1] consists of collections of (1) *symbol structure* types, (2) *operations* that can be applied to the symbol structures, and (3) *integrity rules* that define the legal symbol structures that are allowed by the model. An **information base**[1] is a symbol structure based on a specific information model that describes a *specific* aspect of the real world. It consists of *structured information* and serves a role somewhat analogous to long-term human memory.

Information modeling possesses at least three dimensions: (1) the real world *domain* being modeled, (2) the *symbol structures* used, and (3) the intended *users* of the model. The **domain** may consist of concrete items, abstract concepts, natural-world phenomena, or any type of human or automated activity (e.g., entity knowledge, activity/task knowledge, processes, agents, goals, relationships, positions, roles, etc.). The **symbol structures** may take many possible forms: natural languages, implementation (programming)

[1]The definitions of *information model* and *information base* have been adopted from [Myl98].

languages, mathematical languages, diagrammatic languages, logic-based languages, etc.. Such a variety of languages clearly have varying degrees of formality *and* relationship to machine implementation. The **users** of information models may be people *and/or* other automated systems. The symbol structure dimension determines the *type* of the information model (physical, logical, or conceptual) and its *expressiveness*. Physical models utilize programming-like data structures, with an emphasis on computational efficiency (at the expense of modeling real world structure). Logical models use abstract mathematical data types to hide implementation details from the user, but offer minimal information structuring capability. Conceptual models represent information in a manner analogous to the way humans conceptualize the information, i.e., conceptual models model the *complex structure and relationships* implicit in the real world domains they are modeling.

7.3 Conceptual Modeling

Conceptual modeling is described in [Myl92a] as modeling information in a manner

1. consistent with the way *people* conceptualize the subject matter (e.g., via entities, relationships, actions, and abstraction mechanisms), and
2. independent of machine implementation issues

That is, conceptual models are intended for use by *people* rather than by systems, for the communication and understanding of information; they are characterized by "high-level" complex symbol structures which have no specific relationship to any machine implementation. They are specified by their *ontologies* and *abstraction mechanisms*:

Ontology Aspects of the world to be modeled, expressed in terms of the *primitive* concepts or *semantic* terms used within the area being modeled (e.g., entity, activity, relationship, process, state, agent, goal, belief, actor, role, etc.)

Abstraction mechanism Means of *organizing* a model's primitive concepts into abstract *semantic relationships* (e.g., classification, generalization, aggregation, contextualization, etc.)

Ontologies may be arbitrarily classified into various types, depending on the *nature* of the application being modeled. For example, a *static* ontology is concerned with the static aspects of an application by describing concrete or generic *entities* that exist in the application, and their attributes and interrelationships. A *dynamic* ontology is concerned only with the *changes* inherent in an application, such as events, processes, states, and state transitions. An *intentional* ontology is concerned with agents (intelligent autonomous entities) and the goals and beliefs they possess about other concrete or generic entities. A more detailed description of these "coarse-grained" ontologies, based on a survey of several conceptual models, may be found in [Myl98].

Classification is the abstraction mechanism describing the binary relationship of a *single* concept as a *member of* one or more *generic* concepts (classes), with *all* class members sharing common properties. Classification is a non-transitive relation. The converse of classification is *instantiation.*

Generalization is the abstraction mechanism describing the binary relationship of *superset* between two *generic concepts (classes)*; it is a partial order (and hence transitive) relation. The converse of generalization, called *specialization*, provides the inference rule of *inheritance*, where the properties of members of a class are also properties of the members any specialization (subset) of the class. Inheritance may be strict (constraints on properties may be strengthened only) or default (constraints on properties may be arbitrarily changed). In addition, specialized classes may have *additional* properties to those inherited.

Aggregation is the most common abstraction mechanism; it structures a concept in terms of its *component* concepts. Such an aggregation structure is not necessarily unique for a given concept: it is possible that a concept may be described by aggregation in more than one way, depending on the *context* under which the concept is modeled. References to aggregate components may either be "by value" or "by reference" (i.e., components may be other *concepts* or may be *pointers* to other concepts). There is generally no

restriction on the "types" of a concept's components; therefore, if a concept is composed of any components of type Class, aggregation hierarchies of arbitrary nested complexity may result, possibly leading to recursive aggregation.

Omitted from the above discussion are the (potentially complex) issues of multiple inheritance and multiple instantiation; for a discussion of these issues, see [Tho92], and [Myl90] pp. 6-7, respectively.

A more detailed survey of the many abstraction mechanisms that have been adopted by Computer Science from Cognitive Science and Mathematics may be found in [Myl98] (e.g., classification, generalization, aggregation, contextualization, materialization, normalization, and parameterization).

Following [Bor90], as cited in [Myl92a], it is useful to contrast conceptual modeling with *knowledge representation* and *semantic data modeling.* Knowledge representation is the formal axiomatic modeling of information for the purpose of supporting *reasoning and planning* by *automated systems.* Semantic data models (e.g., the *Entity-Relationship* model [Che76], and *Taxis* [Myl80]) attempt to capture the semantics of an application and use this semantics in a manner consistent with current database technology; thus semantic data models make implicit *implementation assumptions* about the manner in which an information model will be represented on a machine, and are intended for use by people *or* systems.

During the period 1974 to the present, a significant amount of research into knowledge representation, semantic data models, and conceptual models has been performed by the Knowledge Representation Group of the Department of Computer Science at the University of Toronto. A comprehensive overview of this work appears in [Myl92b]. The culmination of this work, the conceptual modeling language *Telos* [Kou89], [Myl90], [Myl92a], is the subject of the remainder of the chapter. Section 7.4 discusses related work in knowledge representation, databases, and software engineering, and outlines Telos' genealogy. Section 7.5 provides a brief overview of Telos' distinguishing features. Section 7.6 discusses Telos' representational framework.

7.4 Related Work and Telos Genealogy

It is useful to overview related work in the context of research performed at the University of Toronto from 1974 to the present.

7.4.1 Related Work in Knowledge Representation and Databases

Knowledge representation is concerned with the modeling of knowledge for the purpose of automated reasoning and planning. **Database management systems** are concerned with efficiently managing large amounts of knowledge. In the 1970s, modeling of the "semantic content" of real world domains became an increasingly important focus of research in both of these areas.

In the area of knowledge representation, semantic networks became a popular notation and were adopted by many AI projects. Initial research in the *TORUS* project at the University of Toronto [Myl76] used semantic networks as a means of representing the *semantics* of relational database systems. This research led to the conclusion that semantic networks provide a promising notation for knowledge representation. However, TORUS and other semantic network proposals at the time had serious weaknesses in unambiguously representing natural language and other knowledge. These weaknesses resulted in research into more powerful and greatly expanded semantic network representations. One such research effort at the University of Toronto led to the *general purpose* knowledge representation language *PSN* (Procedural Semantic Network) [Lev77]. PSN introduced several novel features within semantic networks: (1) *object-oriented* semantic network representation, (2) use of metaclasses, (3) *extension* of classification to metaclasses, relations, and programs, (4) use of generalization and aggregation within semantic networks, and (5) built-in class operations (methods).

In the area of database models, attention was directed to the shortcomings of physical and logical database models in modeling real world domains. Abrial's *semantic* model

[Abr74] and Chen's *Entity-Relationship* model [Che76] were early results of this new direction. They were followed at the University of Toronto by the *Taxis* model [Myl80]. Taxis used some of PSN's features, however it was heavily oriented towards maintaining consistency with traditional database technology. Taxis was concerned with semantic modeling for the *specific* purpose of supporting the *design* of information systems. It organizes *all* components of an information system in terms of generalization hierarchies (taxonomies). The strong implementation-oriented emphasis of Taxis is reflected by the fact that *data, attributes*, and *programs* are each classified into *fixed built-in categories*, according to their implementation or execution characteristics. Taxis extends PSN's class operations by allowing arbitrary class operations, in addition to built-in methods.

Object-oriented databases [Ber93] have attracted attention in the database community and share some of the goals of the above cited research; however, they are closer in philosophy to object-oriented programming languages, and strongly emphasize the aggregation abstraction mechanism. The following are their main features: (1) modeling of real world entities as *objects*, (2) aggregation of *complex* non-primitive objects as attributes, (3) encapsulation of method *implementation* (from method specification), (4) single-level classification, and (5) inheritance. They are useful for applications such as CAD where real world entities possessing *complex internal structure* are required to be modeled in a manner that reflects exactly the actual internal structure of the entities.

7.4.2 Related Work in Software Engineering

Information modeling is a key component of **requirements modeling**, the initial analysis phase of software design where the software analyst attempts to *understand* a real world problem before embarking on a software solution to the problem. Significant early work in this area is represented by the techniques of *Structured Analysis* (e.g., Data Flow Diagrams (DFDs) [DeM78] and Structured Analysis and Design Technique (SADT) [Ros77]). These early research efforts emphasized the aggregation abstraction

mechanism. Subsequent research efforts introduced further abstraction mechanisms into requirements modeling, such as generalization and classification. For example, the requirements modeling language *RML* [Gre84] developed at the University of Toronto, and *object-oriented analysis* approaches [Coa90] were highly influential in the evolution of requirements modeling methodologies.

RML was introduced as a *formal* version of SADT for formally specifying the *functional requirements* for proposed software systems. It adopted ideas from knowledge representation and semantic data modeling and features an object-oriented representational framework supporting classification, aggregation, and generalization. RML distinguishes three types of built-in objects: *entities, activities*, and *assertions*, with entities and activities organized into generalization hierarchies. Attributes associated with each of these objects are classified into specific built-in *attribute-categories*, each with a *prespecified* semantic meaning. An assertional sublanguage replaces Taxis' procedural sublanguage and is used to specify constraints and deductive rules on objects.

7.5 Telos

RML's rather rigid built-in semantics for objects and attribute-categories made it inadequate as a *general-purpose* modeling language. Therefore a line of more expressive and general-purpose conceptual modeling languages was developed, beginning with *CML* [Sta86], and culminating with Telos. This line of languages borrowed several concepts from its predecessors and also added new innovations. Telos provides greatly improved modeling flexibility over RML, allowing the modeling of widely disparate real world situations. The following is a brief outline of Telos' distinguishing features:

1. *Proposition "building-block"*: elimination of the distinction between entities and relationships by allowing *all* knowledge base components to be treated in a *consistent manner* as propositions; i.e., in Telos, "everything is a proposition";

2. *Object-oriented* semantic network framework;

3. *Abstraction mechanisms*: classification, aggregation, and generalization;

4. Classification along an *infinite* dimension of metaclasses (cf., the *single*-level classification characteristic of most object-oriented programming languages);

5. *First-class citizenship of attributes*: instantiation *and* attribution (allows the use of attribute *metaclasses* as a means of obtaining *customizable* attribute categories);

6. *Multiple instantiation*: an object may be an instance of more than one class;

7. *Functional capability*: operations for knowledge insertion, removal, and query;

8. *Assertional sublanguage*: domain-specific integrity constraints and deductive rules;

9. *Temporal* representation and reasoning capability.

7.6 Telos Representational Framework

The **Telos representational framework**[1] is the structure provided by Telos for representing knowledge; it is a generalization of the graph-theoretic data structures characteristic of semantic networks and object-oriented programming languages. A Telos knowledge base is composed entirely of a collection of *propositions*. A Telos proposition is defined as a 4-tuple with components which are themselves propositions: *source, label, destination*, and *duration*; the first three components denote a *labeled edge*, which serves as a "relationship" within the represented knowledge (analogous to a semantic-network link); the last component denotes a time-interval proposition, which represents the lifetime of the relationship that is represented by the proposition[2]. Example proposition definitions are provided below.

[1] Several of Telos' important features are omitted from this very brief discussion (e.g., its assertion language and temporal knowledge representation); see [Kou89] for a thorough discussion of Telos' features.

[2] Time-interval propositions are not used in the current work.

The collection of Telos propositions is divided into two disjoint sets: *individuals* and *attributes*. Individuals represent real world *entities, classes, objects, concepts*, etc.; attributes represent real world *binary relationships* between entities or other relationships. Individuals may be considered to be the nodes in a traditional semantic network, and attributes may be considered to be the links between the nodes. The following are example proposition definitions, and their respective types[3]:

Country := [Country, ..., Country, ...] *class individual*

Canada := [Canada, ..., Canada, ...] *token individual*

role_of_city_in_country := [Country, *role*, City, ...] *class attribute*

Canada's_capital_is_Ottawa := [Canada, *capital*, Ottawa, ...] *token attribute*

Telos propositions form the "building blocks" of *all* objects in a Telos knowledge base. This convention, together with the structuring mechanisms discussed below, removes the traditional distinction between entities and relationships, allowing them to be treated in a *uniform* manner.

Propositions are organized along three orthogonal abstraction or structuring dimensions: *classification, aggregation*, and *generalization*.

Classification calls for *every* proposition to be a member of (instance of) one or more generic propositions called *classes*, which are themselves propositions and members of other more abstract classes. Thus there exists an infinite classification dimension: *tokens* (propositions having no instances), *simple classes* (propositions having only tokens as instances), *meta classes* (propositions having only simple classes as instances), *metameta classes* (propositions having only meta classes as instances), etc..

Token propositions represent concrete entities or relationships in the domain of discourse, while class propositions represent abstract or generic concepts.

Instantiation of a structured object (from one or more classes) can be considered a

[3]Ellipses (...) denote proposition components that are not applicable to the specific example.

typing mechanism: it determines the kinds of attributes the object can have and the properties it must satisfy.

At each class level of the classification hierarchy there is a Telos *built-in* class: **Token** (a simple class, having all tokens as instances), **SimpleClass** (a meta class having all simple classes as instances), **MetaClass** (a metameta class having all meta classes as instances), **MetaMetaClass** (a metametameta class having all metameta classes as instances), etc.. In addition, there are a number of *built-in* ω-classes which are used to facilitate the structuring of the knowledge base: **Proposition** (having all propositions as instances), **Class** (having all generic propositions as instances), **Individual** (having all individuals as instances), **Attribute** (having all attributes as instances), **AttributeClass** (having all attribute classes as instances, and **OmegaClass** (having all ω-classes as instances).

Aggregation allows for the creation of *structured* objects consisting of collections of attributes that have a common proposition as source.

Generalization allows for classes along the *same classification* level to be organized into a hierarchy from most general to most specialized (in the form of superclasses and subclasses).

7.7 Implemented Telos Information Bases and Tools

Chapter 8 outlines an information base of a Telos conceptual model of the information domain of the news-document filtering system presented in the preceding chapters. The following are further examples of implemented Telos information bases and tools:

1. **i^* Modeling Tool**

   Automated modeling of an application's *early software requirements* based on the i^* strategic-dependency modeling framework [Yu95] requires large amounts of both graphical modeling information and real-world application-domain information. Automated representation of this information, together with a graphical user in-

terface for visual presentation, has been developed using a Java implementation of a subset of Telos.

2. **ITHACA Software Information Base**

   The ITHACA project [Con91] is a large ESPRIT project devoted to the construction of an object-oriented application development environment. One of ITHACA's main components is a *software information base* intended to facilitate software reuse; automated representation of this information base, together with a graphical user interface, has been developed using a C++ implementation of a subset of Telos.

3. **Generic Telos Browser**

   Currently under development is a Java 2.0 graphical user interface browser that will allow the visual display of the underlying conceptual models of *arbitrary* Telos information bases.

Chapter 8

Modeling the Subject World of a Document Filtering System

8.1 Introduction

This chapter describes an information base of a Telos conceptual model of the *information domain* or subject world of the news-document filtering system presented in the preceding chapters. The static ontological concept of *entity* has implicitly been assumed appropriate for this modeling situation. News-domain subject-areas, topics, documents, and their interrelationships are modeled in a manner consistent with the way in which *people* conceptualize this information, that is, in a strict classification hierarchy. Similarly, the descriptive properties of these entities are modeled precisely in the manner in which people associate attributes with these entities.

In addition, information base *query-capability* is described: the implemented query and subtopic-instantiation capability (described in Sections 8.5 and 8.6) may be considered a fine-grained *deterministic* classification mechanism that complements the coarse-grained *non-deterministic* document filtering process. These sections provide a vivid illustration of an additional tangible benefit of conceptual modeling: the implementation

of complex queries proved quite elegant *and* relatively effortless—queries are abstract concepts that are formulated by *people*, and hence they are implemented most effectively in information bases that are organized in a manner that is consistent with the way in which people conceptualize the subject matter being modeled.

8.2 Subject World Entity Structure

The information domain or **subject world** of a news-document filtering system consists of three principal types of *structured (aggregate) objects*: general **subject-areas**, specific **topics** within subject-areas, and **news-documents** within topics. Therefore, the *static* ontological concept of **entity** is implicitly assumed as appropriate for modeling this information domain. The above information entities do not exist in isolation: human users intuitively impose a hierarchical structure among them. For example, the relationship between a *topic* and a *subject-area* (assuming both are classes) may be *"is a type of"*, *"is part of"*, or *"is a member of"* (i.e., in Telos terminology, is-a, part-of, or instance-of, respectively). Similarly, a given *document-token* may be considered *"a part of"* or *"a member of"* a *topic* (i.e., part-of or instance-of respectively).

8.2.1 Justification for Entity Structuring Choice

Each of Telos' three *abstraction mechanisms* (classification, generalization, and aggregation) was independently considered as a structuring choice for the interrelationships between the generic and concrete concepts of a news-document filtering system (subject-areas, topics, and documents). It was decided that a **uniform** structuring mechanism across these entities is the most appropriate choice (in order to facilitate implementation of user-queries of the information base[1]).

[1] See Sections 8.4 and 8.5.

Generalization was quickly discarded as a choice because the three entities are conceptually dissimilar, i.e., a document is not *"a type of"* topic, nor is a topic *"a type of"* of subject-area. This will become more apparent when the attributes of each of the entities are discussed in Section 8.3.

Aggregation appeared useful at first: subject-areas are *composed* of various topics, and a topic may be considered *composed* of all documents in the domain that discuss that topic. However it was decided, in an abstract sense, that aggregation is more appropriate when an entity is composed of a *fixed* number of well-defined components. Also, under aggregation, removing *any* component from an entity should change the inherent nature of the entity, causing it to be redefined; however, if aggregation was adopted for the current model, "removing" a document from a topic, or a topic from a subject-area, would **not** disqualify that topic or subject-area from retaining its previous identity.

Classification was finally chosen as the most appropriate Telos abstraction mechanism for the current model: documents may be considered *members of* topic generic concepts and, likewise, topics may be considered *members of* subject-area generic concepts. That is, each generic concept may have any number of members (instances). Therefore, instance-of was chosen for both the document/topic and topic/subject-area relationships. This results in the *entire* entity hierarchy being represented as a *classification* structure, as follows:

1. MetaMetaClass **Subject_Area**
   IN MetaMetaClass
   ISA MetaClass

2. MetaClass **Topic_Class**
   IN Subject_Area, MetaClass
   ISA SimpleClass

3. SimpleClass **Topic**
   IN Topic_Class, SimpleClass
   ISA Token

4. Token **Article**
   IN Topic, Token

where MetaMetaClass, MetaClass, SimpleClass, and Token are *built-in* Telos classes.

As an illustration of the above entity classification hierarchy, the following will be used as a *running example* throughout the text: Economics_Topic_Class as an example subject-area Topic_Class metaclass, Antitrust_Lawsuits_Topic as an example Topic simpleclass, and AP980723-0139 as an example Article token (an instance of Antitrust_Lawsuits_Topic). This document is assumed to be a news article which discusses the recent U.S. government antitrust lawsuit action taken against Microsoft corporation (regarding the Windows 95 operating system). Figure 8.1 illustrates the entity classification structure for this example, in graphical form. Also shown are two additional topics, Entrepreneurs_Topic and Operating_Systems_Topic, of which the given document is *also* assumed to be an instance (an example of multiple instantiation).

8.3 Aggregation and Attribute Classification

The definition of the three structured entity types discussed in the preceding section is formalized by assigning to each entity several components or *attributes* via Telos' *aggregation* structuring dimension. Telos' treatment of attributes as first class citizens allows the use of attribute meta and simple classes as customizable attribute categories which may be used to classify the attributes associated with an individual. Aggregation and attribute classification will be discussed next.

The definition of the metametaclass Subject_Area, shown below, introduces three attribute classes with labels *subjectNumber*, *subjectTitle*, and *topicDescription*. The attribute simpleclasses with labels *subjectNumber* and *subjectTitle* have as *destinations* the built-in classes Integer and String, respectively; these provide the two descriptive properties of every Topic_Class instance of Subject_Area. The attribute metaclass with label *topicDescription* acts as an attribute category for the descriptive properties of Topic simpleclass instances of Topic_Class (with component labels *topicNumber*, *topicTitle*, and *topicNarrative*, which represent attribute simpleclasses having *source* Topic and *destinations* Integer, String, and String respectively).

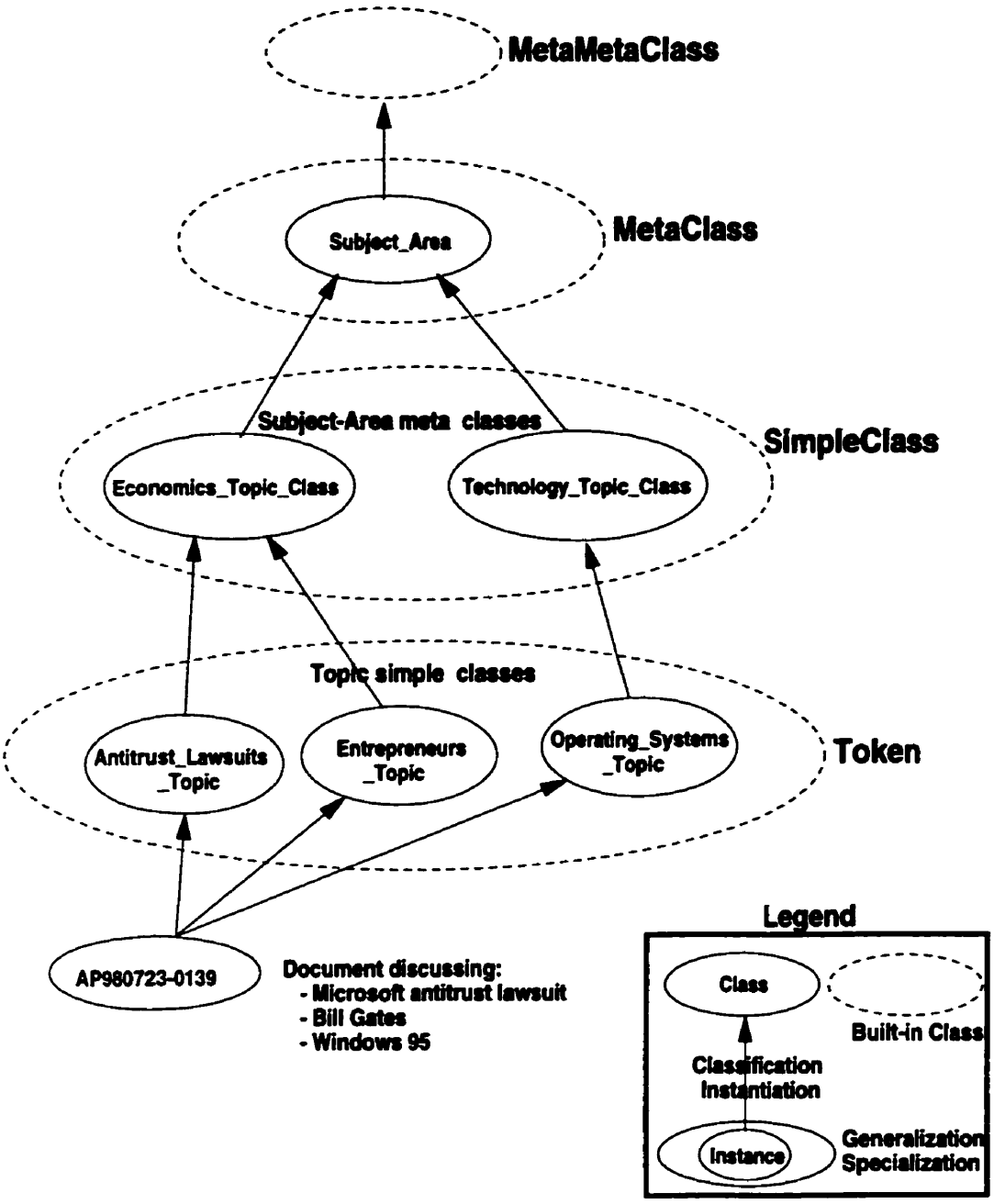


Figure 8.1: Example *entity classification* structure of the news-article filtering model.

```
MetaMetaClass Subject_Area
    IN MetaMetaClass
    ISA MetaClass
    WITH
        attribute
            subjectNumber: Integer;
            subjectTitle: String;
            topicDescription: SimpleClass
End

MetaClass Topic_Class
    IN Subject_Area, MetaClass
    ISA SimpleClass
    WITH
        subjectNumber
            : ...
        subjectTitle
            : "..."
        topicDescription
            topicNumber: Integer;
            topicTitle: String;
            topicNarrative: String
        attribute
            articleDescription: SimpleClass
End
```

Every metaclass subject-area instance Topic_Class of Subject_Area further introduces an attribute metaclass with label *articleDescription* which acts as an attribute category for the two descriptive properties of Article token-instances of Topic (with component labels *articleContent* and *articleSemantics* and destinations String and Semantic_Category).

```
SimpleClass Topic
    IN Topic_Class, SimpleClass
    ISA Token
    WITH
        topicNumber
            : ...
        topicTitle
            : "..."
        topicNarrative
            : "..."
        articleDescription
            articleContent: String;
            articleSemantics: Semantic_Category
End


Token Article
    IN Topic, Token
    WITH
        articleContent
            title: "...";
            head: "...";
            text: "..."
        articleSemantics
            person: ...;
            location: ...;
            organization: ...;
            item: ...;
            concept: ...;
            action: ...;
            role: ...;
            relationship: ...
End
```

The components of the *articleContent* attribute category (with labels *title*, *head*, and *text*) represent the three major descriptive String attributes of an Article token.

The components of the *articleSemantics* attribute category each have user-defined *tokens* as destinations; these represent the *user's* (optional) semantic categorization of important article keywords (for each relevant article and for *each* listed semantic category, the user is prompted to provide an important keyword that is to be associated semantically, in the specified category, with that article). This provides *optional* semantic content for articles, which may be useful during queries of the repository[2]. Such semantic content allows the user to dynamically create arbitrary *subtopic* queries of the repository; these subtopic queries provide a finer query-granularity than that allowed by the original static topic hierarchy.

See figure 8.2 for a graphical depiction of the above *aggregation* and *attribute classification* structures for the current example.

[2]See section 8.5.

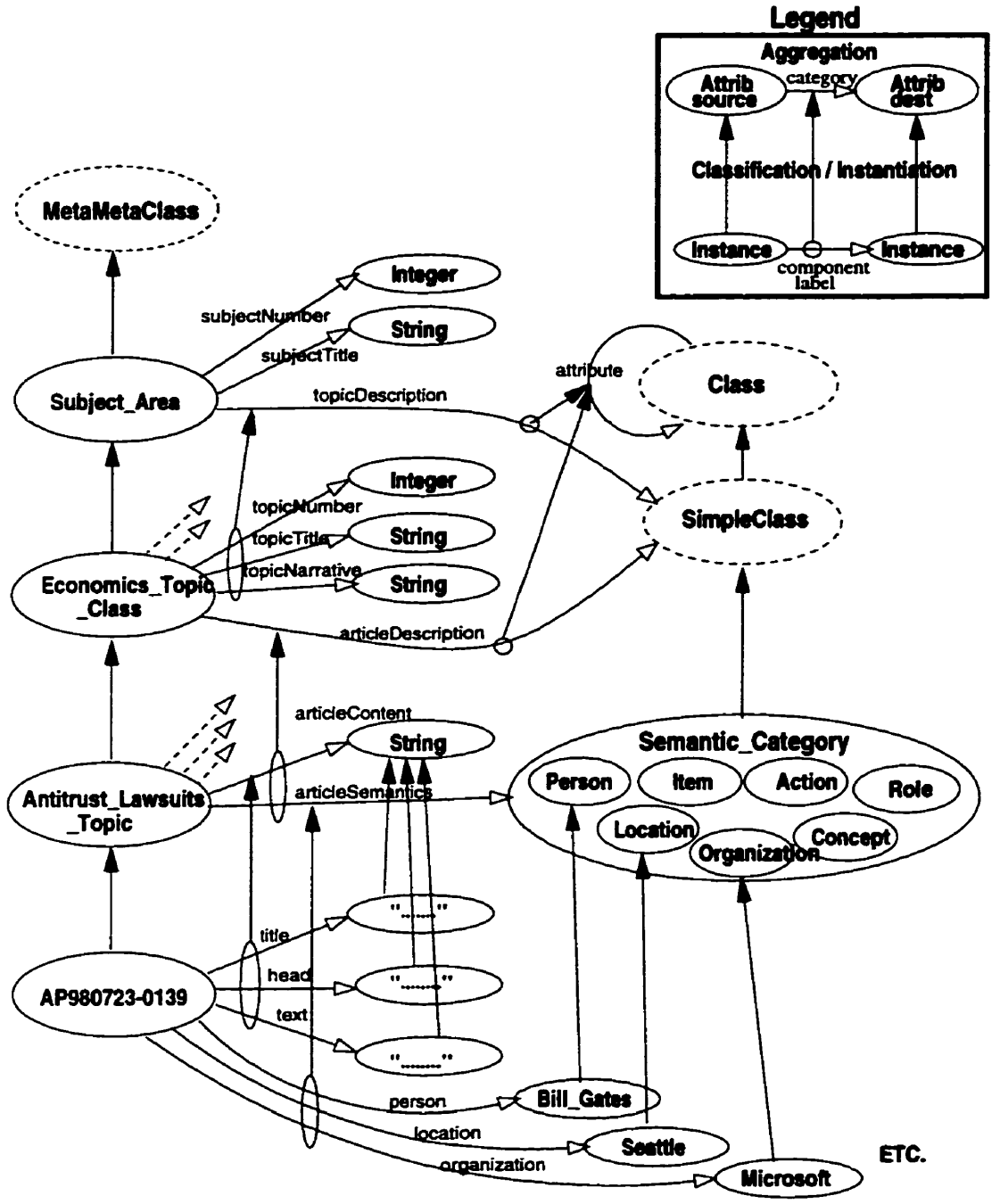


Figure 8.2: Example *aggregation* and *attribute classification* structure of the news-article filtering model.[3]

[3]Dashed arrows indicate attribute-tokens omitted from the diagram that have been instantiated from the attribute-simpleclasses with labels: *subjectNumber*, *subjectTitle*, *topicNumber*, *topicTitle*, and *topicNarrative*.

The following attribute definitions illustrate the *attribute classification* hierarchy for the specific example of a user-provided *person* semantic-keyword, as illustrated in figure 8.2:

1. **AttributeClass** := [Class, *attribute*, Class]
   IN AttributeClass
   ISA Attribute, Class

2. **AttribMeta** := [Economics_Topic_Class, *articleDescription*, SimpleClass]
   IN AttributeClass, MetaClass
   ISA SimpleClass

3. (a) **AttribSimple** := [Antitrust_Lawsuits_Topic, *articleSemantics*, Semantic_Category]
   IN AttribMeta, AttributeClass, SimpleClass
   ISA Token

   (b) **AttribSpecialization** := [Antitrust_Lawsuits_Topic, *articleSemantics*, Person]
   ISA AttribSimple

4. **AttribToken** := [AP980723-0139, *person*, Bill_Gates]
   IN AttribSpecialization, AttributeClass, Token

8.4 Extracting Information from the Information Base

The model was implemented as an information base using the JTelos language[4]. JTelos does not offer Telos' standard built-in Ask or Retrieve query commands; however, Telos' sparse framework provides an elegant means of extracting information from a repository through *custom implementation* of attribute-based queries. The following are the main contributing factors that enable such query-capability:

- Strict classification hierarchy between subject-area, topic, and news-article

[4]JTelos was developed by Techné Knowledge Systems Inc., Toronto. It provides an interface between a Telos information base and the Java programming language.

- Attribute first class citizenship
- Multiple instantiation

The following section describes several types of repository queries that have been implemented and provides suggestions for possible enhancement of the current query capability.

8.5 Query Implementation

Listed below are six example instances of implemented query-capability, where "*" and "NA" are defined as follows:

- "*" represents the knowledge base *target-individuals* that are the objects of the query.
- "NA" indicates that the field is not used by the query.

1. All **antitrust-articles** (topic instances) containing the *person-token* Bill_Gates:

Topic:	Antitrust_Lawsuits_Topic
Article:	*
Article-attribute (label):	*person*
Article-attribute (value):	Bill_Gates

2. All **articles and parent topics** where article contains the *person-token* Bill_Gates:

Topic:	*
Article:	*
Article-attribute (label):	*person*
Article-attribute (value):	Bill_Gates

3. All **parent topics** of the given article, *regardless* of article attributes:

Topic:	*
Article:	AP980723-0139
Article-attribute (label):	NA
Article-attribute (value):	NA

4. All **antitrust articles** (topic instances), *regardless* of article attributes:

Topic:	Antitrust_Lawsuits_Topic
Article:	*
Article-attribute (label):	NA
Article-attribute (value):	NA

5. All **antitrust articles** (topic instances) containing the *text* string "Bill Gates":

Topic:	Antitrust_Lawsuits_Topic
Article:	*
Article-attribute (label):	NA
Article-attribute (value):	"Bill Gates"

6. All **articles and parent topics** where article contains the *text* string "Bill Gates":

Topic:	*
Article:	*
Article-attribute (label):	NA
Article-attribute (value):	"Bill Gates"

Query types 5 and 6 involve *text* searches of the body of retrieved articles; these are useful for the specific cases where the user has declined to provide the system with article semantic-content information (provision of this information is optional, after the user has provided her relevance-judgement of the article).

The above queries are only a "first attempt" at query implementation; a number of enhancements are possible: for example, the article-attribute query field could employ disjunction, conjunction, negation, or any boolean function of article-attributes. This would lead to considerably more powerful query capability.

8.6 Virtual-Topics

An additional implemented feature is the ability to instantiate the documents returned by an arbitrary query into new custom user-defined *virtual-topics* within the information base. This feature provides the user with the ability to *dynamically* create and instantiate complex *subtopics* within the Telos information base; it forms a powerful document classification feature beyond the static topic-document classification capability provided by the filtering process's learned user-profile: users can, at any time, create special subtopics of interest and populate these new subtopics with instances returned from any query.

For example, assuming that the article-attribute query field employs conjunctions, then the documents returned from a query asking for "all Antitrust_Lawsuits_Topic documents that contain *both* of the *organization-tokens*: Microsoft and Netscape" could be dynamically instantiated into a *new topic* in the information base called, for example, Microsoft_Netscape_Dispute.

This capability provides the user with more control over the information base, allowing her to "personalize" the information base by arbitrarily classifying documents into a personal hierarchy, without being limited to the initial static topic-document classification hierarchy.

Chapter 9

Conclusions and Future Work

9.1 Contribution and Summary

The following summarizes the contribution of the current work:

1. A new approach to parametric information filtering that integrates a *multinomial* document model with *incremental* Bayesian methods as a means of learning user-profiles through *sequential data sampling over time*;

2. Integration of a news-document filtering process with a conceptual model *information base* of the process's *information domain.*

9.1.1 Information Filtering

The *probabilistic* information filtering model was chosen for investigation because, as Chapter 5 has shown, it is amenable to mathematical analysis, whereas nonprobabilistic models are primarily heuristic-based. The *multinomial* probabilistic model was chosen because, as discussed in Chapter 4, it is intuitively reasonable that document relevance is proportional to the *frequency* of occurrence of "important" terms in the document (rather than simply whether important terms *occur* in a document); this intuition has been confirmed empirically, for large vocabulary sizes, in [McC98]. The *Dirichlet conjugate prior* approach of estimating multinomial parameters was adopted because it accommodates

the updating of prior to posterior probabilities in *sequential data sampling* environments such as information filtering.

In empirical trials using 164,597 Associated Press news-documents and 50 user-topics, the current interactive probabilistic model was shown to outperform two interactive non-probabilistic models for the TREC F3 performance measure, receiving an average score of 34.38 over the 50 topics (compared to −27.14 and −6.22 for the two reference nonprobabilistic models). The nonprobabilistic models, however, exhibited better performance for the measure consisting of the geometric mean of precision and recall (this may account for their poor F3 performance, as explained in Section 6.2.3). The competitive performance of the probabilistic model verifies the validity of the theoretical approach developed in Chapter 5.

The overfitting discussed in Section 6.2.3 indicates that perhaps the complexity of the news-document filtering task is such that no single model can effectively model the relevance regions of the document space. This possibility gives rise to multiple-model approaches such as the SIGMA approach, discussed in [Kar96].

9.1.2 Conceptual Modeling

Information filtering literature generally treats user information needs as one-time needs only; i.e., relevant documents and their relationships to topics and subject areas are lost after the user has finished reading each document presented by the system. The current work provides for the persistent modeling of the relationships between documents, topics, and subject areas.

The information domain or subject world of a news-document filtering application was modeled with the Telos conceptual modeling language. The resulting information base was integrated with the filtering process to provide users with the ability to store filtering results persistently, in a conceptual manner. People conceptualize news-document domains in a particular structured form; conceptual modeling allows the automated mod-

eling of the semantic relationships among the model's concepts in a manner consistent with the way people view the relationships in the real world. The current implementation allows the user to build an automated model of a news-document information domain that is structured in the same manner as in the real world. In addition, the implemented information base provides a powerful query-capability that allows the user to selectively retrieve information previously returned by the filtering process.

9.2 Directions for Future Research

There are several possible directions for future research that expand upon the current information filtering work:

9.2.1 Document phrases

A refinement that has significant performance potential is the inclusion of word groups or phrases within the current probabilistic filtering model. Phrases clearly have more information content than isolated words. This refinement, however, is a departure from the "bag of words" model and may require a Bernoulli document model, rather than a multinomial model (as important phrases will occur in documents with much *lower* frequencies than important terms; therefore phrase *occurrence*, rather than phrase frequency, may be more appropriate in deciding document relevance).

9.2.2 Equivalent Sample Sizes

More work needs to be done to determine whether there exist *equivalent sample sizes* ess_R and ess_N that optimize performance. Figure 6.2 suggests that globally optimal *ess* values may not exist; however, there is evidence that *each topic* may possess its own optimal values for these parameters. For *very large* document data sources (such as exist in real-world document filtering environments), it may be possible to *learn* such optimal

ess values by topic. For example, as training progresses over time for a given topic, a heuristic search such as a genetic algorithm beam-search on *ess* values over a subset of $\mathcal{N}^2$ may yield higher-performing *ess* values for that topic than is possible using static (global) *ess* values. As topics are processed independently, such *ess* learning would be performed independently for each topic and therefore would be consistent with adaptive filtering requirements.

9.2.3 Alternative Priors Approaches

For certain applications, such as information filtering, the use of Dirichlet conjugate priors is an unwieldy approach for estimating the parameters of a multinomial model. Friedman and Singer [Fri98] provide the following features that characterize such applications:

- The set of possible outcomes (e.g., documents) from draws on a given multinomial distribution is very large, and often not known in advance;
- The number of training examples is small, relative to the number of possible outcomes;
- The outcomes that have non-zero probability constitute a relatively small subset of all possible outcomes, and are not known in advance.

Predictions based on a Dirichlet prior tend to distribute most of the probability mass to outcomes that will *not* occur in the training data. Friedman and Singer provide a *hierarchical prior* that allows *efficient* predictions over an infinite number of outcomes, without losing the ability to explicitly represent the model's posterior distribution (making it suitable for stochastic sampling tasks where explicit representations of approximated distributions are required). Their priors approach has potential value in stochastic modeling environments that are characterized by very large state spaces, such as (1) information filtering, and (2) Markov decision process parameter-estimation in reinforcement learning.

A related, although simpler, priors approach is provided by Ristad [Ris95]. It lacks the flexibility of Friedman and Singer's approach, but outperforms other standard multinomial estimation approaches, and may be of value for multinomial estimation in information filtering.

9.2.4 Alternative Frequency-Based Parametric Models

The Poisson family of distributions provides an alternative means of modeling term frequencies within text documents. Lewis [Lew98] observes that the use of mixtures of Poisson distributions has generally not proven more effective for batch-oriented ad hoc retrieval than the simple binary independence model. Bookstein [Boo83], however, provides an example of successful two-Poisson document modeling within a *sequential* relevance feedback ad hoc retrieval environment. As sequential feedback is inherent in the information filtering task, the latter work raises the question whether the Poisson distribution may be of value for document modeling within information filtering.

Bibliography

[Aal92] Aalbersberg, I., "Incremental Relevance Feedback", *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1992.

[Abr74] Abrial, J-R., "Data Semantics", in Klimbie and Koffeman, eds., *Data Management Systems*, North Holland, 1974. Cited in [Myl92a] and [Myl98].

[All96] Allan, J., "Incremental Relevance Feedback", *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1996. Cited in [Cal98].

[Ber93] Bertino, E., Martino, L., *Object-Oriented Database Systems*, Addison-Wesley, 1993.

[Boo83] Bookstein, A., "Information Retrieval: A Sequential Learning Process", *Journal of the American Society for Information Science*, Vol. 34, No. 5, pp. 331-342, 1983.

[Bor90] Borgida, A., "Knowledge Representation and Semantic Data Modeling: Similarities and Differences", *Proceedings Entity-Relationship Conference*, Geneva, 1990. Cited in [Myl92a].

[Buc95] Buckley, C., Salton, G., "Optimization of Relevance Feedback Weights", *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1995.

[Cal98] Callan, J., "Learning While Filtering Documents", *Proceedings of the Twenty-First Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998.

[Cas97] Castillo, E., Hadi, A., Solares, C., "Learning and Updating of Uncertainty in Dirichlet Models", *Machine Learning*, Vol. 26, 1997.

[Che76] Chen, P. P., "The Entity-Relationship Model: Towards a Unified View of Data", *ACM Transactions on Database Systems*, Vol. 1, No. 1, 1976.

[Coa90] Coad, P., Yourdon, E., "Object-Oriented Analysis", *IEEE*, 1990.

[Con91] Constantopoulos, P., Jarke, M., Mylopoulos, J., Vassiliou, Y., "The Software Information Base: A Server for Reuse". Cited in [Myl92a].

[Cra98] Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., Slattery, S., "Learning to Extract Symbolic Knowledge from the World Wide Web", *AAAI-98: Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pp. 509-516, AAAI Press, 1998.

[DeM78] DeMarco, T., *Structured Analysis and System Specification*, Yourdon Press, 1978.

[Fer99] Ferber, J., *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley, 1999.

[Fri98] Friedman, N., Singer, Y., "Efficient Bayesian Parameter Estimation in Large Discrete Domains", *Advances in Neural Information Processing Systems 11*, 1998.

[Fuh93] Fuhr, N., "Lecture Notes on Information Retrieval", Chapter 6, University of Dortmund, 1993.

[Gel95] Gelman, A., Carlin, J., Stern, H., Rubin, D., *Bayesian Data Analysis*, Chapman and Hall, 1995.

[Gre84] Greenspan, S., "Requirements Modeling: A Knowledge Representation Approach to Software Requirements Definition", Ph.D. Thesis, Department of Computer Science, University of Toronto, 1984.

[Hah98] Hahn, U., Schnattinger, K., "Towards Text Knowledge Engineering", *AAAI-98: Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pp. 524-531, AAAI Press, 1998.

[Har92] Harman, D., "Relevance Feedback Revisited", *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1992.

[Hul98] Hull, D., *TREC-7 Filtering Track Description*, Version 1.2 (Final), david.hull@xrce.xerox.com, May 12, 1998.

[Kar96] Karakoulas, G., Ferguson, I., "A Computational Market Model for Multi-Agent Learning", 1996.

[Kei97] Keim, M., Lewis, D., Madigan, D., "Bayesian Information Retrieval: Preliminary Evaluation", *Preliminary Papers of the Sixth International Workshop on Artificial Intelligence and Statistics*, pp. 303-310, 1997.

[Kiv94] Kivinen, J., Warmuth, M., "Exponentiated Gradient versus Gradient Descent for Linear Predictors", Technical Report UCSC-CRL-94-16, Basking Center for Computer Engineering & Information Sciences, University of California Santa Cruz. Cited in [Lew96].

[Kou89] Koubarakis, M., Mylopoulos, J., Stanley, M., Borgida, A., "Telos: Features and Formalization", Technical Report KRR-TR-89-4, Department of Computer Science, University of Toronto, February 1989.

[Leh92] Lehmann, F., ed., *Semantic Networks in Artificial Intelligence*, Pergamon Press, 1992.

[Lev77] Levesque, H., "A Procedural Approach to Semantic Networks", M.Sc. Thesis, Department of Computer Science, University of Toronto, 1977.

[Lew96] Lewis, D., Schapire, R., Callan, J., Papka., R., "Training Algorithms for Linear Text Classifiers", *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1996.

[Lew98] Lewis, D., "Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval", AT&T Labs—Research, *ECML-98: Tenth European Conference on Machine Learning*, 1998.

[Lyc94] Lycos Inc., http://www.lycos.com/, *The Lycos "Catalog of the Internet"*, Carnegie Mellon University, 1994.

[Mar89] Maritz, J., Lwin, T., *Empirical Bayes Methods*, Chapman and Hall, 1989.

[McC98] McCallum, A., Nigam, K., "A Comparison of Event Models for Naive Bayes Text Classification", *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[Mit97] Mitchell, T., *Machine Learning*, WCB/McGraw-Hill, 1997.

[Myl76] Mylopoulos, J., Borgida, A., Cohen, P., Roussopoulos, N., Tsotsos, J., Wong, H., "A Natural Language Understanding System for Data Management", *Information Systems*, Vol. 2, No. 2, 1976.

[Myl80] Mylopoulos, J., Bernstein, P., Wong, H., "A Language Facility for Designing Interactive Database-Intensive Applications", *ACM Transactions on Database Systems*, Vol. 5, No. 2, 1980.

[Myl90] Mylopoulos, J., Borgida, A., Jarke, M., Koubarakis, M., "Telos: A Language for Representing Knowledge About Information Systems (Revised)", Technical Report KRR-TR-89-1, Department of Computer Science, University of Toronto, August 1990.

[Myl92a] Mylopoulos, J., "Conceptual Modeling and Telos", in Loucopoulos, P., and Zicari, R., eds., *Conceptual Modeling, Databases, and CASE: An Integrated View of Information Systems Development"*, Wiley, 1992.

[Myl92b] Mylopoulos, J., "The PSN Tribe", *Computers & Mathematics with Applications, Vol. 23, No. 2-5*, pp. 223-241, 1992. Published as a special issue in [Leh92].

[Myl98] Mylopoulos, J., "Information Modeling in the Time of the Revolution", *Information Systems*, Vol. 23, No. 3/4, pp. 127-155, 1998.

[Oar96] Oard, D., Marchionini, G., "A Conceptual Framework for Text Filtering", *Journal of User Modeling and User-Adapted Interaction*, 1997.

[Por80] Porter, M., "An Algorithm for Suffix Stripping, Program", *Automated Library and Information Systems*, Vol. 14, No. 3, pp. 130-137, 1980.

[Ris95] Ristad, E., "A Natural Law of Succession", Research Report CS-TR-495-95, Department of Computer Science, Princeton University, 1995.

[Rob76] Robertson, S., Sparck Jones, K., "Relevance Weighting of Search Terms", *Journal of the American Society for Information Science*, Vol. 27, No. 3, pp. 129-146, May-June 1976. Cited in [Lew98].

[Rob94] Robert, C., *The Bayesian Choice: A Decision-Theoretic Motivation*, Springer-Verlag, 1994.

[Roc71] Rocchio, Jr., J., "Relevance Feedback in Information Retrieval", in Salton, G., ed., *The SMART Retrieval System: Experiments in Automatic Document Processing*, pp. 313-323, Prentice-Hall, 1971. Cited in [Lew96].

[Ros77] Ross, D., "Structured Analysis (SA): A Language for Communicating Ideas", *IEEE Transactions on Software Engineering*, Vol. SE-3, No. 1, January 1977.

[Sah98] Sahami, M., "Using Machine Learning to Improve Information Access", Ph.D. Thesis, Department of Computer Science, Stanford University, 1999.

[Sal83] Salton, G., McGill, M., *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983.

[Sal87] Salton, G., Buckley, C., "Term Weighting Approaches in Automatic Text Retrieval", Technical Report 87-881, Department of Computer Science, Cornell University, November 1987.

[Sta86] Stanley, M., "CML: A Knowledge Representation Language with Applications to Requirements Modeling", M.Sc. Thesis, Department of Computer Science, University of Toronto, 1986.

[Tho92] Thomason, R., "NETL and Subsequent Path-Based Inheritance Theories", *Computers & Mathematics with Applications, Vol. 23, No. 2-5*, pp. 179-204, 1992. Published as a special issue in [Leh92].

[vanR79] van Rijsbergen, C., *Information Retrieval*, second ed., Chapter 6, Butterworths, London, 1979.

[Wid85] Widrow, B., Stearns, S., *Adaptive Signal Processing*, Prentice-Hall, 1985. Cited in [Lew96].

[Yah94] Yahoo! Inc., http://www.yahoo.com/, 1994.

[Yu95] Yu, E., "Modelling Strategic Relationships for Process Reengineering", Ph.D. Thesis, Department of Computer Science, University of Toronto, 1995.