

RCV1: A New Benchmark Collection for Text Categorization Research (DRAFT)

David D. Lewis

*Independent Consultant
Chicago, IL, USA*

LYRL2002@DAVIDDLEWIS.COM

Yiming Yang

*Language Technologies Institute & Computer Science Department
Carnegie Mellon University*

YIMING@CS.CMU.EDU

Tony Rose

*Reuters Ltd.
London, UK*

TGR@TONYROSE.NET

Fan Li

*Language Technologies Institute
Carnegie Mellon University*

HOSTLF@CS.CMU.EDU

Editor: ???

Abstract

Reuters Corpus Volume I (RCV1) is an archive of over 800,000 manually categorized newswire stories recently made available for research purposes. Use of this data for research on supervised learning for text categorization requires a detailed understanding of the real world constraints under which the data was produced. Drawing on interviews with Reuters personnel and access to Reuters documents, we describe the coding policy and quality control procedures used in producing the RCV1 data, the intended semantics of the hierarchical category taxonomies, and the corrections necessary to remove errorful data. We refer to the corrected data as RCV1-v2. We benchmark several widely-used supervised learning methods on RCV1-v2, illustrating the collection's properties, suggesting new directions for research, and providing baseline results for future studies. We make available detailed, per-category experimental results, as well as corrected versions of the category assignments and taxonomy structures, via online appendices.

Keywords: test collection, controlled vocabulary indexing, operational systems, applications, evaluation, methodology, effectiveness measures, support vector machines, SVMs, nearest neighbor, Rocchio

1. Introduction

Text categorization is the automated assignment of natural language texts to predefined categories based on their content. It is a supporting technology in several information processing tasks, including controlled vocabulary indexing, routing and packaging of news and other text streams, content filtering (e.g. spam, pornography), information security, help desk automation, and others. Closely related technology is applicable to other classification tasks on text, including classification with respect to personalized or emerging classes (e.g. alerting systems, topic detection and tracking), non-content based classes (e.g. author iden-

tification, language identification), and to mixtures of text with other data (e.g. cross-media classification, text mining).

Research interest in text categorization has been growing in machine learning, as well as in information retrieval, computational linguistics, and other fields. This partly reflects the importance of text categorization as an application area for machine learning, but also results from the availability of *text categorization test collections*. These are collections of documents to which human indexers have assigned categories from a predefined set. Test collections enable researchers to test ideas without hiring indexers, and (ideally) to objectively compare results with published studies.

Existing test categorization test collections (13, 24, 31) suffer from one or more of the following weaknesses: few documents, lack of the full document text, inconsistent or incomplete category assignment, peculiar textual properties, and/or limited availability. These difficulties are exacerbated by a lack of documentation on how the collections were produced and on the nature of their category systems. The problem has been particularly severe for researchers interested in hierarchical text categorization who, due to the lack of good collections and good documentation, have often been forced to invent hierarchies without a clear basis in the data (8, 28).

Even if current collections were perfect, however, there would be an ongoing need for new ones. Just as machine learning algorithms can overfit by tuning a classifier’s parameters to the accidental properties of a training set, a research community can overfit by refining algorithms that have already done well on the existing data sets. Only by periodically testing algorithms on new test collections can progress be verified.

A data set recently made available¹, Reuters Corpus Volume 1 (RCV1) (20), has the potential to address many of the above weaknesses. It consists of over 800,000 newswire stories that have been manually coded using three category sets. However, RCV1 as distributed is simply a collection of newswire stories, not a test collection. It includes known errors in category assignment, provides lists of category descriptions that are not consistent with the categories assigned to articles, and lacks essential documentation on the intended semantics of category assignment.

This paper attempts to provide the necessary documentation, and to describe how to eliminate miscodings where possible. We begin in Section 2 by describing the operational setting in which RCV1 was produced, with particular attention to the categories and how they were assigned. Besides being crucial to understanding the semantics of the category assignments, the insight into operational text categorization may be of independent interest. Section 3 examines the implications of the production process for the use of RCV1 in research, and Section 4 summarizes the changes we recommend to produce a maximally correct test collection, which we call RCV1-v2.

Sections 2 to 4 are based on Reuters documentation, interviews with Reuters personnel, and statistical analysis of the documents and categories. To complement this analysis, we provide benchmark results on RCV1 for well-known supervised learning approaches to text categorization. These results provide future users with a standard for comparison, as well as reassurance that the tasks posed by the collection are neither trivial nor impossible. Section 5 gives the design of our experiments, Sections 6 & 7 discuss the algorithms and

1. <http://about.reuters.com/researchandstandards/corpus/index.asp>

text representation, and Section 8 presents the benchmark results and observations. We end with some thoughts on research directions the new collection may support.

2. Coding the RCV1 Data

Apart from the terrible memories this stirs up for me personally (coding stories through the night etc.), I can't find fault with your account. Most accurate.

– Reuters editor commenting on our summary of coding procedures.

The RCV1 data was produced in an operational setting at Reuters, Ltd., under procedures that have since been superseded. Only later was use of the data in research contemplated. Information that in a research setting would have been retained was therefore not recorded. In particular, no formal specification remains of the coding practices at the time the RCV1 data was produced. However, by combining related documentation and interviews with Reuters personnel we have largely reconstructed those aspects of coding relevant to text categorization research.

2.1 The Documents

Reuters is the largest international text and television news agency. Its editorial division produces some 11,000 stories a day in 23 languages. Stories are both distributed in real time and made available via online databases and other archival products.

RCV1 is drawn from one of those online databases. It was intended to consist of all English language stories produced by Reuters between 20-Aug-1996 and 19-Aug-1997.² The data is available on two CD-ROMs and has been formatted in XML.³ Both the archiving process and later preparation of the XML dataset involved substantial verification and validation of the content, attempts to remove spurious or duplicated documents, normalization of dateline and byline formats, addition of copyright statements, and so on.

The stories cover the range of content typical of a large English language international newswire. They vary from a few hundred to several thousand words in length. Figure 1 shows an example story (with some simplification of the markup for brevity).

2.2 The Categories

To aid retrieval from database products such as Reuters Business Briefing (RBB), category codes from three sets were assigned to stories. The code sets were designed to meet customer requirements for access to corporate/business information, with the main focus on company coding and associated topics. With the introduction of the RBB product the focus broadened to the end user in large corporations, banks, financial services, consultancy, marketing, advertising and PR firms.

2. At least one French story, document 731137, is known to have been included, but the number of such accidents was small.

3. Further details are available at <http://about.reuters.com/researchandstandards/corpus/>

```

<?xml version="1.0" encoding="iso-8859-1" ?>
<newsitem itemid="2330" id="root" date="1996-08-20" xml:lang="en">
<title>USA: Tylan stock jumps; weighs sale of company.</title>
<headline>Tylan stock jumps; weighs sale of company.</headline>
<dateline>SAN DIEGO</dateline>
<text>
<p>The stock of Tylan General Inc. jumped Tuesday after the maker of
process-management equipment said it is exploring the sale of the
company and added that it has already received some inquiries from
potential buyers.</p>
<p>Tylan was up $2.50 to $12.75 in early trading on the Nasdaq market.</p>
<p>The company said it has set up a committee of directors to oversee
the sale and that Goldman, Sachs & Co. has been retained as its
financial adviser.</p>
</text>
<copyright>(c) Reuters Limited 1996</copyright>
<metadata>
<codes class="bip:countries:1.0">
  <code code="USA"> </code>
</codes>
<codes class="bip:industries:1.0">
  <code code="I34420"> </code>
</codes>
<codes class="bip:topics:1.0">
  <code code="C15"> </code>
  <code code="C152"> </code>
  <code code="C18"> </code>
  <code code="C181"> </code>
  <code code="CCAT"> </code>
</codes>
<dc element="dc.publisher" value="Reuters Holdings Plc"/>
<dc element="dc.date.published" value="1996-08-20"/>
<dc element="dc.source" value="Reuters"/>
<dc element="dc.creator.location" value="SAN DIEGO"/>
<dc element="dc.creator.location.country.name" value="USA"/>
<dc element="dc.source" value="Reuters"/>
</metadata>
</newsitem>

```

Figure 1: An example Reuters Corpus Volume 1 document.

2.2.1 TOPIC CODES

Topic codes were assigned to capture the major subjects of a story. They were organized in four hierarchical groups: CCAT (Corporate/Industrial), ECAT (Economics), GCAT (Government/Social), and MCAT (Markets). This code set provides a good example of how controlled vocabulary schemes represent a particular perspective on a data set. The RCV1 articles span a broad range of content, but the code set only emphasizes distinctions relevant to Reuters' customers. For instance, there are three different Topic codes for corporate ownership changes, but all of science and technology is a single category (GSCI).

2.2.2 INDUSTRY CODES

Industry codes were assigned based on types of businesses discussed in the story. They were grouped in 10 subhierarchies, such as I2 (METALS AND MINERALS) and I5 (CONSTRUCTION). The Industry codes make up the largest of the three code sets, supporting many fine distinctions.

2.2.3 REGION CODES

Region codes fell into three groups, Countries, Regional Groupings, and Economic Groupings. Distinctions among the types of groupings were not formally represented in the code-set, however, and there was no hierarchical taxonomy defined.

2.3 Coding Policy

Explicit policies on code assignment increase consistency and compliance with operational goals. The Reuters coding reflected two broad policies⁴:

1. Minimum Code Policy: Each story was required to have at least one Topic code and one Region code.
2. Hierarchy Policy: Coding was to assign the most specific appropriate codes from the Topic and Industry sets, as well as all ancestors of those codes. There was no limit on the number of codes with the same parent that could be applied.

These policies were implemented by a combination of manual and automated means during coding, as discussed below and in Section 3.3.

2.4 The Coding Process

During the years 1996 and 1997, the period from which the corpus is drawn, Reuters produced just over 800,000 English language news stories per year. Coding was a substantial undertaking. At one point Reuters employed 90 people to handle the coding of 5.5 million English language stories per year.⁵ The consistency of coders with each other and with standards was periodically evaluated and found to be high. We have reported details of these evaluations elsewhere (20).

4. We have given these policies names for convenience. They were not named by Reuters.

5. This figure includes both English language stories produced by Reuters and ones obtained from other sources, and included additional code sets not present in the RCV1 data.

		Manually Corrected	
		No	Yes
Manually Edited	No	170,745	334,975
	Yes	288,851	23,289

Table 1: Number of stories in 1997 receiving manual editing and/or manual correction.

Coding of Reuters-produced stories was accomplished in three stages: autocoding, manual editing, and manual correction.

2.4.1 AUTOCODING

Stories first passed through a rule-based text categorization system known as *TIS* (Topic Identification System). Most codes had at least one rule that could assign them, but automated coding was not attempted for some codes believed to be beyond the capability of the technology. Two of the codes perceived to be difficult were GODD (human interest) and GOBIT (obituaries). It is interesting to note that these two categories proved in our experiments to be two of the most difficult to assign effectively.

In addition to rules that assigned codes based on the text of the story, some simple source-processing rules mapped codes that were manually assigned by journalists to equivalent codes in the RBB code set. For example, a story with the Editorial code SPO (Sport) would automatically receive the RBB code GSPO. Other source processing rules triggered on other parts of the markup, for instance assigning any story whose slug contained the string “BC-PRESS DIGEST” to the most general news code (GCAT).

2.4.2 MANUAL EDITING

The output of TIS was automatically checked for compliance with the Minimum Code policy. If so, the story was sent to a holding queue. If not, the story was first sent to a human editor. This editor would assign the codes they felt applied, while ensuring the story got at least one Topic and one Region code. Editors could also delete or change automatically assigned codes. Editors occasionally fixed errors in the formatting of the story during this phase, but their primary responsibility was correction of coding. The edited story then went to the holding queue for final review.

2.4.3 MANUAL CORRECTION IN THE HOLDING QUEUE

Every 6 hours, the holding queue was reviewed by editors, who had the opportunity to correct mistakes in coding. Once stories passed through the holding queue, they were batched up and loaded onto the database in blocks.

2.5 Coding Quality

Human coding is inevitably a subjective process. Studies have shown considerable variation in interindexer consistency rates for different data sets (3). The process described above represented a significant investment in attempting to achieve a high rate of consistency

and correctness. Beyond that, stories were also sampled periodically and feedback given to coders on how to improve their accuracy.

Table 1 attests to considerable success in that effort. It shows, for the year 1997, how many stories had autocoding that failed the Minimum Code test and thus underwent manual editing, as well as how many had at least one code corrected in the holding queue.⁶

A total of 312,140 stories had autocoding that failed the Minimum Code test and were thus manually edited. All were also reviewed by a second editor in the holding queue, but of these only 23,289 or 13.4% had codes changed by that second editor. In contrast, 334,975 (66.2%) of the 505,720 stories whose autocoding passed the Minimum Code test were changed in the holding queue. Admittedly, an annotation let editors reviewing the holding queue know which stories had been manually edited, and this may have influenced their judgment of which to correct. Still, Table 1 provides additional confidence in the consistency of coding.

2.6 The Evolution of Coding at Reuters

It should be mentioned that the above approach, based on TIS and manual correction, has since been superceded at Reuters. The rule-based approach of TIS had several drawbacks:

- Creating rules required specialized knowledge, thus slowing down the addition of new codes and the adaptation of rules to changes in the input.
- The rules did not provide an indication of the confidence in their output. There was thus no way to focus editorial correction on the most uncertain cases, nor any way of indicating (except by violation of coding policy) that new stories were appearing that would suggest changes or additions to the code set.

Consequently, Reuters now, in fact, uses a machine learning-based approach for text categorization. Classifiers are induced from large amounts of training data, with an inbuilt feedback loop to trigger the involvement of human editors (based on autocoding confidence scores) and analysis tools to indicate when new training data/categories may be required.

3. RCV1 and Text Categorization Research

A test collection is more than a corpus. In this section we consider how the production and character of RCV1 impact its use for text categorization research. In Section 4 we go on to describe how to correct errors in the raw RCV1 data (which we call RCV1-v1) to produce a text categorization test collection (which we call RCV1-v2), so in Section 3 we present statistics for both versions of the data, indicating when they are different.

3.1 Documents

RCV1 contains 35 times as many documents (806,791 for RCV1-v1, and 804,414 for RCV1-v2) as the popular Reuters-21578 collection⁷ and its variants (15, 31). Indeed, the only

6. RCV1 contains stories spanning parts of 1996 and 1997, so the number of stories in the corpus is not the same as the number of stories in Table 1.

7. Available from <http://www.daviddlewis.com/resources/testcollections/reuters21578/>

widely available text categorization test collection of comparable size is OHSUMED (5, 13) at 348,566 documents. While useful, OHSUMED has the disadvantages that it does not contain the full text of documents, its medical language is hard for non-experts to understand, and its category hierarchy (MeSH) is huge and structurally complex.

RCV1 is also “cleaner” than previous collections. Stories appear one to a file, and have unique document IDs. IDs range from 2286 to 810597 for RCV1-v1, and 2286 to 810596 for RCV1-v2. There are gaps in the range of IDs in the original RCV1-v1 ($810,597 - 2286 + 1 = 808,312 > 806,791$), and additional gaps (due to deleted documents) in RCV1-v2. The numeric order of IDs corresponds to the date order of stories, but does not have a direct relationship to time of appearance of stories within a day. (Indeed, since RCV1 was generated from a cleaned-up archival database, the notion of time of appearance of a story is not meaningful.)

XML formatting of both text and metadata simplify data preparation. In addition, the stories are from an archival database, rather than a raw newswire feed. This means fewer brief alerts (the infamous “blah, blah, blah” stories of Reuters-21578), corrections to previous stories, exact duplicates, and so on. While classification of raw newswires is certainly of interest, avoiding such anomalies allows more focus on machine learning research issues.

RCV1 contains all stories from an interval of one year. For temporal studies, this is a major advantage over Reuters-21578, which had bursty coverage of a fraction of a year. Unfortunately, only dates, not time stamps were preserved in RCV1, so patterns of stories within a single day cannot be studied.

3.2 Categories

RCV1 documents are categorized with respect to three controlled vocabularies: *Topics*, *Industries*, and *Regions*. In this section, we discuss the three RCV1 category sets and their implications for text categorization experiments. In particular, we describe the intended hierarchical structure (or lack of it) for each code set, something which is not made clear in the documentation accompanying RCV1.

3.2.1 TOPIC CODES

The file *topic_codes.txt* on the RCV1 CD-ROMs lists 126 Topic codes. However, some of these codes were not actually used by editors at the time the RCV1 data was categorized. At most, only the 113 codes in the subhierarchies Markets (codes starting with “M”), Corporate/Industrial (“C”), Economics (“E”, except ENT12), and Government/Social (“G”) were used. Other evidence, including a Reuters document with an alternate description of the Topics hierarchy, suggests that the eight codes in the numeric range G11 to G14, plus GEDU and MEUR, were also not available.

This leaves 103 Topic codes we believe were actually used for coding, and which we therefore recommend be used in text categorization experiments. We provide a list of valid Topic codes as Online Appendix 1. As it happens, all of these 103 codes occur at least once in both the RCV1-v1 and RCV1-v2 datasets. Their corpus frequencies span five orders of

magnitude, from 5 occurrences for GML (MILLENNIUM ISSUES), to 374,316 occurrences (381,327 in RCV1-v2) for CCAT (CORPORATE/INDUSTRIAL).⁸

The code symbols inserted in articles to indicate their membership in categories were internally structured to serve two related, but not identical, purposes:

1. Defining a hierarchy to support automated assignment of general codes on the basis of (manual or automated) assignment of more specific codes (Section 3.3).
2. Imposing an alphanumeric sort order that grouped related codes, aiding manual lookup.

For instance, the code C311 (DOMESTIC MARKETS) is a child in the hierarchy of code C31 (MARKETS/MARKETING), and also appears near related codes, e.g. C32 (ADVERTISING/PROMOTION), in an alphanumeric listing.

The original hierarchy used for automated assignment can be reconstructed as follows:

1. Replace the codes CCAT, ECAT, GCAT, and MCAT with the corresponding single letters C, E, G, and M.
2. To find the parent of a code, remove the minimal suffix such that the result is another code. The codes C, E, G, and M have as parent the root of the tree.

Researchers interested in hierarchical categorization are of course not restricted to using the original category hierarchy. In particular, it may be useful to introduce an additional level of the hierarchy corresponding to the high level numeric groupings that aided lookup. This can be done by first adding to the hierarchy the artificial codes C1-C4, E1-E7, G1, and M1, and then following the above procedure. We do not recommend actually assigning these artificial codes to documents and evaluating text categorization effectiveness on them.

Online Appendix 2 is an XML encoding of the original version of the hierarchy. It contains a total of ??? nodes: 103 valid Topics and 1 root node. Online Appendix 3 is an XML encoding of a hierarchy that includes the two-character truncations as a new intermediate layer. It contains a total of ??? nodes: 103 valid Topics, ??? nodes in the new intermediate layer, and 1 root node.

Online Appendix ? is a simple list of the 103 valid Topic codes. (DDL: Waiting to find out if will host everything at JMLR or everything at UCI KDD.)

Editors were able to assign any of the 103 Topic codes to a story, not just codes at leaf nodes of the hierarchy. They were instructed to use the most specific code applicable to a particular aspect of a story, a common indexing principle (9, pp. 28-30). Codes at internal nodes of the hierarchy thus acted much like named “Other” categories, implicitly forming a contrast set with their child codes.

However, in the RCV1 data a non-leaf code may also be present not because it was directly added, but because one its descendant codes was present and was automatically expanded. We call this “Other + expansion” semantics, to distinguish it from pure “Other” semantics. We discuss the implications of this for research use of RCV1 in Section 3.3.

8. Some Topic category frequencies are higher in RCV1-v2 than in RCV1-v1, despite RCV1-v1 having fewer documents, because RCV1-v2 fills in missing hierarchical expansions of Topic categories (Section 4).

3.2.2 INDUSTRY CODES

The file *industry_codes.txt* on the RCV1 CD-ROMs lists a total of 870 codes. However, only those with 6 (e.g. I98100) or 8 characters (e.g. I9741112) were ever intended for assignment to documents. There are 385 codes with 6 or 8 characters, but these include code I99999 (DUMMY) and eight 6-character codes labeled TEMPORARY, none of which were used during coding. Omitting these nine invalid codes leaves 376 codes that we recommend be used in experiments. We provide a list of these 376 valid Industry codes as Online Appendix 4.

Of the 376 valid Industry codes, 350 have at least one occurrence in the corpus. Nonzero category frequencies range from 2 for I5020030 (RESERVOIR CONSTRUCTION) and I5020050 (SEA DEFENCE CONSTRUCTION) to 34,788 (34,775 in RCV1-v2) for I81402 (COMMERCIAL BANKING).

The Industry categories incorporate many fine-grained distinctions in subject matter. (For instance, there are five variations on the real estate industry.) They therefore provide an excellent test of the ability of text categorization systems to make fine distinctions.

As with Topics, the Industry code symbols encode both a hierarchy and a numeric sort order. The unused codes in *industry_codes.txt* are there to support a legacy interface used by Reuters coders. The interface required Industry codes to have exactly 6 or 8 characters, regardless of hierarchy position. Entering a code of some other length required padding that code to 6 or 8 characters with trailing digits, usually but not always 0's. The intermediate length codes in *industry_codes.txt* were then needed to drive the hierarchy expansion software.

For research purposes the redundant codes should be removed. A simple approach is to delete for each textual description in *industry_codes.txt* all but the shortest code with that description, with these exceptions:

1. The codes I455 (HOUSEHOLD TEXTILES) and I647 (HOUSEHOLD TEXTILES) are distinct codes and should both be retained.
2. Code I65 (RETAIL DISTRIBUTION) should be deleted from the hierarchy, while code I64 (RETAIL DISTRIBUTION) is retained. Codes with the prefix I65 should then be considered children of I64.⁹
3. All codes with description DUMMY or TEMPORARY should be deleted.

Following this transformation, the structure of the hierarchy is clear in the edited file. The parent of a given code is simply the longest code which is a proper prefix of it, (with special handling for I65). The canonical form of any 6- or 8-character code appearing in a document is simply the longest code which is a prefix (proper or improper) of the code as it appears in the document.

Editors could assign any Industry code in the hierarchy (padding them to either 6 or 8 characters), except, apparently, the topmost (2-character) codes. Automated expansion up the hierarchy was done, but the RCV1 data as distributed contains only the expansions of 8 character codes to their immediate parents, not the complete expansions (Section 3.5).

9. It appears there was once a distinction between I64 (Retail - General) and I65 (Retail - Specialist), but this distinction was lost during evolution of the taxonomy.

We include a copy of the Industries hierarchy as Online Appendix 5 to this paper. In contrast to the case of two-character Topic codes, there does not appear to be any meaningful numeric groupings of Industry codes that are not already nodes in the original hierarchy.

3.2.3 REGION CODES

The file *region_codes.txt* on the RCV1 CD-ROMs contains 366 geographic codes, of which 296 occur at least once in the corpus. The Reuters documentation we could obtain suggests that all 366 of these codes were available to Reuters editors, and so are appropriate to use in experiments. We provide a list of these 366 valid Region codes as Online Appendix 6. Nonzero class frequencies span the range from 1 (for 10 codes in RCV1-v1 and 8 codes in RCV1-v2) to 266,239 (265,625 in RCV-v2) for USA.

In addition, 3 codes with a total of four occurrences occur in the RCV1 articles but not in the file *region_codes.txt* (bringing the total number of Region codes in RCV1-v1 to 299). These codes are CZ - CANAL ZONE (1 occurrence), CZECH - CZECHOSLOVAKIA (2 occurrences), and GDR - EAST GERMANY (1 occurrence). These codes appear to be errors, so in producing RCV1-v2 we replaced them by the codes PANA (PANAMA), CZREP (CZECH REPUBLIC), and GFR (GERMANY), respectively.

No hierarchically-based automatic expansion of Region codes was done by Reuters. However, Reuters personnel did view the Region codes as falling into three informal groups: Countries, Regional Groupings, and Economic Groupings. We include as Online Appendix 7 a coarse hierarchy based on this distinction, but researchers may well want to define additional hierarchical structure based on geographic, economic, political, or other criteria.

Whether assigning RCV1 Region codes is a good test of text categorization capability as opposed to named entity recognition capability (4), is debatable. It is clear, however, that assigning Region codes is not *solely* a named entity task. There are many stories that mention the United States, for instance, that are not assigned to the USA code, and there are Region codes which are not named entities, such as WORLD and DEVGCO (DEVELOPING COUNTRIES).

3.3 Coding Policy

Coding policies specify certain requirements for how coding should be done, beyond an editors' judgment of which codes capture the content of a particular text. As mentioned in Section 2.3, at least two coding policies, which we call the Hierarchy Policy and the Minimum Code Policy, were used by Reuters during the period the data in RCV1 was produced. We discuss here their implications for use of RCV1 as a test categorization test collection.

3.3.1 IMPLICATIONS FOR CORPUS PROPERTIES

The Hierarchy Policy required that when a Topic or Industry code was assigned to an article, all the codes which were ancestors of it in the Topic code hierarchy should be assigned as well. This creates some very high frequency codes (CCAT is assigned to 46% of the corpus), as well as strong, partially deterministic, dependencies between hierarchically related codes.

The Minimum Code Policy required that articles get at least one Region code and one Topic code. This policy probably did not greatly affect the codes assigned, since the

code sets themselves were designed to cover the likely content of the newswire. However, unlike the Hierarchy Policy, the Minimum Code Policy did require human coders to change their behavior: in cases where they might otherwise decide that no code applies, they were forced to choose some assignment. From a statistical standpoint, the Minimum Coding Policy introduces a weak dependence among all codes in a set.

3.3.2 IMPLICATIONS FOR ALGORITHM DESIGN

If one knows that the correct categorization of a document obeys coding policies, it is natural to attempt to modify a text categorization algorithm so its output obeys those policies. Whether doing this will actually improve the effectiveness of a given system is, however, less clear.

The obvious approach to implementing the Hierarchy Policy is to run a categorizer as usual, and then add the ancestors of all assigned categories if not already present. This runs the risk, however, of adding a high level category which was rejected by a well-trained classifier, on the basis of a low level category assigned by a less well-trained classifier.

How easy (and desirable) it is to implement the Minimum Code Policy varies with the text categorization method. For instance, a common strategy in text categorization is to create a separate binary classifier for each category. However, this approach is likely to assign no categories to some documents. One possible response is to train classifiers that estimate probability of class membership, and assign the most likely of the rejected categories.

3.3.3 IMPLICATIONS FOR EVALUATION

When testing algorithms on a corpus produced using a particular coding policy, should one disallow outputs that violate that policy? This is often done when testing algorithms for multiclass (1-of- k) categorization: only algorithms that assign exactly one category for each test document are allowed. In an operational setting the data model, APIs, or other constraints might require strict adherence to coding policy.

On the other hand, if we view the system’s output as something which will be reviewed and corrected by a human editor, a more relaxed approach is possible. Rather than forbidding outputs that violate coding policy, the effectiveness measure should measure the cost of correcting these policy violations, along with any other errorful assignments. The common approach in research papers of evaluating binary classification effectiveness separately for each category may be a reasonable way of capturing the cost of correction. This is the approach we adopt in reporting benchmark results in Section 8.

3.4 Were the Codes Manually Assigned?

It is reassuring that each RCV1 document was both autocoded and examined by at least one person, and sometimes two. It is very unlikely that subsets of the collection will be discovered, as was the case with Reuters-22173, that appear to simply have been missed during coding.

A different potential worry for researchers arises from the use of autocoding in preparing RCV1. If the RCV1 codes were largely assigned by rules, then machine learning research

with the corpus would become an exercise in rediscovering those (possibly simple and uninteresting) rules.

Fortunately, every story had its automated coding checked by at least one editor.¹⁰ Stories which failed the Minimum Code test were seen by two editors, the first of whom always added or changed at least one of the original TIS-assigned codes. Further, Table 1 shows that not only were all stories manually reviewed, but 79% had at least one code manually changed. This argues that, despite the use of automated coding, RCV1 can be considered a manually categorized test collection.

3.5 Coding Errors

Table 1 suggests a high degree of interindexer consistency for RCV1, and thus suggests low levels of both disagreements and simple errors. On the other hand, knowledge of the coding policies shows that the level of errors is not zero.

There are 2,377 documents (0.29% of RCV1-v1) which violate the Minimum Code Policy by having either no Topic codes (2,364 documents) or no Region codes (13 documents). There are 14,786 documents (1.8% of RCV1-v1) which violate the Hierarchy Policy on Topic codes, i.e. an ancestor of some assigned Topic code is missing. Of the 103 Topic codes that were used for the RCV1 data, 21 have at least one child in the Topics hierarchy. Each of these 21 codes is missing from at least one document to which the Hierarchy Policy says it should have been assigned. A total of 25,402 occurrences of these 21 codes are missing.

With respect to Industry codes, application of the Hierarchy Policy was also imperfect:

- The immediate parent of an 8-character code was automatically added to the document in most cases, but these cases were missed:
 - There are a total of eight 8-character codes with one or more appearances in the corpus whose immediate parent code is not its 6-character truncation. They are (with parent shown in parentheses): I1610107 (I16100), I1610109 (I16100), I4752105 (I47520), I9741102 (I97400), I9741105 (I97400), I9741109 (I97400), I9741110 (I97400), and I9741112 (I97400). Assignment of parent codes which are not truncations happened in only 7.1% to 46.6% of documents containing the child code, depending on the particular child category. This contrasts with essentially 100% assignment of parent codes which were 6-character truncations of 8-character codes.
 - A single document containing two children of I01001 is missing I01001. This appears to be a simple error. By contrast, all 12,782 other occurrences of children of I01001 are in documents that contain I01001.
- No grandparents or higher level ancestors of 8-character codes appear to have been automatically added, nor any ancestors of 6-character codes. The few cases where both a code and one of these other ancestors are assigned to a document appear to result from a manual editorial decision to assign both.

10. An exception is the code GMIL, for millenium-related stories. This was automatically assigned, possibly without manual checking, sometime after the period of the corpus. There may have been a very small number of other such codes, though we have not found evidence for this.

These violations result from some combination of human error, glitches in the hierarchical expansion software, and/or omissions of some data when producing RCV1. In the next section we propose how these errors should (or should not) be corrected in using the corpus for experimental purposes.

4. RCV1-v2: A New Text Categorization Test Collection

Since all evidence suggests that violations of the Hierarchy Policy and Minimum Coding Policy are simple errors, removing these violations will produce more accurate results in any classification experiments made using RCV1. In the section we describe the procedures necessary to remove these errors. We call the resulting corrected text categorization test collection RCV1-v2 (for version 2), while referring to the uncorrected original version (as used, for instance, in the TREC-10 experiments (17)) as RCV1-v1.

The following corrections convert RCV1-v1 to RCV1-v2:

1. Remove from the corpus the 13 documents that violate the Minimum Code Policy due to missing all Region codes, and the 2,364 documents that violate the policy due to missing all Topics. This leaves a total of 804,414 documents to use in research. Online Appendix 8 provides a list of the IDs of documents to omit.
2. For each Topic code present in a document, add all missing ancestors of the code. This adds 25,402 Topic code assignments (Online Appendix 9).
3. Replace the four errorful occurrences of Region codes, as described in Section 3.2.3 and listed in Online Appendix 10.

We applied these corrections to the corpus before producing the results reported in the next section.

We decided against trying to correct violations of the Hierarchy Policy for Industry codes. Our reasoning was based on considering the following partition of the Industry codes:

- Leaf codes. These would not be affected under any correction scheme.
- Non-leaf 6-character codes with 8-character children. All but 4 of these are assigned in 100% of the cases one or more of their children are present. One (I01001) is missing from only one of the 12,783 documents that contain one or more of its children. The three remaining codes I16100, I47520, and I97400, as mentioned above, are assigned in between 7.1% and 46.6% of the cases that the Hierarchy Policy suggests they should be. These are exactly the three cases where there is a nonstandard child-parent relationship (the numeric code for the parent is not the truncation of the numeric code for the child). We have not been able to determine if the assignments of these codes which are present in the corpus represent a partially successful automated assignment or, conversely, a complete omission of automated assignment in combination with manual decisions to assign the codes in certain cases. If we modified the corpus by assigning these codes when their children are present, it is unclear whether we would be respecting the intended semantics, or washing it out.

- Other non-leaf codes. There appear to have been no automated assignments of these non-leaf codes, so each occurrence corresponds to a manual judgment that this high level code is appropriate. Automated expansion would swamp these manual judgments with large numbers of automated assignments (up to 100-fold more), producing an arguably less interesting classification task.

However, users of the corpus should recognize that leaving the Industry assignments in their original form means that some non-leaf Industry categories have “Other + expansion” semantics (6-character codes with 8-character children, with the semantics of I16100, I47520, and I97400 being unclear), and the rest have pure “Other” semantics (Section 3.2.1).

4.1 Availability of RCV1-v2 Data

Online Appendices 8, 9, and 10 give just the changes necessary to produce RCV1-v2 from RCV1. Alternatively, the *complete* corrected sets of RCV1-v2 category labels are provided in Online Appendices 11, 12, and 13. In addition, two versions of the complete set of RCV1-v2 documents in vector form (described in Section ?) are provided as Online Appendices 14 and 15.

5. Benchmarking the Collection: Methods

An important part of the value of a machine learning data set is the availability of published benchmark results. Among other things, good benchmark results serve to ensure that apparently superior new methods are not being compared to artificially low baselines. We therefore ran several of the most popular supervised learning approaches on the RCV1-v2 data, both to provide such a benchmark, and to verify that our corrections to the data did not introduce any new anomalies.

5.1 Training/Test Split

We split the RCV1-v2 documents chronologically into a training set (articles published from 20-Aug-1996 to 31-Aug-1996; document IDs 2286 to 26150) and test set (1-Sep-1996 to 19-Aug-1997; document IDs 26151 to 810596). The result is a split of the 804,414 RCV1-v2 documents into 23,149 training documents and 781,265 test documents. This chronological boundary is the same used in the TREC-10/2001 filtering track (17). However the TREC-10/2001 filtering track used the raw RCV1 data (???? uncorrected RCV1-v1 documents split into 23,307 training documents and 783,484 test documents), so the TREC results are not comparable with ours.

A chronological split, rather than a random one, is desirable since the majority of operational text categorization tasks require training on currently available material, and then applying the system to material that is received later. The chronological breakpoint we chose has the advantage of giving almost all Topic categories two or more training examples, while still retaining most of a complete year as test data.

5.2 Categories

We provide benchmark data on all categories that evidence indicates were available to Reuters indexers, even those with few or no positive examples. There are 103 Topic categories, 101 with one or more positive training example on our training sets, and 103 with one or more positive test examples. There are 376 Industry categories, 313 with positive training examples, and 350 (including all of the 313) with positive test examples. And there are 366 Region codes, 228 with positive training examples, and 296 (including all of the 228) with positive test examples.¹¹

5.3 Effectiveness Measures

We measure the effectiveness of a text classifier on a single category with the F_β measure (11):

$$F_\beta = \frac{(\beta^2 + 1)A}{(\beta^2 + 1)A + B + \beta^2 C} \quad (1)$$

where A is the number of documents a system correctly assigns to the category, B is the number of documents a system incorrectly assigns to the category (false alarms), and C is the number of documents that belong to the category but which the system does not assign to the category. We report values for $\beta = 1.0$, which corresponds to the harmonic mean of recall and precision. The F-measure as presented above is undefined when $A = B = C = 0$. We treated this as a constant value, $c_F = 0.0$, though arguments can also be made for other values (11).

To measure effectiveness across a set of categories we use both the *macroaverage* (un-weighted average of effectiveness across all categories) and the *microaverage* (effectiveness computed from the sum of per-category contingency tables) (10, 26).

6. Benchmarking the Collection: Classifiers

We tested three supervised learning approaches that have been widely used in text categorization experiments: support vector machines (SVMs) (6), weighted k -Nearest Neighbor (k NN) (29), and Rocchio-style algorithms (1). All classifiers trained by these algorithms consist of a model that assigns scores to documents, and a threshold which such a score must exceed for the category to be assigned.

Our training algorithms (except for SVM.2 – see below) first train the scoring model for a category, and after that train the threshold for the category. The threshold is trained using the *SCut* (32) procedure. The training set is randomly split into 5 subsets. Five runs are done, each using four of those subsets for training and the remaining subset (the *validation* set) for finding the threshold that gives highest effectiveness on the validation set. A final scoring model is then trained on the entire training set, and used with a threshold equal to the mean of the thresholds found on the 5 cross-validation folds. Other thresholding strategies (32) were evaluated on the training data, but *Scut* was consistently superior on RCV1 so only results for it are presented.

11. All counts are the same for RCV1-v1, if the four invalid assignments of three invalid Region categories in RCV1-v1 are ignored.

6.1 SVM

SVM algorithms find a linear decision surface (hyperplane) with maximum margin between it and the positive and the negative training examples for a class (6). SVMs using nonlinear kernel functions are also possible, but have not shown a significant advantage in past text categorization studies, and are not investigated here.

SVMs have outperformed competing approaches in a number of recent text categorization studies, but there has been some suggestion that they choose a poor decision threshold when the numbers of positive and negative examples are very different, as they are for low frequency categories in text categorization (33). We therefore used in our baselines two SVM variants that adjust for category frequency:

- Approach 1 (SVM.1) (33): A single SVM classifier was trained for each category. SVM training used the *SVM_Light* (7) package¹², version 3.50. All parameters were left at default values. The resulting classifier’s threshold was then replaced by an SCut threshold as described above.
- Approach 2 (SVM.2) (12): Multiple SVM classifiers were trained using different weightings (*SVM_Light*’s *-j* parameter) of positive examples to negative examples: 0.1, 0.2, 0.4, 0.6, 0.8, 0.9, 1.0, 1.25, 1.5, 2.0, 3.0, 4.0, 6.0, 8.0, 10.0, and 15.0. Leave-one-out cross-validation (LOO) (*SVM_Light*’s *-x 1* parameter) was then used to compute a training set contingency table corresponding to each setting. The $F_{1.0}$ value for each setting was computed from its contingency table, the best setting for each category chosen, and a classifier trained on all training data using that setting. SVM.2 was the top-ranked approach in the batch filtering and routing tasks in the TREC-10 evaluation (17).

Given the robustness of SVMs to high dimensionality, no feature selection was done for the SVM runs.

6.2 kNN

Weighted k NN classifiers have been consistently strong performers in text categorization evaluations (29, 31). The variant we used here finds, for a given test document, the k nearest training documents using the cosine similarity measure. Then for each category, the scores of the nearest neighbors belonging to that category are summed to produce the score of the category for the document. That is, the score of category c_j with respect to test document $\vec{x}$ (a vector of term weights) is:

$$s(c_j, \vec{x}) = \sum_{\vec{d} \in R_k(\vec{x})} \cos(\vec{x}, \vec{d}) I(\vec{d}, c_j) \quad (2)$$

where $\vec{d}$ is a training document; $R_k(\vec{x})$ is the set of the k training documents nearest to $\vec{x}$; and $I(\vec{d}, c_j)$ is indicator function whose value is 1.0 if $\vec{d}$ is a member of category c_j , and 0.0 otherwise.

12. Available from http://www.joachims.org/svm_light

The resulting score is then compared to the category threshold to determine whether or not to assign the category to the test document. The value of k is tuned using 5-fold cross-validation on the training set.

The kNN method is more sensitive to nonrelevant features than SVMs are, so the vectors used to compute the above scores first had feature selection applied. Features were ranked using the χ^2 -max criterion, i.e. the maximum χ^2 of the feature over all categories in the category set (19, 30). Feature sets of the following sizes were tested:

50, 100, 200, 400, 600, 800, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, 12000, 14000, 16000, 20000, 25000, 30000, 47152 (*all features*)

The optimal feature set size was chosen by 5-fold cross validation on the training set. A different feature set was chosen for each of the three category sets (Topics, Industries, Regions), and for each of two averaged effectiveness measures (microaveraged $F_{1.0}$ and macroaveraged $F_{1.0}$). Thus a total of six distinct feature sets were used in the reported results, drawn from $3 \times 2 \times 23 = 138$ tested.

6.3 Rocchio-Style Prototype Classifier

The Rocchio method was developed for query expansion using relevance feedback in text retrieval (18, 21). Applied to text categorization, it computes a *prototype* vector for each category as a weighted average of positive and negative training examples.

Our Rocchio prototype for category c_j was:

$$\vec{p}_j(\gamma, n) = \frac{1}{|\mathcal{D}(c_j)|} \sum_{\vec{d}_i \in \mathcal{D}(c_j)} \vec{d}_i - \gamma \frac{1}{|\mathcal{D}_n(\bar{c}_j)|} \sum_{\vec{d}_j \in \mathcal{D}_n(\bar{c}_j)} \vec{d}_j \quad (3)$$

where $\vec{d}$ is a training document; $\mathcal{D}(c_j)$ is the set of positive training examples for category c_j ; γ is the weight of the negative centroid; and $\mathcal{D}_n(\bar{c}_j)$ is the “query zone” described below.

Many enhancements have been proposed to original Rocchio algorithm (23). We used the following ones:

1. We restrict the negative examples used to those in a set $\mathcal{D}_n(\bar{c}_j)$, the “query zone” (25), consisting of the n negative examples with highest dot product with the centroid of the positive examples.
2. As with kNN, we do an initial feature selection using the χ^2 -max criterion.
3. We then do a further feature selection on a per-category basis by zeroing out all but the p_{max} largest nonzero coefficients in the Rocchio vector. It is possible, but uncommon, for negative coefficients to remain in the Rocchio vector after this procedure.

We did not incorporate enhancements where the Rocchio parameters are used only as a starting point for an algorithm optimizing some other criterion. We tune p_{max} , γ , n (size of query zone), and the SCut threshold using 5-fold cross validation.

7. Benchmarking the Collection: Text Representation

The same sets of training and test document feature vectors were provided to each algorithm. Feature vectors were produced by downcasing text from the <headline> and <text> XML elements¹³, then applying tokenization, punctuation and stop word removal, stemming, term weighting, feature selection, and length normalization, as described below.

We defined tokens to be maximal sequences of ASCII letter, digit, and underscore characters, followed by an optional “’s” or “’nt”. Tokens consisting of purely digits were discarded, as were common English words found on a stop word list (from the SMART system (22) and included as Online Appendix 17).

Tokens were stemmed with our own implementation of the Porter stemmer¹⁴ (16). Our indexing terms were therefore stemmed words.

Document vectors based on the terms were then created, with each coordinate of the vectors corresponding to a unique term. Only terms which occurred in one or more of the 23,307 RCV1-v1 documents falling before our chronological breakpoint were used¹⁵ That is, terms which had their only occurrences in post-breakpoint documents (i.e. our test documents) did not affect vector formation in any way.

The weight of a term in a vector was computed using $\log TF \times idf$ term weighting (2). This gives term t in document d a initial weight of:

$$w_d(t) = (1 + \log_e n(t, d)) \times \log_e (|\mathcal{D}|/n(t)) \quad (4)$$

where $n(t)$ is the number of documents that contain t , and $n(t, d)$ is the number of occurrences of term t in document d ¹⁶

Some algorithms (see Section 6) then apply feature selection to these vectors. This implicitly replaces $w_d(t)$ with $w'_d(t)$, which is equal to $w_d(t)$ for selected features, but equal to 0.0 for nonselected features.

Finally, differences in document lengths are handled by cosine normalizing (2) the feature vectors (i.e. normalizing them to have a Euclidean norm of 1.0), giving these final weights:

$$w''_d(t) = \frac{w'_d(t)}{\sqrt{\sum_t w'_d(t) \times w'_d(t)}} \quad (5)$$

Despite being simple by IR standards, the above preprocessing would obviously be nontrivial for other researchers to replicate exactly. We therefore have made the exact

13. The <title> element contains a “country code” string that was semi-automatically inserted, sometimes based on the Region codes. The <title> element should therefore not be used in experiments predicting category membership. As part of producing RCV1, Reuters added the <headline> element which contains the same text as the <title> element, but strips out the country code.

14. As discussed at <http://www.tartarus.org/~martin/PorterStemmer>, it is very rare for two implementations of the Porter stemmer to agree with each other, and ours is probably not an exception.

15. We had actually intended to use only our training set, the 23,149 pre-breakpoint RCV1-v2 documents for this purpose, not the 23,307 pre-breakpoint RCV1-v1 documents. The use of the RCV1-v1 documents is not problematic, however, since their errorful category codes were not used, only their text.

16. The *idf* weights used were actually computed from all 23,307 RCV1-v1 documents which fall before our chronological breakpoint, not just the 23,149 RCV1-v2 documents we used for training. As in the previous note, this was not intended, but again is not problematic, since only the document text is used, not the codes.

vectors used in our experiments available in two appendices. Online Appendix 14 contains documents that have been tokenized, stopworded, and stemmed.

Online Appendix 15 contains documents in final vector form, i.e. as $TF \times idf$ weighted, cosine-normalized vectors. Online Appendix 15 uses numeric term IDs rather than the string form of words, but Online Appendix 16 gives the mapping between the two.

The vectors in Online Appendix 15 are based on terms that occurred in our training set, and so should only be used in experiments that use the same training/test split that we did. In contrast, the tokenized representations in Online Appendix 14 contain all non-stopwords, and so can be used with any training/test split.

8. Benchmarking the Collection: Results

Tables 2 and 3 give microaveraged and macroaveraged values of $F_{1.0}$ for the four classification methods, three category sets, and three subsets of each category set. The results largely confirm past studies: SVMs are dominant, weighted k-NN is competitive, and Rocchio is a plausible but somewhat lagging straw man. The choice of averaging, category set, and effectiveness measure affects absolute scores but rarely the ordering of approaches.

We provide not only the averaged data of Tables 2 and 3, but also the full contingency tables for each category as Online Appendix 18. This allows computing alternate effectiveness measures for our classifiers, though of course they were trained to optimize $F_{1.0}$.

8.1 Microaveraging vs. Macroaveraging

Microaveraged measures are dominated by high frequency categories. For RCV1, this effect varies among the category sets. For Topics, hierarchical expansion of the assigned categories inflates the frequency of all non-leaf categories. Non-leaf categories account for only 20% (21/103) of Topic categories but 79% (2,071,530 / 2,606,875) of all Topic code assignments. The four top level Topic categories (CCAT, ECAT, GCAT, and MCAT) alone account for 36% (945,334 / 2,606,875) of all Topic assignments. Thus microaveraged scores for Topics largely measure effectiveness at broad, perhaps less interesting, content distinctions. In contrast, hierarchical expansion for Industry categories affected only the immediate parents of leaf categories, and Regions underwent no hierarchical expansion at all, so the most frequent (and thus dominant) categories for Industries and Regions are not necessarily the broadest categories.

Macroaveraging gives equal weight to each category, and thus is dominated by effectiveness on low frequency categories. For Topics and Industries this is largely the leaf categories in each taxonomy, thus categories with narrow meanings. For Regions, narrowness of meaning is less the issue than amount of news coverage of the geographic entity. Macroaveraged Region effectiveness is dominated by categories corresponding to small countries.

8.2 Averaging Over Categories with No Positive Examples

Past research has varied in how categories with no training or test examples are handled in measuring text categorization effectiveness. We include averages over all categories, over

Category Set	Subset	SVM.1	SVM.2	kNN	Rocchio
Topics	1+ train (101)	0.816	0.810	0.765	0.693
	1+ test (103)	0.816	0.810	0.765	0.693
	all (103)	0.816	0.810	0.765	0.693
Industries	1+ train (313)	0.513	–	0.396	0.384
	1+ test (350)	0.512	–	0.396	0.384
	all (376)	0.512	–	0.395	0.384
Regions	1+ train (228)	0.874	–	0.792	0.794
	1+ test (296)	0.873	–	0.791	0.793
	all (366)	0.873	–	0.791	0.793

Table 2: Effectiveness (microaveraged $F_{1.0}$) of classifiers trained with four supervised learning algorithms, with parameter settings chosen to optimize microaveraged $F_{1.0}$ on cross-validation folds of the training set. Classifiers are trained on our RCV1-v2 training set (23,149 documents) and tested on our RCV1-v2 test set (781,265 documents). The computationally expensive SVM.2 algorithm was run on only one set of categories (Topics). Separate microaverages are presented for (1) categories with one or more training set positive examples, (2) categories with one or more test set positive examples, and (3) all categories. The number of categories in each subset is shown in parentheses.

categories with at least one positive training example, and over categories with at least one positive training and test example.¹⁷ Each average is useful for different purposes:

- *Averaging over all categories:* This best reflects the operational task. Such an average is also important for comparisons with knowledge-based and string-matching approaches which can be used even on categories with no positive training examples.
- *Averaging over categories with one or more positive test examples:* This factors out the impact of choosing a particular arbitrary value to use for F_β when there are no positive test examples (Section 5.3). This impact can occasionally be large. For instance, macroaveraged effectiveness figures for Regions on RCV1-v2 are strongly affected by whether categories with zero positive test examples are included in the average (Table 3).
- *Averaging over categories with one or more positive training examples:* This is appropriate when the primary goal is comparing different supervised learning methods.

8.3 Effectiveness on Individual Categories

Past text categorization research arguably has overemphasized average effectiveness. This was partly a necessity. On the widely used ModApte test set of Reuters-21578, the median

17. No categories in our training/test split of RCV1-v2 have one or more positive training example but zero positive test examples.

Category Set	Subset	SVM.1	SVM.2	kNN	Rocchio
Topics	1+ train (101)	0.619	0.557	0.560	0.504
	1+ test (103)	0.607	0.546	0.549	0.495
	all (103)	0.607	0.546	0.549	0.495
Industries	1+ train (313)	0.297	–	0.235	0.170
	1+ test (350)	0.266	–	0.210	0.152
	all (376)	0.248	–	0.196	0.142
Regions	1+ train (228)	0.601	–	0.588	0.572
	1+ test (296)	0.463	–	0.453	0.441
	all (366)	0.375	–	0.366	0.356

Table 3: Effectiveness (macroaveraged $F_{1.0}$) of classifiers trained with four supervised learning algorithms, with parameter settings chosen to optimize macroaveraged $F_{1.0}$ on cross-validation folds of the training set. See Table 2 for details.

frequency Topic category (of 135 defined Topic categories) has 3 occurrences, so averaging was necessary to produce effectiveness figures that were at all accurate.

In contrast, the median frequency Topic category on our RCV1 test set has 7,250 test set occurrences: bigger than the entire ModApte test set. Only 3 categories have fewer than 100 test set occurrences, so category-level effectiveness figures are more meaningful.

Figure 2 shows smoothed $F_{1.0}$ values for our four classifier training approaches on the 103 Topic categories, sorted by training set frequency of the category. While smoothing aids comparison across the classifiers, it hides a good deal of category-to-category variations. Figure 3 shows raw $F_{1.0}$ values for the SVM.1 approach on all three category sets. Since our focus is methodological we make only a few observations on this data:

- Effectiveness generally increases with increasing class frequency, but the category-to-category variation is very large (Figure 3). This variation has been noted for previous collections, but the large size of RCV1 gives more confidence in this observation. Further, some of the decrease in variation at the right of the graph results from the fact that even a poor classification on a high frequency category yields a moderately high F-measure value (14). For instance, the rightmost Topics category has a test set frequency of 0.465, so a classifier that simply assigned *all* test documents to this category would have an $F_{1.0}$ of 0.635.
- Among the tested approaches, SVMs are dominant at all category frequencies. This fact has been obscured in many previous SVM studies, which restricted experiments to a small set of high frequency categories. However, it is consistent with our previous results on 90 Reuters-21578 Topics categories which had positive examples in both the ModApte training set and ModApte test set (29).
- The SCut approach to threshold tuning for SVMs (SVM.1) is at least as good as the much more computationally expensive leave-one-out procedure (SVM.2), and so should be preferred. It appears that the difference in the methods largely results from

their choice of threshold rather than the orientation of the resulting hyperplanes. The effectiveness of SVM.1 and SVM.2 classifiers is almost identical if their thresholds are set to the test set optimal values. This is somewhat surprising, since SVM.2 often chooses hyperplanes based on uneven weighting of positive and negative examples, and thus with substantially different orientations than those chosen by SVM.1. The angle between the normals of the SVM.1 and SVM.2 hyperplanes (i.e. the arccos of the dot product of the cosine-normalized weight vectors), averaged over the 103 Topic categories, is 19.6 degrees.

- We find some support for previous suggestions (23) that Rocchio-style algorithms are at their best when relatively few positive examples are available, though in all cases they lag the other methods tested. An interesting avenue for future work, now possible with RCV1, would be teasing apart the impact of category narrowness vs. number of positive training examples.

9. Summary

Research in machine learning is heavily driven by available data sets, and supervised learning for text categorization is no exception. We believe RCV1 has the potential to support substantial research advances in hierarchical categorization, scaling of learning algorithms, effectiveness on low frequency categories, sampling strategies, and other areas.

We hope that by documenting the data production process, the nature of the category systems, and the impact of these on the resulting test collection, we have contributed to this progress. We further hope that our benchmark data will encourage replicability and transparency in future text categorization research.

Acknowledgments

We express gratitude to Reuters, Ltd. for making Reuters Corpus Volume 1 available, and for supporting its design and production by Reuters employees Chris Harris and Miles Whitehead. In addition, one of us (Tony Rose) thanks Reuters for supporting his work on the corpus while a Reuters employee. We also acknowledge and thank Stephen Robertson for his work with Reuters on planning of the corpus.

We urge the research community to support this effort by respecting the terms of the license agreement¹⁸, and in particular clause 3.3 on acknowledging Reuters and providing a copy of any publication. We encourage researchers with questions about the corpus to post them to the Reuters Corpora mailing list¹⁹, which both we and Reuters personnel read regularly.

Many current and former Reuters employees provided information on the production of the data or editorial processes at Reuters, including Dave Beck, Chris Harris, Paul Hobbs, Christopher Porter, Jo Rabin, Mark Stevenson, Miles Whitehead, Richard Willis, and Andrew Young. This paper would not be possible without their input. We also thank

18. <http://about.reuters.com/researchandstandards/corpus/agreement.htm>

19. <http://groups.yahoo.com/group/ReutersCorpora/>

???, Tom Ault, Evgeniy Gabrilovich, Mikhail Kreines, Bill Teahan, and Benjy Weinberger for comments on drafts of this paper or sharing their data on RCV1.

This research is sponsored in part by the U.S. National Science Foundation (NSF) under the grant numbers KDI-9873009, IIS-9982226, and DMS-0113236. Any opinions or conclusions in this paper are the authors' and do not necessarily reflect those of the sponsors.

References

- [1] T. Ault and Y. Yang. kNN, Rocchio and metrics for information filtering at TREC-10. In *The Tenth Text REtrieval Conference (TREC 2001)*, pages 84–93, Gaithersburg, MD 20899-0001, 2002. National Institute of Standards and Technology.
- [2] C. Buckley, G. Salton, and J. Allan. The effect of adding relevance information in a relevance feedback environment. In *SIGIR 94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 292–300, London, 1994. Springer-Verlag.
- [3] C. W. Cleverdon. The significance of the Cranfield tests of index languages. In *Fourteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '91)*, pages 3–12, 1991.
- [4] R. Grishman and B. Sundheim. Design of the MUC-6 evaluation. In *Sixth Message Understanding Evaluation (MUC-6)*, San Francisco, CA, November 1995. Defense Advanced Research Projects Agency, Morgan Kaufmann.
- [5] W. Hersh, C. Buckley, T. J. Leone, and D. Hickman. OHSUMED: an interactive retrieval evaluation and new large text collection for research. In *17th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 192–201, 1994.
- [6] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *European Conference on Machine Learning (ECML)*, pages 137–142, Berlin, 1998. Springer.
- [7] T. Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning (ICML'99)*, pages 200–209, San Francisco, CA, 1999. Morgan Kaufmann.
- [8] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *International Conference on Machine Learning (ICML'97)*, pages 170–178, Nashville, US, 1997. Morgan Kaufmann.
- [9] F. W. Lancaster. *Indexing and Abstracting in Theory and Practice*. University of Illinois, Champaign, IL, 1998. Second edition.
- [10] D. D. Lewis. Evaluating text categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 312–318. Defense Advanced Research Projects Agency, Morgan Kaufmann, February 1991.

- [11] D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *SIGIR '95: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, New York, 1995. Association for Computing Machinery.
- [12] D. D. Lewis. Applying support vector machines to the TREC-2001 batch filtering and routing tasks. In *The Tenth Text REtrieval Conference (TREC 2001)*, pages 286–292, Gaithersburg, MD 20899-0001, 2002. National Institute of Standards and Technology.
- [13] D. D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka. Training algorithms for linear text classifiers. In *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 298–306, Konstanz, 1996. Hartung-Gorre Verlag.
- [14] D. D. Lewis and R. M. Tong. Text filtering in MUC-3 and MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 51–66, Los Altos, CA, June 1992. Defense Advanced Research Projects Agency, Morgan Kaufmann.
- [15] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *15th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'92)*, pages 37–50, 1992.
- [16] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
- [17] S. Robertson and I. Soboroff. The TREC 2001 filtering track report. In *The Tenth Text REtrieval Conference (TREC 2001)*, pages 26–37, Gaithersburg, MD 20899-0001, 2002. National Institute of Standards and Technology.
- [18] J. J. Rocchio, Jr.. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1971.
- [19] M. Rogati and Y. Yang. High performing and scalable feature selection for text classification. In *AAAI-02*, page (submitted), 2002.
- [20] T. Rose, M. Stevenson, and M. Whitehead. The Reuters Corpus Volume 1 – from Yesterday’s News to Tomorrow’s Language Resources. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, 2002.
- [21] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of American Society for Information Sciences*, 41:288–297, 1990.
- [22] G. Salton, editor. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1971.
- [23] R. E. Schapire, Y. Singer, and A. Singhal. Boosting and Rocchio applied to text filtering. In *Proceedings of the Twenty-first Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 215–223, New York, 1998. The Association for Computing Machinery.

- [24] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 2002. Forthcoming.
- [25] A. Singhal, M. Mitra, and C. Buckley. Learning routing queries in a query zone. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 25–32, 1997.
- [26] J. M. Tague. The pragmatics of information retrieval experimentation. In K. Sparck Jones, editor, *Information Retrieval Experiment*, chapter 5. Butterworths, London, 1981.
- [27] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.
- [28] A. S. Weigend, E. D. Wiener, and J. O. Pedersen. Exploiting hierarchy in text categorization. *Information Retrieval*, 1(3):193–216, 1999.
- [29] Y. Yang and X. Liu. A re-examination of text categorization methods. In *The 22th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 42–49, 1999.
- [30] Y. Yang and J. P. Pedersen. A comparative study on feature selection in text categorization. In *The Fourteenth International Conference on Machine Learning (ICML'97)*, pages 412–420. Morgan Kaufmann, 1997.
- [31] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):67–88, 1999.
- [32] Y. Yang. A study on thresholding strategies for text categorization. In *The Twenty-Fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, pages 137–145, New York, 2001. The Association for Computing Machinery.
- [33] T. Zhang and F. J. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, 4:5–31, 2001.

Figure 2: Test set $F_{1.0}$ for four classifier approaches on 103 RCV1-v2 Topic categories. Categories are sorted by training set frequency, which is shown on the x -axis. The $F_{1.0}$ value for a category with frequency x has been smoothed by replacing it with the output of a local linear regression over the interval $x - 200$ to $x + 200$.

Figure 3: Raw test set $F_{1.0}$ values for the SVM.1 approach on all three category sets. The line shows corresponding smoothed (as in Figure 2) values. Training set frequency is shown on the x -axis.