

Sorting Email Messages by Topic

Project for CPSC 503 and CPSC 532b

Peter Gorniak

December 2, 1999

Abstract

The problem of categorizing a document by topic can be divided into three steps. First, the document's text must be converted into potential features. From these many potential features a few receive promotion to feature status during the feature selection step. Finally, a classifier learns to categorize documents into topics based upon these features. BUCFE, Bottom-Up Combinatorial Feature Extraction, adds to the commonly used individual words syntactically parsed sentence fragments as potential features. Here, a neural network using relevant term selection and the Winnow algorithm make use of BUCFE's suggestions. The problem domain of categorizing email messages into mailboxes provides a real-world test bed.

1 Introduction

Text categorization assigns a given document to one topic from a given set of topics, based upon samples of documents already assigned topics from the same set. This task poses problems in Computational Linguistics, Statistics and Machine Learning. Conceptually, it can be broken down into three distinct stages: Feature Production, Feature Selection and Classifier Production. Each of these stages attains a unique flavour when seen from the point of view of one of the mentioned disciplines, and often these flavours appear mutually exclusive.

1.1 Feature Production

During the first stage of text categorization the full text of a document to be classified must be parsed to produce a list of potential features that could serve as a basis for categorization. The central tradeoff in this task looms between capturing more linguistic properties of the message text on the one hand and creating features with good statistical properties across documents and topics on the other. For example, the following serve as potential features in varying approaches, listed in order of increasing use of linguistic knowledge.

Single Words from Text - This approach leaves everything to statistics.

Everything separated by whitespace or punctuation is considered a potential feature. Elimination of common words or stemming (using the root form instead of the form that occurs) are not attempted and word

context is ignored, with the reasoning that these things will be dealt with in the feature selection step. Many researchers concentrating on the statistical and learning aspects use this type of feature production, such as (Wiener *et al.*, 1995) and (Dagan *et al.*, 1997).

Word Groups from Text - Here a fixed number of words around any word form part of a potential feature. This takes context into account somewhat more, but still ignores variation in word forms. If the single word approach is replaced by this one statistics for the features produced are likely to tend to the worse because word groups occur more sparsely than single words.

Word Stems - Stemming can either be performed by a morphological algorithm, which requires a lexicon and the morphological rules for the language (Lewis, 1992), or can be approximated (Bayer *et al.*, 1996). In principle, stemming improves feature statistics as it is a many-to-one function for each word, and words thus occur less sparsely.

Phrases - Extracting phrases from the document text requires some sort of partial syntactical parse. Parsed elements can range from unclassified sentence fragments (as in the approach presented here) to noun phrases or even to complete parse-trees of parts of sentences. Most parsers require a lexicon and syntactical rules of the language being parsed. Given the unpredictable domain considered here, that of email messages, it turns out to be impossible to provide a complete lexicon

and pointless to hope for any kind of complete parse. The statistical problems mentioned with word groups are pushed to their extremes with the phrasal approach, because there are many ways to express the same thing in natural language, thus even though the features captured by phrase parsing appear more meaningful and important to classification, their sparse occurrences render them close to unusable due to their statistical properties (Lewis, 1992).

BUCFE, (Bottom-Up Combinatorial Feature Extraction), the method examined in this paper, falls into the last category, but the phrasal parsing used does not rely on a dictionary, nor does it simply replace words with phrases. Instead, it performs a bottom-up parse of sentence fragments bounded by words that either appear on a stop-list of common words or have been identified as too infrequent to be statistically relevant. The parser then offers all possible combinations of word groupings in such a fragment as possible features and relies on the feature selection process to determine their statistical relevance. The hope behind this approach is to blur the line between a strict linguistic parse and a word-based approach by making the tradeoff automatically and based on statistical properties. In essence, the decision of whether to use phrases or not is deferred to the feature selection stage.

1.2 Feature Selection

No matter what kind of feature production algorithm is used, the set of features offered in the end is too large and contains too much redundancy to be useful for text categorization in unmodified form. During the feature selection stage the goal is to rank features according to their relevance in the categorization task, possibly discarding irrelevant ones. Again, several common approaches exist, amongst them:

Hard and Soft Clustering - Clustering in general works by grouping features to decrease their number. Hard clustering does this by simply assigning features to groups and then treating the features in a group identically, whereas soft clustering maintains probabilities for features within a group (Li & Yamanishi, 1997). Two properties of the new approaches applied in this paper lead to the decision not to subscribe to explicit clustering techniques: a) BUCFE already offers a context-based kind of hard clustering and b) the multi-layered neural network used should be able to recognize natural feature clusters during the learning process. The only kind of clustering that could potentially offer significant improvements would be one that groups features with similar meaning, as determined by linguistic knowledge as opposed to statistics, but the linguistic knowledge required to do this renders that approach prohibitive.

Latent Semantic Indexing - LSI works from the fact that a Singular

Value Decomposition of a feature-by-document matrix lets one rank the resulting orthogonal basis vectors according to their contribution in the overall linear transformation. Largely irrelevant dimensions can be discarded (Deerwester *et al.*, 1990). For an application of LSI similar to the one discussed here see (Wiener *et al.*, 1995). It would be interesting to see how LSI rates the features produced by BUCFE, but that combination was not tested for this paper.

Pre-Classification Relevancy - There are several measures that can be used directly to rank the features individually, , such as information gain, before they are used in the categorization task. (Wiener *et al.*, 1995) in reading (Salton & Buckley, 1998) suggest a *relevancy score*, namely

$$\log \frac{w_{tk}/d_t + 1/6}{w_{\bar{t}k}/d_{\bar{t}} + 1/6}$$

where w_{tk} is the number of documents with a topic that contain the term, d_t is the total number of documents that contain the term and \bar{t} signifies the same entities for documents not containing the term. This measure performed well for them in suggesting features for a neural network, thus it seemed a good choice for the approach here.

Classification Relevancy - There are some classifiers, such as Variable-Kernel classifiers (Lowe, 1995), that allow for feature selection during the actual learning task. Lowe’s algorithm may scale up to deal with the several thousand features to select from in text categorization, but for

this paper an algorithm known for its good large scale behaviour, namely Winnow as introduced in (Littlestone, 1988) appeared favourable. Winnow has successfully been applied in text categorization before, and the version used here is Balanced Winnow as described in (Dagan *et al.*, 1997).

1.3 Classifier Production

Once feature selection concludes, any classifier that can learn the required function from features to topics should work, and most supervised learning algorithms can do so and have been applied to this task. Examples include neural networks (Wiener *et al.*, 1995) and maximum likelihood estimation using the EM algorithm (Li & Yamanishi, 1997). Here, a backpropagation neural network (Rumelhart *et al.*, 1986) was chosen as a potentially non-linear classifier, and a mistake-driven algorithm, Winnow (Littlestone, 1988), that is linear, but allows for feature selection during the classifier production stage.

2 The Problem Domain

This project is an extension of MailMind, an agent that uses a neural network to sort email messages based on the sender's address (Gorniak, 1998). The setting is intentionally a very realistic one to evaluate the performance of intelligent agents employing current learning techniques in a real-world,

end-user oriented task. The user is assumed to have accumulated and hand-sorted some hundred messages that arrived consecutively into a small number of mailboxes. The sorting can be based on personal preferences and be inconsistent. For example, a mailbox with messages from this author’s Canadian correspondence was used as data, which contains messages from Canadians visiting Germany that have German email addresses in the *From* header. Along the same lines are the data used to evaluate the currently discussed extension of MailMind samples from real-world mailboxes. The samples are small (usually 100 messages) and all stem from about the same timeframe. This means that they are statistically non-representative, partially due to the small sample size and partially due to the fact that correspondence within the same mailbox can show differing features over time. An obvious and trivial example of the latter that often produces inaccurate relevancy measures of features are the timestamps attached when an email program quotes text. They are often of the form ”so-and-so said such-and-such on such-and-such date”, where the date contains a month. Given that 100 consecutive messages of any mailbox likely fall within the range of a few months, features like ’dec’ or ’jan’ often pop up as relevant in distinguishing the topic of this mailbox from others. This is only one example of the type of problems imposed by this setting.

This problem domain is similar to the usual one for evaluating text categorization approaches in that it produces a large number of potential features and exhibits potentially complex functions from features to topics. It differs

from standard test domains like the Reuter corpus in that

- documents assumed to be sorted by topic are actually often sorted by point of origin, leading to noise within a mailbox,
- messages contain significant amounts of grammatical errors and content with no natural language grammar such as source code or software engineering tools' outputs,
- and test data sets are not randomly chosen and small.

Some of the methods examined here are geared specifically towards this domain. All of the results quoted should only be compared internal to this problem domain and may differ when applied to other tasks.

3 BUCFE - Bottom-Up Combinatorial Feature Extraction

(Lewis, 1992) indicates the result that syntactic parsing, while linguistically attractive and intuitively relevant, does not help with the text categorization problem due to the poor statistical properties of the phrases it creates, which clustering attempts do not alleviate. BUCFE comprises an attempt to reintroduce a platform for syntactic parsing into text categorization. It is a platform for two reasons:

- It is only involved in the feature production step, and amongst the features it produces are the individual words of the document. If during feature selection no phrases appear relevant, they can simply be discarded. If, however, they appear useful to the categorization task they can be used. The number of features offered by BUCFE is on the same order of magnitude as that of individual words, meaning that feature selection is equally hard with BUCFE in place as without.
- BUCFE's parser can be supplied with as much syntactic information as seems useful for a given feature generation task. In the form used here BUCFE only combines the words between terms known to be insignificant in all possible way, but with the simple addition of rules to the parser's context-free grammar more complex terms can be formed and offered as potential features.

Many text categorization approaches discard certain words immediately: those on a stop-list of words known to be redundant for topic distinction, such as conjunctions and prepositions, and those that occur too infrequently to be of any statistical value. Instead of discarding these words, BUCFE uses them as its lexicon. Any word it does not know is labeled as UNKNOWN. It uses a bottom-up chart parser and in its basic form works with an overly simple context-free grammar:

TERM \rightarrow UNKNOWN

TERM \rightarrow TERM TERM

concerning that server next to other mailboxes that speak of the game of othello and of servers as well, but rarely in one term, the feature 'othello server' can be used to distinguish between these mailboxes, whereas the individual word features are of no help here.

In this form BUCFE is language independent except for its list of about 200 stop words, and thus ideally suited for the domain of e-mail messages examined here. It also deals well with the common spelling and grammatical mistakes of email messages, because its parsing is forgiving enough to never fail, but still produce the correct features if they can be identified.

It is easy to add more syntactical features to BUCFE by expanding its context-free grammar. For example, adding

$\text{TERM} \rightarrow \text{TERM P TERM}$

allows for terms connected by prepositions. However, the number of features produced increases exponentially with the length of contiguous terms parsed, so it may be preferable to write grammars that do not offer all possible combinations of words as TERMS when more syntactical knowledge is added to the parses. In the current form terms are delimited to small enough sizes to produce an adequate number of potential features.

4 Feature Selection Approaches

4.1 Relevancy Scores

MailMind's neural network was readily available and has proven extremely useful in sorting email message based upon the sender's address, so in a first attempt it served as the classifier. However, BUCFE produces about 4000 features for two mailboxes whereas MailMind used about 50 inputs for its neural network, making feature selection an important task.

The first approach to feature selection uses the relevancy score described in 1.2. BUCFE parses all example messages and offers possible features. The feature selection algorithm Comparing two mailboxes with closely related topics, one with messages from Apple's Macintosh Java Runtime mailing list and the other with messages about the Java version of MAGOO, an othello playing agent, the following are deemed the most relevant features that distinguish the two:

Most entries should be self-explanatory in their relevancy. 'Codewarrior' and 'Cafe' are names of Java development environments and 'jar' is the name for Java library files. Next to the obviously relevant terms like 'magoo' and 'mrj' some other selection choices are notable. It catches the eye that with the limited amount of examples and the small number of mailboxes statistics cannot take the place of word stemming, as can be seen by 'play' and 'playing' or 'game' and 'games' both being present. Word stemming would cluster these terms, but it can be suspected that with more examples a form like

| MAGOO | MRJ |
|--------------|----------------|
| magoo | jun |
| othello | mrj |
| game | apple |
| play | development |
| games | jar |
| beat | mac |
| playing | codewarrior |
| level | cafe |
| move | com |
| board | end |
| moves | dev |
| played | digest |
| source code | mrj dev |
| try | mrj dev digest |
| source | windows |

Table 1: Top Scoring Terms According to Relevancy Scores for a mailbox about MAGOO, a Java Othello Playing Program and a mailbox about the Macintosh Runtime Java Environment

'play' that can be several forms of the verb as well as a noun would be selected over 'playing'. Visible is also the artifact of the limited time-scope of the example, giving rise to features like 'jun'.

On the phrasal side it is clear that in general individual words provide better statistical distinction and are therefore selected. The presence of phrases like 'mrj dev digest' is due to titles prepended to the mailing list messages that tend to appear very often (and are thus correctly classified as important features). Most interesting is the selection of 'source code' and 'source' but not of 'code' for MAGOO. Examining the actual mailbox indeed reveals that most of the MRJ messages never bother to use the full phrase 'source code', but simply say 'code' very often, whereas people tend to ask for 'source code' in MAGOO. This is an example where using BUCFE introduces a feature more relevant than 'source' alone that would not be available using single word features. It stands to reason that with more closely related topics the number of phrases selected would increase.

The terms thus selected then serve as binary on-off inputs to a neural network with one hidden layer. The network is organized to support feature formation as pictured in figure 1, where features for one topic are visibly grouped together.

4.2 Balanced Winnow

In the approach to feature selection described in the last section the feature selection stage is completely decoupled from the classifier production stage.

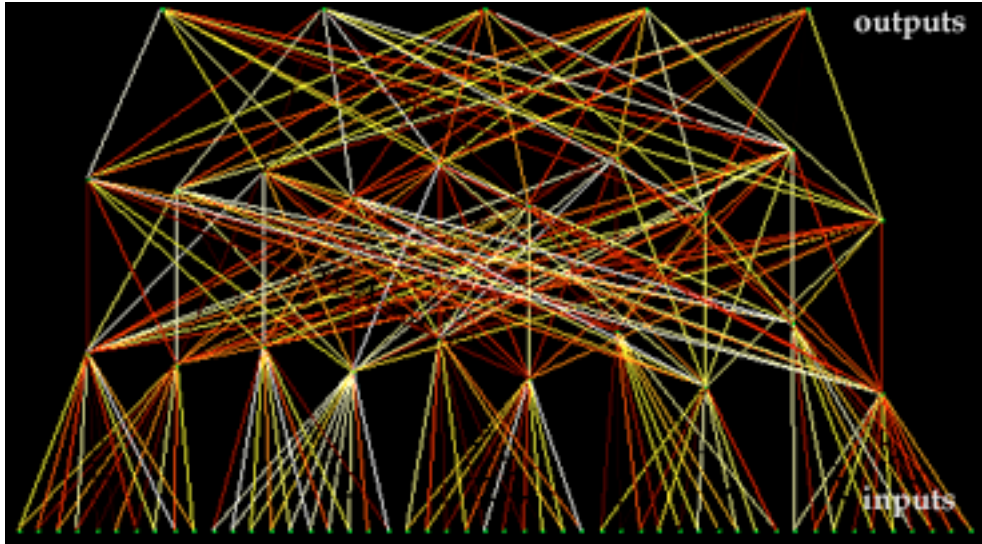


Figure 1: An Example MailMind Network Configuration

This seems problematic, because the classifier, especially a non-linear classifier such as the neural network used, might differ in how relevant features are for its learning process from the estimate given in the pre-processing stage. However, it is hard to determine which features are actually important after the neural network has learned, so one must simply hope that the feature preferences exhibited during selection and learning are somewhat similar.

Winnow, a mistake driven algorithm that can learn with a large number of sparse features, tightly couples the feature selection and classifier production stages. For this task it takes exactly the form of Balanced Winnow described in (Dagan *et al.*, 1997), except for the fact that features are discarded when the difference between their positive and negative weights is less than it was set to originally, and neither a threshold range nor feature repetition are

used. Winnow only performs classification into two categories, so the task discussed here require a separate instance of the algorithm for each mailbox. To decide which mailbox a message should go into, the maximum sum a Winnow instance produces is taken as the winning candidate. Thus, correct decisions are possible even when one or more Winnow instances are actually suggesting the wrong answer.

5 Results

The results here quoted should be taken with a grain of salt in that the problem domain is a very specific one as described in section 2 and especially because there was insufficient time to tune many of the dozens of parameters involved. It may well be that different settings would provide different rankings, especially for BUCFE. However, the settings used do provide good results here, and some are fixed to simulate the real-world situation (e.g. number of training instances.) Table 2 summarizes the results. Training data are comprised of the first 100 messages of each mailbox and the training data is presented 5 times. The classifier is then tested on the remaining messages, which number 1750 for the three mailbox runs. The mailboxes used are MAGOO as discussed in 4.1, a mailbox containing correspondence with Glenayre, a paging infrastructure company, and a mailbox containing messages from a music discussion list. The neural network used is the back-propagation network from (Gorniak, 1998) with a learning coefficient of 0.2

and 15 features per mailbox. It uses the top scoring terms according to section 4.1. Winnow ran with a promotion parameter of 1.3, a demotion of 0.7 and a threshold of 1.0.

| Three Mailboxes | % correct training | % correct testing |
|-----------------|--------------------|-------------------|
| NN - Words | 93.5 | 83.9 |
| NN - BUCFE | 94.1 | 82.4 |
| Winnow - Words | 67.0 | 69.6 |
| Winnow - BUCFE | 79.1 | 73.4 |

Table 2: Results using three mailboxes, a backpropagation neural network (NN) and the Winnow algorithm both with and without BUCFE

Due to the rare selection of phrases as features with the given mailboxes and relevancy scoring the negligible effect of BUCFE on performance using the neural network is to be expected. Overall the neural network approach turns out to perform rather well in the set task. BUCFE’s effect is notable using Winnow, which retains more features and thus more phrases than the other approach. However, performance overall is worse with Winnow. As mentioned before, trials runs were very limited in scope due to time constraints and more runs are needed with parameters changes as well as more mailboxes to make the results conclusive.

6 Conclusion

The overall task of text categorization of email messages can be accomplished to high success rates by selecting relevant features and training common machine learning classifiers on them. The phrasal parsing approach suggested

here, BUCFE, in principle does offer more valuable features than single words that can be important in distinguishing topics. With an algorithm like Winnow that weighs features during the learning process this effect can be identified, but more trial runs are needed to establish this as a general result. Especially with more mailboxes results will likely look quite different. Further interesting directions to be taken are applying BUCFE to different feature selection and learning strategies, as well as expanding BUCFE's grammar and lexicon to offer more complex phrasal structures as potential features.

References

- Bayer, Thomas, Renz, Ingrid, Stein, Michael, & Kressel, Ulrich. 1996. Domain and Language Independent Feature Extraction for Statistical Text Categorization. *Computation and Language*, **7**.
- Dagan, Ido, Karov, Yael, & Roth, Dan. 1997. Mistake-Driven Learning in Text Categorization. *Computation and Language*, **6**.
- Deerwester, Scott, Dumais, Susan T., & Harshman, Richard. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, **41**, 331–407.
- Gorniak, Peter J. 1998. *MailMind - A Connectionist E-Mail Sorting Agent*. Honours Thesis.
- Lewis, David D. 1992. *Representation and Learning in Information Retrieval*. Ph.D. thesis, University of Massachusetts.
- Li, Hang, & Yamanishi, Kenji. 1997. Document Classification Using a Finite Mixture Model. *Computation and Language*, **5**.
- Littlestone, N. 1988. A New Linear-Threshold Algorithm. *Machine Learning*, **2**, 285–318.
- Lowe, David G. 1995. Similarity Metric Learning for a Variable-Kernel Classifier. *Neural Computation*, **7**, 72–85.

- Rumelhart, D., Hinton, G.E., & Williams, R.J. 1986. Learning internal representations by error propagation. *Chap. 8, pages 318–362 of: Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press.
- Salton, G., & Buckley, C. 1998. Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, **24**, 513–523.
- Wiener, Eric, Pedersen, Jan O., & Weigend, Andreas S. 1995. A Neural Network Approach to Topic Spotting. *In: Fourth Annual Symposium on Document Analysis and Information Retrieval.*