Proc. 5th International Conference on Soft Computing as Transdisciplinary Science and Technology
CSTST'08, Paris, France, October 27-31, 2008, pp. 218-225

Recursive Data Mining for Role Identification

Vineet Chaoji
Rensselaer Polytechnic Institute
Troy, New York 12180
chaojv@cs.rpi.edu

Apirak Hoonlor
Rensselaer Polytechnic Institute
Troy, New York 12180
hoonla@cs.rpi.edu

Boleslaw K. Szymanski
Rensselaer Polytechnic Institute
Troy, New York 12180
szymansk@cs.rpi.edu

ABSTRACT

We present a text mining approach that enables an extension of a standard *authorship assessment* problem (the problem in which an author of a text needs to be established) to *role identification* in communications within some Internet community. More precisely, we want to recognize a group of authors communicating in a specific role within such a community rather than a single author. The challenge here is that the same author may participate in different roles in communications within the group, in each role having different authors as peers. An additional challenge of our problem is the length of communications. Each individual exchange in our intended domain, communications within an Internet community, is relatively short, in the order of several dozens of words, so standard text mining approaches may fail. An example of such a problem is recognizing roles in a collection of emails from an organization in which middle level managers communicate both with superiors and subordinates. To validate our approach we use the Enron email dataset which is such a collection.
Our approach is based on discovering patterns at varying degrees of abstraction in a hierarchical fashion. Such discovery process allows for certain degree of approximation in matching patterns, which is necessary for capturing non-trivial structures in realistic datasets. The discovered patterns are used as features to build efficient classifiers. Due to the nature of the pattern discovery process, we call our approach `Recursive Data Mining`. The results show that a classifier that uses the dominant patterns discovered by Recursive Data Mining performs well in role detection.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*; I.2.7 [**Artificial Intelligent**]: Natural Language Processing—*Text analysis*; I.5.2 [**Pattern Recognition**]: Design Methodology—*Feature evaluation and selection, Pattern analysis*

General Terms

Algorithms, Experimentation

Keywords

Data Mining, Feature Extraction, Text classification

1. INTRODUCTION

The problem of understanding characteristics of data has attracted keen interest of scientists from early years of computer science. Specifically, characteristics that represent a certain "style" within the data have been widely used to analyze and discriminate between different attributes of data, their sources and the underlying generative models.
Within the data mining community, the term *feature extraction* is commonly used for techniques that identify features relevant to the application at hand. Within this context, the term feature has been loosely used for attributes of data that can range, for instance, from keywords for text documents to principle eigenvectors for high dimensional genetic data. Feature extraction is broadly considered to be composed of two sub-tasks – *feature construction* and *feature selection* [11], each addressing one of the two main challenges of the problem. The first challenge arises when a lot of noise is present in the data resulting in construction of ineffective features. The second challenge results from the large number of features usually generated. The features are ranked based on optimality criteria – such as information gain, usefulness for novelty detection – and only the top-ranked features are used to avoid the curse of dimensionality [7] and to enhance generalization capabilities.
Each human communication carries not only semantic content defining its meaning but also a unique word and pattern of words structure characteristic of its author. Hence, proper feature selection and extraction may enable reliable attribution of a communication's authorship in the process of *authorship assessment.* In this paper, we present a more complex problem of detecting roles of communicators within a communicating group based on styles of their communications. To solve this new problem, we apply a general feature extraction method, termed *Recursive Data Mining* (RDM), that uses statistically significant, approximately matched sequential patterns.
Identifying patterns using significance tests was used extensively in biological sequence analysis [16]. In this paper, we focus on extracting patterns from electronic media; those patterns are later used to build a classifier. The approach is independent of any semantic information contained in the

communication, making it amenable to text documents written in different languages. The method also controls degree of flexibility by allowing different degree of inexactness in matching of patterns. Otherwise, presence of noise (in the form of spelling mistakes, use of abbreviations, etc.) would lead to very few matches.

Even though RDM can be applied to data of any nature (time series data, genome data, etc.), we focus here on text documents that are traces of electronic communication, and specifically, we consider the Enron email dataset introduced in [17] as a benchmark for email classification. In our work, the features obtained from Enron email dataset are used to identify the organizational role (e.g., manager, president, administrative assistant, etc.) of the sender of an email. Potential applications of our approach include analysis of groups on the Internet of which structure is not clear or not explicitly defined and, in some cases intentionally obstructed or hidden by the group members. Identifying the leaders and followers in such informal groups is of great value for social sciences, network science and security.

The basic premise behind our approach is that within any form of communication, not only traces of personal style can be identified but also its variations can be detected. Moreover, the variations of the personal style based on the stature or relationship to the recipient of the communication are similar among large groups of users sharing the common cultural heritage. This similarity is so strong that such a style variation can be detected for a group of the authors and not only for the specific individual author. For instance, the manner in which a person communicates with his/her superior is quite different from the manner in which one would communicate to his/her friend.

To address the above mentioned challenges, RDM uses an approach that extracts syntactic patterns. These patterns capture the stylistic characteristics, which are in turn used to attribute a role to an individual. We built a framework for discovering statistically significant sequence patterns from a stream of data. The methods developed can be applied for finding significant sentences in a stream of text such as email, blog or chat-room session. The methods can also be applied for finding significant regions in a DNA sequence. Previously, RDM was presented in [9, 22] for social network analysis and masquerade detection.

Some of the key contributions of RDM approach are as follows:

- The patterns formed do not have any length restriction. This allows arbitrary size patterns to be discovered. Most of the other published techniques work on a fixed size window.
- The method is hierarchical in nature. This enables us to capture patterns at various levels of abstractions. Moreover, the hierarchical nature allows us to remove noisy symbols from the stream as we move from a lower to a higher level in the hierarchy. This ultimately leads to discovery of *long range patterns* that are separated by long noisy intermediate text segments.
- The method is also able to discover approximate (similar) patterns with the level of inexactness of matching controlled by a user provided parameter.

The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 introduces the basic terminology for this work and is followed by Section 4 containing a detailed description of our methodology. The experimental results are presented in Section 5, while Section 6 offers conclusions.

2. RELATED WORK

There is a large body of work that deals with extracting patterns and features from unstructured raw text data. Experts in the fields of information retrieval, natural language processing, data mining and statistical learning have focused on a diverse set of techniques for feature extraction and feature selection to solve the author identification problem. Linguists use statistical techniques to obtain a set of significant words that would help identify authors. In 1964 Mosteller and Wallace [18] solved the Federalist Papers problem by identifying 70 *function words* and applying statistical inference for the analysis. In the Natural Language Processing (NLP) community, the Hidden Markov Model has greatly influenced many techniques used for speech recognition [3]. Other complementary techniques used by the NLP community include part of speech tagging [12], which is used for assigning a syntactic category to each word in a text document. In [11], authors provide a comprehensive introduction to feature selection techniques. Many other applications also benefit from feature extraction and feature selection, which include automatic painting classification [6], detection of malicious alterations of images [20], grouping proteins and genes into classes based on function, location and biological process in bioinformatics [25] and characterizing the behavior of an individual or a group of individuals in e-Commerce [23].

Our work is a continuation of an approach introduced in [8] which was developed concurrently and independently with the approaches presented in [19, 21]. All these papers discuss how the underlying structure in a communication in a human language can be learned in a hierarchical manner. In [8] we have shown that a hierarchical structure can model an emergence of basic speech capabilities in human infants. In [19] the authors extract a hierarchical nested structure by substituting grammar for repeated occurrences of segments of tokens. Similarly, in [21] the authors present a data independent hierarchical method for inferring significant rules present in natural languages and in gene products. Our efforts differ in that we provide certain flexibility in the patterns found by allowing gaps. This enables us to work with much smaller datasets as compared to [21]. In recent works [1, 2], the frequent mining was modified to obtain useful patterns which are used for classification in various domains.

3. PRELIMINARIES

Consider a set of sequences, denoted as $\mathcal{SEQ}$. Each sequence consists of a series of *tokens* from a set $\mathcal{T}$. Thus, a sequence $S \in \mathcal{SEQ}$ of length n can be represented as $t_1, t_2, \ldots, t_n$, where $t_i \in \mathcal{T}$. Depending on the application, a token may represent a different entity. For instance, in the domain of text documents, a token can either represent a character or a word and a sequence S would then correspond to the whole document. For stock market data, each token could represent a numeric value (price and volume) while the sequence would represent the entire time series of purchases (or sales) of a certain stock. A special token, called

the *gap token*, corresponds to a blank entry and is represented by the symbol $\perp$. The gap token mimics the '.' character in regular expressions - it can be matched with any other token. A *sequence pattern* $\mathcal{P}$ is an ordered sequence of tokens from $\mathcal{T} \cup \{\perp\}$. Formally, $\mathcal{P}$ can be denoted as $\{s_i : s_1, s_{l(\mathcal{P})} \in \mathcal{T} \wedge s_i \in \mathcal{T} \cup \{\perp\}, i = 2 \ldots l(\mathcal{P}) - 1\}$, where i is the index of a token in the sequence and $l(\mathcal{P})$ is the *length* of the pattern $\mathcal{P}$. It should be noted that the first and last tokens are never the gap tokens. This restriction is useful for combining contiguous patterns.
Two patterns are said to have an *exact match* if they consist of the same sequence of tokens. Given a *similarity function*, $sim(\mathcal{P}_1, \mathcal{P}_2)$ a similarity score between 0 and 1 is assigned to each pair of patterns. Exact matching restricts the similarity score to binary values - $sim(\mathcal{P}_1, \mathcal{P}_2) = 1$ if $\mathcal{P}_1 = \mathcal{P}_2$, 0 otherwise. The presence of a gap token in a sequence pattern relaxes the exact match constraint, allowing it to match a wider set of patterns with $sim(\mathcal{P}_1, \mathcal{P}_2) \in [0, 1]$. A match with similarity score greater than $\alpha \in (0, 1)$ is called a *valid match*. The set $\mathcal{M}_{\mathcal{P}}(\alpha)$ is the *s*et of valid matches for a pattern $\mathcal{P}$ at the level of similarity α. A pattern $\mathcal{P}$ of length l and g gaps is termed as a $(l, g) - pattern$. If $\mathcal{P}$ has a match at index i in sequence S, then it belongs to the set of $S_i(l, g)$*-patterns*. The set of patterns $S_i(l)$, given by the expression $\cup_{g=0}^{max\_gap} S_i(l, g)$ represents all patterns of length l starting at index i in S. *max_gap*, as the name indicates, is the maximum number of gaps allowed in a pattern. In the rest of the paper, the term pattern would always imply a sequence pattern and terms pattern and feature would be used interchangeably, unless stated otherwise.

4. RECURSIVE DATA MINING

Recursive Data Mining (RDM) is an approach for discovering features from sequences of tokens. Given a set of sequences as input, the algorithm captures statistically significant patterns from the initial sequences. The patterns obtained are assigned new tokens. The initial sequences are re-written by collapsing each sequence pattern to its newly assigned token, while retaining the rest of the tokens. The algorithm now operates on the re-written sequences and continues to iterate through the pattern generation and sequence re-writing steps until either the sequences cannot be re-written further or a predefined number of iterations is reached. Each generation of sequences in the above process is termed a ***level***, with the initial set of sequences called ***level(0) sequences***. The patterns obtained at each level form a set of features. The term "recursive" in the name refers to this iterative step that obtains the next level by operating on the current level. In the RDM process, we claim that the recursive (hierarchical) processing of the data captures distinctive features at varying levels of abstraction. Intuitively, at lower levels the patterns obtained are more specific, resulting is a smaller set of valid matches, $\mathcal{M}(\alpha)$. At higher levels, the patterns are more general, resulting is a larger $\mathcal{M}(\alpha)$ set. On the other hand, with increasing levels, the number of patterns found decreases monotonically.
In this section we present the details of an RDM based classifier. Like most supervised learning tools, RDM has two stages of processing – training and testing. The *training phase* starts with *pattern generation*, followed by pattern selection through the *pattern significance* step. Out of the significant patterns, the *dominant patterns* form the feature set for the current level. The overall RDM process is outlined in Algorithm 1. The input is a set of sequences $\mathcal{SEQ}_0$ (*level(0) sequences*), which also forms the initial set of sequences in the iterative procedure. The set of sequences for level $(i+l)$ are generated from the sequences in level i and the corresponding set of dominant patterns $\mathcal{D}$. $\mathcal{P}_{ALL}$ and $\mathcal{P}_{SIG}$ represent the sets of all and significant patterns, respectively. Dominant patterns (denoted by $\mathcal{D}$) for the current level are generated by the *get_dominant_patterns* method. The union of dominant patterns at each level is collected in $\mathcal{L}$.

Algorithm 1 Recursive Data Mining

```
Input: Set of sequences $\mathcal{SEQ}_0$
Output: Sets of patterns (features) $\mathcal{L}$, one for each level
 1: $\mathcal{L} = \{\}$, $i = 0$
 2: repeat
 3:     if $i > 0$ then
 4:         $\mathcal{SEQ}_i = make\_next\_level(\mathcal{SEQ}_{i-1}, \mathcal{D})$ // Level(i)
 5:     end
 6:     $\mathcal{P}_{ALL} = pattern\_generation(\mathcal{SEQ}_i)$
 7:     $\mathcal{P}_{SIG} = sig\_patterns(\mathcal{SEQ}_i, \mathcal{P}_{ALL})$
 8:     $\mathcal{D} = get\_dominant\_patterns(\mathcal{SEQ}_i, \mathcal{P}_{SIG})$
 9:     $\mathcal{L} = \mathcal{L} \cup \mathcal{D}$
10:     $i{+}{+}$
11: until $\mathcal{D} == \varnothing \vee i == max\_level$
12: return $\mathcal{L}$
```

4.1 Pattern Generation

Each input instance is converted into a sequence of tokens. The initial sequence of tokens, *level(0) sequence*, will be referred to as S_0. A sliding window of length l_w moves over S_v ($v = 0$ initially). At each position p of the window, all possible (l_w, max_gap)*–sequence patterns* are generated. The number of patterns generated equals the number of combinations of tokens covered by the window along with the gap token. A bounded hash keeps count of the number of occurrences of each pattern at level v, as the sliding window moves over S_v. This processing forms the first pass over the sequence S_v. Figure 1, shows the patterns generated at position 1 and 2 of the sequence.

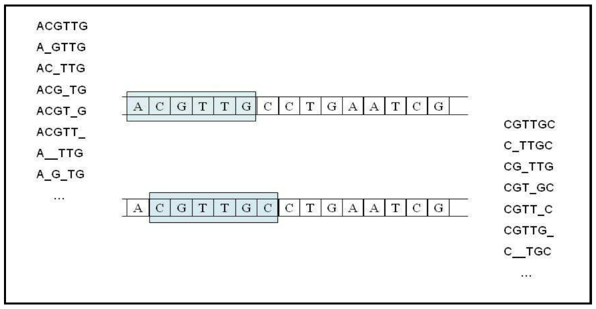


Figure 1: Pattern Generation Step ($l_w = 6, max\_gap = 2$)

4.2 Pattern Significance

The number of (l_w, max_gap)-patterns uncovered in the sequences is generally large. Many of those patterns are either very specific to a certain sequence or insignificant because they contain commonly occurring tokens. In either case, they are ineffective in capturing any stylistic attributes while adding to the computation cost of the algorithm. The "usefulness" of a pattern is computed with a statistical significance test. Patterns that are deemed insignificant are eliminated from further consideration. Recall that the set of unique tokens appearing in a set of sequences $\mathcal{SEQ}$ is denoted by $\mathcal{T}$. The frequency of a token t_i appearing in $\mathcal{SEQ}$ will be denoted by f_{t_i}. So the probability of token t_i appearing in $\mathcal{SEQ}$ denotes as $P(t_i)$ is

$$P(t_i) = \frac{f_{t_i}}{\sum_{j=1}^{|T|} f_{t_j}}. \tag{1}$$

For a pattern $\mathcal{P}$ of length l_w, the probabilities of tokens appearing in the pattern can be represented as a vector $(p_{t_1}, p_{t_2}, \cdots, p_{t_{l_w}})$. Recall that a gap is represented by a special token $\perp$. The probability of pattern $\mathcal{P}$ is thus given by the expression

$$\begin{aligned} \mathbf{P}(\mathcal{P}) &= P(RV_1 = t_1, RV_2 = t_2, \cdots, RV_{l_w} = t_{l_w}) \quad (2) \\ &= p(t_1)p(t_2 \mid t_1) \cdots p(t_{l_w} \mid t_1, \cdots t_{l_w - 1}) \end{aligned}$$

where RV_i is a random variable for appearance of token t_i. Assuming that the words appear independently of each other (this assumption is just for the purpose of measuring pattern significance, because if they are not, co-appearing words will eventually be merged into a single token at the higher level of RDM abstraction), just the marginal probabilities for the words need to be computed, resulting in

$$\mathbf{P}(\mathcal{P}) = \prod_{i=1}^{l_w} p_{t_i}. \tag{3}$$

The probability of a gap token, denoted as ϵ, is a user defined constant (see 4.3 for details). The probability of occurrence of $\mathcal{P}$ under the independent appearance of tokens assumption is given by

$$\mathbf{P}_R(\mathcal{P}) = P(RV_1 = t_1, RV_2 = t_2, \cdots, RV_{l_w} = t_{l_w}). \tag{4}$$

Further assuming equal probability for each token to appear (we will call a model satisfying this assumption a *random appearance model*), the above expression simplifies to

$$\mathbf{P}_R(\mathcal{P}) = \Big(\frac{1}{\mid \mathcal{T} \mid}\Big)^{l_w}. \tag{5}$$

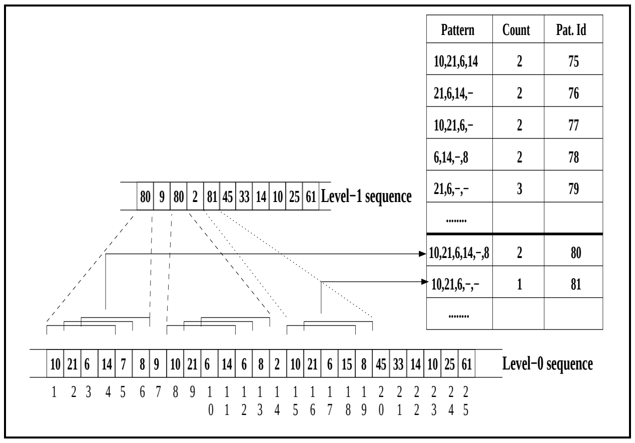
The ratio $\frac{\mathbf{P}_R(\mathcal{P})}{\mathbf{P}(\mathcal{P})}$ is used to determine significance of the pattern. If the above ratio is smaller than 1, then the pattern is considered significant, otherwise it is considered insignificant. The ratio indicates the likelihood of pattern occurrence under the random appearance model as compared to its occurrence under the unknown observed distribution. This is similar in essence to the *log-likelihood ratio test*, with null hypothesis (H_0), that the observed distribution is similar to the random appearance distribution. The alternate hypothesis H_1 states otherwise. The log-likelihood ratio is given by the expression

$$\mathbf{LRT} = -2log_e\Big(\frac{\mathcal{L}_R(\theta)}{\mathcal{L}_O(\theta)}\Big), \tag{6}$$

where $\mathcal{L}_R(\theta)$ is the likelihood function under the random appearance model and $\mathcal{L}_O(\theta)$ is the likelihood for the observed distribution. H_0 is a special case of H_1, since it has fewer parameters (captured by θ) as compared to the more general alternate hypothesis. Applying the significance test to the set of patterns $\mathcal{P}_{ALL}$ gives us a smaller set of *significant patterns*, $\mathcal{P}_{SIG}$.

Other significance tests for sequence patterns have been proposed. Permutation tests [10] provide a simple approach for comparing the observed occurrences of a pattern with the number of likely occurrences over a random sequence. The practical application of this method requires generating a large number of random permutations of the input sequence and computing the statistics on the random permutations. If the input sequence is long, this operation can be computationally expensive. Karlin et al. [16, 15] have proposed significance tests for identifying relevant regions in protein sequences.

4.3 Dominant Patterns



Pattern	Count	Pat. Id
10,21,6,14	2	75
21,6,14,–	2	76
10,21,6,–	2	77
6,14,–,8	2	78
21,6,–,–	3	79
....		
10,21,6,14,–,8	2	80
10,21,6,–,–	1	81
....		

Figure 2: Sequence Re-writing Step

After the significant patterns at level v are determined, a second pass is made over the sequence of tokens S_v. At each position in the sequence, the tokens in the significant patterns are matched against the tokens in the sequence. The matching score is defined as the conditional probability of a match given two symbols, i.e., if $\mathcal{P}[i]$ and $S_v[j]$ are the same then the conditional probability of a match is 1. On the other hand, if $\mathcal{P}[i] = \perp$ then the conditional probability is ϵ. The matching score can be computed as follows:

$$score(\mathcal{P}[i], S_v[j]) = \begin{cases} 1 & \text{if } \mathcal{P}[i] = S_v[j] \\ \epsilon & \text{if } \mathcal{P}[i] = \perp, 0 < \epsilon < 1 \\ 0 & \text{otherwise} \end{cases}, \tag{7}$$

where $\mathcal{P}[i]$ is the i^{th} token of the pattern and j is the corresponding index over sequence S. ϵ is intended to capture the notion that a $\perp$ symbol is not as good as an exact match but much better than a mismatch. The value of ϵ is user defined. It was set to 0.95 in our experiments to favor a match with the gap token. The total score for a pattern, starting at index j in S, is given by

$$score(\mathcal{P}, S_v[j]) = \sum_{i=1}^{|\mathcal{P}|} score(\mathcal{P}[i], S_v[j+i]). \tag{8}$$

The pattern that has the highest score starting at location j in the input sequence is termed as the ***dominant pattern*** starting at position j. In other words, this is a pattern x defined by the expression $\text{argmax}_{x \in S_v}\, score(x, S_v[j])$. The term dominant pattern reflects the fact that this pattern

dominates over all other significant patterns for this position in the sequence. Two dominant patterns that are placed next to each other can be merged to form longer dominant patterns. The merging process is continued till no further dominant patterns can be merged.
An example of the merging process is shown in Figure 2. After the first pass of the sequence at level v, a new token is assigned to each dominant pattern found. During the second pass at this level, the sequence for level $v + 1$ is generated. In this process, each subsequence corresponding to a dominant pattern is replaced by the new token for this dominant pattern. When no dominant pattern can be matched at position j, the original token is copied from sequence S_v to the new sequence S_{v+1}. Figure 2 illustrates this step.

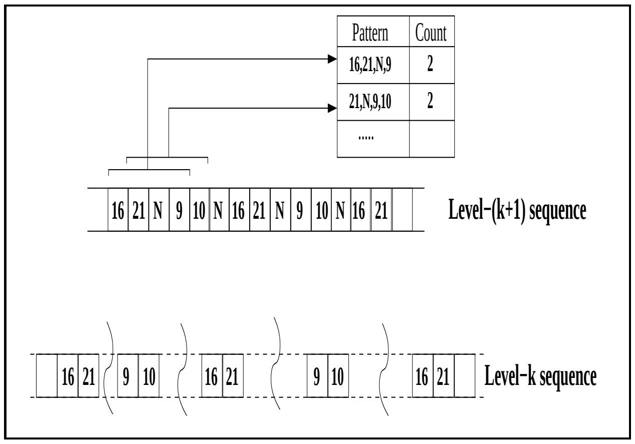


Figure 3: Removing Noisy Tokens from Long Range Patterns

As the RDM algorithm generates subsequent levels, certain tokens get carried over from lower levels without participating in any dominant patterns at higher levels. Such tokens are termed "**noisy**" for the following reasons. First, they do not contribute to any patterns at these levels. Second, they obstruct the discovery of patterns that are separated by a long sequence of noisy tokens. Patterns separated by noisy tokens are called *long range patterns*. These long range patterns can be captured only if the noisy tokens lying in between them can be collapsed. As a result, at each level, we collapse contiguous sequence of tokens that have not resulted in new dominant patterns for the last k levels, into a special noise token. Value of constant k is selected using the tuning dataset (see Section 5). Figure 3 illustrates the process of collapsing noise tokens into a single special token N. Once the noise tokens are collapsed, distant tokens can now fall within the same window, leading to more patterns being discovered at higher levels.
The set of dominant patterns D_v at level v form the features for this level. This iterative process of deriving level $v + 1$ sequence from level v sequence is carried on till no further dominant patterns are found or $v + 1$ has reached a user predefined maximum value. The sets of features extracted are utilized by an ensemble of classifiers.

4.4 Training Phase and Testing Phase

The training phase involves using dominant patterns generated at each level to construct an ensemble of classifiers $(\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_{max\_level})$, one for each level. The dominant patterns reflect the most relevant patterns, ignoring the highly frequent and infrequent patterns (upper and lower cut–offs in the pattern frequency distribution). The upper and lower cut–offs are intended to prevent the use of insignificant patterns as features. The classifiers can be created from any machine learning method, such as Naïve Bayes or Support Vector Machine.
Given a set of text documents $\mathcal{SEQ}_{tr}$, along with the labels $r_1, r_2, \cdots, r_v$ of all possible classes, dominant patterns are generated for each document starting at level 0 up to level *max_level*. The union of all tokens in $\mathcal{T}$ and dominant patterns at a level v across all documents in $\mathcal{SEQ}_{tr}$ forms the set of feature for classifier $\mathcal{C}_v$. For the ensemble of classifiers, the final prediction value is the weighted sum of the class prediction of individual classifier. Each classifier is assigned a weight that reflects the confidence of the classifier. In order to determine this confidence value, the set $\mathcal{SEQ}_{tr}$ is further split into a tuning set $\mathcal{SEQ}_{tu}$ and a training set $\mathcal{SEQ}_{tr} \backslash \mathcal{SEQ}_{tu}$, where $\mathcal{A} \backslash B$ denotes the set difference operation $A - B$ (A minus B). Each classifier in the ensemble trains its model based on $\mathcal{SEQ}_{tr} \backslash \mathcal{SEQ}_{tu}$. The accuracy of the classifier on the tuning set determines the confidence of classifier $\mathcal{C}_i$ as

$$conf(\mathcal{C}_i) = \frac{accuracy(\mathcal{C}_i)}{\sum_{j=1}^{max\_levels} accuracy(\mathcal{C}_j)}, \tag{9}$$

where $accuracy(\mathcal{C}_j)$ is computed over the tuning set $\mathcal{SEQ}_{tu}$.
After the training phase discovers features from the training data, the testing phase finds occurrences of those features in the test data. Hence, the testing phase follows the training phase in terms of the level by level operating strategy. If a dominant pattern X was discovered at level(v) during the training phase, then it can be only applied to level(v) in the testing phase. Initially, the frequencies of tokens and level(0) dominant patterns are counted over the level(0) test sequence. This vector of frequencies forms the feature vector at level(0). Once the feature vector for level(0) is obtained, the next level sequence is generated. This is achieved by substituting the token of the best matching pattern at every position in the level(0) test sequence. It should be noted that if the best match has a score below the user specified threshold α, then the token at level(0) is carried over to level(1). Now, the occurrences of the dominant patterns at level(1) are counted over level(1) test sequence. This process continues till all levels of dominant patterns are exhausted. Each classifier in the ensemble classifies the test data and the final prediction value is according to the following weighing scheme:

$$\mathbf{P}(C \mid x) = \sum_{i=1}^{max\_levels} conf(\mathcal{C}_i) \times \mathbf{P}_{\mathcal{C}_i}(C \mid x), \tag{10}$$

where x is a test sequence and $\mathbf{P}_{\mathcal{C}_i}(C \mid x)$ is the prediction value assigned by classifier $\mathcal{C}_i$.

5. EXPERIMENTS AND RESULTS

We show that RDM performs better than comparable classifiers such as Naïve Bayes (NB), Support Vector Machines (SVM) and Predictive Association Rule based (CPAR [24], which combines the advantages of associative and traditional rule-based classifiers). Support Vector Machines based classifiers have been shown by [13] to perform well for text classification tasks. We used SVMLight as the SVM implemen-

Table 1: Dataset for Role Identification

	Training Set	Testing Set	Total	# Sent folders
CEO	1010	250	1260	3
Manager	1403	349	1752	4
Trader	654	162	816	4
VP	1316	327	1643	4
Total	4383	1088	5471	15

tation [14], and IlliMine package for CPAR [24]. RDM does not use any semantic tools (part-of-speech tagging or synonym groups) in order to extract patterns that later serve as features for the classifiers. As a result, we compare RDM with other techniques that do not utilize domain or semantic knowledge either. A brief introduction to the Enron email dataset used for running the experiments is provided before the discussion of the experimental setup.

5.1 Data Preparation and Experimental Setup

Experiments are performed on the March 2, 2004 version of Enron email dataset, distributed by William Cohen [4]. The dataset was cleaned to eliminate attachments, quoted text and tables from the body of the email messages and header fields from the email. No effort was made to correct spelling errors or to expand abbreviations in an attempt to reduce the noise in the data. However, stemming was performed on the dataset.

For our purpose of identifying roles, employees were partitioned into groups based on their organizational role in Enron, as suggested in [5]. Only the roles of *CEO, Manager, Trader and Vice-President* were used in our experiments since a large number of employees were designated with these roles. Since we are concerned with identifying roles based on messages sent by employees, we only deal with the messages in the *Sent* folder of each participant. For each of the roles, the emails are divided into two sets as summarized in Table 1. Finally, each stemmed word in an email is considered a token, and each email represents one sequence.

The RDM algorithm requires a few parameters to be set for the classification model. The parameters to be selected include 1) the size of the window, 2) the maximum number of gaps allowed in the window, 3) the weights assigned to the classifier at each level, 4) the parameter k used to eliminate noisy tokens. A greedy search over the parameter space is conducted to determine the best set of parameter values. To compute the parameter values, the training set is further split into two parts. A classifier is trained on the larger part, and tuned on the smaller part (called the tuning set).

5.2 Results

We compare the classifiers – Naïve Bayes, RDM with NB, SVM, RDM with SVM and CPAR – under two classification settings: binary and multi-class. RDM was used both with Naïve Bayes, and SVM as the ensemble classifiers. For both classification settings, *F-measure*, which is the harmonic mean of precision and recall, is used to compare performance of the classifiers. In the binary classification setting, given a test message m, the task is to answer the question *"Is message* m *sent by a person with role r"?* where $r \in \mathcal{R} = \{\text{CEO, Manager, Trader, Vice-President}\}$.

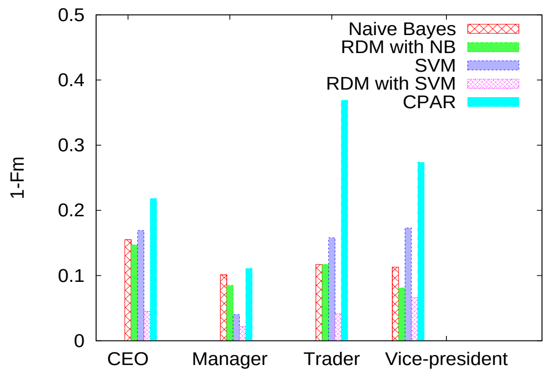


Figure 4: Binary Classification – F-measure

The training set is divided in such a way that all messages belonging to role r form the positive class and all messages belonging to $\mathcal{R}\backslash r$ form the negative class. The performance for the five classifiers is shown in Figure 4, where the values of 1 - *F-measure* are presented to highlight the differences in performances. Note that a smaller value of 1 - *F-measure* indicates a better classifier. In terms of the F-measure, RDM with SVM performs better than NB, SVM or CPAR for all tested roles while RDM with NB performs better for most of the roles. To further analyze the results, we computed the Root Mean Square Error (RMSE) for NB and RDM with NB. The RMSE is computed using the expression

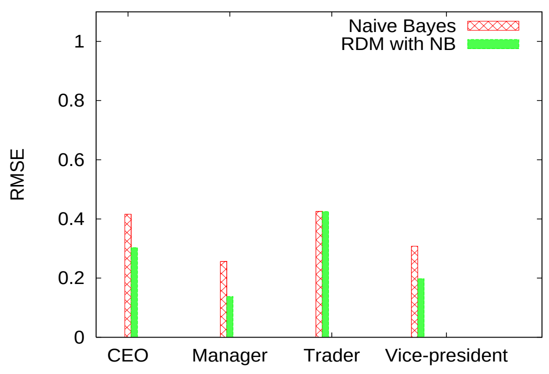


Figure 5: Binary Classification – RMSE Comparison

$$RMSE(\mathcal{T}_{test}) = \sqrt{\frac{\sum_{i=1}^{|\mathcal{T}_{test}|}(1 - P(r \mid \mathcal{T}_{test}{}^{i}))^2}{\mid \mathcal{T}_{test} \mid}}, \qquad (11)$$

where $\mathcal{T}_{test}{}^{i}$ is the i^{th} document in the test set and $r = \operatorname{argmax}_c P(c \mid \mathcal{T}_{test}{}^{i})$. Since the decision function value from SVMLight could not be converted to an error term, the plot in Figure 5 does not show comparison with SVM. Similarly, CPAR does not provide any comparable measure. The lower the RMSE value is, the more confident is the classifier in its prediction. Figure 5 shows that RDM with NB is more confident in its predictions even when the F-scores for RDM with NB and NB are very close for a certain role.

The second set of results compares the performance under the multi-class classification setting, wherein the task is to answer the question *"Which is the most likely role, out of* *roles $R_1, \ldots, R_n$, for sender of message* m*?"* For NB and RDM, the training data is split into four groups and probabilities computed for each of the roles. For SVM, four sets of datasets are generated, one for each pair of roles $(r, \mathcal{R}\backslash r)$. The comparison for the classifiers is shown in Figure 6. RDM convincingly outperforms the other classifiers in this comparison.

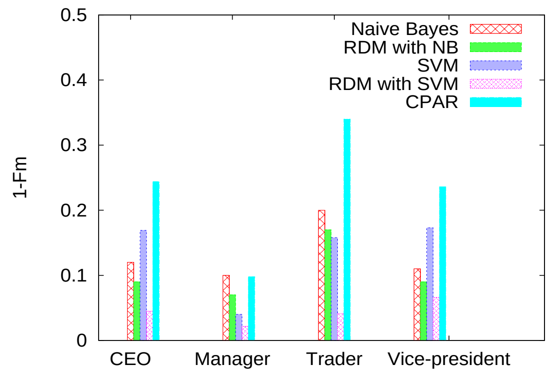


Figure 6: Multi-class Classification – F-measure

To further investigate the results obtained for the multi-class scenario, we performed the *paired t-test* for statistical significance. The paired t-test provides a hypothesis test of the difference between population means for a pair of random samples whose differences are approximately normally distributed. Note that a pair of samples, each of which may not be from a normal distribution, often yields differences that are normally distributed. The null hypothesis H_0 states that the difference in performance between RDM and the other methods is not significant. In other words, H_0 states that the two methods perform equally well. The alternate hypothesis, states otherwise.

A 20-fold cross validation was performed on the data. The accuracy results obtained therein are used for the t-test, where SVM and CPAR are compared against RDM with SVM (denoted as RDM-SVM), and NB is compared against RDM with NB (denoted as RDM-NB). The results are shown in Table 2. Based on the p-value in Table 2 we reject the null hypothesis, indicating a definite improvement provided by RDM. The confidence interval for the mean difference shows that the improvement lies between 1.8% and 3% as compared to NB, whereas compared to SVM (and CPAR), it is between 8% and 10%.

For the final test we divide each role into two parts based on the user identities. For instance, the folders of *Jeffrey Skillings, David Delainey* and *John Lavorato* form the CEO group, because CEO's of Enron subsidiaries are also considered as Enron CEO's for our experiments. The first part, namely training set, contains messages from *John Lavorato* and *David Delainey*, while messages from *Jeffrey Skillings* form the second part (test set). An RDM based classifier is trained using messages in the first part and tested on messages in the second part. In this experiment, we analyze the performance of the classifier for a member whose messages are not in the training set. The results for different roles are shown in Figure 7. The test set size is gradually increased and the accuracy is noted. Notice that for the roles of Manager, Trader and Vice-President, the accuracy increases with larger number of message. The converse is observed for the role of CEO. After examining the messages for the CEO, we observed that most of the messages were written by CEO's administrative assistants. This explains the relatively poor performance of classifiers for this role.

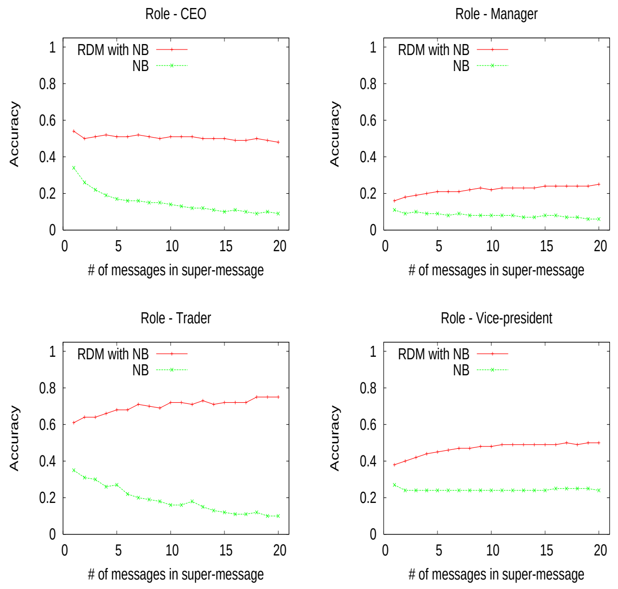


Figure 7: Frequency of Accurate Classification over Unseen Message Folder

6. CONCLUSIONS

We propose a general framework for feature extraction from a sequence of tokens. The framework is based on the idea of capturing statistically significant sequence patterns at increasing levels of generalization. These patterns act as features for an ensemble of classifiers, with one classifier defined at each level of generalization. The proposed method is simple and flexible, hence, it can be applied to a range of applications. We applied it to capture stylistic patterns in the Enron email dataset and used those patterns for identifying the organizational roles of authors. The method, in its current state, is devoid of any semantic knowledge, which can be easily incorporated to identify semantically related patterns. Techniques such as part of speech tagging and synonym dictionaries can augment our approach. Based on the success of the method on a noisy dataset, we believe that the method can perform better on cleaner datasets and on other application areas such as grouping gene products by their families.

For our future work, we plan to conduct experiment to demonstrate the broad applicability of this method on cleaner and more structured text categorization datasets, such as gene datasets, including GenBank database, and foreign language datasets, including the Reuters dataset and Russian Blogosphere data.

7. ACKNOWLEDGMENTS

This work was partially supported by the ONR Contract N00014-06-1-0466. The content of this paper does not nec-

Table 2: Results of paired t-test

Classifier Pair	Mean difference	Std. Dev. of ($\bar{d}$)	t-statistic (df=19)	p-value	95% confidence interval
NB vs RDM-NB	0.02393	0.002525	9.48	1.23E-08	(0.0186 - 0.0292)
SVM vs RDM-SVM	0.08927	0.00434	20.55	1.94E-14	(0.0818 - 0.0984)
CPAR vs RDM-SVM	0.09329	0.00535	17.45	3.74E-13	(0.0821 - 0.1045)

essarily reflect the position or policy of the U.S. Government, no official endorsement should be inferred or implied.

8. REFERENCES

[1] B. Bouqata, C. Carothers, B. Szymanski, and M. Zaki. Vogue: A novel variable order-gap state machine for modeling sequences. In *Proc. Tenth European Conf. Principles and Practice of Knowledge Discovery in Databases, PPKDD'06*, pages 42–54, 2006.

[2] H. Cheng, X. Yan, J. Han, and C. Hsu. Discriminative frequent pattern analysis for effective classification. In *Proc. Int. Conf. Data Engineering*, pages 716–725, 2007.

[3] K. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proc. Second Conf. Applied Natural Language Processing*, Austin, Texas, 1988.

[4] W. Cohen. http://www.cs.cmu.edu/˜enron/.

[5] W. Cohen. http://isi.edu/˜adibi/Enron/Enron Employee Status.xls.

[6] A. I. Deac, J. C. A. van der Lubbe, and E. Backer. *Feature Selection for Paintings Classification by Optimal Tree Pruning*. Springer Verlag, Berlin, Germany, 2006.

[7] P. F. Evangelista, M. J. Embrechts, and B. K. Szymanski. Taming the curse of dimensionality in kernels and novelty detection. In *Applied Soft Computing Technologies: The Challenge of Complexity, Abraham, A.; Baets, B.d.; Koppen, M.; Nickolay, B. (Eds.)*, pages 425–438, Berlin, Germany, 2006. Springer Verlag.

[8] K. Fialkowski and B. K. Szymanski. Conceptor: a model of selected consciousness features including emergence of basic speech structures in early childhood. In *Art, Technology, Consciousness mind@large, Roy Ascott (ed.)*, pages 185–190, Bristol, U.K., 2000. Intellect Press.

[9] M. Goldberg, M. Hayvanovich, A. Hoonlor, S. Kelley, Malik, Magdon-Ismail, K. Mertsalov, B. K. Szymanski, and W. Wallace. Discovery, analysis and monitoring of hidden social networks and their evolution. In *Proc. IEEE Int. Conf. Technologies for Homeland Security*, Waltham, MA, 2008.

[10] P. Good. *Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypotheses*. Springer, Berlin, Germany, 2000.

[11] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[12] H. Halteren, J. Zavrel, , and W. Daelemans. Improving accuracy in NLP through combination of machine learning systems. *Computational Linguistics*, 27(2):199–229, 2001.

[13] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proc. Tenth European Conf. Machine Learning, ECML'98*, 1998.

[14] T. Joachims. *Making large-Scale SVM Learning Practical*. MIT-Press, Cambridge, Massachusetts, 1999.

[15] S. Karlin. Statistical significance of sequence patterns in proteins. *Current Opinion in Structural Biology*, 5:360–371, 1995.

[16] S. Karlin and V. Brendel. Chance and statistical significance in protein and dna sequence analysis. *Science*, 257:39–49, 1997.

[17] B. Klimt and Y. Yang. The enron corpus: A new dataset for email classification research. In *Proc. Sixteenth European Conf. Machine Learning, ECML'04*, pages 217–226, 2004.

[18] F. Mosteller and D. L. Wallace. *Inference and disputed authorship:The Federalist*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1964.

[19] C. G. Nevill-Manning and I. H. Witten. Identifying hierarchical structure in sequences. *Journal of Artificial Intelligence Research*, 7:67–82, 1997.

[20] C. Rey and J. Dugelay. Blind detection of malicious alterations on still images using robust watermarks. In *Proc. IEEE Seminar Secure Images and Image Authentication*, pages 7/1–7/6, 2000.

[21] Z. Solan, D. Horn, E. Ruppin, and S. Edelman. Unsupervised learning of natural languages. *Proc. Natl. Acad. Sci. U S A*, 102(33):11629–11634, 2005.

[22] B. Szymanski and Y. Zhang. Recursive data mining for masquerade detection and author identification. In *Proc. Fifth IEEE Information Assurance Workshop*, pages 424–431, 2004.

[23] H. Yang, Z. Pan, X. Wang, and B. Xu. A personalized products selection assistance based on e-commerce machine learning. In *Proc. Int. Conf. Machine Learning and Cybernetics*, volume 4, pages 2629–2633, 2004.

[24] X. Yin and J. Han. Cpar: Classification based on predictive association rules. In *Proc. SIAM Int. Conf. Data Mining (SDM'03)*, San Francisco, CA, 2003.

[25] M. Yousef, S. Jung, L. C. Showe, and M. K. Showe. Recursive cluster elimination (rce) for classification and feature selection from gene expression data. *BMC Bioinformatics*, 8, 2007.