# Recognizing End-User Transactions in Performance Management

**Joseph L. Hellerstein, T.S. Jayram, Irina Rish**
IBM Thomas J. Watson Research Center
Hawthorne, New York
{hellers, jayram, rish}@us.ibm.com

## Abstract

Providing good quality of service (e.g., low response times) in distributed computer systems requires measuring end-user perceptions of performance. Unfortunately, such measures are often expensive or impossible to obtain. Herein, we propose a machine-learning approach to recognizing end-user transactions consisting of sequences of remote procedure calls (RPCs) received at a server. Two problems are addressed. The first problem is labeling an RPC sequence that corresponds to one transaction instance with the correct transaction type. This is akin to text classification. The second problem is transaction recognition, a more comprehensive task that involves segmenting RPC sequences into transaction instances and labeling those instances with transaction types. This problem is similar to segmenting sounds into words as in speech understanding. Using Naive Bayes approach, we tackle the labeling problem with four combinations of feature vectors and probability distributions: RPC occurrences with the Bernoulli distribution and RPC counts with the multinomial, geometric, and shifted geometric distributions. Our approach to transaction recognition uses a dynamic-programming Viterbi algorithm that searches for a most likely segmentation of an RPC sequence into a sequence of transactions, assuming transaction independence and using our classifiers to select a most likely transaction label for a given RPC sequence. For both problems, good accuracies are obtained, although the labeling problem achieves higher accuracies (up to 87%) than does transaction recognition (64%).

## Introduction

Providing good quality of service (e.g., low response times) to end-users of distributed computer systems is essential for e-Commerce, among other applications. A first step is to characterize *end-user transactions*. An end-user transaction is a sequence of interactions between the end user and his/her workstation that reflects a logically complete unit of work, such as opening a database, opening a view, reading several records and closing the database. Characterizing end-user transactions (or simply transactions) is needed to (a) better quantify end-user perception of performance, (b) create representative workloads, and (c) provide better resource management. This paper describes a machine-learning approach to recognizing transactions.



Transactions consist of sequences of commands that end-users issue to their workstation. In distributed systems, these commands typically cause *remote procedure calls (RPCs)* to be sent from the user's workstation to one or more tiers of servers that process the RPCs. To illustrate the forgoing, we use the Lotus Notes email system. Common RPCs include OPEN_DB, READ_ENTRIES, and FIND_BY_KEY. Examples of the transactions in Lotus Notes include: replication, search for a note, update notes, and re-sort view.

Because end-user workstations are so numerous and since they are often not the responsibility of the administrative staff, there is often little opportunity to collect measurement data from the workstation itself. Rather, it is at the servers where measurements (e.g., RPCs received) are collected. Unfortunately, little information about end-user transactions is present at the server. In principle, client-server protocols could be instrumented to mark the beginning and end of user interactions. However, this is not sufficient to identify transactions since users often view a sequence of application interactions as a single unit of work. In current practice, this quandary is addressed either by using surrogates for transactions (e.g., synthetic transaction generated by probing stations) or by labeling transactions manually for post-processing. The former often leads to incorrect assessments of service quality. The latter is extremely time consuming. Indeed, it took multiple experts several weeks to segment and label the data we use in this paper.

Our objective is to automatically identify transactions that correspond to a sequence of RPCs from a user. This *transaction recognition* problem involves *segmenting* a sequence of RPCs into transaction instances and *labeling* these instances with the correct transaction types. Labeling alone is a classification problem akin to text classification. The segmentation problem is similar to that faced in speech understanding where an acoustic model is used to partition sounds into words.

Fig. 1 depicts the two problems we address. In problem 1 (labeling), the sequence of RPCs has been separated into sessions (the figure shows one session), which in turn have been segmented into transaction instances. The task here is to label each instance with the correct transaction type based

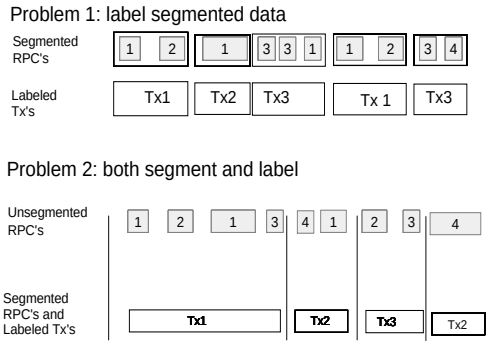


Figure 1: Illustration of labeling and transaction recognition problems.

on the RPCs in the instance. For example, in the figure, the third transaction instance, which consists of two RPCs of type 3 and one RPC of type 1, is labeled as Tx3. In problem 2 (transaction recognition), the task is to both segment RPCs into transaction instances and to label these instances. For example, the RPC sequence $(1, 2, 1, 3, 4, 1, 2, 3, 4)$ is segmented into four transaction instances that are labeled Tx1, Tx2, Tx3, and Tx2.

Herein, we propose the use of machine-learning techniques to recognize transactions. For labeling, we use Naive Bayes classifiers specified by the choice of feature vector and by conditional probabilities of each feature given a class (a Naive Bayes model assumes that features are mutually independent given a class). Our approach to transaction recognition uses a dynamic-programming Viterbi algorithm that searches for a most likely segmentation of an RPC sequences into a sequence of transactions, assuming transaction independence and using our classifiers to select a most likely transaction label for a given RPC sequence.

The results herein reported are of three types. First, we provide insight into a new problem domain for machine learning — recognizing end-user transactions to aid in performance management. Second, we demonstrate that Naive Bayes works well for the labeling problem in that it provides an accuracy of approximately 85-87% (with over 30 transaction classes). Third, we describe an approach to the transaction recognition problem that attains an accuracy close to 64%.

The remainder of this paper is organized as follows. Section 2 describes the data characteristics and discusses probabilistic models used later for labeling and for transaction recognition. Section 3 details our results for labeling transaction instances, and Section 4 does the same for labeling with segmentation. Section 5 discusses related work. Our conclusions are contained in Section 6.

## Data Characteristics

This section describes the data characteristics and discusses probabilistic models that we use in subsequent sections.

Our data are obtained from a Lotus Notes email server at a large oil company. The data consist of traces of individual RPCs collected during two one-hour measurements of the email interactions of several hundred users. Included in the trace is the type of RPC (e.g., OPEN_COLLECTION), the identity of the server connection (which identifies a single user), and the time (in seconds) at which the request is made. In addition, we have the results of the segmentation and labeling done by Lotus Notes experts, a process that took several weeks.

The data are organized into two data sets, data set 1 and data set 2. Each data set contains approximately 1,500 transaction instances and about 15,000 RPC instances. There are 32 different types of transactions and 92 RPC types. Fig. 2 displays the marginal distributions of transaction and RPC types. Note that the distributions are highly skewed.

The segmentation and labeling done by Lotus Notes experts allows us to structure the data into transaction instances labeled with their types. Instances have a variable number of RPCs. The transaction type is used as the class variable. Our feature vector is a function of the RPC's within a transaction instance.

Our choice of feature vectors is based on what has been employed in related literature (e.g., text classification) and ease of computation. Two feature vectors are considered. Both are represented as a vector of length $M$ (the number of RPC types). The first is the occurrence of each RPC type. Referring to Fig. 1, $M = 4$ and the value of the occurrence feature vector for the third transaction instance is $(1, 0, 1, 0)$. A second feature vector is RPC counts. Again referring to Fig. 1 and the third instance, the value of the count feature vector is $(1, 0, 2, 0)$.

Applying Naive Bayes requires estimating the conditional probability of each feature given a transaction type. For occurrences, the Bernoulli distribution is used. (A Bernoulli random variable takes value 1 with probability $p$ and value 0 with probability $1 - p$.) Thus, for each combination of RPC type and transaction type, we estimate $P(o_{ij} = 1|T_i)$, where $o_{ij} = 1$ if RPC of type $j$ occurred in a transaction instance of type $i$, and 0 otherwise.

For counts, we need to estimate $P(n_{ij}|T_i)$ for each $i, j$ and each value of $n_{ij}$, where $n_{i,j} = 0, 1, 2, ...$ is the number of RPCs of type $j$ in a transaction instance of type $i$ (RPC count). Our approach to these estimation problems is to use several parametric distributions. First, we consider the multinomial distribution over the RPC counts:

$$P(n_{i1}, ..., n_{iM}|T_i) = \frac{n!}{\prod_{j=1}^{M} n_{ij}!} \prod_{j=1}^{M} p_{ij}^{n_{ij}},$$

where $p_{ij}$ are the parameters of the distribution satisfying $\sum_{j=1}^{M} p_{ij} = 1$, and $n$ is the total number of RPCs in a transaction, $\sum_{j=1}^{M} n_{ij} = n$. The multinomial distribution has been successfully used for text classification with word counts as features (Nigam *et al.* 1998; McCallum & Nigam 1998). Note, however, that the multinomial distribution goes beyond the Naive Bayes assumption since it implies a dependency between the RPC counts (they must sum to the total number of RPCs in a transaction instance). Another parametric distribution we consider is the geometric distri-

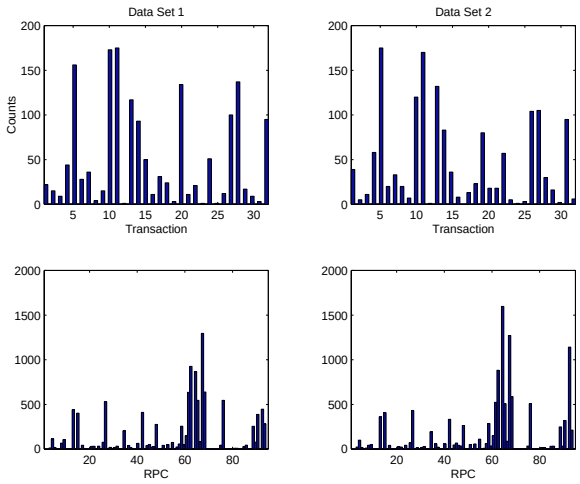


Figure 2: Marginal Distributions of RPC Types and Transaction Types

bution:

$$P(n_{ij}|T_i) = p_{ij}^{n_{ij}}(1-p_{ij}), n_{ij} = 0, 1, 2, 3, ....$$

This distribution is widely used to describe performance characteristics of queuing systems (Kleinrock 1975), a perspective that is consistent with our application domain.

A closer look at the nature of client-server protocols suggests a third distribution that is a variation on the geometric. Specifically, client-server interactions are broadly of two types. The first are fixed overheads, such as opening a database or accessing a collection of objects. Once this has been done then "payload" operations may take place, such as reads and writes. This suggests that we should mix deterministic distributions with distributions that have substantial variability. It turns out that a variant of the geometric distribution can accommodate these requirements. The variant, which we call the shifted geometric distribution, includes a shift parameter $\nu_{ij}$ that specifies the minimum count for RPCs of type $j$ in a transaction instance of type $i$. Namely,

$$P(n_{ij}|T_i) = p_{ij}^{n_{ij}-\nu_{ij}}(1-p_{ij}),$$

where $P(n_{ij}|T_i) = 0$ if $n_{ij} < \nu_{ij}$. Thus, a shifted geometric distribution with a fixed shift parameter and a probability parameter of 0 is a deterministic distribution. Details of the shifted geometric, including its maximum likelihood estimators, are contained in Appendix A.

Which distribution function best fits the RPC counts in our data? Answering this question is complicated by the fact that we have a mixture of distributions. For the multinomial, there is a distribution for each transaction type. For geometric and shifted geometric, there are 308 distributions—one for each combination of transaction type and RPC type that occurs in our data.

We proceed as follows. For each distribution function, we calculate its parameters for each combination of transaction and RPC type using standard maximum-likelihood estimators. Then, we use Monte Carlo techniques to generate synthetic transaction instances in accordance with the parameters of the distribution function. A large number of synthetic instances is generated in order to achieve a very low variance in the estimation of distribution quantiles. Using a Chi-square goodness of fit test, we compare the empirical distribution of each function's synthetic instances with the empirical distribution of our data. The Chi-square statistics have the interpretation that a lower number indicates that the distribution family better approximates the data. The results are reported in Fig. 3. While no distribution provides a very good fit (primarily due to the frequency of zero counts), it is clear that the shifted geometric provides the best fit and the multinomial has the worst fit.

| | **Data Set 1** | **Data Set 2** |
|---|---|---|
| **Multinomial** | 2238 | 1928 |
| **Geometric** | 490 | 398 |
| **Shifted Geometric** | 192 | 178 |

Figure 3: Chi-Square Statistics for Fits of Parametric Distributions

## Labeling

This section describes our approach to assigning transaction types to previously segmented transaction instances. This is a classification task, where $C = \{T_1, ..., T_n\}$ is the set of possible labels (transaction types), $F_k = (f_k^1, ..., f_k^M)$ is the feature vector computed for the $k$th transaction instance, and $f_k^j$ denotes the feature corresponding to RPC of type $j$. We consider two feature types: RPC occurrences and RPC counts. More complex features, such as those related to sequencing, are beyond the scope of this paper.

We use Naive Bayes classifiers. Given transaction instance $k$, we seek $T_i$ that maximizes $P(T_i|F_k)$. Applying Bayes rule gives $P(T_i|F_k) = \frac{P(F_k|T_i)P(T_i)}{P(F_k)}$. Since Naive Bayes assumes conditional independence between features given class, we get $P(F_k|T_i) = \prod_j P(f_k^j|T_i)$.

We consider several classifiers, each specified by a feature type and a feature distribution: RPC occurrences with the Bernoulli distribution, and RPC counts with the multinomial, geometric, and shifted geometric distributions. These choices are based on previous work in text classification and the data characteristics described in the previous section. We also consider two different parameter estimators for the Bernoulli model: (1) the maximum-likelihood estimator and (2) a non-standard estimator that coincides with the maximum-likelihood estimator of the parameter of the geometric distribution.

Why not use the empirical probability distribution functions (PDFs)? The empirical PDF estimates $P(f = v|c)$ from the frequency of the value $v$ given the feature $f$ and the class $c$ in the training data set. For occurrences, the empirical PDF coincides with the Bernoulli distribution. However, for counts, the situation is more complex. Since the range of count values is potentially unbounded, there may be feature values in the test data for which no probability es-

Table 1: Results for the data set 1: mean $\mu$ and standard deviation $\sigma$ of the classification accuracy (in percents).

| N | BEG | BE | MN | GE | SG |
|---|---|---|---|---|---|
| | $\mu \pm \sigma$ | $\mu \pm \sigma$ | $\mu \pm \sigma$ | $\mu \pm \sigma$ | $\mu \pm \sigma$ |
| 160 | $73 \pm 6\%$ | $72 \pm 5\%$ | $73 \pm 5\%$ | $74 \pm 6\%$ | $61 \pm 7\%$ |
| 320 | $80 \pm 5\%$ | $80 \pm 4\%$ | $81 \pm 4\%$ | $81 \pm 5\%$ | $70 \pm 8\%$ |
| 480 | $83 \pm 3\%$ | $83 \pm 3\%$ | $83 \pm 3\%$ | $83 \pm 3\%$ | $73 \pm 8\%$ |
| 799 | $84 \pm 4\%$ | $84 \pm 3\%$ | $84 \pm 4\%$ | $85 \pm 3\%$ | $78 \pm 5\%$ |
| 1119 | $85 \pm 4\%$ | $85 \pm 3\%$ | $85 \pm 3\%$ | $86 \pm 3\%$ | $79 \pm 5\%$ |
| 1439 | $86 \pm 2\%$ | $85 \pm 3\%$ | $86 \pm 2\%$ | $87 \pm 2\%$ | $79 \pm 3\%$ |

timate can be obtained in the training data. This difficult can be addressed by grouping together ranges of values in the test data, but doing so creates a yet another problem–how to form these groups. Because of these complexities, we do not consider the empirical PDF for counts.

The accuracy of the classifiers is gauged in two ways. The first is the fraction of correctly labeled transactions. The second approach is to measure the fraction of correctly labeled RPCs. This approach puts more weight on longer transaction instances. It turns out that the two metrics produce results that are within a few percent of each other. So, we only report results for the first metric.

Our methodology sets aside 10% of the transaction instances for testing, and uses a subset of the remaining data for training the classifier. We varied the size of the training data set from 10% to 90% of the original (input) data set size, to see the effect of the training set size on classification accuracy. We did 30 runs for each subset size, randomly selecting the test set and then randomly selecting the training subset from the remaining data. In each run, the training and the test sets were fixed for all classifiers.

Figure 4 and Table 1 present the results of experiments for data set 1. The x-axis in the figure is training set size, and the y-axis is average labeling accuracy. Table 1 presents both means and standard deviations of the accuracy. The first column, $N$, is the number of training instances, while the columns 2 to 6 show the classification results for the following distributions: Bernoulli with geometric estimator (**BEG**), Bernoulli with ML estimator (**BE**), multinomial (**MN**), geometric (**GE**), and shifted geometric (**SG**). Note that accuracy generally improves with the size of the training set. All of the classifiers have an accuracy in excess of 75%. This is quite competitive with the literature on text classification (e.g., (Nigam *et al.* 1998; McCallum & Nigam 1998)) and is much better than a classifier that always chooses the transaction type that occurs with highest probability in the data (which would provide an accuracy of about 10%). Note that the counts-with-shifted-geometric classifier has accuracies that are consistently lower than the others. This is in contrast to Fig. 3 that shows the shifted geometric provides the best fit to the data. Therefore, the best-fitting distribution may not necessarily provide the best classifier.

Another interesting observation is that occurrences-with-Bernoulli, counts-with-multinomial, and counts-with-geometric yield very similar results. This is usually not the case in text classification. For example, others have

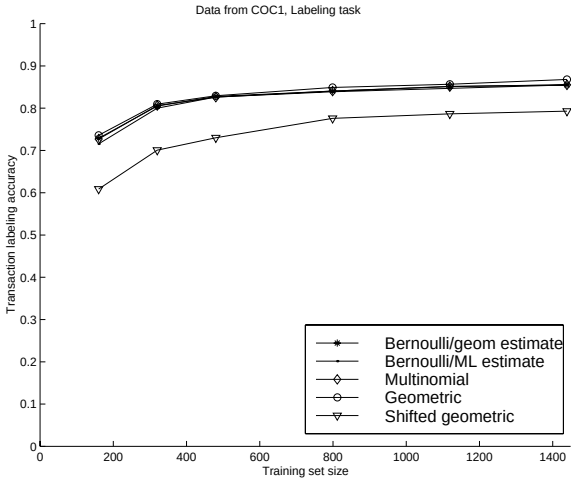


Figure 4: Classification results for the data set 1.

Table 2: Classification results for three data sets using 90% of the data for training and the rest for testing.

| Data set | BEG | BE | MN | GE | SG |
|---|---|---|---|---|---|
| | $\mu \pm \sigma$ | $\mu \pm \sigma$ | $\mu \pm \sigma$ | $\mu \pm \sigma$ | $\mu \pm \sigma$ |
| 1 | $86 \pm 2\%$ | $85 \pm 3\%$ | $85 \pm 3\%$ | $86 \pm 2\%$ | $79 \pm 3\%$ |
| 2 | $83 \pm 2\%$ | $84 \pm 3\%$ | $81 \pm 3\%$ | $82 \pm 3\%$ | $78 \pm 4\%$ |
| 3 | $84 \pm 1\%$ | $85 \pm 1\%$ | $85 \pm 1\%$ | $84 \pm 1\%$ | $77 \pm 1\%$ |

shown that the counts-with-multinomial classifier is significantly better than occurrences-with-Bernoulli (McCallum & Nigam 1998). That our data do not abide by this principle suggests that its characteristics differ from those of text classification.

Table 2 presents similar results for the data set 2 and for an additional data set 3 obtained from a different customer. Data set 3 contains 73 RPC types, 37 transaction types, and 16210 transaction instances. Here we only report the results for 90% of the data used for training. These data are consistent with the preliminary conclusions stated above. The shifted geometric classifier is significantly less accurate than that for the other distributions. Further, accuracies do not change much between data sets if the same classifier is used.

## Transaction Recognition

Here, we consider the more difficult problem of transaction recognition that includes segmentation and labeling. We proceed by assuming that transactions are mutually independent. Then, using the dynamic-programming approach known as Viterbi search (Jelinek 1998; Fu 1982) and the Naive Bayes models described in the previous section, we compute the probability of RPC sequences constituting a transaction.

The transaction recognition algorithm can be derived as follows. Let $R_{1n} = (r_1 r_2 \ldots r_n)$ be the input sequence of RPCs, and let $R_{ij}$ denote the subsequence $(r_i r_{i+1} \ldots r_j)$. Any sequence of integers $V = (i_1, ..., i_m)$, where $i_1 = 1$, $i_m \leq n$, and $i_j < i_{j+1}$, is called a *segmentation* of $R_{1n}$. The interpretation is that each $i_j$ marks the start of $j$-th transac-

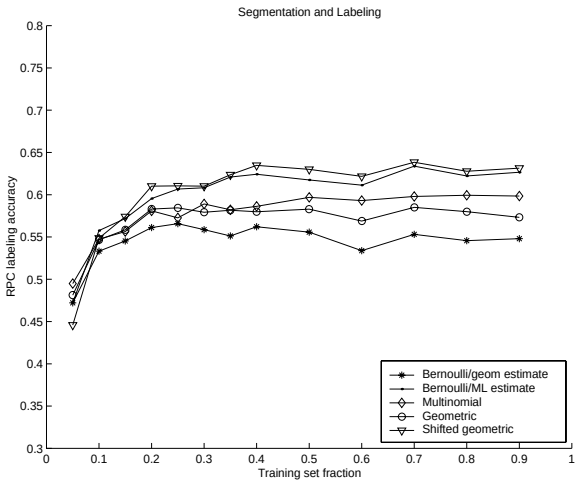


Figure 5: Transaction recognition results for data set 1.

tion instance within $R_{1n}$. Our objective is to find a most likely segmentation

$$V^* = \arg\max_V P(V|R_{1n}) = \arg\max_V P(V, R_{1n}),$$

where $V$ is a possible segmentation of the sequence $R_{1n}$. Let $\alpha_k = \max_V P(V, R_{1k})$, where $0 < k \leq n$ and $\alpha_0 = 1$. Using the assumption of transaction independence, we obtain the following dynamic-programming equation:

$$\alpha_k = \max_{0 \leq j < k} \alpha_j \cdot \max_T P(T, R_{j+1k}),$$

where $T$ is a transaction type. In order to find $P(T, R_{j+1,k})$, we compute the feature vector (using either occurrences or counts) for $R_{j+1,k}$ and apply one of the previously discussed Naive Bayes models. Doing so allows us to compute $V^*$ in $O(n^2)$ time.

Assessing the accuracy of our transaction recognition algorithm is complicated by the fact that the recognizer may not always find the correct transaction boundaries. As a result, our accuracy metric here is the fraction of correctly labeled RPCs. Fig. 5 displays the results of running our algorithm in combination with all four classifiers for data set 1. The x-axis is the fraction of the data set used for training, and the y-axis shows the percent of correctly labeled RPCs. As expected, accuracies are lower for transaction recognition than for labeling alone – 64% vs. 87% for the best case. Also, in contrast to the labeling problem where 4 of the 5 classifiers performed equivalently, here we see a clear ranking of the classifiers: counts-with-shifted geometric is the best, followed by occurrences-with-Bernoulli. It is interesting that the latter, which uses a more restrictive feature vector, performs better than counts-with-multinomial and geometric. Further, that the ordering of the classifiers in Fig. 5 differs from that in Fig. 4 suggests that the characteristics of a good classier for transaction recognition may differ from that for classification alone.

## Related Work and Discussion

To the best of our knowledge, the problem of recognizing end-user transactions has not yet been addressed in the literature. However, the problem of recognizing end-user transactions is closely related to several well-studied machine-learning domains such as text classification, speech recognition, and pattern recognition.

In text classification, a text is represented by a set of features such as word occurrences or word counts, and a classification algorithm, trained on a set of labeled examples, assigns topics (class labels) to previously unseen text instances. Examples include classification of Web pages (Craven *et al.* 1998; Nigam *et al.* 1998), sorting electronic mail (Sahami *et al.* 1998) or news articles (Lewis & Gale 1994; Joachims 1998). A common approach views a text as a "bag of words" (i.e. information about word sequence is ignored), using word occurrences or word counts as features. We employed this approach for assigning a transaction type to a given RPC sequence (labeling problem).

Various learning approaches, such as kNN, Naive Bayes, maximum entropy, neural nets, support-vector machines (SVMs), and many others have been compared on existing benchmarks (Yang 1999; Yang & Pedersen 1997; Joachims 1998; McCallum & Nigam 1998). Surprisingly, the Naive Bayes classifier performs very well in many domains in which its independence assumption is violated. This has been noted elsewhere (e.g., Domingoes-Pazzani97,McCallum98a). In our domain, RPCs within the same transaction are clearly not independent. However, Naive Bayes achieves quite high accuracy.

Despite similarities with text classification, our domain is inherently more complex because it requires solving the segmentation problem. Related areas include speech recognition (Jelinek 1998) (a stream of sounds must be segmented into words), statistical natural language processing (Manning & Schutze 1999) (segmenting a sequence of words into phrases), and general syntactic pattern recognition (Fu 1982). Commonly used models include Hidden Markov Models (HMMs) and stochastic context-free grammars (SCFGs). A SCFG consist of probabilistic production rules that can be used for constructing a stochastic language with an embedded rich syntactic structure. For example, they have been used as generative models for producing hierarchical workloads in order to evaluate the performance of distributed systems (Raghavan, Joseph, & Haring 1995).

Our approach to segmentation and labeling is closely related to the Viterbi search employed in speech recognition. Further investigation of segmentation techniques that would exploit the structure of our problem is an important direction for future work.

## Conclusions

This paper applies machine-learning techniques to a new domain — recognizing end-user transactions. Two problems are addressed. The first is labeling transaction instances with the transaction type. This is similar to problems studied in text classification. The second is transaction recognition, i.e. segmenting RPC sequences into transaction instances and

labeling these instances. This more difficult problem is akin to segmenting sounds into words in speech understanding.

We provide three kinds of results. The first characterizes the domain itself. We indicate that there are a substantial number of classes–more than 30 transaction types in the data we obtained from a large oil company. The vocabulary size–the RPC types–is modest, in the range of 50 to 100. There is a highly skewed distributions of both classes and vocabularies (i.e., RPCs). Further, we show that the distribution of RPC counts fits the shifted geometric distribution much better than either the multinomial or the geometric distributions.

Using Naive Bayes, we tackle the labeling problem by employing four combinations of feature vectors and probability distributions: RPC occurrences with the Bernoulli distribution, RPC counts with the multinomial distribution, RPC counts with the geometric distribution, and RPC counts with the shifted geometric distribution. Our experimental results indicate that the classifier using the shifted geometric distribution classifies correctly 78% of transactions. While this is competitive with the literature on text classification, it is considerably lower than the 85-87% achieved by the other classifiers. What is puzzling here is that the shifted geometric provides a much better description of the underlying distribution of RPCs.

Our dynamic-programming approach to transaction recognition uses the Viterbi algorithm, assuming transaction independence and using our classifiers to select most likely transaction label for a given RPC sequence. As an accuracy measure, we use the percent of correctly labeled RPCs rather than transactions, since the transaction recognizer may not always find the correct transaction boundaries. Here, our empirical results indicate that using occurrences with the Bernoulli distribution (assuming ML estimator) and counts with the shifted geometric distribution achieve higher accuracies than the other classifiers (approximately 64%). An interesting observation is that using a better classifier does not always result into a better transaction recognizer (e.g., shifted geometric is the worst classifier, but leads to the best recognizer).

Are our results to date sufficient for the intended applications of recognizing transactions? These applications include: (a) quantifying end-user perceptions of performance (b) creating representative workloads, and (c) anticipating resource management requests. Existing approaches to (a) and (b) attempt to obtain the requisite information through measurement. Item (c) is mostly addressed in an ad hoc manner. Thus, there is little appreciation of the implications of approximate results. This is clearly an area in need of further investigation.

Our results to date provide encouragement that it is possible to recognize end-user transactions using Naive Bayes or other machine-learning techniques. We view this as a starting point. One area of future work is characterizing domain properties that result in a high accuracy for Naive Bayes, especially when the independence assumption does not hold. We also plan to compare Naive Bayes with other state-of-the-art learning techniques, particularly with a general Bayes net classifier and SVMs.

Another direction for further research is the feature selection. By this, we mean both selecting a feature type (e.g., occurrences or counts for single RPCs, or functions on subsets/subsequences of RPCs) and selecting a subset of features of a given type. A commonly used approach to the second problem is selecting a fraction of features having the highest mutual information with the class variable (Yang & Pedersen 1997). This approach is quite effective in text classification where the feature set size is usually large (e.g., when a dictionary contains thousands of words). Since our domain has only about 100 RPCs, there may be little the advantages to employing of feature selection for counts or occurrences. However, if features employing sequence information are considered, then the feature space could grow exponentially and hence feature selection might be essential.

Still other areas of future work include more sophisticated transaction recognition algorithms and an investigation of the properties of a classifier that yield a better transaction recognizer.

## Appendix A: Shifted Geometric Distribution

This appendix provides details on the shifted geometric distribution and the estimation of its parameters in our data. This distribution extends the geometric distribution by having a shift parameter, $\nu_{ij}$, that specifies the minimum count for RPCs of type $j$ in a transaction instance of type $i$. Thus,

$$P(n_{ij}|T_i) = p_{ij}^{n_{ij}-\nu_{ij}}(1-p_{ij})$$

where $P(n_{ij}|T_i) = 0$ if $n_{ij} < \nu_{ij}$.

Let $m_i$ be the number of type $i$ transactions and $q_i = P(T_i)$. Let $n_{ilj}$ be the number of type $i$ RPCs in the $l$-th segment that is labeled as a transaction of type $i$. Then the likelihood function is:

$$\begin{aligned} P(\{T_{il}\},\{n_{ilj}\}) &= \prod_{i=1}^{I}\prod_{l=1}^{m_i}\prod_{j=1}^{J} p_{ij}^{n_{ilj}-\nu_{ij}}(1-p_{ij})q_i \\ &= \prod_i q_i^{m_i}\prod_{j_i} p_{ij}^{n_{ij}-m_i\nu_{ij}}(1-p_{ij})^{m_i} \end{aligned}$$

The maximum likelihood estimators can be found in a straight-forward way. These are: $\hat{p}_{ij} = \frac{n_{ij}-m_i\nu_{ij}}{n_{ij}-m_i\nu_{ij}+m_i}$, $\hat{q}_i = \frac{m_i}{\sum_{i'} m_{i'}}$, and $\hat{\nu}_{ij} = \min_l n_{ilj}$.

## References


Craven, M.; DiPasquo, D.; Freitag, D.; McCallum, A. K.; Mitchell, T. M.; Nigam, K.; and Slattery, S. 1998. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, 509–516.

Fu, K. S. 1982. *Syntactic Pattern Recognition and Applications*. Engelwood Cliffs, NJ: Prentice Hall.

Jelinek, F. 1998. *Statistical Methods for Speech Recognition*. MIT Press: Cambridge.

Joachims, T. 1998. Text categorization with support vector machines: learning with many relevant features. *European Conf. Mach. Learning, ECML98*.

Kleinrock, L. 1975. *Queueing Systems, Volume 1*. John Wiley and Sons.

Lewis, D. D., and Gale, W. A. 1994. A sequential algorithm for training text classifiers. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, 3–12.

Manning, C. D., and Schutze, H. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press: Cambridge.

McCallum, A. K., and Nigam, K. 1998. A comparison of event models for naive Bayes text classification. In *Proceedings of the 1st AAAI Workshop on Learning for Text Categorization*, 41–48.

Nigam, K.; McCallum, A. K.; Thrun, S.; and Mitchell, T. M. 1998. Learning to classify text from labeled and unlabeled documents. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, 792–799.

Raghavan, S. V.; Joseph, P. J.; and Haring, G. 1995. Workload models for multiwindow distributed environments. *Lecture Notes in Computer Science* 977:314–326.

Sahami, M.; Dumais, S. T.; Heckerman, D.; and Horvitz, E. 1998. A Bayesian approach to filtering junk E-mail. In *Proceedings of the 1998 Workshop on Learning for Text Categorization*, 55–62.

Yang, Y., and Pedersen, J. O. 1997. A comparative study on feature selection in text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, 412–420.

Yang, Y. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval* 1(1-2):69–90.