

# Higher Order Naïve Bayes: A Novel Non-IID Approach to Text Classification

Murat Can Ganiz, *Member, IEEE*, Cibin George, and William M. Pottenger, *Member, IEEE*

**Abstract**—The underlying assumption in traditional machine learning algorithms is that instances are Independent and Identically Distributed (IID). These critical independence assumptions made in traditional machine learning algorithms prevent them from going beyond instance boundaries to exploit latent relations between features. In this paper, we develop a general approach to supervised learning by leveraging higher order dependencies between features. We introduce a novel Bayesian framework for classification termed Higher Order Naïve Bayes (HONB). Unlike approaches that assume data instances are independent, HONB leverages higher order relations between features across different instances. The approach is validated in the classification domain on widely used benchmark data sets. Results obtained on several benchmark text corpora demonstrate that higher order approaches achieve significant improvements in classification accuracy over the baseline methods, especially when training data is scarce. A complexity analysis also reveals that the space and time complexity of HONB compare favorably with existing approaches.

**Index Terms**—Machine learning, statistical relational learning, naïve bayes, text classification, IID.

## 1 INTRODUCTION

A well-known problem in real-world applications of machine learning is that expert labeling of large amounts of data for training a classifier is prohibitively expensive. Often, in practice, only a small amount of labeled data is available for training. In this case, however, making an adequate estimation of the model parameters of a classifier is challenging. Underlying this issue is the traditional assumption in machine learning algorithms that instances are IID: independent and identically distributed [1]. This assumption simplifies the underlying mathematics of statistical models, but, in fact, does not hold for many real-world applications [2]. As a result of this assumption, models constructed under the IID assumption do not leverage connections between the attributes in different instances. A well-known example is market basket analysis, which forms sets of items that are purchased together in a given context such as a market basket. The advantage to this approach is that classification of a single instance of previously unseen data is possible because no additional context is needed to infer class membership [3]. However, such a context-free approach does not exploit valuable information about relationships between instances in the data set [4].

Recently, a new domain termed *link mining* has emerged [2]. The focus of the research in this new domain is on data

sets of instances that are explicitly linked, for example, by hyperlinks. Data sets with explicit links of this nature have been termed networked data [3]. One of the areas of algorithm research in this domain is link-based object classification, where the task is to classify instances connected to each other via a set of explicit links of this nature. Such approaches leverage both the attributes contained in the instances and the explicit links between instances, making the underlying assumption that labels of the linked objects tend to be correlated. Collective classification algorithms operate on networked data by jointly inferring class labels of a set of instances based on this underlying assumption. Several classification algorithms of this nature have been proposed that exploit the explicit links between instances (e.g., [5], [6], [7], and [8]). As noted, they utilize class labels of related instances when labeling a test instance and usually reduce classification error compared to flat (i.e., IID) classifiers. Models built in this way can be thought of as *second order* because explicit links between instances are leveraged during model construction. Such second-order connections are based on explicit links between instances such as hyperlinks between web pages, citation links between scientific papers, etc.

In this work, we focus on the development and analysis of a supervised machine learning algorithm that does not make the IID assumption, but instead moves beyond instance boundaries to exploit the latent information in higher order co-occurrence paths between features within data sets. Unlike existing approaches, however, we do not rely on explicit links between instances when learning a model, but rather leverage implicit links. In what follows, we define these implicit links.

We often refer to the terms “higher order path,” “higher order link” or “higher order co-occurrence” in this paper. In order to understand the meaning of these terms, consider Fig. 1 (reproduced from [9]). This figure depicts three documents, D1, D2, and D3, each containing two terms, or

• M.C. Ganiz is with the Department of Computer Engineering, Dogus University, Acibadem, Kadiköy, Istanbul 34722, Turkey. E-mail: mcganiz@dogus.edu.tr.

• C. George is with the Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854-8019. E-mail: cibin@cs.rutgers.edu.

• W.M. (Bill) Pottenger is with the Department of Computer Science and DIMACS, Rutgers University, 96 Frelinghuysen Road, Piscataway, NJ 08854. E-mail: drwmp@cs.rutgers.edu.

Manuscript received 7 Dec. 2008; revised 16 Nov. 2009; accepted 14 Jan. 2010; published online 2 Sept. 2010.

Recommended for acceptance by B.C. Ooi.

For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2008-12-0640. Digital Object Identifier no. 10.1109/TKDE.2010.160.

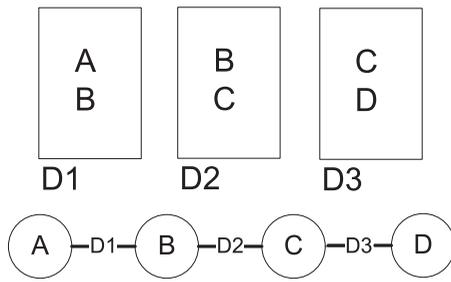


Fig. 1. Higher order co-occurrence.

entities represented by the letters A, B, C, and D. Above the three documents, there is a higher order path that links entity A with entity D through B and C. This is a third-order path since three links or “hops,” connect A and D.

A higher order path can also be represented as a chain of co-occurrences of entities (attribute values, words, terms, etc.) in different records (instances, documents, etc.). Actually, we can extract co-occurrence relations from virtually any data set as long as there is a meaningful context of entities.

In general, we are interested in exploiting the latent information in such higher order paths. We are motivated by the Latent Semantic Indexing (LSI) algorithm [10], which is a widely used technique in text mining and IR based on Singular Value Decomposition (SVD) matrix factoring. This approximation reduces the noise and sparsity of the original term vector space and partially solves the synonymy problem. The authors note that the fundamental problem in IR is that existing methods try to match words of queries with words of documents, but these individual words do not provide reliable evidence for the conceptual content the user is seeking because there are many ways to express a concept and words usually have several meanings. Similar to the IID assumption in machine learning, in traditional vector space models used in IR, documents are assumed to be independent since only documents that include query terms are retrieved and higher order relations between words are not considered. However, LSI takes advantage of implicit higher order (or latent) structure in the association of terms and documents. For example, [9] proved mathematically and demonstrated empirically that LSI is based on the use of higher order relations, in particular higher order co-occurrences. The authors also demonstrated that the retrieval performance of LSI is correlated with higher order relations. Higher-order relations in LSI capture “latent semantics” [9]. Our second motivation comes from the statistical relational learning domain. The work in this domain focuses on the limitations of the IID assumption in traditional machine learning and as noted reveals that such a context-free approach does not exploit the available information about relationships between instances in the data set [4].

Motivated by this prior work, we developed a novel classification algorithm termed Higher Order Naïve Bayes (HONB) that leverages higher order paths, which implicitly link instances in a data set. Unlike approaches that assume instances are IID, HONB leverages implicit co-occurrence relationships between attributes in different instances. Attributes (e.g., words in documents in text collections)

are richly connected by such higher order paths, and the generative model built by HONB exploits this rich connectivity pattern.

The power of HONB is most visible, when we have only a small amount of training data. This is important because, for example, labeling documents by category or class is an expensive process and in many real-world applications the amount of labeled data is far from adequate [11]. In order to demonstrate the utility of this approach, we vary the amount of the input (i.e., labeled training data) in our experiments. Our results on several textual data sets show that when training data is scarce (i.e., a small number of labeled instances), HONB significantly reduces the generalization error by leveraging higher order paths.

In what follows, we first discuss background and related work, then present our approach to learning based on higher order paths. We follow this with a section summarizing our results for widely used text classification data sets. Next, we discuss these results including aspects of future work, and finally draw conclusions.

## 2 BACKGROUND AND RELATED WORK

### 2.1 Higher Order Co-Occurrences

Higher order co-occurrences play a key role in the effectiveness of systems used for information retrieval and text mining. One example is Literature Based Discovery (LBD), which employs second-order co-occurrence to discover connections between concepts (entities). A well-known example is the discovery of a novel migraine-magnesium connection in the medical domain. In [24] Swanson found that in the Medline database some terms co-occur frequently with “migraine” in paper titles, e.g., “stress” and “calcium channel blockers.” They also discovered that “stress” co-occurs frequently with “magnesium” in other titles. As a result, they hypothesized a link between “migraine” and “magnesium,” and some clinical evidence has been obtained that supports this hypothesis. In LBD a second-order link of this nature is represented as  $A \rightarrow B \rightarrow C$ , where in this example A is “migraine,” C is “magnesium,” and B is one of several possible connecting terms such as “stress.”

Another example is LSI, a well-known approach to information retrieval. Kontostathis and Pottenger [9] mathematically prove that LSI implicitly depends on higher order co-occurrences. They also demonstrate empirically that higher order co-occurrences play a key role in the effectiveness of systems based on LSI. LSI can reveal hidden or latent relationships among terms, as terms semantically similar lie closer to each other in the LSI vector space. This can be demonstrated using the LSI term co-occurrence matrix as the following example shows.

Let’s consider a simple document collection given in Fig. 2, where document c1 has the words {human, interface} and c3 has {interface, user}. As can be seen from the co-occurrence matrix in Fig. 3, the terms “human” and “user” do not co-occur in this example collection. After applying LSI, however, the reduced representation co-occurrence matrix in Fig. 4 has a nonzero entry for “human” and “user” thus implying a similarity between the two terms. This is an example of second-order co-occurrence; in other words, there is a

Titles:	
c1:	<i>Human machine interface</i> for Lab ABC computer applications
c2:	<i>A survey of user opinion of computer system response time</i>
c3:	<i>The EPS user interface management system</i>
c4:	<i>System and human system engineering testing of EPS</i>
c5:	<i>Relation of user-perceived response time to error measurement</i>
m1:	The generation of random, binary, unordered <i>trees</i>
m2:	The intersection <i>graph</i> of paths in <i>trees</i>
m3:	<i>Graph minors</i> IV: Widths of <i>trees</i> and well-quasi-ordering
m4:	<i>Graph minors</i> : A <i>survey</i>

Fig. 2. Example document collection (Deerwester et al., 1990).

second-order path between “human” in c1 and “user” in c3 through “interface” (common to both c1 and c3). This second-order path implicitly links c1 to c3, violating the IID assumption. The results of experiments reported in [9] show that there is a strong correlation between second-order term co-occurrence, the values produced by the SVD algorithm used in LSI, and the performance of LSI measured in terms of F-measure, the harmonic mean of precision, and recall. As noted, the authors also provide a mathematical analysis, which proves that LSI does, in fact, depend on higher order term co-occurrence.

There are several research efforts that use LSI in text classification [25], [26], [27], [28], [29]. Although, LSI is essentially an unsupervised method, several authors have developed modifications to LSI such that it makes use of class labels during training [27], [28], [29]. Of these approaches [25] is particularly relevant to our work since the focus is also on small training sets. For example, for the 20 newsgroup data set the training set sizes range from one instance per class to 20 instances per class in this work. In addition, the algorithm employs a large number of unlabeled instances to aid in learning. The kNN classifier presented in [25] uses cosine similarity and combines the similarity scores of the 30 closest-neighbors using a noisy-OR operator.

A related issue in LSI is the choice of the truncation parameter  $k$ , which can be very important for supervised learning systems based on LSI. Previous work shows that values of  $k$  ranging from 100 to 300 give the best results [25]. However, this value is dependent on the size of the corpora. On the other hand, there are methods for choosing a “good” number of singular values for the dimensionality reduction

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
human (t1)	x	1	1	0	2	0	0	1	0	0	0	0
interface (t2)	1	x	1	1	1	0	0	1	0	0	0	0
computer (t3)	1	1	x	1	1	1	1	0	1	0	0	0
user (t4)	0	1	1	x	2	2	2	1	1	0	0	0
system (t5)	2	1	1	2	x	1	1	3	1	0	0	0
response (t6)	0	0	1	2	1	x	2	0	1	0	0	0
time (t7)	0	0	1	2	1	2	x	0	1	0	0	0
EPS (t8)	1	1	0	1	3	0	0	x	0	0	0	0
survey (t9)	0	0	1	1	1	1	1	0	x	0	1	1
trees (t10)	0	0	0	0	0	0	0	0	0	x	2	1
graph (t11)	0	0	0	0	0	0	0	0	1	2	x	2
minors (t12)	0	0	0	0	0	0	0	0	1	1	2	x

Fig. 3. Deerwester term-to-term matrix (adapted from Kontostathis and Pottenger [9]).

in LSI, some of which are discussed in [30]. Such methods can be used to build an automated LSI-based classifier, which adapts the  $k$  value as the training set size changes.

## 2.2 Statistical Relational Learning

The vast majority of statistical machine learning algorithms operate on “flat” data and traditionally assume that instances are independent and identically distributed (IID). As noted, however, this context-free approach does not exploit the available information about relationships between instances in the data set [4]. In statistical relational learning, models operate on relational data that includes explicit links between instances (e.g., hyperlinks between web pages or citation links between scientific papers). These relations provide rich information that can be leveraged to improve classification accuracy because attributes of linked instances are often correlated, and links are more likely to exist between instances that have some commonality [2]. Given a set of test instances, relational models simultaneously label all instances in order to exploit the correlations between class labels of related instances. This is also called collective classification (or collective inference), and violates the traditional IID assumption. Several studies (e.g., [5], [6], [7], and [8]) have shown, however, that by making inferences about multiple data instances simultaneously, classification error can be significantly reduced [12].

In related work, Macskassy and Provost [3] formulate linked data as networked data and classify machine learning methods for networked data into two categories; within-network inference and across-network inference. Networked

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
human (t1)	x	0.54	0.56	0.94	1.69	0.58	0.58	0.84	0.32	-0.32	-0.34	-0.25
interface (t2)	0.54	x	0.52	0.87	1.5	0.55	0.55	0.73	0.35	-0.2	-0.19	-0.14
computer (t3)	0.56	0.52	x	1.09	1.67	0.75	0.75	0.77	0.63	0.15	0.27	0.2
user (t4)	0.94	0.87	1.09	x	2.79	1.25	1.25	1.28	1.04	0.23	0.42	0.31
system (t5)	1.69	1.5	1.67	2.7	x	1.81	1.81	2.3	1.2	-0.47	-0.39	-0.28
response (t6)	0.58	0.55	0.75	1.25	1.81	x	0.89	0.8	0.82	0.38	0.56	0.41
time (t7)	0.58	0.55	0.75	1.25	1.81	0.89	x	0.8	0.82	0.38	0.56	0.41
EPS (t8)	0.84	0.73	0.77	1.28	2.3	0.8	0.8	x	0.46	-0.41	-0.43	-0.31
survey (t9)	0.32	0.35	0.63	1.04	1.2	0.82	0.82	0.46	x	0.88	1.17	0.85
trees (t10)	-0.32	-0.2	0.15	0.23	-0.47	0.38	0.38	-0.41	0.88	x	1.96	1.43
graph (t11)	-0.34	-0.19	0.27	0.42	-0.39	0.56	0.56	-0.43	1.17	1.96	x	1.81
minors (t12)	-0.25	-0.14	0.2	0.31	-0.28	0.41	0.41	-0.31	0.85	1.43	1.81	x

Fig. 4. Deerwester term-to-term matrix truncated to two dimensions (adapted from Kontostathis and Pottenger [9]).

data is relational data, where instances are interconnected such as web pages or research papers. In our case, however, the data set does not have to be innately relational. We extract relational information (i.e., links) from a standard machine learning data set by using higher order co-occurrences. These links are called higher order paths. Each such higher order path implies a relation between different records (instances).

### 2.3 Naïve Bayes (NB)

NB is a very popular algorithm due to its simplicity and efficiency, especially in the text-mining domain. The primary reason for its simplicity and efficiency is the attribute independence assumption. Although, NB assumes that attributes are independent, it nonetheless performs well in numerous domains, including on real-world data sets. A detailed analysis of NB can be found in [13]. There has been a focused effort, however, to relax NB attribute independence assumption while retaining its desirable simplicity and efficiency. Two techniques that exemplify this effort are Lazy Bayesian Rules (LBR) and Super-Parent TAN (SP-TAN). Both of these techniques have achieved remarkable accuracies, but at the cost of high-computational overhead. SP-TAN, for example, has high-computational complexity at training time and LBR has high-computational complexity at classification time. The complexity of such approaches detracts from their usefulness as alternatives to NB [14]. A detailed analysis of these approaches can be found in [14].

### 3 ENUMERATION OF HIGHER ORDER PATHS

We focus on discovering higher order association patterns in a labeled machine learning data set based on relations between items, or entities. Entities can be attribute-value pairs in a standard machine learning data set, words in a textual data set, etc. A higher order association is represented as a chain of co-occurrences of such entities in different instances. As noted, we also refer to such associations as higher order paths. Given a supervised learning data set (i.e., labeled training data), we attempt to discover patterns in sets of higher order associations that distinguish between the classes in the labeled data. In order to accomplish this, we first need to enumerate all higher order paths in a given class of instances. In this section, we present our definitions for higher order paths and data structures to represent and enumerate them. In the following section, we present a theoretical framework for the closed-form (analytical) enumeration of higher order paths.

Our definition of a higher order path is similar to that found in graph theory, which states that given a nonempty graph  $G = (V, E)$  of the form  $V = \{x_0, x_1, \dots, x_k\}$ ,  $E = \{x_0x_1, x_1x_2, \dots, x_{k-1}x_k\}$  with nodes  $x_i$  distinct, two vertices  $x_i$  and  $x_k$  are linked by a path  $P$ , where the number of edges in  $P$  is its length. Such a path is often referred to by the natural sequence of its vertices  $x_0x_1 \dots x_k$  [15]. Our definition of a higher order path differs from this in a couple of respects. First, vertices  $V = \{e_0, e_1, \dots, e_k\}$  represent entities, and edges  $E = \{r_0, r_1, \dots, r_m\}$  represent records, documents or instances. Finally and most importantly, in a higher order path both vertices and edges must be distinct. We are interested in enumerating all such paths.

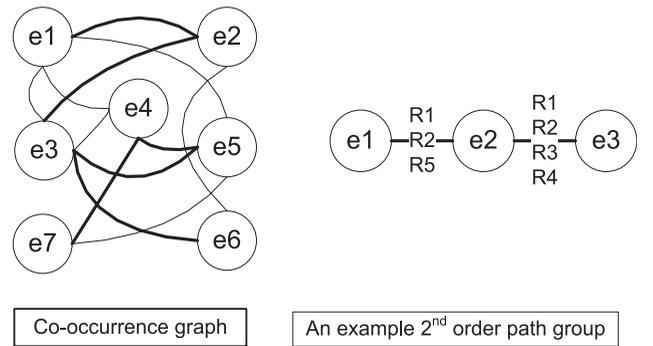


Fig. 5. Path Group structure based on a simple path in co-occurrence graph.

It is not straightforward, however, to represent higher order paths in conventional graph structures. In order to use conventional graph structures and algorithms, we divided the above representation into two structures. First, we form a co-occurrence graph  $G_c = (V, E)$ , in which the vertices are the entities and there is an edge between two entities, if they co-occur in one or more records. A path (length  $\geq 2$ ) extracted from  $G_c$  satisfies the first requirement of our higher order path definition since the vertices in this path are distinct. The second requirement entails that records on a path must be distinct, and another data structure that contains lists of records for each edge is needed. We term this structure a *path group*. Note that the second requirement results in more intuitive paths. For example, with LSI in view the latent semantics of  $e_1 \sim R_1 \sim e_2 \sim R_2 \sim e_3$  is more clear than  $e_1 \sim R_1 \sim e_2 \sim R_1 \sim e_3$ . We are interested in paths of this nature that implicitly link entities through different records because they have the potential to reveal latent information. As noted previously, such implicit links between records violate the IID assumption typically made in traditional machine learning algorithms.

However, we are not necessarily looking for the shortest paths between two entities. In the second-order path between  $e_1$  and  $e_3$  in Fig. 5 below, there is a shorter (first order) path between  $e_1$  and  $e_3$  as well. Although, we use particular order paths (e.g., only second order or only third order) in our algorithms, given our definition of a higher order path, we enumerate many more higher order paths than just the shortest paths. These higher order paths include latent connection information for number of different records.

Using the path group representation it is possible to satisfy the second requirement of our higher order path definition. In effect, we need to identify the systems of distinct representatives (SDRs) from the sets of records in a path group. An SDR of the sets  $S_0, \dots, S_{n-1}$  is defined as a sequence of  $n$  distinct elements  $r_0, \dots, r_{n-1}$  with  $r_i \in S_i, 0 \leq i \leq n-1$  [16]. Each distinct representative corresponds to a record in a higher order path. In order to enumerate all the distinct representatives in a given higher order path, a bipartite graph  $G_b = (V_1 \cup V_2, E)$  is formed such that  $V_1$  represents the sets of records ( $S_0, S_1, \dots$ ) in a given path group and  $V_2$  represents the records themselves. A maximum matching with cardinality  $|V_1|$  in this bipartite graph yields the SDR for a single higher order path. All possible maximum matchings in the bipartite graph

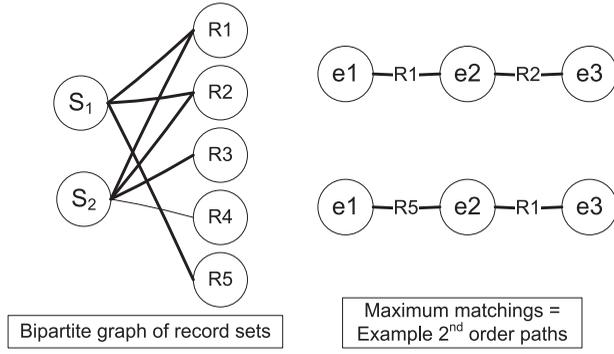


Fig. 6. Extracting/enumerating higher order paths from path group structure.

together comprise all the SDRs for all higher order paths in the path group. This process is summarized in Figs. 5 and 6. In Fig. 5, we can see an example of the second-order path group  $(e_1-\{1, 2, 5\}-e_2-\{1, 2, 3, 4\}-e_3)$ , that is, extracted from the co-occurrence graph  $G_c$ . This particular second-order path group includes two sets of records:  $S_0 = \{1, 2, 5\}$  and  $S_1 = \{1, 2, 3, 4\}$ .  $S_0$  corresponds to the records in which  $e_1$  and  $e_2$  co-occur, and  $S_1$  is the set of records in which  $e_2$  and  $e_3$  co-occur. As noted, the path group may be composed of several higher order paths. In Fig. 6, a bipartite graph  $G_b = (V_1 \cup V_2, E)$  is formed where  $V_1$  represents the two sets of records and  $V_2$  represents all records in these sets. Enumerating all maximum matchings in this graph yields all higher order paths in the path group. The second diagram (depicted in Fig. 6) shows two examples of the many paths in this path group. In the first such path, edge labels  $r_1$  and  $r_3$  are records in  $S_0$  and  $S_1$ , and the path corresponds to maximum matching in the bipartite graph.

Although, this framework gives us the ability to algorithmically enumerate higher order paths of any length, in fact, we have developed closed forms for enumerating the systems of distinct representatives of record sets in second, third, and fourth order path groups. Our approach is motivated by the inclusion-exclusion principle in set theory [16].

**Definition 1.** An SDR of the sets  $S_0, \dots, S_{n-1}$  is defined as a sequence of  $n$  distinct elements  $r_0, \dots, r_{n-1}$  with  $r_i \in S_i, 0 \leq i \leq n-1$  [16].

As noted, we wish to discover all of the systems of distinct representatives of  $n$  sets. This requires that we enumerate all possible combinations of  $n$  distinct records, each coming from one of these  $n$  sets. Motivated by the inclusion-exclusion principle, we have proven three theorems stating formulas for calculating the number of systems of distinct representatives of two, three, and four sets, respectively. Note that in what follows, we use the notation “SDRs” to refer to systems (plural) of distinct representatives (i.e., one or more higher order paths).

**Theorem 1.** The number of SDRs for two sets  $A_1$  and  $A_2$  is

$$\#SDR(A_1, A_2) = \sum_{i|j \in \lambda(1,1)} S_i S_j - \sum_{ij \in \lambda(2)} S_{ij}$$

where  $S_I = |\cap_{i \in I} A_i|$ .

**Proof of Theorem 1.** Each term can be represented by a partition of two, which is shown as  $\lambda(1, 1)$  and  $\lambda(2)$ . The first term represents and counts all sequences of representatives from  $A_1$  and  $A_2$  by multiplying the number of elements in each. In the simplest case, consider that we have two disjoint sets  $A_1$  and  $A_2$ , and  $A_1 \cap A_2 = \emptyset$ . Since there are no intersections, an element from  $A_1$  can be combined with any element in  $A_2$  to form a system of representatives and all the representatives in all the systems will be distinct. However, if  $A_1 \cap A_2 = \emptyset$  then the elements in the intersection can represent both  $A_1$  and  $A_2$  because they belong to both sets. In this case, it is possible to have a sequence with repeated or nondistinct representatives. This obviously violates the definition of a SDR and needs to be excluded from the total count. This kind of exception will occur for all elements in the intersection. Therefore, we need to subtract the number of elements in the intersection, which corresponds to the second term in the closed-form formula.  $\square$

**Theorem 2.** The number of SDRs for three sets  $A_1, A_2$ , and  $A_3$  is

$$\#SDR(A_1, A_2, A_3) = \sum_{i|j|k \in \lambda(1,1,1)} S_i S_j S_k - \sum_{ij|k \in \lambda(2,1)} S_{ij} S_k + 2 \sum_{ijk \in \lambda(3)} S_{ijk}.$$

**Proof of Theorem 2.** For three sets the proof is more involved. A SDR of these sets consists of three distinct representatives. This time we have three sets and we refer to the terms using a partition of three  $\lambda(1, 1, 1)$ ,  $\lambda(2, 1)$ , and  $\lambda(3)$ . We again start by taking the product of the number of elements in the sets. There are a total of  $\sum_{i|j|k \in \lambda(1,1,1)} S_i S_j S_k = |A_1| |A_2| |A_3|$  of these sequences. Then, in the second term, we subtract the cases where two or more representatives are nondistinct or identical. In this step, we are considering the two set intersections and since we have three sets, there are  $C(3, 2) = 3$  combinations of them. For the sequences, where all three elements are identical, we count them once in the first term and subtract them three times in the second term. As a result, we need to add twice the number of these sequences back in to the sum to compensate for over counting.  $\square$

**Theorem 3.** The number of SDRs for four sets  $A_1, A_2, A_3$ , and  $A_4$  is

$$\begin{aligned} \#SDR(A_1, A_2, A_3, A_4) &= \sum_{i|j|k|l \in \lambda(1,1,1,1)} S_i S_j S_k S_l - \sum_{ij|k|l \in \lambda(2,1,1)} S_{ij} S_k S_l \\ &+ 2 \sum_{ijk|l \in \lambda(3,1)} S_{ijk} S_l + \sum_{ij|kl \in \lambda(2,2)} S_{ij} S_{kl} - 6 \sum_{ijkl \in \lambda(4)} S_{ijkl}. \end{aligned}$$

**Proof of Theorem 3.** We refer to all possible sequence patterns by a partition of size four  $\lambda(1, 1, 1, 1)$ ,  $\lambda(2, 1, 1)$ ,  $\lambda(3, 1)$ ,  $\lambda(2, 2)$ ,  $\lambda(4)$  with respect to the formula above. Since  $\lambda(1, 1, 1, 1)$  represents all the sequences obviously its coefficient will be one. In the second term, we subtract  $\lambda(2, 1, 1)$ ; two set intersections, which produce sequences with two or more nondistinct representatives. As with the previous theorem, the second term includes the

sequences of  $\lambda(3, 1)$ , which are three set intersections that produce three or more nondistinct representatives. Thus, we add twice  $\lambda(3, 1)$  back into the sum in the third term. Similarly, each summand of  $\lambda(2, 2)$  occurs in two different summands of  $\lambda(2, 1, 1)$ , so we add  $\lambda(2, 2)$  once to compensate in the fourth term. Finally,  $\lambda(4)$  is a four set intersection, which produces sequences with four nondistinct representatives, which occur once in every summand of the previous terms. This means the formula is too large by  $1 - C(4, 2) + 2C(4, 3) + 3 = 6$ , so we need to subtract  $\lambda(4)$  six times in the fifth term.  $\square$

These closed-form formulas (Theorems 1, 2, and 3) are used in enumerating the second, third, and fourth-order paths in the path group structures exemplified in Fig. 6 above. When it is necessary to enumerate paths of order greater than four, we use the algorithmic approach outlined previously to enumerate all maximum matchings in the bipartite graph.

## 4 HONB

Our approach is based on a NB algorithm, that is applied to the problem of text classification. NB is commonly used in text classification because it is fast and easy to implement. NB is the simplest of Bayesian classifiers in that it assumes that all attributes of the examples are independent of each other given the context of the class [17]. Although, this assumption does not hold for most real-world data sets, overall NB performs fairly well. In our case, the relative simplicity of the NB classifier allows for detailed analysis of the effect of using higher order paths.

It is important to note that the focus in this work is on the independence assumption between instances. Per the definition in the previous section, higher order paths link attributes from different instances thereby implicitly linking different instances. In this way, we violate the IID assumption on instances. For this reason, our approach is similar to the work in link mining that leverages explicit links between instances during training. We *do not*, however, modify the NB algorithm to relax its assumption of independence between attributes. (As noted in the background and related work section, this latter approach is taken in work such as [14].) In contrast, in our framework although priors for NB are estimated from higher order paths, we do not alter the NB algorithm itself. That is why, we name our approach HONB.

NB assumes that a document is generated by a parametric model. We use training data to calculate estimates of model parameters. By using these estimates with Bayes rule, we can calculate the probability that a class generated a given test document. We classify the document in the most probable class in the usual way.

There are two different generative models commonly used for NB classification, the multivariate bernoulli model and the multinomial model. A multinomial model is a unigram language model with integer term counts [17]. In this research, we implemented a multivariate bernoulli model (also known as a binary independence NB model). In this model, an event is a document and is represented by a vector of binary attributes  $X_1, \dots, X_d$ , which take values

$\{0, 1\}$  indicating occurrence of terms in the document. The number of times a term occurs in a document is not captured.  $X_0$  represents the class label of the document and takes values in  $\{1, \dots, C\}$ , where  $C$  is the number of classes.

One can calculate the probability of an event (i.e., a document) given the probability of a document  $d$  belonging to class  $C$  using Bayes rule as follows:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)}. \quad (1)$$

In a multiclass classification scenario, one assigns document  $d$  to the class with the highest  $P(c|d)$ . In addition, since  $P(d)$  is independent of the class it is not needed during classification and we have

$$P(c|d) \propto P(d|c)P(c). \quad (2)$$

A document consists of words  $d \equiv \{w_1, w_2, \dots, w_n\}$ . Here, we can understand the document to be the "event," and the absence or presence of words to be attributes of the event [17]. The NB algorithm assumes that each word in a document is independent of others and independent of its position in the document. As a result, when calculating the probability of a document, the product of the probabilities of all the attribute values, including the probability of nonoccurrence for terms that do not occur in the document, is taken

$$P(d|c) = \prod_{w \in d} P(w|c) \prod_{w \notin d, w \in W} (1 - P(w|c)). \quad (3)$$

However, during classification performance is improved by not calculating the second product for each test document [18], so the equation is rewritten as

$$P(d|c) = \prod_{w \in d} \frac{P(w|c)}{1 - P(w|c)} \prod_{w \in W} (1 - P(w|c)). \quad (4)$$

Here, the second product is precomputed and stored for each class.

To estimate a class probability of a document  $p(c|d)$ , the priors must be estimated from labeled training documents. Probabilities of all words in class  $c$  are estimated by

$$\hat{P}(w|c) = \frac{n(c, w)}{n(c, d)}, \quad (5)$$

where  $n(c, w)$  is the number of documents in class  $c$  including word  $w$  as in (6) and  $n(c, d)$  is the total number of documents in class  $c$

$$n(c, w) = \sum_{d \in D_c} n(d, w). \quad (6)$$

Another parameter of the system is the class prior probability, which is estimated using Maximum Likelihood Estimation (MLE) as follows:

$$\hat{P}(c) = \frac{n(c, d)}{\sum_{c \in C} n(c, d)}. \quad (7)$$

In HONB, we form a co-occurrence graph for each class of labeled training documents and enumerate specific higher order (e.g., second order) paths. Our system parameters are

the same as in the traditional multivariate NB model; however, we estimate the parameters using the higher order paths instead of documents. Similarly, probabilities of all words in class  $c$  are estimated by

$$\hat{P}_{\sim}(w|c) = \frac{\tilde{n}(c, w)}{\tilde{n}(c, h)}, \quad (8)$$

where  $\tilde{n}(c, w)$  is the number of higher order paths in class  $c$  including word  $w$  as in (9), and  $\tilde{n}(c, h)$  is the total number of higher order paths in class  $c$

$$\tilde{n}(c, w) = \sum_{h \in H_c} \tilde{n}(h, w). \quad (9)$$

The class prior is estimated by dividing the number of higher order paths in class  $c$  by the sum of the higher order paths in all classes as shown in (10)

$$\hat{P}_{\sim}(c) = \frac{\tilde{n}(c, h)}{\sum_{c \in C} \tilde{n}(c, h)}. \quad (10)$$

Finally, by using the parameter estimates described above, we can estimate the class probability of a test document using

$$\hat{P}(c|d) \propto \hat{P}_{\sim}(c) \prod_{w \in d} \frac{\hat{P}_{\sim}(w|c)}{1 - \hat{P}_{\sim}(w|c)} \prod_{w \in W} (1 - \hat{P}_{\sim}(w|c)). \quad (11)$$

By using the same parameters—e.g., words—HONB has the advantage that (just like NB) a single test instance can be classified without the need for additional context or a mapping function. However, by using higher order paths instead of documents, we also exploit the latent connections between documents within a given class. For both NB and HONB, we employ Laplace smoothing to avoid zero probabilities.

As noted above, although, we do not assume that documents are IID during model construction, we still make the “naïve” assumption that attributes are independent from each other during classification because we are using NB.

For all the experiments reported in this paper, we used second-order paths. This choice was based on the empirical observation that although models built from third order paths show the same pattern of performance, the performance for second-order paths is slightly better. As noted previously a second-order path connects three different words (or entities) in two different documents (or records). For example,  $w_1 \sim D_x \sim w_2 \sim D_y \sim w_3$ .

## 4.1 Complexity Analysis

During training, NB assembles a simple table of class probability estimates and a table of conditional attribute value probability estimates for each class. Therefore, the space complexity is  $O(cnv)$ , where  $c$  is the number of classes,  $n$  is the number of attributes, and  $v$  is the average number of values per attribute [14]. In the case of text classification, the number of attribute values corresponds to the word dictionary size in both the multinomial and multivariate NB models. As a result the space complexity can be defined as  $O(cd)$ , where  $d$  is the dictionary size. Attribute value conditional probability estimates can be calculated by a simple scan through the data; thus the time complexity of

the learning phase is  $O(td)$ , where  $t$  is the number of training examples and  $d$  is the dictionary size. During classification, classifying a single instance has time complexity  $O(cd)$  using the tables formed during training.

During training with HONB, we first need to enumerate second-order paths before calculating probability estimates. As mentioned before, for each class in  $k$  classes, we first form a co-occurrence graph, and then, enumerate all length two paths from this graph. All length two paths in a graph can be enumerated in time  $O(n^3)$  (based on Stirling’s approximation of  $O(n^2)$  for the number of paths of length two [19]<sup>1</sup>), where  $n$  is the number of nodes (words) in the co-occurrence graph of the words in the documents in a given class, and is bounded by  $d$ . After this step, we form path groups based on each of these paths. We enumerate higher order paths from the path groups using a closed-form formula, which in the worst case has time complexity  $O(u \log(r))$ , where  $u$  and  $r$  are the set sizes, both of which are bounded by  $t$ , the number of training examples (see Figs. 5 and 6). In our work with small samples of training data in text classification,  $t \ll d$ . We saw previously that there are  $O(n^2)$  path groups, yielding a worst-case time complexity of  $O(t \log(t)n^2)$  for path enumeration. In addition, we saw that the time complexity of training for NB is  $O(td)$ , where  $d$  is equivalent to  $n$  in the worst case (when the documents in a class have the entire dictionary in them). As a result, during training HONB has an overall worst-case time complexity of  $O(cn^3)$ , where  $c$  is the number of classes and as noted  $n$  is bounded by  $d$ . This compares favorably with the time complexity for training in SP-TAN as well as the time complexity for classification in LBR, both of which require  $O(tkn^3)$ , where  $t$  is the number of training examples,  $c$  as before is the number of classes and  $n$  is the number of attributes (equivalent to the number of words in the dictionary of size  $d$  in a text classification application) [14].

In terms of space complexity, the only difference between HONB and NB is that in HONB, we need to store the co-occurrence graph (e.g., as an adjacency list). In the worst case for a complete graph, we have  $O(d^2/2)$  additional space complexity since this is a undirected graph. We don’t need to store higher order paths themselves since the attribute value statistics can be updated while enumerating paths. Finally, the time complexity of HONB is exactly the same as NB in the classification phase, which is  $O(cd)$ .

## 5 EXPERIMENTS

### 5.1 Comparing HONB with Traditional Classifiers

In this section, we present the results and analysis of our experiments comparing HONB with three traditional classifiers, NB, Support Vector Machines (SVM) and an LSI-based kNN classifiers (LSI kNN). NB is our baseline classifier. As stated by Chakrabarti [18] “SVMs are some of

1. An intuition for the  $O(n^{P+1})$  bound for paths of length  $P$  can be seen by comparison to the generation of a tree for path traversal. The number of children formed is the total number of paths generated. Such a tree has a fan out of  $(n - 1)$  at the first step (in the worst case of a clique, meaning a node can generate a path to every other node excluding itself),  $n - 2$  at the second step (excluding itself and the first vertex), and so on down to one for paths of all lengths up to  $P$ . Therefore, the total number of children is  $n - 1 * n - 2 * \dots * n - P$ , yielding  $O(n^P)$  complexity for each of  $n$  vertices.

TABLE 1  
Data Sets Used in Experiments

Dataset	Class Names
COMP (5)	comp.graphics, comp.window.x, comp.sys.mac.hardware, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware
SCIENCE (4)	sci.crypt, sci.electronics, sci.med, sci.space
POLITICS (3)	talk.politics.mideast, talk.politics.guns, talk.politics.misc
RELIGION (3)	alt.atheism, talk.religion.misc, soc.religion.christian
Citeseer (6)	AI, Agents, DB, HCI, IR, ML
Cora (6)	Case Based, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Theory
WebKB (5)	Student, Faculty, Course, Project, Staff

the most accurate classifiers for text; no other kind of classifier has been known to outperform it across the board over a large number of document collections." For example, SVMs usually perform better than NB. Therefore, we included SVM in our experiments as the state-of-the-art classifier in general in text classification domain. In addition to SVM, we also included an LSI-based k-Nearest Neighborhood (kNN) classifier since in our prior work we proved that LSI implicitly uses higher order co-occurrence paths [9]. Both SVM and LSI-based kNN classifiers were optimized to achieve the best performance on our data sets. Experiments were done using second-order paths for HONB. We have conducted additional experiments using third-order paths, but the results did not differ significantly. Therefore, given the lower execution time of second-order path enumeration, we chose to use second-order paths. For the experiments in this section, we used four subsets of the 20 newsgroups data set that are also used in related work using word clusters for text classification (e.g., [11]). This related work is also focused on improving the efficiency of the classifiers when the training data is scarce, and these data sets provide an appropriate platform to demonstrate the usefulness of our algorithm. The description of these data sets is given in Table 1. The 20 newsgroups data set is a collection of approximately 20,000 newsgroup documents, partitioned evenly across 20 different newsgroups. It is a commonly used benchmark data set for text classification. We used the 18,828 version of the data set, which has cross postings (duplicates) removed and all headers filtered except for "From" and "Subject" headers.<sup>2</sup> The data set was preprocessed using the Text Mining Infrastructure (TMI) [20]. We used the stemmer and stop word list embedded in the TMI, and filtered words that occurred in less than three documents. Furthermore, for comparison purposes, we employed Information Gain (IG) to rank the words in the dictionary and select the top-ranked 2,000 words for each subset. To observe the scalability of HONB, we also conducted experiments on several additional data sets Cora [21], Citeseer [23], and WebKB [22]. For Cora, Citeseer, and WebKB, we downloaded<sup>3</sup> and used the same processed versions of these data sets that were used in the experiments reported in [23]. The Cora data set is

composed of a number of scholarly research papers on machine learning, in all comprising seven classes: Case-Based, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning and Theory. In [23] Sen and Getoor chose documents such that each document is either cited or cites one of the other documents in the corpus. Stop words were removed and words with document frequency less than 10 were also removed. The final corpus contains 2,708 documents with a vocabulary of 1,433 distinct words and 5,429 links. Similarly, the Citeseer data set is composed of a number of scholarly research papers from computer science domain and contains 3,312 documents with a vocabulary of 3,703 terms. The WebKB data set is a hypertext data set consisting of the web pages of the computer science departments of four different universities. There are five classes: Course, Student, Faculty, Project, and Staff. Sen and Getoor [23] selected only those documents, which either link to or are linked by at least one other document in the data set and extracted a corpus of 877 documents. After stemming and stop word removal the dictionary size is 1,703 distinct words, and there are 1,608 links in this corpus.

As mentioned, we compare our results to those from three different traditional classifiers, NB, LSI kNN, and SVM. All our data sets have binary word features, which indicate only the occurrence or nonoccurrence of a word in a document. We employed a multivariate binary event model in a NB framework as described above. The SVM classifier, we used is implemented in the R statistical tool kit [31] in package e1071, which is the R interface to the libsvm library [32]. For the comparison to SVM, we optimized parameters to obtain the best possible result including a range of experiments with different kernels (e.g., polynomial, radial basis function). In addition, when selecting the value of the soft margin cost parameter C for SVM, we considered the set  $\{10^{-4}, 10^{-3}, \dots, 10^4\}$  of possible values. On every trial, we picked the smallest value of C, which resulted in the highest accuracy obtained on the training set. During optimization of the parameters for the SVM runs, we observed that the C-values were mostly  $10^{-4}$  and  $10^{-3}$ , and the RBF kernel performed the best about 60 percent of the time.

For the LSI kNN classifier, based on the related work in [25], we used the 25 nearest neighbors combined using the noisy-OR operator and cosine similarity. The LSI kNN

2. <http://people.csail.mit.edu/jrennie/20Newsgroups/>.

3. <http://www.cs.umd.edu/~sen/lbc-proj/LBC.html>.

TABLE 2  
Comparison of HONB with Traditional Classifiers on Five Percent Training Data Samples

Dataset	Accuracy %				% improvement of HONB		
	NB	SVM	LSI kNN	HONB	over NB	over SVM	over LSI
COMP (5)	51.43	58.53	61.09	<b>65.06</b>	26.51	11.16	6.50
SCIENCE (4)	59.56	75.09	74.34	<b>84.32</b>	41.58	12.29	13.43
POLITICS (3)	67.86	77.36	82.15	<b>83.34</b>	22.82	7.73	<i>1.45</i>
RELIGION (3)	64.13	69.88	69.71	<b>74.18</b>	15.66	6.15	6.40
Citeseer (6)	31.30	<b>60.94</b>	60.61	51.52	64.62	-15.45	-15.0
Cora (6)	30.19	<b>55.31</b>	44.71	50.33	66.68	-9.00	12.6
WebKB (5)	54.82	<b>67.00</b>	48.58	64.06	16.86	<i>-4.39</i>	31.9

classifier we employed is implemented in the LSA package, also in the R statistical tool kit. The truncation parameter  $k$  of LSI can be set using various recommender routines in this package. Similar to SVM, we optimized the LSI kNN classifier by choosing the best performing recommender routine in this package, which was based on Kaiser-Criterion that retains the singular values larger than 1.0.

In order to simulate a real-world scenario, where only a few labeled data instances are available, in this set of experiments, we selected five percent of the data for training from each data set. In the 20 Newsgroups subsets, five percent corresponds to 25 documents per class. The remaining 475 documents per class were used as our test set. The other data sets, however, do not have an equal number of documents in each class so the number of instances selected varied depending on the distribution of instances. In all experiments, we conducted eight random trials and averaged the results over these trials. The detailed results are depicted in Table 2. The performance is presented in terms of accuracy, and the number of classes per data set is given in parentheses in the first column. Additionally, the largest absolute performance is highlighted in bold. Percent improvements are statistically significant at the five percent level unless marked in red italics, in which case the performance difference is not statistically significant.

For all data sets, HONB significantly outperformed the baseline NB classifier. The performance of HONB is most visible on the 20 Newsgroups data sets. On these data sets HONB outperforms all the other classifiers including state-of-the-art SVM and LSI kNN. The improvement of HONB over SVM is statistically significant for these 20 newsgroups data sets. Similarly, HONB performs statistically significantly better than LSI kNN on three of the four data sets. Nonetheless, for the Citeseer, Cora, and WebKB data sets, SVM is the best performing classifier. However, the difference in performance between SVM and HONB on the WebKB data set is not statistically significant. Additionally, HONB statistically significantly outperforms LSI kNN on both the Cora and WebKB data sets.

These results suggest that under the conditions where training data is scarce, HONB is a significantly better classifier than NB. In addition, for five out of seven of these

data sets HONB is comparable or better than SVM, which is as noted one of the best performing algorithms for text classification. Finally, for six of the seven data sets HONB is comparable or better than LSI kNN.

We contend that the main reason behind the superior performance of HONB is that it makes explicit use of the most valuable latent information that exists in higher order paths. Just as prior research revealed for LSI [9], higher order paths provide additional latent information. Unlike LSI, however, HONB explicitly leverages this higher order information. This, in fact, was one of the motivating purposes for our research—from our prior work, we knew LSI leveraged higher order information, but wanted to get an answer to the question “How is this information best leveraged?” These results represent a significant step forward in answering this question.

In summary, in the case of text classification documents and words in a document set are richly connected by such higher order paths. Higher order paths as noted go beyond instance boundaries. In contrast, traditional classifiers only leverage relationships between attributes within instances (e.g., words within documents), or as with the case of LSI kNN is a black box in terms of how they leverage latent information. When training data is scarce, these approaches do not exploit sufficient information from individual instances, and thus, do not obtain reliable parameter estimates. In contrast, HONB explicitly exploits the rich connectivity between words in different documents that belong to a given class. Consequently, HONB is able to obtain better parameter estimates and performs better.

## 5.2 Evaluating HONB by Varying the Sparsity of Input

In the previous section, we provided an analysis of HONB performance compared to the traditional classifiers NB, SVM, and LSI kNN. We observed that for a five percent subset of the training data, HONB outperforms the baseline NB classifier for all the data sets and outperforms the other traditional classifiers in most cases. In this section, we drill down more deeply to explore the range of performance of HONB as a function of the scarcity of input training data. As before, we compare HONB’s performance to traditional

TABLE 3  
Summary of Data Sets Used in Experiments Varying Scarcity of Training Data

Dataset	Documents	Dictionary	Classes
COMPUTER	500	2000	5
SCIENCE	500	2000	4
POLITICS	500	2000	3
RELIGION	500	2000	3
Citeseer	3312	3703	6
Cora	2708	1433	7
WebKB	877	1703	5

classifiers including NB, SVM, and LSI kNN. The summary of data sets is repeated for convenience in Table 3.

Although, there are explicit links in some of these data sets, as noted previously HONB does not use them. We are only interested in modeling the textual data so that we can assess the impact of leveraging implicit higher order information. For the following experiments, we use the standard evaluation metric of accuracy. As mentioned, in the analysis in this section, we vary the scarcity of the input (i.e., labeled training data) in the experiments in order to demonstrate the utility of our approach. Scarcity is measured in terms of the size of the input training set, where the number of documents per class ranges from five percent to 90 percent.

Figs. 7a, 7b, 7c, 7d confirm that HONB consistently outperforms the baseline method, traditional NB, in terms of classification on the COMPUTER, SCIENCE, POLITICS, and RELIGION data sets. All these accuracy improvements of HONB over NB are statistically significant at the five percent level for all the percentages shown from five percent to 90 percent. HONB performs exceptionally well on 20 newsgroups subsets, and in addition to outperforming the baseline classifier it also outperforms the state-of-the-art SVM classifier, as well as LSI kNN. Improvement over SVM is statistically significant at the five percent level for all the percentages above from five percent to 90 percent. Improvement over LSI kNN is also statistically significant at the five percent level except for the 80 percent and 90 percent samples on the COMPUTER data set, the five percent, 10 percent and 60 percent to 90 percent samples for the POLITICS data set, and the 90 percent sample for the RELIGION data set.

A slightly different pattern of performance can be seen on Citeseer (see Fig. 7e) and Cora (see Fig. 7f) data sets. While classification accuracy of each method monotonically increased with increasing training set size, HONB drastically outperformed NB when training sets were small, reaching close to SVM and LSI kNN. Interestingly, LSI kNN performs comparably with SVM for all the training set sizes, and after about 50 percent NB also achieves similar performance. For our last data set, WebKB, we see a similar performance pattern as that of the 20 newsgroups subsets in that HONB outperforms the baseline NB classifier for all training set percentages. However, here again improvement is statistically significant at the five percent level only up to the 50 percent sample size. In addition, although LSI kNN performs comparably with SVM on all the other data sets, SVM performs exceptionally well on this data set and

outperforms LSI kNN by a large margin. Nonetheless, the performance of HONB is close to SVM in the five percent to 10 percent sample size range, and the difference between the two is not significant at five percent. WebKB is a much smaller data set than the others (see Table 3) resulting in many fewer training documents per class for each subset of the training data selected. Additionally, we should note that the class distribution of WebKB is quite different from that of the other data sets. One of the five classes (the student class) has many more documents than the others. Despite these differences HONB still achieves better performance compared to the baseline model. This highlights the robustness of HONB over different data sets with different class distributions.

These results are especially encouraging in that they suggest that higher order methods were able to construct robust models even when training data was particularly scarce. We speculate that changes in the distribution of highly discriminative terms across classes as a result of increasing training set size allowed NB to outperform HONB on the Citeseer and Cora data sets once the amount of training data reached a certain point. We are currently investigating this hypothesis and intend to report our findings as part of future work.

We conclude that the advantage of HONB, especially at the five percent to 10 percent sample size range, for all data sets is due to the lack of sufficient information in these small training sets to estimate the parameters for a traditional model. Simply put, there are too few documents per class. Even with a very small number of documents per class, however, HONB leverages a large number of second-order paths. These paths enable HONB to generalize even under conditions of extremely scarce input.

In order to verify the value of latent higher order information, in an additional experiment, we used what we termed *pure second-order paths*. In this case, we enumerated only length two paths between entities (words) that did not co-occur elsewhere in the collection. This means we exploited *only* the higher order latent information. Table 4 compares the accuracy of HONB versus HONB with pure second-order paths. As before eight random samples were chosen for each newsgroup category and the average accuracies computed. As can be seen in Table 4, there is no statistically significant difference between the results.

We conclude that the performance improvement of HONB is due to the fact that the algorithm leverages the latent information present in higher order paths. By exploiting this latent information we obtain significantly better results than traditional IID classifiers on several data sets.

Overall, these results indicate that when there is insufficient information for traditional NB to estimate parameters, HONB exploits the latent information in higher order paths to achieve significantly better generalization.

## 6 DISCUSSION

By taking higher order paths as our base unit of “semantics”—our “documents,” if you will—we reveal the latent semantics that distinguish instances of different classes. These results also support our hypothesis that we can capture class-specific latent information in a generative

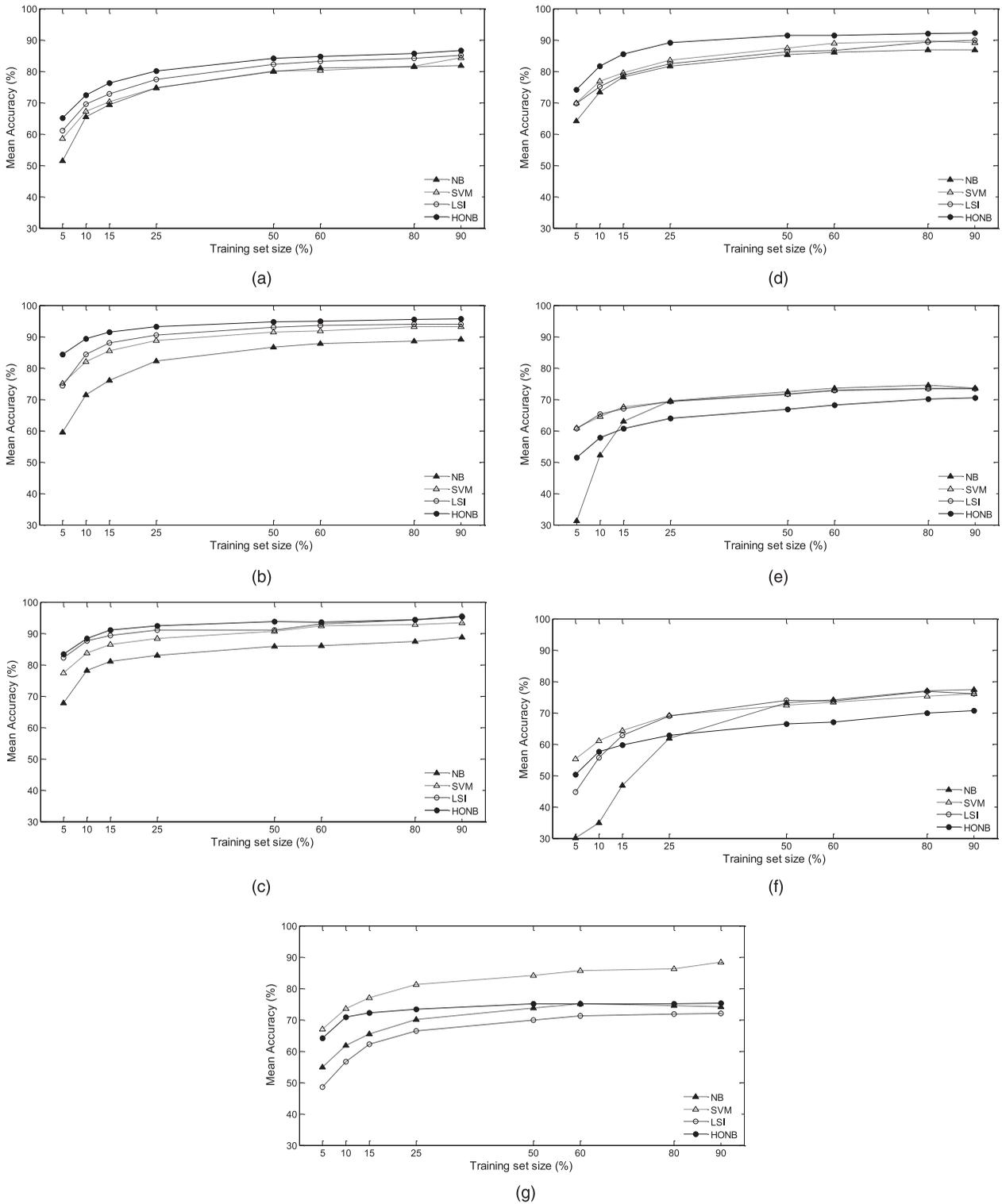


Fig. 7. Classifier accuracy from five percent to 90 percent of training set. (a) Computer. (b) Science. (c) Politics. (d) Religion. (e) Citeseer. (f) Cora. (g) WebKB.

classification model like NB under certain sparse data conditions. Indeed, for some applications all one can get is a very small number of labeled examples: precisely the conditions under which HONB performs well.

One of the limitations of HONB is the constraint to binary data, which indicates only the presence or absence of a word in a document. In consequence, our results may not generalize to multinomial data, which includes word frequency

information. However, one direction of our future work is to adapt HONB to use a multinomial model instead of a multivariate model. It may be possible to incorporate word frequency information in a higher order path. For example, in the second-order path  $w_1 \sim D_x \sim w_2 \sim D_y \sim w_3$ , the word frequencies in  $D_x$  and  $D_y$  could be used, respectively, for  $w_1$  and  $w_3$ .  $w_2$  poses a problem because it occurs in two documents:  $D_x$  and  $D_y$ , and we could choose to use either

TABLE 4  
Accuracy of HONB versus HONB with Pure Higher Order Paths

Dataset	HONB (Avg.)	HONB pure (Avg.)	T-test p-value
COMP (5)	65.1	64.9	0.73
SCI (4)	84.3	84.2	0.82
POL (3)	83.3	83.6	0.76
REL (3)	74.2	74.5	0.88

frequency or some combination. In any case, by leveraging word frequency information in higher order paths it may be possible to exploit more information and obtain improvements on nonbinary data sets. This is especially interesting given the observations of the difference in performance of multivariate versus multinomial models in [17].

## 7 CONCLUSIONS

In prior work [9], we gave a mathematical proof supported by empirical results of the dependence of LSI on higher order paths. In this work, a general approach to leveraging higher order dependencies for supervised learning was developed. We presented a novel classification method termed HONB.

Higher order paths allow a classifier to operate on a much richer data representation than the conventional feature vector form. This is especially important when working with small training sets, where accurate parameter estimation becomes very challenging for traditional IID approaches [11]. Experimental results affirmed the value of leveraging higher order paths on binary data, resulting in significant improvements in classification accuracies on benchmark text corpora across a wide range of training set sizes.

Based on these and other similar results, we conclude that for binary data sets, higher order paths provide valuable information that can improve model performance over traditional models. This work moves the field a step closer to understanding how higher order information should be leveraged in a systematic way for greatest impact.

## ACKNOWLEDGMENTS

The authors wish to thank Rutgers University, the National Science Foundation and the Department of Homeland Security. This material is based upon work partially supported by the National Science Foundation under Grant Numbers 0703698 and 0712139 as well as by the Department of Homeland Security Science and Technology Directorate. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Department of Homeland Security or Rutgers University. The authors also wish to thank colleague Shenzhi Li for her many insightful comments during the development of HONB. Coauthor Murat Can Ganiz wishes to thank Aydin Gerek from the Lehigh University Mathematics Department for his help in developing the closed-form formulas. They also would like

to thank the anonymous reviewers for their input; it improved this paper significantly. Finally, coauthor W.M. Pottenger gratefully acknowledges the continuing help of his Lord and Savior, Yeshua the Messiah (Jesus the Christ) in his life and work.

## REFERENCES

- [1] B. Taskar, P. Abbeel, and D. Koller, "Discriminative Probabilistic Models for Relational Data," *Proc. 18th Ann. Conf. Uncertainty in Artificial Intelligence (UAI '02)*, 2002.
- [2] L. Getoor and C.P. Diehl, "Link Mining: A Survey," *SIGKDD Explorations*, vol. 7, no. 2, pp. 3-12, 2005.
- [3] S.A. Macskassy and F. Provost, "A Brief Survey of Machine Learning Methods for Classification in Networked Data and Application to Suspicion Scoring," *Proc. Workshop Statistical Network Analysis at 23rd Int'l Conf. Machine Learning*, 2006.
- [4] R. Angelova and G. Weikum, "Graph-Based Text Classification: Learn from Your Neighbors," *Proc. ACM SIGIR '06*, 2006.
- [5] S. Chakrabarti, B. Dom, and P. Indyk, "Enhanced Hypertext Classification Using Hyper-Links," *Proc. ACM SIGMOD '98*, pp. 307-318, 1998.
- [6] J. Neville and D. Jensen, "Iterative Classification in Relational Data," *Proc. Workshop on Statistical Relational Learning, 17th Nat'l Conf. Artificial Intelligence*, pp. 42-49, 2000.
- [7] B. Taskar, E. Segal, and D. Koller, "Probabilistic Classification and Clustering in Relational Data," *Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI '01)*, pp. 870-878, 2001.
- [8] Q. Lu and L. Getoor, "Link-Based Classification," *Proc. Int'l Conf. Machine Learning (ICML '03)*, pp. 496-503, 2003.
- [9] A. Kontostathis and W.M. Pottenger, "A Framework for Understanding LSI Performance," *Information Processing and Management*, vol. 42, no. 1, pp. 56-73, 2006.
- [10] S. Deerwester, S.T. Dumais, and R. Harshman, "Indexing by Latent Semantic Analysis," *J. the Am. Soc. for Information Science*, vol. 41, no. 6, pp. 391-407, 1990.
- [11] N. Slonim and N. Tishby, "The Power of Word Clusters for Text Classification," *Proc. 23rd European Colloquium on Information Retrieval Research*, pp. 191-200, 2001.
- [12] D. Jensen, J. Neville, and B. Gallagher, "Why Collective Inference Improves Relational Classification," *Proc. ACM SIGKDD '04*, 2004.
- [13] P. Domingos and M. Pazzani, "Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier," *Proc. Int'l Conf. Machine Learning (ICML '96)*, pp. 105-112, 1996.
- [14] G.I. Webb, J.R. Boughton, and Z. Wang, "Not So Naive Bayes: Aggregating One-Dependence Estimators," *Machine Learning*, vol. 58, no. 1, pp. 5-24, 2005.
- [15] R. Diestel, *Graph Theory*. Springer Press, 2000.
- [16] J.H. Van Lint and R.M.A. Wilson, *A Course in Combinatorics*. Cambridge Univ. Press, 1993.
- [17] A.K. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification," *Proc. AAAI/ICML-98 Workshop Learning for Text Categorization*, pp. 41-48, 1998, also Technical Report WS-98-05, 1998.
- [18] S. Chakrabarti, *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, 2002.
- [19] J. Stirling, *Methodus Differentialis*. William Boywer, 1730.
- [20] L.E. Holzman, T.A. Fisher, L.M. Galitsky, A. Kontostathis, and W.M. Pottenger, "A Software Infrastructure for Research in Textual Data Mining," *Int'l J. Artificial Intelligence Tools*, vol. 14, no. 4, pp. 829-849, 2004.

- [21] A.K. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the Construction of Internet Portals with Machine Learning," *Information Retrieval*, vol. 3, pp. 127-163, 2000.
- [22] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, K. Nigam, and S. Slattery, "Learning to Extract Symbolic Knowledge from the World Wide Web," *Proc. 15th Nat'l Conf. Artificial Intelligence Applications of Artificial Intelligence (AAAI '98)*, 1998.
- [23] P. Sen and L. Getoor, "Link-Based Classification," Technical Report CS-TR-4858, Univ. of Maryland, Feb. 2007.
- [24] D.R. Swanson, "Migraine and Magnesium: Eleven Neglected Connections," *Perspectives in Biology and Medicine*, vol. 31, no. 4, pp. 526-557, 1998.
- [25] S. Zelikovitz and H. Hirsh, "Using LSI for Text Classification in the Presence of Background Text," *Proc. Conf. Information and Knowledge Management (CIKM '01)*, pp. 113-118, 2001.
- [26] S. Zelikovitz and H. Hirsh, "Improving Short-Text Classification Using Unlabeled Background Knowledge to Assess Document Similarity," *Proc. Int'l Conf. Machine Learning (ICML '00)*, pp. 1183-1190, 2000.
- [27] J. Sun, Z. Chen, H. Zeng, Y. Lu, C. Shi, and W. Ma, "Supervised Latent Semantic Indexing for Document Categorization," *Proc. Int'l Conf. Data Mining (ICDM '04)*, pp. 535-538, 2004.
- [28] T. Liu, Z. Chen, B. Zhang, W. Ma, and G. Wu, "Improving Text Classification Using Local Latent Semantic Indexing," *Proc. Int'l Conf. Data Mining (ICDM '04)*, pp. 162-169, 2004.
- [29] S. Chakraborti, R. Mukras, R. Lothian, N. Wiratunga, S. Watt, and D. Harper, "Supervised Latent Semantic Indexing Using Adaptive Sprinkling," *Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI '07)*, pp. 1582-1587, 2007.
- [30] F. Wild, C. Stahl, G. Stermsek, G. Neumann, and Y. Penya, "Parameters Driving Effectiveness of Automated Essay Scoring with LSA," *Proc. Int'l Computer Assisted Assessment (CAA '05)*, pp. 485-494, 2005.
- [31] F. Wild, "An LSA Package for R," *Proc. First Int'l Conf. Latent Semantic Analysis in Technology Enhanced Learning (LSA-TEL'07)*, pp. 11-12, 2007.
- [32] A. Karatzoglou, D. Meyer, and K. Hornik, "Support Vector Machines in R," *J. Statistical Software*, vol. 15, pp. 1-28, 2006.



**Murat Can Ganiz** received the PhD degree in computer science from the Lehigh University, in 2008. He is currently an assistant professor in Department of Computer Engineering at Dogus University, Turkey, which he joined in 2009. His research interests include machine learning, text mining, data mining, link analysis, and social network analysis. He is a member of the IEEE.



**Cibin George** received the BTech degree in computer science and engineering from the National Institute of Technology, Calicut, India, in 2003. He is currently working toward the MS degree in computer science at Rutgers University. His research interests include text mining, information retrieval, and machine learning.



**William (Bill) M. Pottenger** is the CEO and founder of Intuidex, a manufacturer of solutions in the visual and data analytics space. He is also the director of Transition for CCICADA, the USA Department of Homeland Security Command, Control and Interoperability Center for Advanced Data Analysis. He is also an associate research professor at Rutgers University at DIMACS and in the Computer Science Department. His research interests include the fields of statistical

relational learning and information extraction as applied in higher order learning, a framework he developed for both supervised and unsupervised learning based on higher order feature relations. He is active in research in visual and data analytics and parallel and distributed computing as well. His company, Intuidex, Inc., creates leading-edge visual and data analytics and collaborative information management technology for use both at home and in business. He is a member of the IEEE.

► For more information on this or any other computing topic, please visit our Digital Library at [www.computer.org/publications/dlib](http://www.computer.org/publications/dlib).