A Higher Order Collective Classifier for Detecting and Classifying Network Events

Vikas Menon
Department of Computer Science
Rutgers University, New Brunswick
New Jersey 08854
Email: menon@cs.rutgers.edu

William M. Pottenger
Department of Computer Science & DIMACS
Rutgers University, New Brunswick
New Jersey 08854
Email: drwmp@cs.rutgers.edu

***Abstract*—Labeled Data is scarce. Most statistical machine learning techniques rely on the availability of a large labeled corpus for building robust models for prediction and classification. In this paper we present a Higher Order Collective Classifier (HOCC) based on Higher Order Learning, a statistical machine learning technique that leverages latent information present in co-occurrences of items across records. These techniques violate the IID assumption that underlies most statistical machine learning techniques and have in prior work outperformed first order techniques in the presence of very limited data.**

We present results of applying HOCC to two different network data sets, first for detection and classification of anomalies in a Border Gateway Protocol dataset and second for building models of users from Network File System calls to perform masquerade detection. The precision of our system has been shown to be 30% better than the standard Naive Bayes technique for masquerade detection. These results indicate that HOCC can successfully model a variety of network events and can be applied to solve difficult problems in security using the general framework proposed.

I. Introduction

Solving any problem in general involves three phases; detection, classification and resolution in that order. In the absence of sound detection and classification, it is improbable to get the correct resolution. Detection has historically been an easier problem while providing a correct solution has been the most difficult. The problem of providing a resolution is more challenging because it is likely that one may come across a completely new problem which would require creative thought. In this paper we present HOCC a Higher Order based Collective Classifier that is capable of addressing the first two phases of problem solving, detection and classification in particular to security problems for network data. We use this general frame work to solve two difficult and different problems in security; detection & classification of attacks/anomalies and masquerade detection.

To detect and classify attacks/anomalies we use the Border Gateway Protocol (BGP) data. BGP forms the backbone of Internet routing functionality, and any large scale anomalous events (such as worms, misconfiguration and power failure) have a direct impact on BGP performance. Identification and classification of such events can be employed to avoid problems such as route flapping [7] and potentially mitigate damage [2] caused by the events (such as quarantining infected machines). Therefore, identification and classification of BGP events in real time is an important task. There exist several approaches in literature that are capable of identifying anomalous activity in BGP control/data planes. Classifying anomalous events is a more challenging problem. In [5], a collective classification algorithm was developed that distinguishes different events with high accuracy, based on patterns of connectivity in graphs of BGP control-plane data. However, one of the issues in the approach in this prior work is the very large time complexity of the classification process. In addition, although the approach in [5] does address classification of events, it does not address the detection of anomalous events. In this paper, we present a novel algorithm for identification and classification of anomalous BGP events.

A masquerade is where a person disguises his behavior as another user in order to gain the privileges of that user. For masquerade detection we utilize the Harvard Network File System (NFS) trace dataset. The dataset consists of network file system accesses of two groups, the Electrical Engineering and Computer Science graduate students and CAMPUS, the central computing facility at the university. With this dataset we model the behavior of each user using very limited labeled data. We compare the user's network accesses with this model to identify masquerades. The advantage of this dataset is that it consists of a very large corpus of labeled data which helps to quantify classification performance better.

As discussed, we show how HOCC can be applied to two completely different problems. We provide a more general frame work where HOCC can be extended to solve a variety of network security related problems. We utilize this framework to solve two difficult problems in this paper. In the next section we present related work to Higher Order Learning and statistical machine learning with respect to HOCC. We follow this up with a brief overview of HOCC, its application to the BGP and NFS data sets and finally present our conclusions.

II. Related Work

Modern data mining algorithms primarily leverage first-order relations, aiming to discover patterns such as association or classification rules only within records. The underlying assumption is that records are independent and identically distributed [19]. This assumption simplifies the underlying

mathematics of statistical models, but in fact does not hold for many real world applications [6]. Although useful models can be learned, in many cases more accurate models can be learned if higher-order associations are leveraged by the data mining algorithm: i.e., associations between records. For example, in supervised learning a classification model is applied to a single record and a prediction is made based on the items (i.e., attribute-value pairs, or features) of the given record in a context-free manner, independent of the other records in the dataset. However, this context-free approach does not exploit the available information about relationships between records in a dataset [1]. Coupled with this is a growing need for data fusion due to increasing data fragmentation [16]. Consequently, there is a greater need to incorporate contextual information - i.e., higher-order associations that cross record boundaries. In general, our research falls into the field of Statistical Relational Learning (SRL). SRL models operate on relational data that includes explicit links between instances. These relations provide rich information that can be used to improve classification accuracy of learned models since attributes of linked instances are often correlated, and links are more likely to exist between instances that have some commonality [6]. Given a set of test instances, relational models typically label related instances in order to exploit the correlations between class labels. This is called collective classification (or collective inference), and as noted violates the traditional independence assumption. Several studies [3], [15], [19] have shown, however, that by making inferences about multiple data instances simultaneously collective inference can significantly reduce classification error [8].

In our prior work [10], we mathematically proved that Latent Semantic Indexing (LSI), a well-known approach to information retrieval, implicitly depends on higher-order co-occurrence associations. We also demonstrated empirically that higher-order associations 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. In a related effort, [4] uses higher-order co-occurrence to solve a component of the problem of lexical choice, which identifies synonyms in a given context. In another effort, [24] use second-order co-occurrence to improve the runtime performance of LSI. Second and higher-order co-occurrence have also been used in other applications including word sense disambiguation [17] and stemming [22].

In summary, several recent studies including those mentioned above show that collective classification can significantly improve classification accuracy compared to the traditional classification techniques. Our approach, which we term Higher Order Collective Classification, is based on a collective classification algorithm that leverages Higher Order associations.

III. HOCC: A Higher Order Collective Classifier

HOCC creates a multi-graph from aggregated instances over a *window* of time. The vertices of the graph are unique attribute value pairs while the edges are generated based on co-occurrence of two items in any record. This graph represents the model of an ongoing event. From the labeled instances we generate graphs for the *Event Model* and compare them with the *Real Time Models*. A comparison of the graph can be performed using various techniques such as item set enumeration [5] or record set enumeration [14]. HOCC uses the latter novel Higher Order record set enumeration technique which is both time and space efficient. For more details on the HOCC algorithm we refer you to [14].

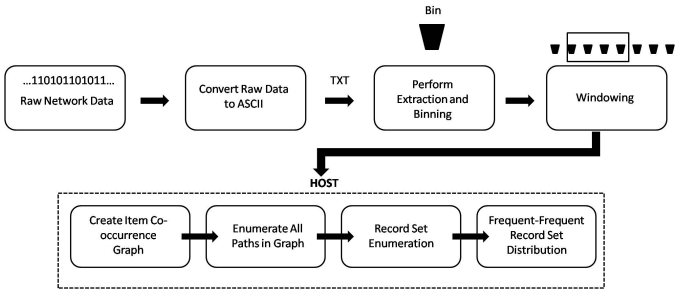

Fig. 1. HOCC

The Record-Set enumeration technique generates a distribution of integers from the graph. We compare two graphs by performing the Student's T-test between two distribution and use p-values for comparison.

We define two models that we shall refer to frequently:

- **Real Time Model** The sliding windows that are obtained from real time network event data are referred to as the Real Time Model.
- **Event Model** This is the model obtained from labeled event bins. This model represents the signature of a previously known event and is used for classification. These models are pre-computed for comparison with the Real Time Models.

Figure 1. shows how HOCC can be applied in a generic manner to different problems. We use this same framework to solve the problems of anomaly detection and classification and masquerade detection in the next sections.

IV. BGP Anomaly Detection and Event Classification:

[13] provides an excellent survey of general anomaly detection problems and techniques. There have been several successful systems developed for the detection of anomalies in BGP traffic. Some previous techniques applied to BGP data include using neural networks and rule mining techniques that can detect novelty (or anomalies) such as those presented in [12] and [11] respectively. In [11], Li et al. use attributes derived from BGP traffic to detect Internet routing anomalies. They employ data mining techniques, in particular a decision tree machine learning algorithm, to train a model using labeled data. The authors use the counts of different types of BGP messages divided into one minute bins. Their model consists of the rules learned, and is used to detect occurrences of abnormal

events. Basically their system can distinguish between two classes - event and normal - but cannot identify the exact type of an event. Thus one important drawback in their approach is that it cannot distinguish between different anomalous events such as worms. In fact, in her public review, Dina Katabi from MIT points out the importance of identifying whether an abnormal event is caused by a worm, blackout or misconfiguration [11]. Similarly [12] achieves success in detecting novel events in BGP but is unable to differentiate between them.

Several other research efforts of a similar nature have been conducted in [25], [22], [24] and [23]. [24] proposes two approaches, signature-based and statistics-based detection. In summary, previous work has been focused on the detection of anomalies rather than the classification of events, and is thus in general unable to differentiate between events.

One significant exception that succeeded in the classification of events is [5] which is able to distinguish between events with very high accuracy. However, the authors were unable to build a model for non-anomalous events (normal behavior). As a result, the approach cannot be used for anomaly detection *per se*. In addition, the approach has a high time complexity which prevents it from being used in practice. This then forms the basis for our work. We build on the work completed in [5] and develop a novel approach capable of precisely classifying events *as well as* identifying anomalies. More significantly, we achieve this in real-time based on a novel approach that leverages our research in Higher Order Learning discussed below.

The BGP control plane data is in binary. We use free tools provided by [20] to convert the binary files into plain text representing packet level information. This information is continuous and each packet has a time stamp. We use these time stamps to create three second bins. Each three second bin consists of a variable number of packets. We extract the following information for each three second bin:

1) Number of Announcements
2) Number of Withdrawals
3) Number of Announcements and Withdrawals
4) Number of AS prefixes announced. Each announcement (1) can announce several AS prefixes
5) Number of AS prefix withdrawals. Each withdrawal (2) can withdraw several Autonomous System (AS) prefixes
6) Number of AS prefix announcements and withdrawals

The bins are generated every three seconds. We create a sliding window over 120 of such three second bins. HOCC is applied to each of these sliding windows in turn. Each window is compared to existing models of both events and non-events (i.e., the normal event model). The Event model was created using a single sliding window containing 120 contiguous three-second bins taken from the event period.

To perform anomaly detection we compare the Real Time Model to the normal Event Model while for classification we compare the Real Time Model to the signature event Models of previously known events. As noted above we compare models by performing the student's T-test between the record set distributions of the models generated by HOCC.

TABLE I
EVENT VS. EVENT COMPARISON

Event 1	Event 2	T-test p-values
Slammer	Witty	0.00016127
Blackout	Witty	0.031218
Slammer	Blackout	0.036645
Normal	Slammer	1.06×10^{-5}
Normal	Witty	0.035918647
Normal	Blackout	0.000401

We tested HOCC with three completely different events. Slammer, which was non-malicious, spread very quickly and caused network outages because it flooded the network but did not damage the infected machines. Witty was similar to Slammer but malicious. Any infected machine was eventually brought down. This meant the spread of Witty was curbed in comparison to Slammer. It only infected a tenth of the number of machines as Slammer. Finally, the last event we consider is Blackout which was caused due to electricity failure and brought down a large number of BGP servers.

For Witty and Slammer, a data set of 600 bins was used, 300 of which are pre-event and 300 post, while for Blackout a dataset of 800 bins was used, 400 of which are pre-event and 400 post. We created sliding windows each composed of 120 consecutive three second instances. From each window we created the item co-occurrence graph and enumerated paths of length three ($p = 3$). In all, for Slammer and Witty 480 windows (600 - 120 bins) were formed and 680 for Blackout.

In *Figure* 2 the sliding window numbers are on the X-axis while the Y-axis corresponds to the value of the T-test p-values ranging between 0 and 1. The horizontal red line near the X-axis is a measure of significance set to 5%. The thin vertical green line indicates the point in time where the bins from the various events start filtering into the sliding window and the thick vertical green line indicates the point where the sliding window has completely entered into the event.

A. Supervised Classification

As noted, HOCC statistically distinguishes between events using a T-test between event models for each known event. Thus the events themselves must be different. The results in *Table* I indicate that each of the event model classes are statistically significantly different based on T-test p-values.

In *Figure* 2 $(a), (c) and (e)$ the results of the T-test comparisons between the Real Time Model and the event models of Witty, Slammer and Blackout are shown across all sliding windows. In Witty and Slammer, HOCC is able to detect a peak almost exactly at the point the Real Time Model moves into the event. For Blackout, the event is detected about 30 seconds later. Larger values of the T-test indicate that the distribution of the Real Time Model matches the distribution of the given Event Model. Small values (red line at bottom) indicate that the models are dissimilar. The results of the T-test remain consistently greater than 5% for Slammer and Blackout

once the Real Time Model moves into the event and less than 5% otherwise. For Witty, the T-test p-values rise sharply and significantly at the start of event but immediately after, decline sharply. Although Witty is positively identified by HOCC, the rapid dropoff is consistent with the fact that Witty has proven difficult to model in prior work [18]. Yet, even though the drop off is very rapid, the magnitude of the difference between the T-test p-values is much larger than [5] indicating that HOCC is more sensitive in classification.

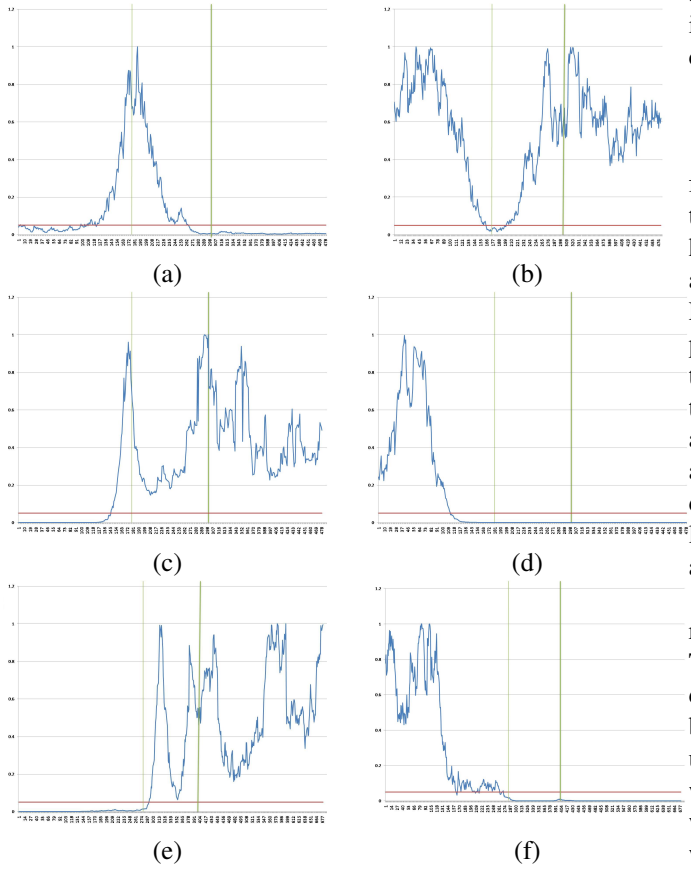

Fig. 2. Figures (a), (c) and (e) depict the results of supervised classification and (b), (d) and (f) the results of anomaly detection for Witty, Slammer and Blackout respectively.

B. Unsupervised Anomaly/Novelty Detection

In this section we present the results of using HOCC for anomaly detection in which we compared the Real Time Model with the Normal Event Model. In *Figure* 2, (b), (d) and (f), HOCC is used to identify the Slammer, Witty and Blackout as *anomalies*, not events. As noted, here the T-test p-values were obtained by comparing the Real Time Model to the Normal Model. We observe large T-test p-values (closer to 1) before the events begin. This is consistent with the fact that the pre-event periods exhibit normal behavior. As the sliding windows move into the event, the T-test p-values fall significantly (to less than 5%). This drop is very large for Slammer and Blackout where the T-test p-values are of the order of 10^{-5} and 10^{-4} respectively. For Witty the drop though not this large, is enough to be significant (less than 5%). The T-test p-values in anomaly detection are roughly the inverse of those seen in classification. This is to be expected, and is an indication that the various models (Normal, Event) have accurately captured the behavior. This is significant for these results support the conclusion that we now have a technique whereby we can successfully model both normal BGP behavior as well as specific targeted events using a single approach based on higher-order path statistics: HOCC. As stated earlier, to the best of our knowledge no other work in literature has been capable of both anomaly detection and classification.

V. Harvard NFS Trace & Masquerade Detection

Masquerade detection has been a difficult problem. Data for masquerade detection is difficult to obtain primarily due to privacy concerns [9]. For this reason, the Harvard dataset has been anonymized. The anonymization process replaced all UID (User Identity), GID (Group Identity) and IP (Internet Protocol) address by arbitrary but consistent values. Another problem that makes masquerade detection difficult is the fact that user behavior changes over time, so a user's behavior today can be potentially completely different a year later. In addition, data collection efforts have not focused on security applications. This has made quantifying performance across different systems difficult. Using the Harvard NFS traces, a large corpus of labeled data it should, however, be possible to address this problem.

The data set contains a wealth of information and can be replayed in real time to generate/simulate the entire traffic. This data contains information of users, groups, calls, addresses, ports and many more albeit anonymized. It contains both network and file system level information. One important underlying assumption that we had to make right at the start was that the data contained absolutely no masquerades. This way we could assume that each user was himself/herself and we could test for masquerades by comparing with other users. Another assumption was that the users behavior changes over time and the change is of a nature where the core behavior of the user remains most like him/herself. This means that even though a user's behavior might have changed with respect to himself, this new behavior is more similar to the user's past behavior. This assumption is important and helps us frame a maximum likelihood problem. Our experimental results confirm the validity of this assumption. In the following sections we explain the use of HOCC for detection of masquerades and compare results with first order systems for the NFS dataset.

A. Experimental Methodology

We randomly selected 10 users from the NFS file traces. For each of these 10 users we randomly choose a start point in time from the traces and collected 500,000 raw NFS calls. These calls were divided into two categories for training and testing divided in time. The initial calls were used for training and the remaining for testing. The time span for these calls was approximately two weeks and we created different signature

model sizes for each user ranging from window sizes of 20 to 20480 seconds (or 5.6 hours). Our best overall results were achieved using two second bins over a window of length 80 for a total of 160 seconds.

From the training set we created 10 different trained models of each of these users, while from the test set we picked up 20 windows for the same users.

Each NFS Call consisted of Time stamp, Source Address, Source port, User ID, Group ID, Call Type (which could be 1 of 21 different NFS call types) , Acc (Access permission type) and Stable_how (this is a bit that indicates the nature of a write). Even though more information was available, we choose these attributes based on the results of attribute selection for first order techniques using the Weka workbench [21] .

Using the Time stamps we binned this data into 2, 4, 8, 16, 32, 64, 128 and 256 second intervals. Each bin consisted of the following information; Total Number of Calls, Total number of Source Addresses, Total number of Source Ports, Total number of calls of each of the 21 types of calls, Total number of Groups, Total number of Acc and Total number of stable_how.

Each instance thus represented binned information over the 500,000 raw instances. We then divided these into windows with sizes ranging from 10, 20, 40, 80, 120 and 160 bins. Each of these windows models user activity for that period. HOCC was used for the comparison of these models. It is expected that the trained models of the user should conform with the test models. Any significant deviations in T-test p-values are considered as anomalies. We present results obtained for models with two second bins and window size of 80.

Table III presents the results of our comparison of user trained models (in column 1) versus the test models (in row 1). Each comparison is the average of p-values of 200 comparisons (10 training models $\times$ 20 test models). For example, the first value in the table corresponds to the comparison of column user 18a88 training set versus row user 18a88 test set. A large value reflects greater similarity. To be precise, although anything larger than a value of p=0.05 would be sufficient to conclude that two distributions are not statistically significantly different, a larger p-value indicates "lesser dissimilarity". The ideal comparison table would then look like a unit matrix with 1.0 along the diagonal and 0 for the rest. But we are dealing with a lot of noise in the form of low level calls that are more often made by programs than users themselves. As a result, we end up modeling a users choice of programs. Also, there is a large amount of noise due to background daemons and services. This makes modeling any user a difficult problem, in particular for first order techniques which end up over fitting and in the worst cases, performing at the baseline.

As can be seen in Table III, our results show large values along the diagonal with an average of 0.8 whereas the non-diagonal elements average 0.13.

One of the concerns with this technique is that some users may have small p-values when compared to themselves, for user 18ad2 the p-value is 0.541. In other cases, we might see users that might not be significantly different from the user under consideration. To resolve these cases, we use a maximum likelihood framework for one class classification [9]. In such cases, we apply the MAX operator on the corresponding row. The MAX operator sets the largest average p-value to one and zero to the rest. In case there are two equally large, MAX randomly chooses one. In the case of user 18ad2 it tells us that the user indeed is most like himself (largest values highlighted in bold). This can also be confirmed by applying the MAX operator along each column. A notable instance is user 18ad6. This user even though most similar to himself is also quite similar to user 18a88. Even though this instance satisfies the MAX operator property and is correctly identified, it is expected that a good technique should strongly differentiate between the two. For this reason we also tried comparing users with different models. For example using 64 second bins and windows of size 80 we can strongly distinguish between user 18ad6 and 18a88 as shown in Table II. Thus, a second model optimized toward a particular user could be used for validation.

TABLE II
USER VS. USER COMPARISON. 64 SECOND BINS WITH WINDOW SIZE 80

User	18a88	18ad6
18a88	**0.7455**	0.1308
18ad6	0.1437	**0.9028**

When we choose to use the most optimal models for each user for comparison, the diagonal had a larger average p-value of 0.89 and the average of non-diagonal elements falls to 0.11. For the simple MAX operator both precision and recall is defined as the ratio of the sum of the diagonal to the total number of users (since there can only exist one maximum in each row and the rest are zero). By building custom models for each user in the maximum likelihood framework we were able to get results with 100% recall and precision for this small set of users, similar to the result shown in Table III. Similarly, for the example shown in Table III, by applying the MAX operator we were able to get ideal performance with 100% precision and recall using just 160 seconds of labeled data to build our models.

In complete contrast to these results, first order techniques were able to achieve low accuracy. We ran the Naive Bayes algorithm on the same dataset. The class for each instance was set to UID and the classifier was trained using 10 fold cross validation, implying 90% of the data was used to train the models. The Naive Bayes classifier achieved an accuracy of 70.07% on both training and test data. Also, as the number of users increase (or number of classes increased) the quality of models drops significantly. For instance for a 20 user data set Naive Bayes had a classification accuracy of only 50%.

VI. CONCLUSION

Algorithms are often specific to an application, making it difficult to extend the same techniques to new domains. In this paper we have presented a novel Higher Order Learning-based record set enumeration technique, HOCC, capable of

TABLE III
USER VS. USER COMPARISON. TWO SECOND BINS WITH WINDOW SIZE 80

User	18a88	18a8a	18a89	18a8e	18a9f	18aa9	18aaa	18ad2	18ad6	18b17
18a88	**0.9535**	0.3811	0.1618	0.0556	0.0623	0.0031	0.1891	0.0072	0.8206	0.2367
18a8a	0.2666	**0.7349**	0.1648	0.1464	0.0003	5.19E-07	0.2425	3.06E-06	0.3757	0.0056
18a89	0.1616	0.1637	**0.9339**	0.1741	0.1582	0.1563	0.1908	0.1568	0.1621	0.1593
18a8e	0.0576	0.1036	0.1769	**0.8357**	0.0146	0.0068	0.5584	0.0083	0.0685	0.0227
18a9f	0.0840	0.0008	0.1585	0.0114	**0.7974**	1.31E-06	0.1382	0.0006	0.0607	0.1659
18aa9	0.0054	8.73E-07	0.1565	0.0044	4.15E-06	**0.6855**	0.1143	0.0001	0.0043	2.00E-07
18aaa	0.1868	0.2230	0.1928	0.4781	0.1357	0.1133	**0.9543**	0.1188	0.1953	0.1513
18ad2	0.0109	6.11E-06	0.1570	0.0055	0.0008	8.05E-06	0.11953	**0.5410**	0.0083	2.05E-05
18ad6	0.8560	0.4896	0.1621	0.0663	0.0911	0.0081	0.1941	0.0159	**0.8608**	0.2616
18b17	0.2206	0.0087	0.1594	0.0174	0.1806	1.40E-07	0.1508	2.37E-05	0.1621	**0.7312**

identifying and classifying network events in real time. To the best of our knowledge, HOCC is the only system capable of doing both anomaly detection and event classification, as demonstrated for the BGP data. HOCC is also capable of performing masquerade detection by building consistent models of users across time. These applications are quite diverse, and showcase the variety of problems amenable to modeling with the Higher Order Learning-based approach in HOCC. Furthermore, HOCC is able to do this by building robust models using very little labeled data. HOCC is a next generation system capable of explicitly utilizing latent higher order information in data sets with the added advantage of being faster and more space efficient than previous generation Higher Order Learning-based systems.

ACKNOWLEDGMENT

The authors wish to thank Rutgers University and the National Science Foundation. This material is based upon work partially supported by the National Science Foundation under Grant Numbers 0703698 and 0712139. 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 or Rutgers University.

We are also grateful for the help of co-workers, family members and friends. Co-author W. M. Pottenger also gratefully acknowledges the continuing help of his Lord and Savior, Yeshua the Messiah (Jesus the Christ) in his life and work.

REFERENCES

[1] Angelova, R., Weikum, G. *Graph-based text classification: learn from your neighbors.* SIGIR 2006:485-492.

[2] Caesar,M. , Subramanian, L. , Katz, R.H. *Route Cause analysis of Interdomain routing Dynamics*. Tech Report, UCB/CSD-04-1302, 2003.

[3] Chakrabarti, S., Dom, B., P. Indyk. *Enhanced Hypertext Classification Using Hyper-Links.* In Proceedings of ACM SIGMOD Conference, pp. 307-318.

[4] Edmonds, P. *Choosing the word most typical in context using a lexical co-occurrence network.* In Proceedings of the Thirty-fifth Annual Meeting of the Association for Computational Linguistics, 1997, pp. 507-509.

[5] Ganiz, M.C., Kanitkar, S., Chuah, M.C., Pottenger, W.M.*Detection of Inter domain Routing Anomalies Based on Higher-Order Path Analysis.* Proceedings of the Sixth International Conference on Data Mining (ICDM) 2006.

[6] Getoor, L., Diehl, C. P. *Introduction to the special issue on link mining.* SIGKDD Explorations (SIGKDD) 7(2):1-2 (2005)

[7] Giffin, T. *What is the Sound of One Route Flapping?* IPAM 2002.

[8] Jensen, D., Neville, J., Gallagher, B. *Why collective inference improves relational classification.* KDD 2004:593-598

[9] Ke Wang, Salvatore J. Stolfo.*One Class Training for Masquerade Detection.* ICDM Workshop on Data Mining for Computer Security (DMSEC). Nov. 19, 2003, Melbourne, FL.

[10] Kontostathis, A., Pottenger, W. M. *A Framework for Understanding LSI Performance*, Information Processing & Management, Volume 42, Issue 1, Pages 56-73. 2006.

[11] Li, J., Dou, D., Wu, Z., Kim, S., Agarwal, V.*An internet Routing Forensics Framework for Discovering Rules of Abnormal BGP events*. ACM SIGCOMM Computer Communication Review. Vol 35, Number 5, pg 57-66. 2005.

[12] Marais, E., Marwala, T.*Predicting the Presence of Internet Worms using Novelty Detection*. Technical Report.

[13] Markou, M., Singh, S. *Novelty detection: A review - parts 1 and 2*. Signal Processing,2003 volume 83.

[14] Menon, V. *A Higher Order Collective Classifier.* A thesis submitted to the Graduate School Rutgers, The State University of New Jersey in partial fulfillment of the requirements for the degree of Master of Science. February 2009.

[15] Neville J., Jensen, D. *Dependency Networks for Relational Data.* ICDM 2004:170-177

[16] Rahm, E., Bernstein, P. A. *A survey of approaches to automatic schema matching.* VLDB J. (VLDB) 10(4):334-350 (2001)

[17] Schutze, H. *Automatic word sense discrimination*, Computational Linguistics, 24(1):97-124, 1998

[18] Shannon, C.; Moore, D., *The spread of the Witty worm*. Security & Privacy, IEEE Volume 2, Issue 4, July-Aug. 2004 Page(s): 46 - 50

[19] Taskar, B., Abbeel, P., Koller, D. *Discriminative Probabilistic Models for Relational Data.* UAI 2002:485-492

[20] The Co-operative Association for Internet Data Analysis. *http://www.caida.org/*

[21] Witten, I. H., and Frank, E.*Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[22] Xu, J., Croft, W. B. *Corpus-based stemming using co-occurrence of word variants.* ACM Transactions on Information Systems 16 (1), 1998, pp. 61-81.

[23] Zhang, J., Rexford, J., Feigenbaum, J. *Learning-Based Anomaly Detection in BGP Updates.* Proceeding of the 2005 ACM SIGCOMM Workshop on Mining Network Data. 219 - 220, 2005.

[24] Zhang, X., Berry, M., Raghavan, P.*Level search schemes for information filtering and retrieval.* Information Processing and Management 37 (2), 2000, pp. 313-334.

[25] Zhao, X., Pei, D., Wang, L., Massey, D., Mankin, A., Wu, S., and Zhang, L.*Detection of Invalid Routing Announcements in the Internet.* Proceedings of Dependable Systems and Networks, 2002.