Using Randomized Projection Techniques to Aid in Detecting High-Dimensional Malicious Applications

Jan Durand
Department of Computer Science
Louisiana Tech University
Ruston, LA 71270
jrd037@latech.edu

Travis Atkison
Department of Computer Science
Louisiana Tech University
Ruston, LA 71270
atkison@latech.edu

ABSTRACT
This work is part of an on-going effort in using randomized projection as a feature extraction and reduction method to improve a cosine similarity, information retrieval technique to enhance the detection of known malicious applications and their variations. We follow a standard information retrieval methodology that allows software to be regarded as documents in the corpus. This provides the ability to search the corpus with a query, malicious software, and retrieve/identify potentially malicious software and other instances of the same type of vulnerability. In our experiments, we compare Gaussian-distributed random matrix randomized projection to two alternative methods of randomized projection, sparse matrix randomized projection and Linial-London-Rabinovich random set randomized projection, and assess their performance when applied to features of malicious applications extracted via the information retrieval technique of *n*-gram analysis. In our results, the Gaussian distributed random matrix approach outperformed the other methods with generally higher values for each observed performance metric, however, each algorithm showed promise in selected scenarios. These results support the hypothesis that applying the technique of random matrix projection as a dimensionality reduction method for the cosine similarity metric has merit in determining if an application may contain a malicious application.

Categories and Subject Descriptors
D.2.0 [**Software Engineering**]: General – Protection mechanisms

General Terms
Security

Keywords
Malicious software detection, information retrieval, n-gram analysis, cosine similarity, randomized projections

1. INTRODUCTION
The increasing ubiquity of the Internet in the 21st century has allowed for global connectivity and information sharing on an unprecedented scale. Along with the productivity afforded by such technology innovations comes the potential for malicious attacks from hackers and infection by malware. Cyber security has been an ever growing and crucial field in the defense of computer systems and networks from attack. In particular, considerable focus has been placed on the development of anti-virus software packages, which detect and incapacitate malicious applications such as viruses, worms and trojan horses. According to the 2007 CSI Computer Crime and Security Survey [1], 54.3 percent of the total budget for industry software security in 2005 was allotted to anti-virus software. In addition, the 2008 CSI Computer Crime and Security Survey [2] reported that 97 percent of the surveyed respondents, including U.S. corporations, government agencies, financial institutions, medical institutions and universities, used anti-virus software. Anti-virus software is a signature-based solution that relies on a database of signatures to detect known malicious applications. Such signature-based systems are practical [3] but inherently limit the detection of new and previously unknown types of malicious attacks [4]. Some attempts to overcome this limitation have been based in the fields of information retrieval and data mining [5-8] and have shown some success. However, they were all subject to the "curse of dimensionality", first introduced by Bellman in [9]. This typically refers to the computational challenges associated with mathematical operations in a high-dimensional space [4]. By moving to a low-dimensional space that preserves the underlying properties of our data, the effect of the "curse of dimensionality" can be mitigated. Principal Component Analysis (PCA) and Latent Semantic Analysis (LSA) mentioned in [10] are examples of such dimension reduction techniques. In this paper, we focus on the feature extraction and reduction technique of randomized projection introduced in [11, 12] to produce a low-dimensional embedding of our high-dimensional data. We extend the work done in [4, 13] using randomized projection as a feature extraction and reduction technique in the context of malicious application detection. We explore two alternative methods of randomized projection and assess their performance, in addition to the method used in [4], to determine which gives the best results when working with features of malicious applications extracted via the information retrieval technique of *n*-gram analysis.

The following section provides a short background description of information retrieval and randomized projection, related work and discusses malicious software vulnerabilities. In Section 3, the experimental design of this work is discussed including the software and data used. Section 4 discusses the results obtained from the experiments. Finally, section 5 presents our conclusions and identifies future directions.

2. BACKGROUND
Evaluating the effectiveness of a potential solution to the malicious software detection problem, in which a low-

49th ACM Southeast Conference, March 24-26, 2011, Kennesaw, GA, USA. Copyright 2011 ACM 978-1-4503-0686-7 $5.00.

dimensional embedding is used to reduce the dimensions of an information retrieval technique, is an important direction in host security research and has been shown to be promising by the results obtained in [4]. Below, we give a brief description of the information retrieval technique of *n*-gram analysis used previously, as well as an overview of the randomized projection technique. We also describe the type of malicious applications and vulnerabilities that are the target of our detection model.

2.1 Information Retrieval

Information retrieval, traditionally, is the "part of computer science which studies the retrieval of information (not data) from a collection of written documents." [21] These retrieved documents' aim is to "satisfy a user's information need." [21] The process can be thought of as combing through a set of documents, called the corpus, to find a certain piece of information that has a relationship to a given entity, called the query. That piece of information can either be an entire document, set of documents or a subset of a document. Within the information retrieval community several methods exist for finding these pieces of relevant information. These methods include vector space models, latent semantic indexing models and statistical confidence models as well as others. "Vector space models are the first approach to represent a document as a set of terms." [22] As their name implies vector space models represent their data as a vector with each dimension being defined as a term which may or may not have a weight associated with it [23]. One of the most common vector space models is cosine similarity. Cosine similarity determines the similarity between two data vectors by measuring the angular distance between them. "Cosine has the nice property that it is 1.0 for identical vectors and 0.0 for orthogonal vectors." [24] The following is the formula used in this work for computing cosine similarity:

$$\text{Cosine Similarity (Q, D)} = \frac{\sum_i w_{Q,i} w_{D,i}}{\sqrt{\sum_i w_{Q,i}^2} \sqrt{\sum_i w_{D,i}^2}} \quad (1)$$

This formula computes the similarity between a query *Q* and a document *D* by summing the individual components of the two entities represented in the formula as *w*. The individual components for this research, *w*, are defined as *n*-grams. An *n*-gram is "any substring of length *n*." [21] Here the gram (which will be the composite of the substring) is a byte in hexadecimal form. Therefore, $w_{Q,i}$ is the weight of the *ith n*-gram in the query and $w_{D,i}$ is the weight of the *ith n*-gram in the document. There have been other efforts [5, 6, 8, 25, 26, 27] to use the information retrieval concept of *n*-grams as a potential for features. Henchiri et al. [8] and Abou-Assaleh et al. [5, 25] both use the Common N-Gram (CNG) analysis method, which uses the most frequent *n*-grams to represent a class, to detect malicious applications. Henchiri [6] further limits the number of features by imposing a "hierarchical feature selection process". Marceau [27] puts an interesting twist on the problem of using *n*-grams as features by having "multiple-length" grams instead of the tradition single *n*-length gram. Marceau does this by first creating and then compacting a suffix tree to a DAG [27]. Reddy et al. [8] develop their own unique *n*-gram feature selection measure called 'class-wise document frequency'. Santos et al. in [35] create *n*-gram-based signatures for malicious executables that they used to classify unknown executables using the *k*-nearest-neighbors approach.

2.2 Randomized Projections

Malicious application detection, following the genre of information retrieval, suffers from the problem that the data, once processed, is encoded in extremely high dimensions. This high-dimensional data limits the kind and amount of analysis that can be preformed. One method for dealing with the reduction of this type of high-dimensional data is known as feature extraction. Feature extraction transforms, either linearly or non-linearly, the original feature set into a reduced set that retains the most important predictive information. Examples of this type include principal component analysis, latent semantic analysis and randomized projection. In randomized projection, "the original high-dimensional data is projected onto a lower-dimensional subspace using a random matrix whose columns have unit lengths." [14] This type of projection attempts to retain the maximum amount of information embedded in the original feature set while substantially reducing the number of features required. By reducing the number of features, greater amounts of analysis can be performed. The core concept has been developed out of the Johnson-Lindenstrauss lemma [28] which states that any set of n points in a Euclidean space can be mapped to $\mathbb{R}^t$ where $t = O(\frac{\log n}{\varepsilon^2})$ with distortion $\leq 1 + \varepsilon$ in the distances. Such a mapping may be found in random polynomial time. A proof of this lemma can be found in [29].

Researchers have used randomized projection in several different applications [12, 14, 15] to reduce the dimensionality of high-dimensional data. "Randomized projection refers to the technique of projecting a set of points from a high-dimensional space to a randomly chosen low-dimensional subspace or embedding." Minnila et al. [12] are using randomized projection techniques to map sequences of events and find similarities between them. Their specific application is in the telecommunication field looking at how to better handle network alarms. Their goal is to "show the human analyst previous situations that resemble the current one" [12] so that a more informed decision about the current situation can be made. Though their proposed solution is not perfect, it does show the promise of using randomized projections in a similarity based application. Bingham et al. [14] applies randomized projections to an image and text retrieval problem. In comparison to this research problem, their dimensions are not as large (2500 for images and 5000 for text), but the results are still significant. The purpose of their work was to show that compared to other more traditional dimensionality reduction techniques, such as principal component analysis or singular value decomposition, randomized projections offered a greater detail of accuracy. The authors were also able to show a significant computation saving by using randomized projections over other feature extraction techniques, such as principal component analysis. In another text retrieval application, Kaski [10] successfully applied randomized projections in his text retrieval application that used WEBSOM, a graphical self-organizing map. Again Kaski turned to randomized projection as a method to overcome the computation expense that made other dimensionality reduction techniques infeasible when handling high-dimensional data sets. After incorporating randomized projection into their tool, the authors gained an additional 5% increase in classification and topic separation than in previous methods used [17].

The following efforts [15, 18, 19] use randomized projection in conjunction with latent semantic indexing. Papadimitriou et al. [15], looking at another information retrieval technique, show positive results in using randomized projections as a preprocessor to the computationally expensive Latent Semantic Indexing. By simply applying randomized projection to their data before computing the Latent Semantic Indexing, their asymptotic running time for the overall system improved from O(mnc) to O(m($\log^2 n$ + c log$n$)), where m and n are the matrix size, c is the average number of terms per document [15]. Varmuza et al. [20] apply randomized projection to data from the fields of chemoinformatics and chemometrics, and the effects were assessed by performing cluster analysis, classification or calibration on the resulting lower-dimensional data. The results indicated a drastic reduction in data size and computing time compared to principle component analysis, while preserving performance [20]. However, it was noted that though the randomized projection technique was very effective with data sets of large dimensionality, it showed limited performance in calibration tasks with low-dimensional data typical to chemical applications [20]. Zhang et al. [35] use randomized projection as a feature extraction and reduction technique in producing features for weed seed classification.

3. EXPERIMENT

The following provides a description of the components of the experimental methodology that was used to detect malicious applications using the information retrieval technique of n-gram analysis and the dimension reduction technique of randomized projections. All of the experiments were run on commodity hardware running either the Fedora or Ubuntu Linux operating system. It is very significant that these experiments could be completed on commodity hardware. It shows that large specialized machines are not needed to perform malicious application detection and that this work can be broadly applied across almost any level of architecture that researchers/developers may have and still gain the significantly positive results that were obtained and discussed below. In addition, this software and the methods that it supports can easily take advantage of commodity cluster hardware for substantial gains in performance. Details of the software application developed as well as a description of the data set used in the experiments are also described below. This section concludes with an overall experimental design description that provides a description of how the experiments were conducted.

3.1 Similarity Software

The software created for this experiment provides functionality to ingest Windows formatted, binary executables and creates an m-dimensional data space containing vectors that represent those applications. In these experiments, m is the number of total possible n-grams that can be extracted from the ingested applications, one dimension for each possible n-gram. The information stored in each of the dimensions can take on one of several possible values: the absolute total number of occurrences of the particular n-gram in the application, the normalized value of the total number of occurrences of the particular n-gram in the application, or finally, the binary values of '1' if the application contains the particular n-gram or '0' if it does not. Once the m-dimensional vectors have been created, we can then apply the randomized projection technique via matrix multiplication with a randomized matrix [14], or via the extended Linial-London-Rabinovich (LLR) algorithm found in [32].

In the method of randomized projection via matrix multiplication, the original d-dimensional data is projected to a k-dimensional ($k << d$) subspace through the origin, using a random $d \times k$ matrix R whose columns have unit lengths [14]. The random matrix used for the dimensionality reduction can be populated in several ways. Of these, the similarity software can use two methods: 1) by selecting vectors that are Gaussian distributed, random variables with a mean of 0 and a standard deviation of 1 or 2) by selecting vectors that take on the values of 0, +1 or -1 following a probability distribution of 2/3, 1/6 and 1/6 respectively [14]. In matrix notation, where A_{Nxd} is the original set of N d-dimensional observations, $A_{Nxk} = A_{Nxd}R_{dxk}$ is the projection of the data onto a lower k-dimensional subspace [14]. The result is a low-dimensional embedding of the original high-dimensional features. The first method, which uses a normal or Gaussian distributed random matrix, was applied in [4]. In this paper, we will utilize the second method, suggested by Achlioptas [36], which uses sparse matrices to produce a mapping that satisfies the Johnson-Lindenstrauss lemma [14]. This method also offers computational savings through the use of integer arithmetic over the Gaussian distributed random matrix method [14].

LLR randomized projection or, as we refer to it, random set projection, is based on the LLR algorithm, which is an extension of the Johnson-Lindenstrauss [28] and Bourgain [33] algorithms. It is described as follows: For each cardinality $k < n$ which is a power of 2, randomly pick O(log n) sets $A \subset V(G)$ of cardinality k. Map every vertex x to the vector (d(x, A)) (where $d(x, A) = \min\{d(x, y) | y \in A\}$) with one coordinate for each A selected [11]. In short, the algorithm randomly selects k = O(log n) subsets of the original data set, and uses the minimum distances from each vector to each subset as coordinates to create a k-dimensional vector projection. We use a variation of this algorithm found in [32] which basically allows us to select the number of features k to be a value other than log n.

For this set of experiments, the cosine similarity algorithm, shown in Eq. (1), was then applied to the query application's vector and the corpus applications' vectors for each set of reductions. This application of the algorithm produced the prediction results.

3.2 Data Set

The data set that was compiled together for the experiments described in this section consisted of 1544 Windows formatted binary executable files. None of the files in the data set were larger than 950 KB. Of these files, 303 were extracted from a fresh installation of the Windows XP operating system. Another 406 were extracted from a fresh installation of the Windows Vista operating system. Both of these sets were obtained by installing the respective operating system in a virtual environment that was installed on a commodity PC. These virtual environments were not connected to the Internet and therefore provided a safe location. This ensured that it would allow for application extraction without the worry of malicious infiltration during the gathering phase of the research effort. This process provided a total of 709 files that were in the data set and that were considered benign. The remaining 835 files for the data set were malicious Trojan horse applications that were downloaded from various websites on the Internet including http://www.trojanfrance.com and http://vx.netlux.org. A Trojan horse, similar to the myth, may provide a useful service (for example, a calculator or Notepad) but once executed performs harmful actions. Symantec reported in their bi-annual threat report for the first half of 2005, that "six of

the top ten spyware (information leakage) programs were delivered to their victim by being bundled with some other program." [30] Trojans are a very popular and effective way of infiltrating user systems. To give an idea of their prevalence, in 2009, Trojans accounted for 6 of the top 10 new malicious code families detected; 51 percent of the volume of the top 50 malicious code samples reported; four of the top 10 staged downloaders; and eight of the top 10 threat components downloaded by modular malicious software [31].

3.3 Design

This section describes the overall design of this experiment. The size of the *n*-grams was varied from a 3-byte, 5-byte and a 7-byte window. Only the binary value-weighting scheme described above was used for this effort. For the dimensionality reduction portion, the Gaussian distributed random matrix, sparse random matrix, and LLR random set randomized projection methods described above in section 3.1 were applied to the original high-dimensional data set to produce three separate new low-dimensional embeddings each, which contained 500, 1000 and 1500 features. The cosine similarity algorithm was then applied to these reduced dimensional data sets over a range of threshold values, from 0 to 1.0 in 0.05 increments, to produce prediction values. The results obtained from these experiments are presented below.

4. RESULTS

The following is a snippet of the results generated throughout these experiments, shown here as evidence that using randomized projection as a feature extraction technique has the potential to improve the cosine similarity algorithm when applied to the problem of malicious software detection.

4.1 Validation

As with any new method, technique or technology that is introduced, a system for determining its accuracy or validity must also be presented. Validation is a key component to providing feasible confidence that any new method is effective at reaching a viable solution, in this case a viable solution to the malicious application detection problem. Validation is not only comparing the results to what the expected result should be, but it is also comparing the results to other published methods.

To that end several performance values were used to measure and compare the performance of the experiments conducted in this research effort. These values include true positive rate (TPR), false positive rate (FPR), accuracy and precision. TPR, also known as recall, "is the proportion of relevant applications retrieved, measured by the ratio of the number of relevant retrieved applications to the total number of relevant applications in the data set." [22] FPR is the ratio of negative instances that were incorrectly identified. Accuracy is the ratio of the number of positive instances, either true positive or false positive, that were correct. "Precision is the proportion of retrieved applications that are relevant, measured by the ratio of the number of relevant retrieved applications to the total number of retrieved applications," [22]. All of these values are derived from information provided from the truth table. A truth table, also known as a confusion matrix, provides the actual and predicted classifications from the predictor. The following are the mathematical definitions of the performance formulas as well as the truth table (Table 1) where, a (true positive) is the number of malicious applications in the data set that were classified as malicious applications, b (false positive) is the number of benign applications in the data set that were classified as malicious applications, c (false negative) is the number of malicious applications in the data set that were classified as benign applications, and d (true negative) is the number of benign applications in the data set that were classified as benign applications [24]. Below are the formulas for the four performance calculations that were used in this research effort for validation of the predicted results.

Table 1. Definition of Truth Table

		Actual	
		Positive	**Negative**
Predicted	**Positive**	a	b
	Negative	c	d

$$TPR = \frac{b}{a + c} \quad (2)$$

$$FPR = \frac{b}{b + d} \quad (3)$$

$$Accuracy = \frac{a+d}{a+b+c+d} \quad (4)$$

$$Precision = \frac{a}{a+b} \quad (5)$$

It is important to note that most methods used in previous research, report only accuracy value ratings. However, a high accuracy rate may not tell the entire story. For example, assume that the data set contains a high number of true negatives but a low number of true positives predicted. If the value of negative instances is much greater than the number of positive instances then using the formula for Accuracy (Eq. 4) above would produce a high value and using the formula for TPR (Eq. 2) above potentially would produce a low value.

4.2 Experimental Performance

The calculated performance values described above were used to validate and show that the randomized projection method proposed in [4] added a performance increase to the malicious detection algorithm presented. The performance increase was defined in terms of absolute comparison of the validation methods. We used this same validation methodology to evaluate the randomized projection methods of sparse matrix randomized projection and LLR random set randomized projection. Note that due to space limitations we only present one sample of the results obtained from the entire breadth of experiments that were performed on this data set.

Table 2 depicts the performance values for the entire data set after sample dimensionality reduction of 1500 features, using sample n-gram values of 7, for each of the 3 aforementioned randomized projection methods: Gaussian distributed matrix randomized projection (RM1), sparse matrix randomized projection (RM2), and LLR random set randomized projection (RS). It must be noted that the non-dimensionality reduced, original data set had upwards of 10^9 features; thus the reductions presented here are significant reductions.

Table 2. Performance Values for *n*-gram size of 7 and Dimension Reduction to 1500 features

Performance Metric	Cosine Similarity Threshold								
	0.1			0.15			0.20		
	RM1	RM2	RS	RM1	RM2	RS	RM1	RM2	RS
TPR	0.95	0.78	0.54	0.99	0.84	0.54	1	0.87	0.56
FPR	0.02	0.16	0.014	0.02	0.19	0.011	0.08	0.23	0.013
Accuracy	0.956	0.80	0.74	0.99	0.83	0.75	0.96	0.83	0.76
Precision	0.98	0.85	0.98	0.98	0.84	0.98	0.93	0.82	0.98

The results for the RM1 method discussed in [4], for all dimensionality reduction sizes and various *n*-gram feature sizes are extremely positive. Each result has high accuracy, precision and TPR values, approaching 1, while maintaining a low FPR. This means that the applications that are presented to the analyst are, with a high confidence, applications that contain malicious functionality. Furthermore, because of the low FPR and high TPR an analyst will be presented for examination much fewer applications before the malicious applications are scrutinized.

On average, the sparse matrix (RM2) and LLR random set (RS) method performed marginally compared to the Gaussian distributed random matrix approach (RM1), generally having lower accuracies, precisions and TPRs, and higher FPRs. However, under certain parameter values, each of the alternative methods had moments where they were comparable to or even outperformed the RM1 method. For example, the RM2 method on average achieved TPR results within 3% of the RM1 method for *n*-grams of size 5 with a data set reduced to 1000 features. Likewise, the RS method had comparable results for precision and lower results for FPR, with *n*-grams of size 7 and dimension reduction to 1500 features.

It must be noted that the results for the entire data set without randomized projection applied acquired similar, though consistently lower, accuracies, in the range of an average of over 10% lower. Experimental results found that there was also a substantially lower TPR, up to 30% lower, lower precision and higher FPR. This accentuates the validity that by applying the randomized projection algorithm the malicious software detection algorithm has improved performance. This can be attributed to the 'curse of dimensionality' complicating the prediction method. Significant gains were also made from a computational performance standpoint. The addition of computing the matrix multiplication to acquire the reduced dimensional data set was minimal and can be improved with further refinements and taking advantage of advances in fast matrix multiplication. Furthermore, obtaining a prediction result for an individual application saw over a 100-time increase. Over a small number of predictions, the minimal time to compute the matrix was absorbed. The data space required to contain the non-reduced feature vectors was a factor of 3 greater than that required to hold the reduced data set.

5. CONCLUSIONS

In this paper we compared the performance of the Gaussian distributed random matrix randomized projection method, explored in [4], to the sparse matrix randomized projection method and the LLR random set randomized projection method, in an effort to explore new and alternative feature extraction methods in the context of malicious application detection. From the results, we can conclude that the Gaussian distributed random matrix approach outperformed the other methods with generally higher values for each observed performance metric. However, we cannot simply discount the sparse matrix and LLR random set randomized projection methods as unpromising or useless. Each algorithm showed promise in selected scenarios and also offers possible computational savings over the Gaussian distributed random matrix method. It is worthy to note that the LLR random set method seemed to only start producing significant results when dealing with *n*-grams of size 7. It is possible that the performance of this technique may increase with the size of the *n*-grams used as features; this is a hypothesis that needs to be looked into. Perhaps with further investigation and experimentation we can improve the performance and usefulness of these two methods. The results presented here along with the entire set of results gained from the experiments support the hypothesis that applying the technique of random matrix projection as a dimensionality reduction method for the cosine similarity metric has merit in determining if an application may contain a malicious application. This conclusion has been validated using traditional validation measures as well as through performance gains in both size and time derived through experimentation.

Future efforts for this research include investigating ways of improving the randomized projection techniques discussed in this paper such as the varying *n*-gram length and observing its effect on the LLR random set method, as well as exploring different implementations of both algorithms. We also plan to consider incorporating the k-nearest-neighbors algorithm with the current randomized projection and cosine similarity model in predicting malicious applications.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the U.S. Air Force, Air Force Research Laboratory under Award No. FA9550-10-1-0289. The authors would also like to thank Mr. Richard Libby, from Intel for equipment donation and Dr. Box Leangsuksun for high performance computing services.

7. REFERENCES

[1] R. Richardson, *2008 CSI Computer Crime and Security Survey*, Computer Security Institute 2008.

[2] R. Richardson, *2007 CSI Computer Crime and Security Survey*, Computer Security Institute 2007.

[3] M. T. Dlamini, J. H. P. Eloff, and M. M. Eloff, "Information security: The moving target," *Computers & Security*, vol. 28(3-4), pp. 189-198, 2009.

[4] T. Atkison, "Aiding Prediction Algorithms in Detecting High-Dimensional Malicious Applications Using a Randomized Projection Technique," in *Proceedings of the*

48th Annual ACM Southeast Regional Conference, Oxford, MS, USA, Apr. 2010.

[5] T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, "Detection of New Malicious Code Using N-grams Signatures," in *Proceedings of the 2nd Annual Conference on Privacy, Security and Trust*, New Brunswick, Canada, 2004, pp. 193 – 196.

[6] O. Henchiri and N. Japkowicz, "A Feature Selection and Evaluation Scheme for Computer Virus Detection," in *Sixth International Conference on Data Mining*, ICDM'06, 2006, pp. 891-895.

[7] J. Z. Kolter and M. A. Maloof, "Learning to Detect and Classify Malicious Executables in the Wild," *The Journal of Machine Learning Research*, vol. 7, pp. 2721-2744, 2006.

[8] D. K. S. Reddy and A. K. Pujari, "N-gram analysis for computer virus detection," *Journal in Computer Virology*, vol. 2, no. 3, pp. 231 – 239, 2006.

[9] R. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, 1961.

[10] P. Cunningham, "Dimension Reduction," University College Dublin, Belfield, Ireland, Tech Rep. UCD-CSI-2007-7, Aug. 2007.

[11] N. Linial, E. London, and Y. Rabinovich, "The geometry of graphs and some of its algorithmic applications," *Combinatorica*, vol. 15, no. 2, pp. 215-245, 1995.

[12] H. Mannila and J. K. Seppänen, "Finding similar situations in sequences of events via random pro," in *Proceedings of First SIAM International Conference on Data Mining*, Chicago, IL, Apr. 2001.

[13] T. Atkison, "Applying Randomized Projection to Aid Prediction Algorithms in Detecting High-Dimensional Rogue Applications," in *Proceedings of the 47th ACM Southeast Conference*, Clemson, SC, USA, 2009.

[14] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 245-250.

[15] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent Semantic Indexing: A Probabilistic Analysis," *Journal of Computer and System Sciences*, vol. 61, no. 2, pp. 217-235, 2000.

[16] S. Vempala, *The Random Projection Method*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 2004.

[17] S. Kaski, "Dimensionality Reduction by Random Mapping: Fast Similarity Computation for Clustering," in *The 1998 IEEE International Joint Conference on Neural Networks*, 1998.

[18] M. Kurimo, "Indexing Audio Documents by using Latent Semantic Analysis and SOM," *Kohonen Maps*, 1999, pp. 363-374.

[19] J. Lin and D. Gunopulos, "Dimensionality reduction by random projection and latent semantic indexing," in *Proceedings of the Text Mining Workshop at the 3rd SIAM International Conference on Data Mining*, 2003.

[20] K. Varmuza, P. Filzmoser, and B. Liebmann, "Random projection experiments with chemometric data," *Journal of Chemometrics*, vol. 24, no. 3-4, pp. 209-217, Mar. 2010.

[21] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Harlow, England, Addison Wesley, 1999.

[22] N. Liu, B. Zhang, J. Yan, Q. Yang, S. Yan, Z. Chen, F. Bai, and W.-Y. Ma, "Learning Similarity Measures in Non-Orthogonal Space," in *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management Washington*, D.C., USA, 2004, pp. 334 – 341.

[23] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613-620, 1975.

[24] A. Singhal, "Modern Information Retrieval: A Brief Overview," *Bulletin of the Technical Committee on Data Engineering*, vol. 24, no. 4, pp. 35-43, 2001.

[25] T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, "N-gram-based Detection of New Malicious Code," in *Proceedings of the 28th Annual International Computer Software and Applications Conference*, COMPSAC., 2004.

[26] J. O. Kephart, G. B. Sorkin, W. C. Arnold, D. M. Chess G. J. Tesauro, and S. R. White, "Biologically inspired defenses against computer viruses," in *Proceedings of the 14th International Joint Conference on Artificial Intellgence*, San Francisco, CA, 1995, pp. 985-996.

[27] C. Marceau, "Characterizing the Behavior of a Program Using Multiple-Length N-grams," in *Proceedings of the 2000 Workshop on New Security Paradigms*, Ballycotton, County Cork, Ireland, 2000.

[28] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," *Contemporary Mathematics*, vol. 26, pp. 189-206, 1984.

[29] S. Dasgupta and A. Gupta, *An elementary proof of the Johnson-Lindenstrauss Lemma*, Technical Report TR-99-006, International Computer Science Institute, Berkley, California, USA, 1999.

[30] Symantec, Symantec Internet Security Threat Report: Trends for January 05 - June 05, Sep. 2005.

[31] Symantec, Symantec Internet Security Threat Report: Trends for 2009, Apr. 2010.

[32] T. Atkison, H. Kargupta, and C. Nicholas, *Dimensionality Reduction Using a Randomized Projection Algorithm: Preliminary Results*, Technical Report TR-CS-01-11, University of Maryland, Baltimore, MD, USA, 2001.

[33] J. Bourgain, "On Lipschitz embedding of finite metric spaces in Hilbert space," Israel J. Math. 52, 46-52, 1985.

[34] I. Santos, Y. Penya, J. Devesa, P. Bringas, "N-Grams-based file signatures for malware detection," in *Proceedings of the 11th International Conference on Enter-prise Information Systems (ICEIS)*, vol. aidss. pp. 317–320, 2009.

[35] M. Zhang, C. Cai, and J. Zhu, "Sparse representation for weed seeds classification," in *2010 International Conference on Green Circuits and Systems (ICGCS)*, June 2010, pp. 626-631.

[36] D. Achlioptas, "Database-friendly random projections," in *Proceedings of ACM Symposium on the Principles of Database Systems*, 2001, pp. 274–28