Automatically Securing Machine Learning Classifiers

Project Summary

Machine learning is increasingly used in security-sensitive applications such as malware detection or network intrusion detection. Due to their intrinsic adversarial nature, these applications differ from the classical machine learning setting in which the underlying data distribution is assumed to be stationary. To the contrary, in security-sensitive applications, samples (and, thus, their distribution) can be actively manipulated by an intelligent, adaptive adversary to confound learning. This has led to a competition between the designers of learning systems and their adversaries. Paradigms from security and cryptography have been adapted to the machine learning field, to better understand the security properties of machine learning systems in adversarial settings. However, previous studies mostly had unrealistic assumptions under which attackers can modify training data, trained models or directly conduct the evasion in the feature space without generating real evasive samples.

The research objective of this proposal is to design and develop generic methods to automatically secure machine-learning classifiers in security-sensitive applications. Specifically, the PIs focus this proposal on three fundamental research tasks: (1) first, the PIs will develop a general framework simulating adversarial efforts to evade classifiers, to automatically assess the robustness of learning classifiers against motivated adversaries; (2) second, to improve the robustness of classifiers, a new unified architecture based on deep representation learning will be designed to automatically learn better features for security-sensitive applications ; (3) third, the PI team will design a novel and efficient framework to automatically fix machine-learning classifiers under evasion attacks. Given the participants rich research experience in machine learning, secure computation and evolutionary algorithms, the PI team is in a unique position to carry out this project and make these ambitious goals feasible.

Intellectual Merit: The proposed framework for automatically protecting machine-learning classifiers in security contexts is unique and novel with respect to the combination of machine learning, security computation and evaluation algorithm-based software simulation. Existing efforts have not fully modeled attackers abilities and underestimate the computational challenges when working with security-sensitive machine learning classifiers. In addition, by leveraging both evolutionary algorithms and a deep learning architecture, this proposal aims to improve the robustness of classifiers under attack. Combining rigorous simulation analysis and state-of-the-art learning technology, the proposed framework is designed to be accurate, robust, efficient and scalable.

Broader Impacts: The proposed research is innovative and crucial not only because it enhances security-sensitive applications, but also because it improves machine learning. The PI team will make the developed computational methods and tools available to the public online. The proposed algorithms and tools are expected to help applications in broader scientific domains like software engineering with security-concerns or forensic data mining with potential adversaries.

This project will also facilitate the development of novel educational tools and new courses at the undergraduate and graduate levels at the University of Virginia (UVA). The PI team is committed to, and have actively participated in, engaging minority students and under served populations in research activities to introduce them to advances in scientific research.

Automatically Securing Machine Learning Classifiers

Project Description

Yanjun Qi, David Evans, Westley Weimer
University of Virginia

[Dave's Note: *need a better title... Current: "Automatically Securing Machine Learning Classifiers" ("securing" doesn't really work here)*

- Automatically Hardening Machine Learning Classifiers Against Adversaries

- Understanding Machine Learning Classifiers in Adversarial Contexts

- Machine Learning Classifiers in the Presence of Adversaries

- Adversarial Machine Learning: Automatic Techniques for Evaluating and Strengthening Classifiers

- ??? **]**

1 Introduction and Significance of Research

At the intersection between computer security and machine learning, this proposal aims to address the computational challenges in securing machine-learning classifiers in security-sensitive contexts. This proposal will develop novel and robust algorithms built on two emerging techniques: evolution algorithm and deep feature learning. To this end, the project focuses on designing robust and generic algorithms and tools to make machine learning methods more secure under security contexts. The results will potentially lead to new and effective tools for security computation. The proposed methods will potentially also influence relevant data mining studies with concerns to be attacked.

Why Making Machine Learning Classifiers Secure is Necessary? Machine learning models are popular in security tasks such as malware detection, network intrusion detection and spam detection. From the data scientists' perspective, these models are effective since they achieve extremely high accuracies on test datasets. For example, Dahl et al. reported achieving 99.58% accuracy in classifying Win32 malware using an ensemble deep neural network with dynamic features [11]. Šrndic et al. trained a SVM-RBF model with accuracy of 99.958% in a PDF malware classification task by using structural path features [44].

However, it is important to realize that machine-learning techniques were not originally designed to withstand manipulations made by intelligent and adaptive adversaries. Unlike when machine learning is used in other fields, security tasks involve adversaries responding to the classifier. For example in malware detection, attackers may try to generate new malware samples that have different patterns to evade existing classifiers. This breaks the assumption of machine learning models that the training data and the operational data share the same data distribution, consequently decreasing the accuracy in practice.

Why Existing Efforts on Securing Machine Learning are not Sufficient? To better understand the security properties of machine learning systems in adversarial settings, paradigms from security

and cryptography have been adapted to the machine learning field [5]. Following common security protocols, machine-learning systems should use proactive protection mechanisms that anticipate and prevent the adversarial impacts. This requires (i) finding potential vulnerabilities of learning before they are exploited by the adversary; (ii) investigating the impact of the corresponding attacks (i.e., evaluating classifier security); and (iii) devising appropriate countermeasures if an attack is found to significantly degrade the classifiers' performance.

According to [5], attacks against learning algorithms in practical scenarios, can be classified, into poisoning attack (manipulation of training data) and evasion attack (exploitation of the classifier).

- Poisoning attacks contaminate the learner's training data to mislead it, i.e., specially crafted attack points are injected into the training data. For instance, web-based repositories and honeypots often collect malware examples for training, which provides an opportunity for the adversary to poison the training data.
- Evasion attacks circumvent the learned classifier During its deployment phase. For instance, spammers and hackers often attempt to evade detection by obfuscating the content of spam emails and malware code. In the evasion setting, malicious samples are modified at test time to evade detection; that is, to be misclassified as benign. No influence over the training data is assumed. A clear example of evasion is image-based spam in which the spam content is embedded within an attached image to evade the textual analysis performed by anti-spam filters. Another example of evasion is given by spoofing attacks against biometric verification systems.

This proposal focuses on evasion attacks since they are encountered more frequently in adversarial settings during system operation. Though evasion attacks have been partially explored by classifier authors as well as third-parties, most of them significantly under-estimate the attackers' ability in manipulating samples. They often mistakenly assume the attackers can only insert new contents because removing existing contents would easily corrupt maliciousness [44, 22, 8]. In addition, previous works are ad hoc and only work on particular target classifiers or specific types of samples [45, 22]. Other than suggesting point solutions, they do not provide any way to automatically evaluate and improve the effectiveness of a classifier against an adversary.

In this proposal, we focus on investigating classification algorithms under evasion attacks in which the adversary aims to avoid detection by manipulating malicious test samples. We seek to design and apply principled algorithms to systematically assess classifier security in attack scenarios. Our system aims to allows a classifier designer to understand how the classification performance of each considered model degrades under attack, and thus, to make more informed design choices.

Task I: Automatically Evading Classifiers.

Machine learning is widely used to develop classifiers for security tasks. However, the robustness of these methods against motivated adversaries is uncertain. In this task, we propose a generic method to evaluate the robustness of classifiers under attack. The key idea is to stochastically manipulate a malicious sample to find a variant that preserves the malicious behavior but is classified as benign by the classifier. Using population-based sample search algorithms, the

proposed system is not coupled to a specific classifier and could search for evasive variants in many security applications.

Task II: Learning Robust Features Automatically. It is well understood that the performance of machine learning methods is heavily dependent on the choice of data representation (or features) on which they are applied. The preliminary results obtained from "Task-I" has raised serious doubts about the effectiveness of classifiers based on superficial features in the presence of adversaries. In this task, we propose a new machine-learning architecture, especially based on deep representation learning technique to learn deeper features to build classifiers that are resistant to evasion attempts by adversaries. Such features will depend on higher-level semantic analysis of the input file, in ways that are difficult to change without disrupting the malicious behavior.

[Dave's Note: ******** *Dave, 2 to 3 sentences about the static and dynamic labels ??* ********]

Task III: Automatically Fixing Classifiers. The ultimate goal of our proposal is to allow a classifier designer to understand and improve the classification performance under attack. To speed-up the search of evasive samples, we propose to develop novel search algorithms driven by probability modeling. This should help to learn which operations are more effective for generating evasive variants to direct the search more efficiently.

[West's Note: ******** *wes, 2 to 3 sentences about how to speed up evolaution algorithm to quickly find evasive samples...***]**

Tool development, dissemination, and evaluation. We will evaluate and package the proposed methods into a data mining toolkit and release it using GNU General Public License for wide dissemination. Our new framework can be widely applied to solve many other scientific machine learning problems with adversary concerns.

Intellectual Merit and Potential: **[Dave's Note:** ******** *Dave, anything to add here?* ********]

Broader Impacts: We anticipate that the proposed research will help create an exciting interdisciplinary environment for graduate and undergraduate students. As part of this proposal, the PIs will develop an interdisciplinary course "Secure Machine Learning and Data Mining": for beginning graduate students and senior undergraduate students from Computer Science. The PI will also contribute a number of small project topics to UVA CS senior capstone projects, and attract female and under-represented minority students to explore advanced data mining technologies. Interested, senior undergraduate students will be assigned with designed project topics and to work with the PIs for finishing their undergraduate theses. Currently the leard PI is mentoring two undergraduates (both female), **[Dave's Note:** *co-PI Evans mentoring XXXX***]** and **[West's Note:** *co-PI Weimer is mentoring XXX***]**). The REU supplement will be submitted immediately if the proposal is selected. The proposed work will also be suitable for generating summer research projects for undergraduate students in science and engineering. The results will be disseminated: (1) by publications in leading journals and top conferences; (2) by making materials and tools available on the project website for timely public access; and (3) by teaching, research seminars, and public forums such as workshops and tutorials on related areas.

PIs' Qualification : The proposed project builds on the PIs's expertise from statistical machine learning, secure computation and evolution algorithm. Throughout the lead PI's PhD (SCS@CMU),

she has worked on biomedical data mining through information integration of heterogeneous sources [26, 36, 32, 35, 46, 25, 28, 34, 29, 33]. Afterwards during the five years of industrial research in the machine learning department at NEC Labs America, the lead PI has focused primarily on deep-learning based data modeling for better feature representations. She has innovated and published a number of multi-task learning, semi-supervised learning, sparse learning and deep learning studies under this framework [15, 31, 48, 23, 7, 16, 37, 27, 6, 10, 24, 3, 9, 19, 18, 38, 39, 2, 4, 30, 35].

[Dave's Note: ******** *Dave, anything to add here?* ********]

[West's Note: ******** *Wes, anything to add here?* ********]

The University of Virginia engineering school has a particularly strong focus on "secure computation" and relevant studies [**?**]. Thus, the research project outlined here will allow the PIs to build stronger local collaborations with groups from UVA school of ??? . The PIs group is well-positioned to push this effort forward by developing secure machine learning algorithms and open-source software development.

2 Background

In this section, we present basic backgrounds of machine learning, evasion attack model and evolution algorithms.

2.1 Machine Learning Classifiers

Machine learning learns from and makes predictions on data. A machine learning-based classification algorithm attempts to find a hypothesis function f that maps data points into different classes. For example, a malware classification system would find a hypothesis function f that maps a data point (a piece of malware sample) into either *benign* or *malicious*.

The effort to train a machine learning system starts with feature extraction. As most machine learning algorithms cannot operate on highly structured data, the data samples are usually represented in a specially designed feature space. For example, a malware classifier may extract the file size and the function call traces as features. Each feature is a dimension in the feature space, consequently every sample is represented as a vector. An extra step of feature selection may be performed to reduce the number of features when the number of features is too large for the classification algorithm.

The most widely used machine learning algorithms in security tasks are *supervised learning*, in which the training dataset comes with labels identifying the class of every training sample. The hypothesis function f is trained to minimize the prediction error on the training set. This function usually results in a low error rate on the operational data under the *stationarity* assumption that the distribution over data points encountered in the future will be the same as the distribution over the training set.

Machine learning has provided promising results and is widely deployed for specific security tasks including malware classification. Without examining the behavior of suspicious malware in a real system, it often employs static properties to predict the maliciousness, for example the file structure, file size, metadata, grams of tokens or system calls. Although this approach often

achieves high accuracy in validation, the classifier may learn properties that are superficial artifacts of the training data, rather than properties that are inherently associated with malware. This is because malware samples in the training data are likely to differ from the benign samples in many ways that are not essential to their malicious behavior.

2.2 Evasion Attacker

We assume an attacker starts with a desired malicious sample that is (correctly) classified by a target classifier as malicious, and wants to create a sample with the same malicious behavior, but that is (mis-)classified as benign. The attacker is capable of manipulating the malicious sample in many ways, and is likely to have knowledge of samples that are (correctly) classified as benign.

We assume the attacker has black-box access to the target classifier, and can submit many variants to that classifier. For each submitted variant, the attacker learns its classification score. The classification score is a number (typically a real number between 0 and 1) that indicates the classifier's prediction of maliciousness, where values above some threshold (say 0.5) are considered malicious and samples with lower classification scores are considered benign. We do not assume the attacker has any internal information about the classifier, only that it can use it as a black-box that outputs the classification score for an input sample.

2.3 Evolutionary Algorithm

[West's Note: ******** *Wes, anything to add here?* ********]

Genetic programming (GP) is a type of evolutionary algorithm, originally developed for automatically generating computer programs tailored to a particular task [13, 17]. It is essentially a stochastic search method using computational analogs of biological mutation and crossover to generate variants, and modeling Darwinian selection using a user-defined *fitness function*. Variants with higher fitness are selected for continued evolution, and the process continues over multiple generations until a variant with desired properties is found. Genetic programming has been shown to be effective in many tasks including fixing legacy software bugs [21], software reverse engineering [14], and software re-engineering [40].

3 Task I: Automatically Evading Classifiers

We propose a generic method to assess the robustness of a classifier by simulating attackers' efforts to evade that classifier. Drawing ideas from *genetic programming* (GP) [13, 17], we do not assume the attacker has any detailed knowledge of the classifier or the features it uses, or can use targeted expert knowledge to manually direct the search for an evasive sample. Instead, the attacker treats the classifier as a black box. We perform stochastic manipulations to the structure of samples, producing variant samples that may or may not be malicious and may or may not evade the classifier. We then evaluate these generated variants, selecting and retaining the most promising individuals. By repeating this procedure iteratively, we generate variants that are effective but avoid detection. If such variants can be generated, they serve as witnesses to a lack of robustness in the original classifier.

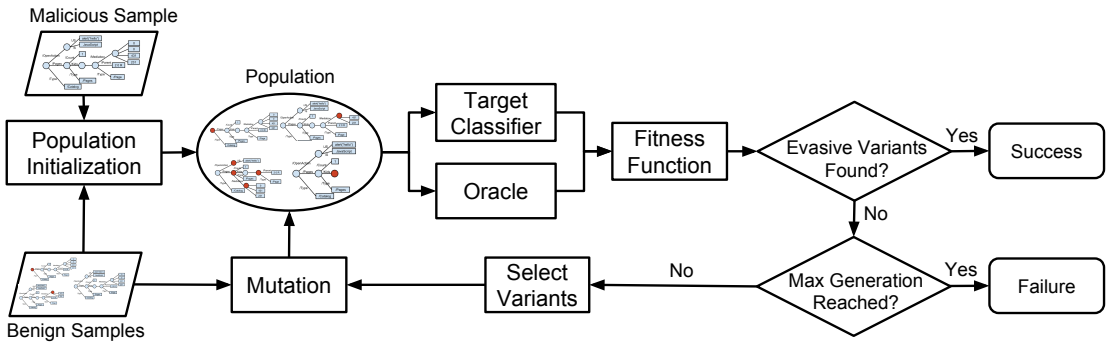

Figure 1: Generic classifier evasion method.

3.1 Previous Work:

Though evasion attacks have been partially explored by classifier authors as well as third-parties, most of them significantly under-estimate the attackers' ability in manipulating samples. They often mistakenly assume the attackers can only insert new contents because removing existing contents would easily corrupt maliciousness [44, 22, 8]. In addition, previous works are ad hoc and only apply to particular target classifiers or specific types of samples [45, 22]. Other than suggesting point solutions, they do not provide any way to automatically evaluate the effectiveness of a classifier against an adversary.

3.2 Preliminary Work:

Our procedure is illustrated in Figure 1. We start with a seed sample that exhibits malicious behavior, and is classified as malicious by the target classifier. Our method aims to find an evasive sample that preserves the malicious behavior but is misclassified as benign by the target classifier.

First, we initialize a population of variants by performing random manipulations on the malicious seed. Subsequently, the population is evaluated by a target classifier as well as an oracle. The target classifier is a black box that outputs a number that is a measure of predicted maliciousness of an input sample. There is some threshold used to decide if it is malicious or benign. The *oracle* is used to determine if a given sample exhibits particular malicious behavior.

If a variant is classified as benign by the target classifier, but found to be malicious by the oracle, it is an evasive sample. If no evasive samples are found, a subset of the generated variants are selected for the next generation based on a fitness measure designed to reflect progress towards finding an evasive sample. Since it is unlikely that the transformations will re-introduce malicious behaviors into a variant, corrupted variants that have lost the malicious behavior are replaced with other variants or the original seed.

Next, the selected variants are randomly manipulated by mutation operators to produce next generation of the population. The process continues until an evasive sample is found or a threshold number of generations is reached.

To improve the efficiency of the search, we collect traces of the mutation operations used and reuse

effective traces. If a search ends up finding any evasive variants, the mutation traces on the evasive variants will be stored as successful traces. Otherwise, the mutation trace of a variant with the highest fitness score is stored. These traces are then applied to other malware seeds to generate variants for their population initialization. Because of the structure of PDFs and the nature of the mutation operators, the same sequence of mutations can often be applied effectively to many initial seeds.

In preliminary work, we constructed variant of PDF malware samples and thus demonstrated a lack of robustness in existing PDF malware classifiers. Our prototype system thus automatically evades structural feature-based PDF malware classifiers. This involves GP operators which perform manipulations on PDF files, an oracle that determines if a generated variant preserves maliciousness, a selection mechanism that promotes promising variants during the evolutionary process, and a fitness function for each target classifier. This approach does not rely on any specific classification algorithms or assume detailed knowledge of feature extraction, but only needs the classification score feedback on generated variants (i.e., the ability to run the classifier as a black box) and a rough knowledge on the feature set.

We evaluated the proposed method on two PDF malware classifiers, PDFrate [43] and Hidost [44], and found that it could automatically find evasive variants for all 500 samples from the Contagio PDF malware archive [1]. The evasive malware exhibits the same malicious behaviors as the original ones, but has sufficiently different patterns in feature spaces to be classified as benign by the machine learning-based models.

A sophisticated attacker, of course, can perform manipulations that would not be found efficiently by a stochastic search, so we cannot claim that a classifier that resists such an attack is necessarily robust. On the other hand, if the automated approach finds evasive samples for a given classifier, it is a clear sign that the classifier is not robust in an adversarial environment.

Our analysis of the discovered evasive variants reveals that both classifiers are vulnerable because they employ *non-robust features*, which can be manipulated without disrupting the desired malicious behavior. Superficial features may work well on test datasets, but if the features used to classify malware are shallow artifacts of the training data rather than intrinsic properties of malicious content, it is possible to find ways to preserve the malicious behavior while disrupting the features.

An observed 100% success rate in finding evasive variants against both classifiers in a large experiment demonstrates the promise of our approach.

3.3 Proposed Work:

We propose a general framework to produce evasive samples and thus assess the robustness of classifiers. Our approach takes as input a classifier, a number of existing malicious samples, and a structural modeling of samples that is loosely related to the feature space of the classifier.

FIXME: Wes thinks we should have an architecture diagram here.

Our algorithm is iterative, repeatedly generating and considering variant samples. Given a sample, we first convert it to a structural representation. For example, this might be the HTML parse

tree for a webpage or the COS Object Tree for a PDF file. We then apply mutation operators, producing a new structural representation. These mutation operators may change the meaning of the sample, but need not. For example, an operator might change `<b><u>text</u></b>` to `<u><b>text</u></b>` in an HTML sample, which might defeat certain naive regular expression classification features. Once the mutations have been applied, we convert the structural representation back to a reified sample (e.g., a binary PDF file). We then evaluate the suitability of that variant sample in two ways. First, we determine if the sample is still effectively malicious. We do this by FIXME-Dave-Jane. If the sample is malicious, we then determine if it is evasive. We do this by using the input classifier as a black box. Unsuitable variants, such as those that are neither malicious nor evasive, are discarded. Suitable variants are retained in the population, and may be combined to form new variants. This combination takes place at the level of structural mutations: for example, a variant produced by applying mutation m_1 to the original and a variant produced by applying mutations m_2 and m_3 to the original might produce a variant containing mutations m_1, m_2 and m_3. This iterative process repeats until either enough malicious, evasive variants have been discovered or until a search budget is exhausted.

To be useful in an adversarial environment or security setting, such a framework must have three critical properties. First, it must effectively explore the search space of candidate variants and have the potential to produce malicious, evasive samples. Second, it must be efficient, admitting the rapid and parallelizable evaluation of candidate variants. Third, it must be general, making minimal assumptions about the amount of information available from conventional classifiers. We three research thrusts to directly improve our prototype algorithm along these dimensions.

- Design and evaluate new security- and classifier-centric structural representations and mutation operators. In previous work, we demonstrated that design decisions related to representations and operators [20] as well as the evaluation functions [12] are crucial to search efficacy of evolutionary algorithms applied in a software context.
- Design and evaluate new iterative algorithms that trade off exploration, exploitation and search efficiency. In previous work, for non-security contexts, we have adapted both steady-state algorithms [42] and island-based genetic algorithms [41] to meet such needs.
- Design and evaluate new evaluation functions and selection strategies that operate when the input classifier gives a binary answer rather than a real-valued detection confidence. FIXME-Dave. In previous work we have used prioritization, on-line learning and static analyses to tackle the problem of binary evaluation functions [47], and we propose to investigate similar strategies in this domain.

3.3.1 Structure, representation, and operators.

3.3.2 Population-based search.

3.3.3 Evaluating candidate samples.

4 Task II: Automatically Learning Better Features

The rapidly developing field of representation learning is concerned with questions surrounding how we can best learn meaningful and useful representations of data.

The performance of machine learning methods is heavily dependent on the choice of data representation (or features) on which they are applied. For that reason, much of the actual effort in deploying machine learning algorithms goes into the design of preprocessing pipelines and data transformations that result in a representation of the data that can support effective machine learning. Such feature engineering is important but labor-intensive and highlights the weakness of current learning algorithms: their inability to extract and organize the discriminating information from the data.

4.1 Previous Work:

4.2 Preliminary Work:

4.3 Proposed Work:

5 Task III: Automatically Fixing Classifiers

5.1 Previous Work:

5.2 Preliminary Work:

5.3 Proposed Work:

6 Evaluation

7 Education, Outreach, and Transfer Plans

The proposer has a strong commitment to education, outreach, and to making useful tools available to the research community, government, and industry.

Graduate and undergraduate student research. The proposers views mentoring research students as among

Student and professional development.

Development

All the PIs have been heavily involved in curriculum development and developing influential

courses and textbooks.

Public Outreach

Tool Distribution. Results from this work will be released as open source tools,

8 Timetable and Impact Summary

Our work is driven by ....

The major milestones for the project are summarized below.

Year 1

-

Year 2

- Evaluate the

Year 3

- Develop

9 Results from Prior NSF Support

10 Budget Justification

Funding is requested to support the work

The requested budget includes:

-
-
-
- Travel support to cover the expenses for the PIs and research assistants to attend relevant meetings.

References Cited

[1] http://contagiodump.blogspot.de/2010/08/malicious-documents-archive-for.html.

[2] B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, O. Chapelle, and K. Weinberger. Supervised semantic indexing. In *Proceeding of the 18th ACM conference on Information and knowledge management (CIKM)*, pages 187–196. ACM, 2009.
Acceptance rate = 15% (123/847).

[3] B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, O. Chapelle, and K. Weinberger. Learning to rank with (a lot of) word features. *Information retrieval*, 13(3):291–314, 2010.

[4] B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, C. Cortes, and M. Mohri. Polynomial semantic indexing. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 64–72, 2009.
Acceptance rate = 23% (263/1105).

[5] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D Joseph, and J Doug Tygar. Can machine learning be secure? In *ASIACCS*. ACM, 2006.

[6] D. Bespalov, B. Bai, Y. Qi, and A. Shokoufandeh. Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 375–382. ACM, 2011.
Acceptance rate = 15% (137/917); [CS].

[7] D. Bespalov, Y. Qi, B. Bai, and A. Shokoufandeh. Sentiment classification with supervised sequence encoder. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2012.
Acceptance rate = 23% (/443); [PS].

[8] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *ECML-KDD*. 2013.

[9] X. Chen, B. Bai, Y. Qi, Q. Lin, and J. Carbonell. Learning preferences with millions of parameters by enforcing sparsity. In *Proceedings of IEEE International Conference on Data Mining (ICDM)*, pages 779–784, 2010.
Acceptance rate = 19% (155/797); [PS].

[10] X. Chen, Y. Qi, B. Bai, Q. Lin, and J.G. Carbonell. Sparse latent semantic analysis. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2011.
Acceptance rate = 25% (86/343); [CS].

[11] George E Dahl, Jack W Stokes, Li Deng, and Dong Yu. Large-scale malware classification using random projections and neural networks. In *ICASSP*, 2013.

[12] Ethan Fast, Claire Le Goues, Stephanie Forrest, and Westley Weimer. Designing better fitness functions for automated program repair. In *Genetic and Evolutionary Computation Conference*, pages 965–972, 2010.

[13] Stephanie Forrest. Genetic algorithms: principles of natural selection applied to computation. *Science*, 261(5123):872–878, 1993.

[14] Mark Harman, William B Langdon, and Westley Weimer. Genetic programming for reverse engineering. In *WCRE*. IEEE, 2013.

[15] Y. He, K. Kavukcuoglu, Y. Wang, A. Szlam, and Y. Qi. Unsupervised feature learning by deep sparse coding. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2014.
Acceptance rate = 15% (60/389); [PS].

[16] Y. He, Y. Qi, K. Kavukcuoglu, and H. Park. Learning the dependency structure of latent factors. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2012.
Acceptance rate = 25% (370/1467); [PS].

[17] John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.

[18] P. Kuksa and Y. Qi. Semi-supervised bio-named entity recognition with word-codebook learning. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2010.
Acceptance rate = 23% (82/351); [PS].

[19] P. Kuksa, Y. Qi, B. Bai, R. Collobert, J. Weston, V. Pavlovic, and X. Ning. Semi-supervised abstraction-augmented string kernel for multi-level bio-relation extraction. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2010.
Acceptance rate = 17% (110/658); [PS].

[20] Claire Le Goues, Stephanie Forrest, and Westley Weimer. Representations and operators for improving evolutionary software repair. In *Genetic and Evoluationary Computation Conference*, pages 959–966, 2012.

[21] Claire Le Goues, ThanhVu Nguyen, Stephanie Forrest, and Westley Weimer. Genprog: A generic method for automatic software repair. *IEEE Trans. on Software Engineering*, 2012.

[22] Davide Maiorca, Igino Corona, and Giorgio Giacinto. Looking at the bag is not enough to find the bomb: an evasion of structural methods for malicious pdf files detection. In *ASIACCS*, 2013.

[23] R. Min and Y. Qi. Sparse higher-order markov random field, Jun 2013. US Patent App. 13/908,715.

[24] X. Ning and Y. Qi. Semi-supervised convolution graph kernels for relation extraction. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2011.
Acceptance rate = 25% (86/343); [PS].

[25] Y. Qi. *Learning of Protein-Protein Interaction Networks*. PhD thesis, Carnegie Mellon University, May 2008.

[26] Y. Qi. Random forest for bioinformatics. In C. Zhang and Y. Ma, editors, *Ensemble Machine Learning: Methods and Applications*, page 307. Springer, 2012.

[27] Y. Qi, B. Bai, X. Ning, and P. Kuksa. Systems and methods for semi-supervised relationship extraction, Apr 2014. US Patent 8,874,432; Type: Grant.

[28] Y. Qi, F. Balem, C. Faloutsos, J. Klein-Seetharaman, and Z. Bar-Joseph. Protein complex identification by supervised graph local clustering. *Bioinformatics*, 24(13):i250, 2008. Impact Factor 5.468.

[29] Y. Qi, Z. Bar-Joseph, and J. Klein-Seetharaman. Evaluation of different biological data and computational classification methods for use in protein interaction prediction. *Proteins: Structure, Function, and Bioinformatics*, 63(3):490–500, 2006. Impact Factor 3.39.

[30] Y. Qi, R. Collobert, P. Kuksa, K. Kavukcuoglu, and J. Weston. Combining labeled and unlabeled data with word-class distribution learning. In *Proceeding of the 18th ACM conference on Information and knowledge management (CIKM)*, pages 1737–1740. ACM, 2009.
Acceptance rate = 20% (294/847).

[31] Y. Qi, S. Das, J. Weston, and R. Collobort. A deep learning framework for character-based information extraction. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, 2014. [PS].

[32] Y. Qi, H.K. Dhiman, N. Bhola, I. Budyak, S. Kar, D. Man, A. Dutta, K. Tirupula, B.I. Carr, J. Grandis, et al. Systematic prediction of human membrane receptor interactions. *Proteomics*, 9(23):5243–5255, 2009. Impact Factor 5.479.

[33] Y. Qi, J. Klein-Seetharaman, and Z. Bar-Joseph. Random forest similarity for protein-protein interaction prediction from multiple sources. In *Proceedings of Pacific Symposium on Biocomputing*, volume 10, pages 531–542, 2005.

[34] Y. Qi, J. Klein-Seetharaman, and Z. Bar-Joseph. A mixture of feature experts approach for protein-protein interaction prediction. *BMC bioinformatics*, 8(Suppl 10):S6, 2007. Impact Factor 3.78.

[35] Y. Qi, P. Kuksa, R. Collobert, K. Sadamasa, K. Kavukcuoglu, and J. Weston. Semi-supervised sequence labeling with self-learned features. In *Proceedings of the 9th IEEE International Conference on Data Mining (ICDM)*, pages 428–437, 2009.
Acceptance rate = 9% (70/786).

[36] Y. Qi and W. Noble. Protein interaction networks: Protein domain interaction and protein function prediction. In H. Lu, B. Schlkopf, and H. Zhao, editors, *Handbook of Computational Statistics: Statistical Bioinformatics*. Springer, 2011.

[37] Y. Qi, M. Oja, J. Weston, and W.S. Noble. A unified multitask architecture for predicting local protein properties. *PLoS ONE*, 7(3):e32235, 2012. Impact Factor 4.411; [PS].

[38] Y. Qi, O. Tastan, J.G. Carbonell, J. Klein-Seetharaman, and J. Weston. Semi-supervised multitask learning for predicting interactions between HIV-1 and human proteins. *Bioinformatics*, 26(18):i645, 2010. Impact Factor 5.468.

[39] Y. Qi, O. Tastan, J.G. Carbonell, J. Klein-Seetharaman, and J. Weston. Semi-supervised multi-task learning for predicting interactions between HIV-1 and human proteins. In *Proceedings of the 9th European Conference on Computational Biology (ECCB)*, page i645, 2010. Acceptance rate = 17% (36/215).

[40] Conor Ryan. *Automatic re-engineering of software using genetic programming*, volume 2. Springer Science & Business Media, 2012.

[41] Eric Schulte, Jonathan DiLorenzo, Stephanie Forrest, and Westley Weimer. Automated repair of binary and assembly programs for cooperating embedded devices. In *Architectural Support for Programming Languages and Operating Systems*, 2013.

[42] Eric Schulte, Jonathan Dorn, Stephen Harding, Stephanie Forrest, and Westley Weimer. Post-compiler software optimization for reducing energy. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2014.

[43] Charles Smutz and Angelos Stavrou. Malicious pdf detection using metadata and structural features. In *Available at http://cs.gmu.edu,*. Department of Computer Science, George Mason University, Tech.Rep., 2012.

[44] Nedim Šrndic and Pavel Laskov. Detection of malicious pdf files based on hierarchical document structure. In *NDSS*, 2013.

[45] Nedim Šrndic and Pavel Laskov. Practical evasion of a learning-based classifier: A case study. In *Oakland*, 2014.

[46] O. Tastan, Y. Qi, J.G. Carbonell, and J. Klein-Seetharaman. Prediction of interactions between HIV-1 and human proteins by information integration. In *Proceedings of Pacific Symposium on Biocomputing*, volume 516, 2009.

[47] Westley Weimer, Zachary P Fry, and Stephanie Forrest. Leveraging program equivalence for adaptive program repair: Models and first results. In *Automated Software Engineering*, pages 356–366, 2013.

[48] F. Xiong, M. Kam, L. Hrebien, and Y. Qi. Ranking with distance metric learning for biomedical severity detection. In *Proceedings of SIAM International Conference on Data Mining (SDM), 3rd Workshop on Data Mining for Medicine and Healthcare (DMMH)*, 2014. [PS].