CS259D: Data Mining for Cybersecurity

Problem

- Diverse network environments
- Dynamic attack landscape
- Adversarial environment
- IDS performance strongly depends on chosen classifier
  - Perform differently in different environments
  - No Free Lunch Theorem

Solution: Multiple Classifier Systems

- Combine outputs of several IDSs
  - Example: Majority voting
- Adapt to dynamic adversarial environments

Adaptive Intrusion Detection System

- Base classifiers:
  - NaïveBayes
  - BayesNetwork
  - Decision Stump
  - RBFNetwork
- Supervised Framework:
  - Combine results of base IDSs
  - Receive the true label of the current sample
  - Measure losses between IDS outputs and true label
  - Maintain weights for base IDSs
- Fusion steps:
  - Loss update
  - Mixing update

Adaptive Intrusion Detection System

- T: number of time instances
- n: number of base IDSs
- At time $1 \leq t \leq T$:
  - IDS outputs: $X_t = (x_{t,1}, x_{t,2}, \ldots, x_{t,n})$
    - $x_{t,i} = 0$ (normal) or 1 (attack) ($1 \leq i \leq n$)
  - Ensemble's prediction: pred(t)
  - True label: y_t (0 or 1)
  - Loss of i-th IDS: $L_{t,i} = (y_t - x_{t,i})^2$
  - Weight vector: $vt = (v_{t,1}, v_{t,2}, \ldots, v_{t,n})$
    - Weights are non-negative, sum up to 1

Adaptive Intrusion Detection System

- Parameters: $\eta > 0, 0 \leq \alpha \leq 1$
- Initialization: $v_1 = v_0^m = (1/n, \ldots, 1/n)$
- At time $1 \leq t \leq T$:
  - Prediction:
    - Compute inner product: $z_t = (v_t, x_t)$
    - Pred(t) = 0, if $0 \leq z_t \leq 0.5$
    - Pred(t) = 1, if $0.5 < z_t$
  - Loss update:
    - Scale weight of each IDS i by $\exp(-\eta L_{t,i})$
    - Compute v_t^m by normalizing scaled weights
  - Mixing update:
    - Compute av_t as average of past vectors v_t^m
    - Compute $v_{t+1} = \alpha * av_t + (1-\alpha) * v_t^m$

Adaptive Intrusion Detection System

- Loss update keeps ensemble competitive with the best base IDS
  - Issue: Hard to recover if an IDS temporarily performs poorly and then performs well
    - Slow adaptation to changes in IDS performances
    - Vulnerable to adversarial changes in the attack pattern
- Mixing update
  - Keeps once-good IDSs around for quick recovery

Experiment: Data Sets

- Dataset 1:
  - Web queries
  - 50,000 samples, 20% attacks
  - Attacks: XSS, SQL Injection, Path Traversal, Command Execution, etc.
- Dataset 2:
  - Traffic targeted to a realistic e-commerce web app
  - 61K requests; 36K normal, 25K abnormal
    - Attacks: SQL Injection, buffer overflow, XSS, etc.

Experiment: Features

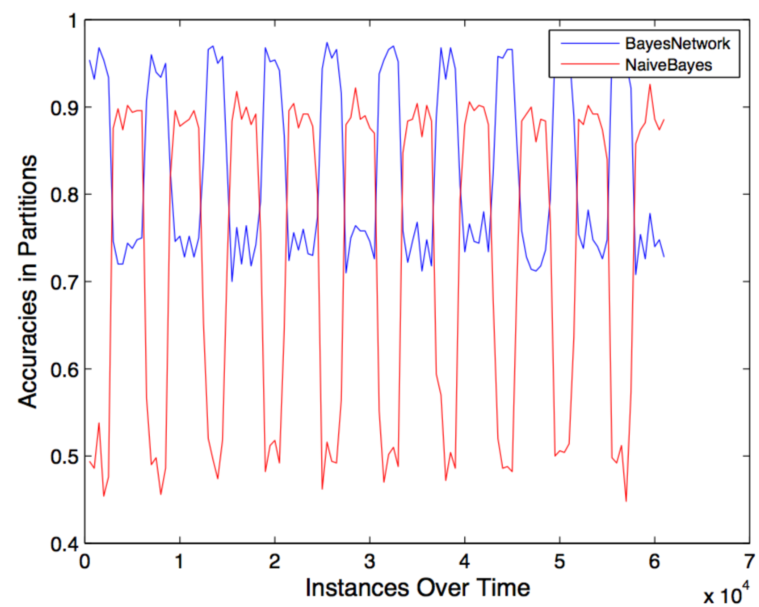
- 30 features
- Length of the request, path, headers
  - Example: buffer overflow
- Four types of characters:
  - Letters
  - Digits
  - Special characters: non-alphanumeric characters with special meanings in programming languages
  - Others
- Entropy of the bytes in the request
- Programming language keywords

Experiment: Features

Feature Name	Feature Name
Length of the request	Length of the path
Length of the arguments	Length of the header “Accept”
Length of the header “Accept-Encoding”	Length of the header “Accept-Charset”
Length of the header “Accept-Language”	Length of the header “Cookie”
Length of the header “Content-Length”	Length of the header “Content-Type”
Length of the Host	Length of the header “Referer”
Length of the header “User-Agent”	Method identifier
Number of arguments	Number of letters in the arguments
Number of digits in the arguments	Number of ’special’ char in the arguments
Number of other char in the arguments	Number of letters char in the path
Number of digits in the path	Number of ’special’ char in the path
Number of other char in path	Number of cookies
Minimum byte value in the request	Maximum byte value in the request
Number of distinct bytes	Entropy
Number of keywords in the path	Number of keywords in the arguments

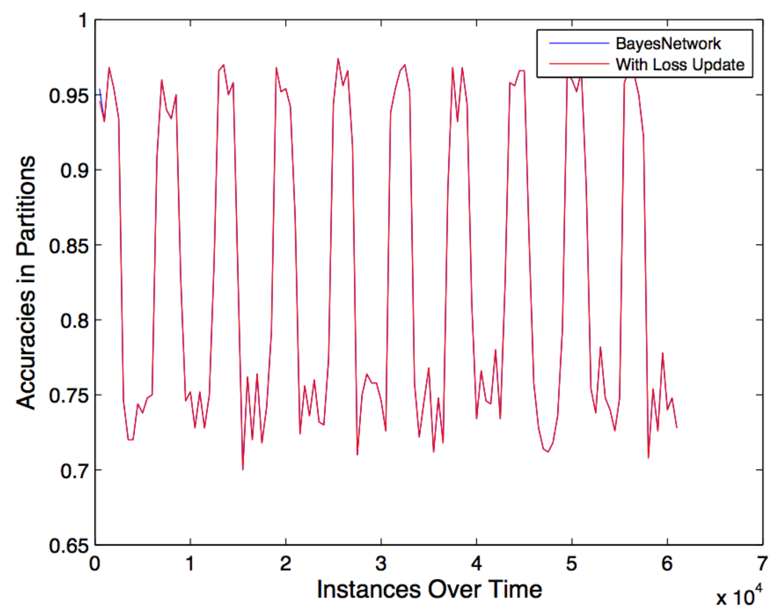
Experiment: Expert Setting

- NaïveBayes
- BayesNetwork
- Decision Stump
- RBFNetwork
- 10-fold cross-validation



Experiment: Loss Update

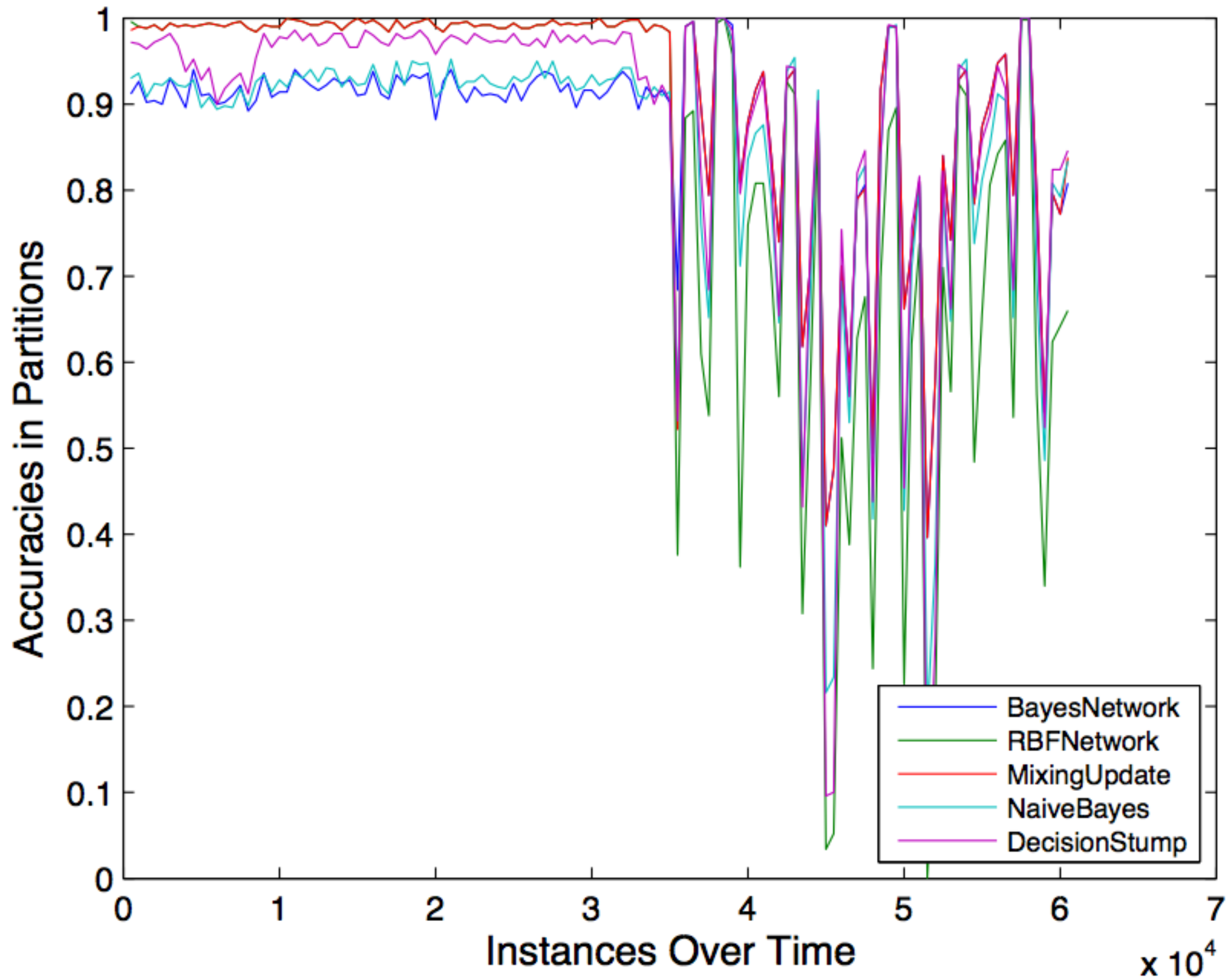
- η =0.1
- No Mixing Update
  - $v_{t+1,i} = v_{t,i}{}^m$
- Performs like the best base IDS
  - Does not adapt to varying IDS performances



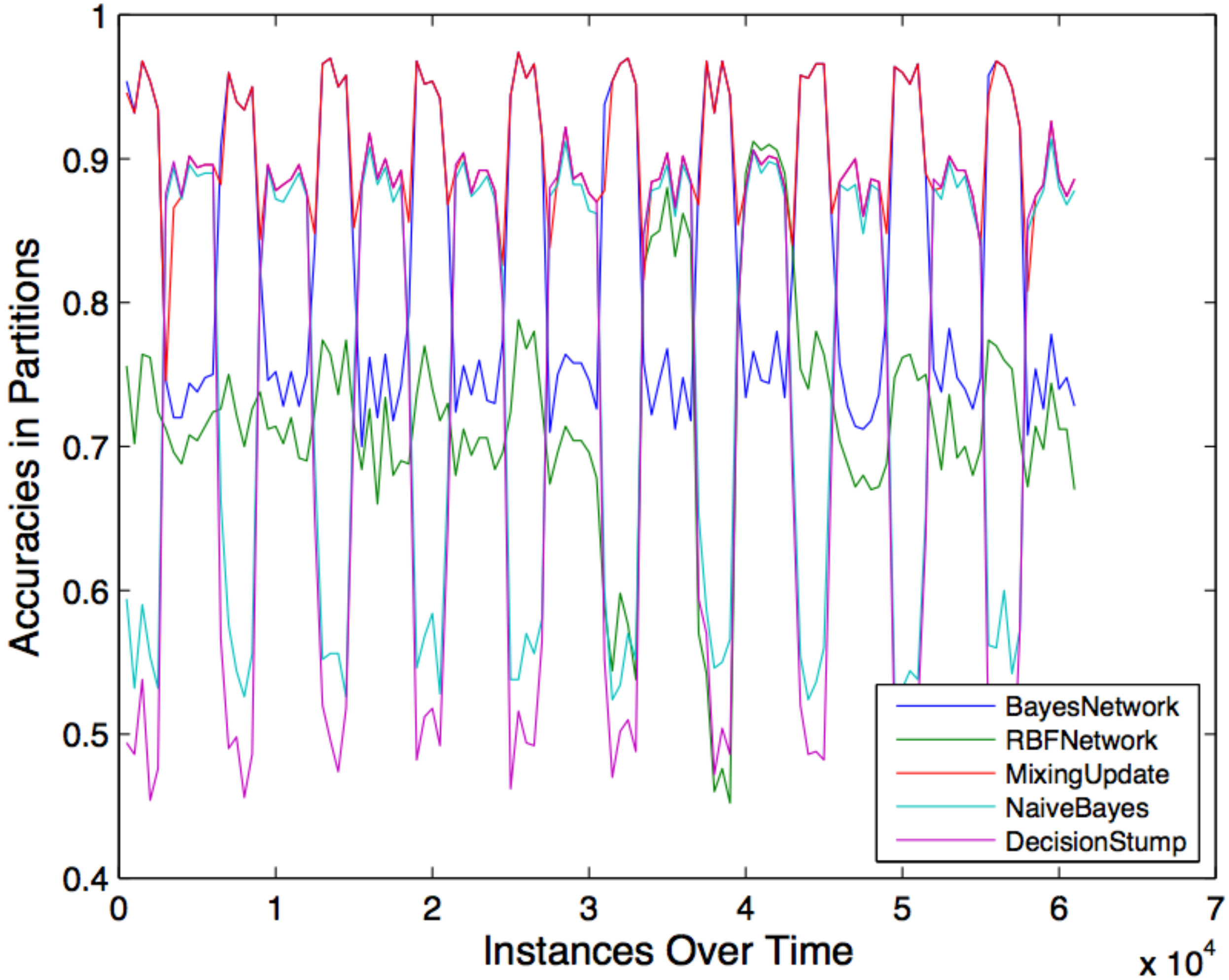
Experiment: Mixing Update

- Simulate adversarial environment
  - Randomly permute data 10 times
- Run each base IDS & Adaptive IDS on each permutation
  - η =0.1, α =0.001
- Use 10-fold cross-validation

Experiment: Mixing Update



Experiment: Mixing Update



Experiment: Accuracies

Algorithm	Dataset1	Dataset2
NaiveBayes	85.12± 0.03	72.78± 0.01
BayesNetwork	86.95± 0.025	**82.79± 0.03**
Decision Stump	84.27± 0.07	74.73± 0.05
RBFNetwork	**87.69± 0.04**	72.46± 0.01
Majority Voting	83	81
Hedge/Boosting	86.3± 0.05	82.1± 0.04
Adaptive IDS	**91.27± 0.01**	**90.52± 0.06**

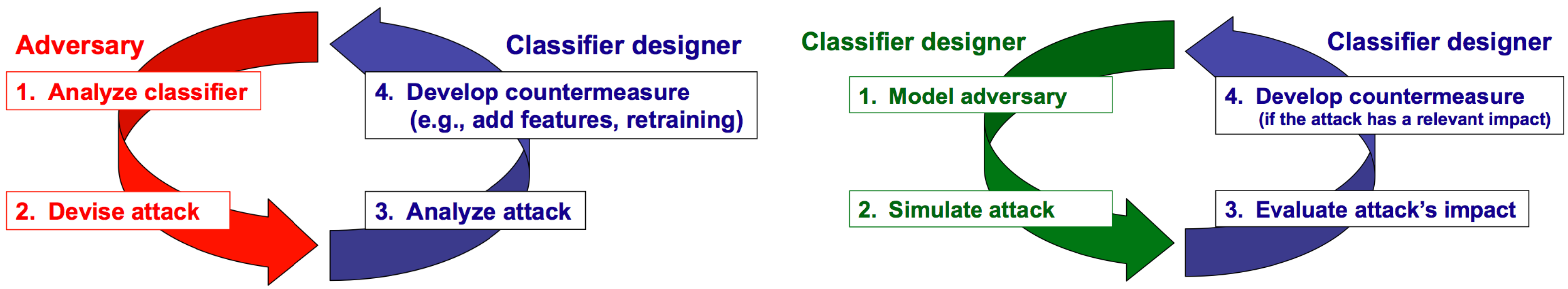
Administrativia

- HW2: Due Wed. 11/5
- Mid-quarter feedback survey
- Invited talk tomorrow: Union Bank
- Guest lecture on Tue: Google

ML for security: Adversarial challenge

- Adversaries adapt
  - ML assumptions do not necessarily hold
    - I.I.D, stationary distributions, linear separability, etc.
- ML algorithm itself can be an attack target

ML for security: Reactive vs Proactive



Adversarial machine learning

"If you know the enemy and know yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will also suffer a defeat. If you know neither the enemy nor yourself, you will succumb in every battle."

– Sun Tzu, The Art of War

Taxonomy of attacks against ML

Influence	Security violation	Specificity
Causative	Integrity	Targeted
Exploratory	Availability	Indiscriminate

Causative integrity attack: The spam foretold

- Send non-spam resembling the desired spam
  - “What watch do you want? Really, buy it now!”
  - “Watch what you buy now! Do you really want it?”
- Learner mistrained
  - misses eventual spam(s)

Causative integrity attack technique: Red herring

- Introduce spurious features into all malicious instances used by defender for training
- Defender learns spurious features as necessary elements of malicious behavior
- At attack time, malicious instances lack the spurious features and bypass the filter

Causative availability attack example: The rogue filter

- Send spam resembling benign messages
  - Include both spam words and benign words
- Learner associates benign words with spam

Causative availability attack technique: Correlated outlier

- Add spurious features to malicious instances
- Filter blocks benign traffic with those features

Causative availability attack technique: Allergy

- Autograph: worm signature generation

	Defense	Attack
Phase I	Identify infected nodes based on behavioral (scanning) patterns	An attack node convinces defense of its infection by scanning
Phase II	Observe traffic from infected nodes, infer blocking rules based on observed patterns	Attack node sends crafted packets, causes ML to learn rules blocking benign traffic (DoS)

Exploratory integrity example: The shifty spammer

- Craft spam so as to evade classifier without direct influence over the classifier itself
  - Exchange common spam words with less common synonyms
  - Add benign words to sanitize spam

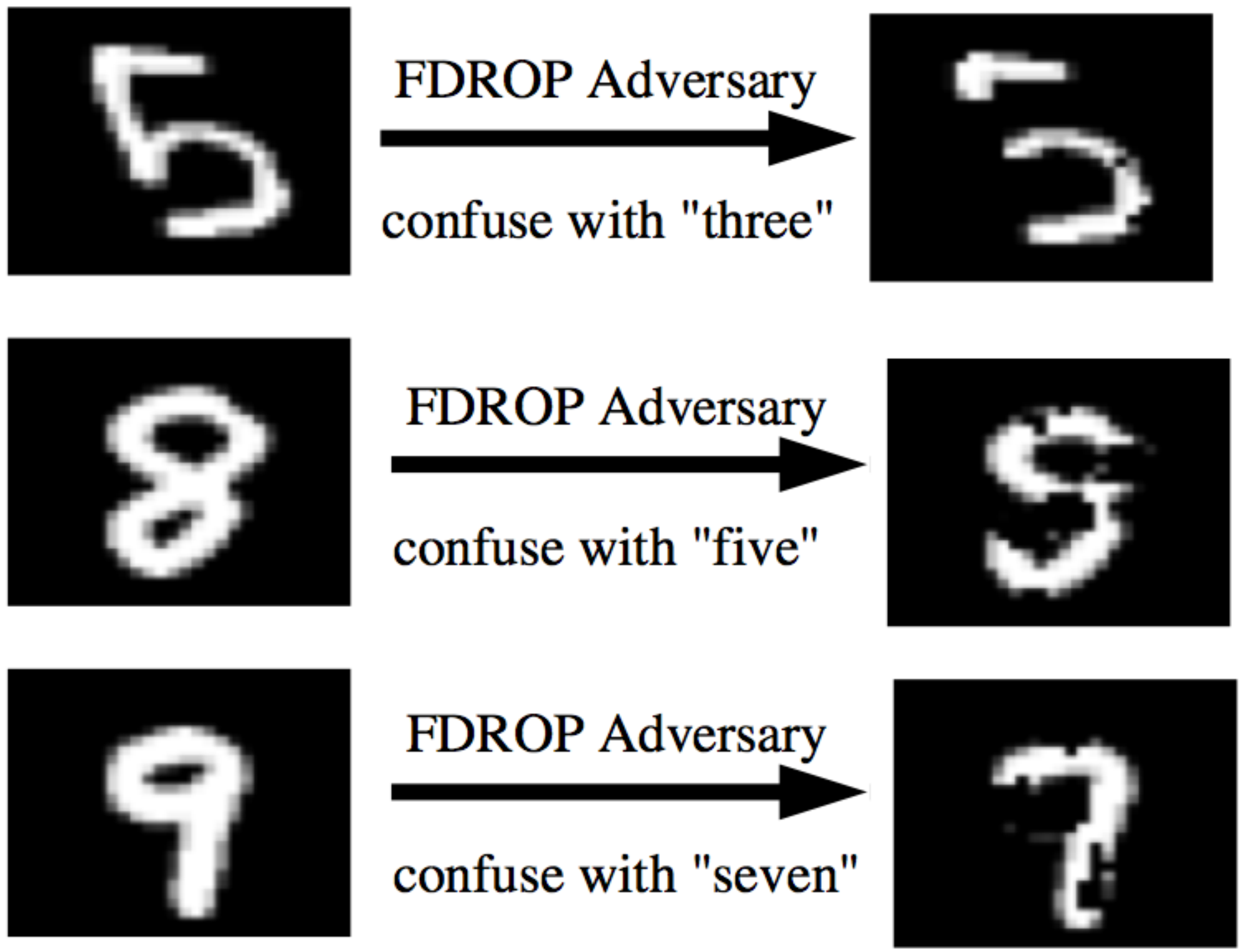
Exploratory integrity attack technique: Polymorphic blending

- Encrypt attack traffic so it appears statistically identical to normal traffic

Exploratory integrity attack technique: Mimicry

- Example: attacking sequence-based IDS
  - Shortest malicious subsequence longer than IDS window size

Exploratory integrity attack technique: Feature drop



Exploratory integrity attack technique: Reverse engineering

- Attacker seeks the highest cost instance that passes the classifier

Exploratory availability example: The mistaken identity

- Interfere with legitimate operation without influence over training
  - Launch spam campaign with target's email address as the From: address of spams
  - Flood of message bounces, vacation replies, angry responses, etc. fill target's inbox

Exploratory availability attack technique: Spoofing

- Example:
  - IPS trained on intrusion traffic blocks hosts that originate intrusions
  - Attack node spoofs legitimate host's IP address

Exploratory availability attack technique: Algorithmic complexity

- Example: sending spams embedded in images

Defense: Exploratory attacks without probing

- Training data
  - Limit information accessible to attacker
- Feature selection
  - Example: use inexact string matching in feature selection to defeat obfuscation of words in spams
  - Avoid spurious features
  - Regularization: smooth weights, defend against feature deletion
- Hypothesis space/learning procedurs
  - Complex space harder to decode, but also harder to learn
  - Regularization: balance complexity and over-fitting

Defense: Exploratory attacks with probing

- Randomization
  - Random decision instead of binary decision
- Limiting/misleading feedback
  - Example: eliminating bounce emails

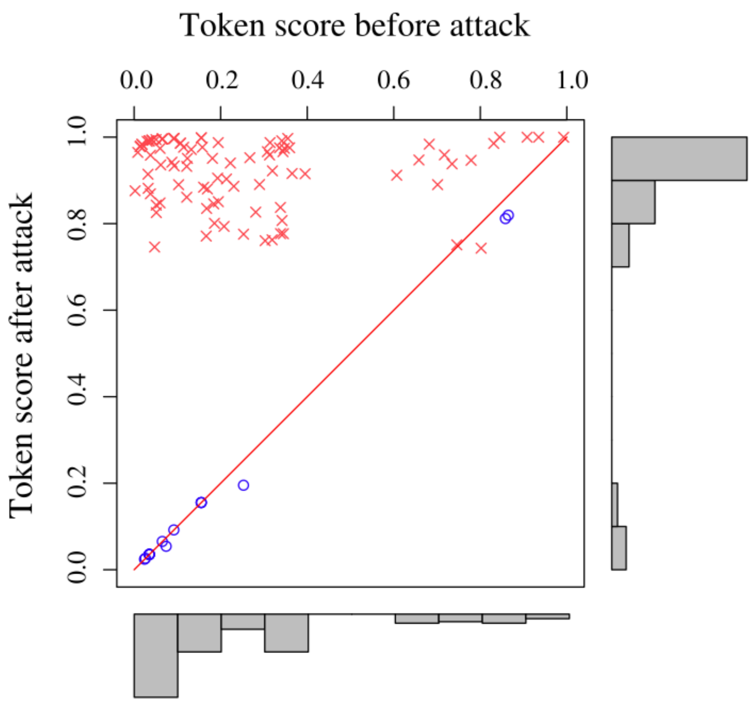
Defense: Causative attacks

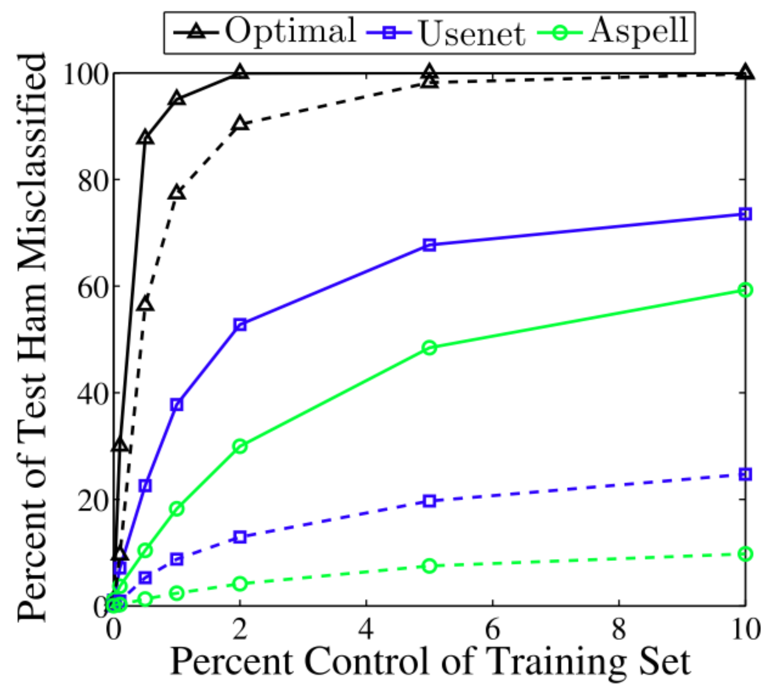
- Data sanitization
  - Example: Reject On Negative Impact (RONI)
- Robust learning
  - Robust statistics
    - Example: Median instead of Mean
- **Multi-classifier systems**
  - Online prediction with experts

Example: Causative availability attack on Naïve Bayes spam filter

- Method:
  - Send attack emails with legitimate words
  - Legitimate words receive higher spam scores
  - Future legitimate emails more likely filtered
- Types:
  - Indiscriminate: Dictionary attack
  - Targeted: Focused attack
- Goals:
  - Get target to disable spam filter
  - DoS against a bidding competitor

Performance





RONI

Before the RONI defense				
		Predicted Label		
		ham	spam	unsure
True Label	ham	97%	0.0%	2.5%
	spam	2.6%	80%	18%

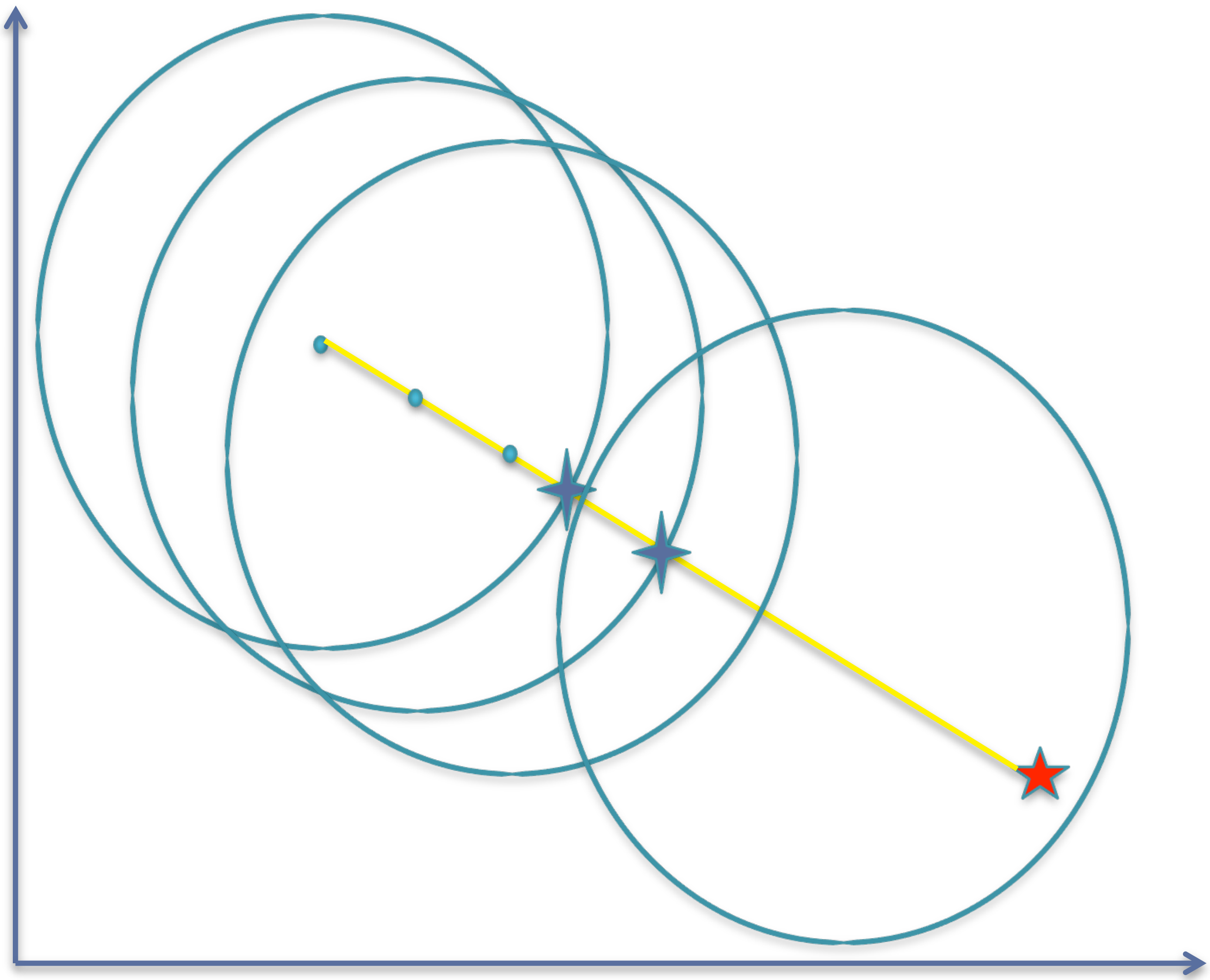
After the RONI defense				
		Predicted Label		
		ham	spam	unsure
True Label	ham	95%	0.3%	4.6%
	spam	2.0%	87%	11%

RONI

Dictionary Attacks (Before the RONI defense)				
		Predicted Label		
		ham	spam	unsure
Optimal				
True Label	ham	4.6%	83%	12%
	spam	0.0%	100%	0.0%
Aspell				
True Label	ham	66%	12%	23%
	spam	0.0%	98%	1.6%
Usenet				
True Label	ham	47%	24%	29%
	spam	0.0%	99%	0.9%

Dictionary Attacks (After the RONI defense)				
		Predicted Label		
		ham	spam	unsure
Optimal				
True Label	ham	95%	0.3%	4.6%
	spam	2.0%	87%	11%
Aspell				
True Label	ham	95%	0.3%	4.6%
	spam	2.0%	87%	11%
Usenet				
True Label	ham	95%	0.3%	4.6%
	spam	2.0%	87%	11%

Poisoning: Boiling frog attack



Boiling frog defense: Robust statistics

Mean: $\bar{r} = \frac{1}{n}\sum_{i=1}^{n} r_i$	Median: $\hat{r} = Median\{r_1, r_2, \ldots, r_n\}$
Variance: $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(r_i - \bar{r})^2$	Median Absolute Deviation: $MAD = Median\{\lvert r_i - \hat{r} \rvert\}$

Conclusion

- This is a game!
- Anticipate the adversary
- Constant arms race

References

- Adaptive Intrusion Detection System via Online Learning, 2012

  http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6421346
- The security of machine learning, 2010

  http://bnrg.cs.berkeley.edu/~adj/publications/paper-files/SecML-MLJ2010.pdf