463.3.1 Machine Learning in Security

Computer Security II
CS463/ECE424
University of Illinois

Security News of the Day

ANDY GREENBERG SECURITY 08.05.2021 05:03 PM

Apple Walks a Privacy Tightrope to Spot Child Abuse in iCloud

Today Apple introduced a new set of technological measures in iMessage, iCloud, Siri, and search, all of which the company says are designed to prevent the abuse of children. A new opt-in setting in family iCloud accounts will use machine learning to detect nudity in images sent in iMessage. The system can also block those images from being sent or received, display warnings, and in some cases alert parents that a child viewed or sent them.

Will Apple child safety gift governments the keys to comms?

https://www.wired.com/story/apple-csam-detection-icloud-photos-encryption-privacy/
https://telecoms.com/510864/will-apple-child-safety-gift-governments-the-keys-to-comms/

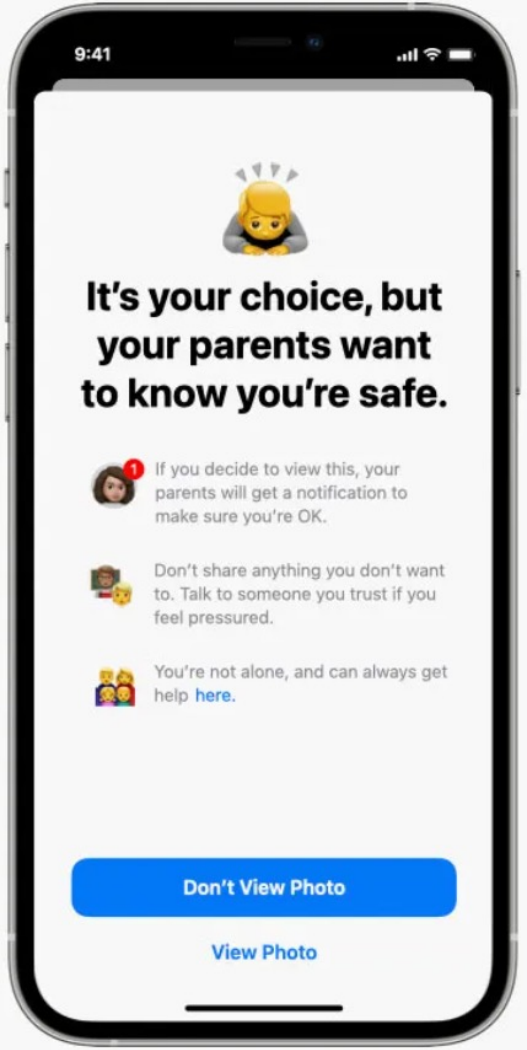
Security News of the Day

ANDY GREENBERG SECURITY 08.05.2021 05:03 PM

Apple Walks a Privacy Tightrope to Spot Child

Today Apple introduced a new set of technological measures in iMessage, iCloud, Siri, and search, all of which the company says are designed to prevent the abuse of children. A new opt-in setting in family iCloud accounts will use machine learning to detect nudity in images sent in iMessage. The system can also block those images from being sent or received, display warnings, and in some cases alert parents that a child viewed or sent them.

Will Apple child safety gift governments the keys to comms?



https://www.wired.com/story/apple-csam-detection-icloud-photos-encryption-privacy/
https://telecoms.com/510864/will-apple-child-safety-gift-governments-the-keys-to-comms/

Definitions
Spam Classification using Logistic Regression
Anomaly Detection through Deep Learning
Challenges for Machine Learning in Security

What is Machine Learning?

The complexity in traditional computer programming is in the code.
In machine learning, algorithms (programs) are in principle simple, and the complexity (structure) is in the data. Is there a way that we can automatically learn that structure? That is what is at the heart of machine learning.

-- Andrew Ng

What is Machine Learning?

Traditional Programming

When you know "how" to do things


```
int addition (int a, int b)
{
        int r;
        r = a + b;
        return r;
}
```

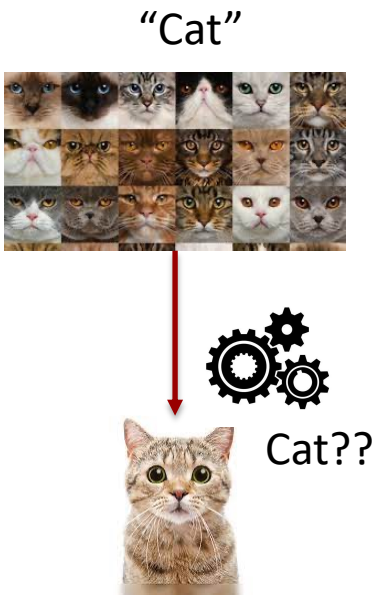
What is Machine Learning?

Traditional Programming

When we know <u>how</u> to do things

Machine Learning

When we don't know how to do it, but we have <u>some examples</u>



 Credit to David Meyer

What is Machine Learning?

A computer program is said to learn from **experience E**
with respect to some class of **tasks T** and **performance measure P**,
if its performance at tasks in T, as measured by P, improves with experience E.

-- Tom Mitchell, *Machine Learning*

Mitchell, T. (1997). *Machine Learning*, McGraw Hill. ISBN 0-07-042807-7, p.2.

Steps towards Designing a ML System

- Step 1: Choosing the Training Experience (i.e., training dataset)
- Step 2: Choosing the Target Function
- Step 3: Choosing a Representation for the Target Function
- Step 4: Choosing a Function Approximation Algorithm
- Step 5: Evaluation

When To Use Machine Learning?

- When patterns exist in the data
  - Even if we don't know what they are
- We cannot pin down the functional relationships mathematically
  - Else we would just code up the algorithm
- When we have lots of (unlabeled) data
  - Labeled training sets harder to come by
  - Data is of a high-dimension
  - Want to discover lower-dimension representations

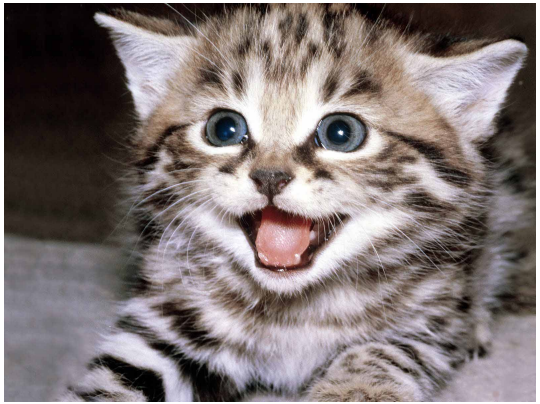
Example: Spam Filtering

- **Task T**: classifying emails into two categories (spam, ham)
- **Performance measure P**: percent emails correctly classified
- **Training Experience E**: a database of emails

Step 1: Choosing the Training Experience

- Training Experience E:
  - A database of emails
- What feedbacks can be provided to the learner?
  - A database of labeled emails
- How well does the training experience represent the distribution of examples over which the final system performance P must be measured?
  - A database of labeled emails that represent the distribution of all the emails

Step 1: Choosing the Training Experience



Cats v.s. Dogs

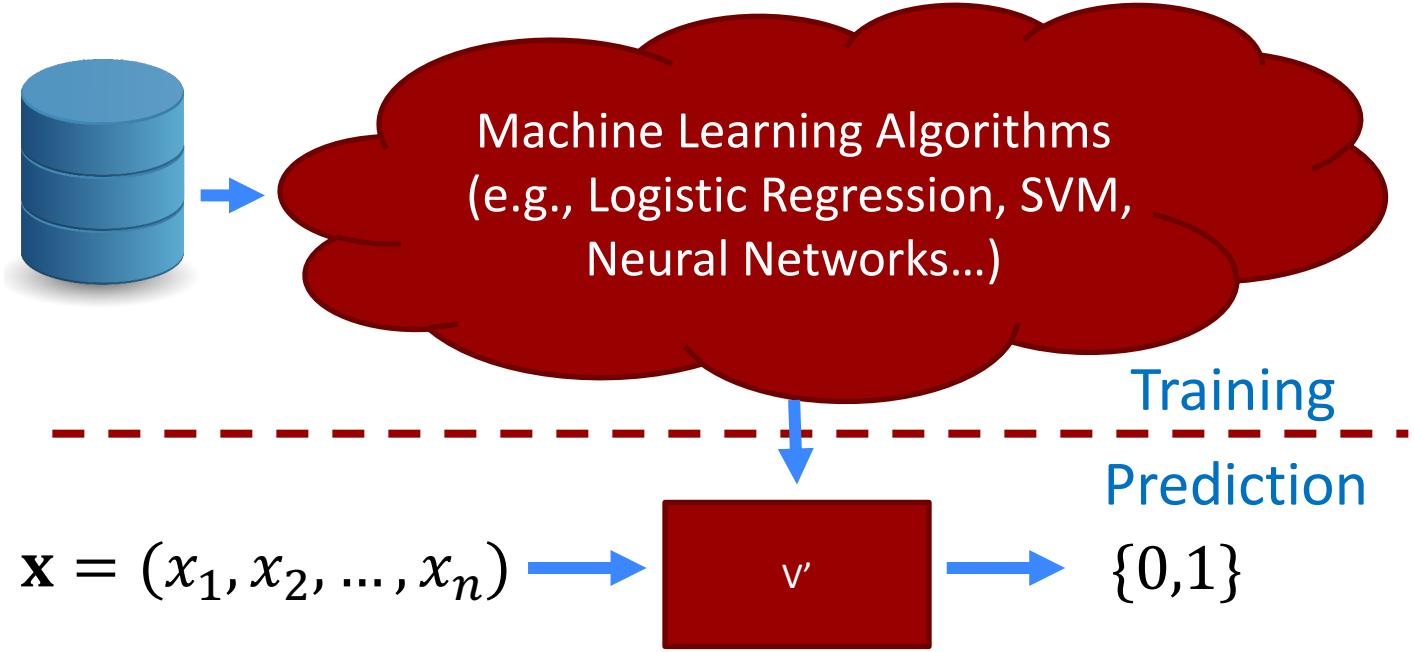
Step 2: Choosing the Target Function

- **Task T:** classifying emails into 2 categories (Spam, Ham)
- Target function $V: A \rightarrow B$
  - What's in the training examples?
    - A: Email contents
  - What should be the output?
    - B: {Spam (1) , Ham (0)}

Step 3: Choosing a Representation for the Target Function

- How do we represent the model inputs and outputs?
  - Numerical, nominal, binary, sequential …
- Feature generation
  - V: Email contents → {0,1}
  - V': $\mathbf{x} = (x_1, x_2, \dots, x_n) \mapsto y \in \{0,1\}$
    - $x_i \in \{0,1\}$ represents whether a word w_i is in the email
- Feature selection
  - To simplify the model (save time, avoid overfitting…)

Step 4: Choosing a Function Approximation Algorithm

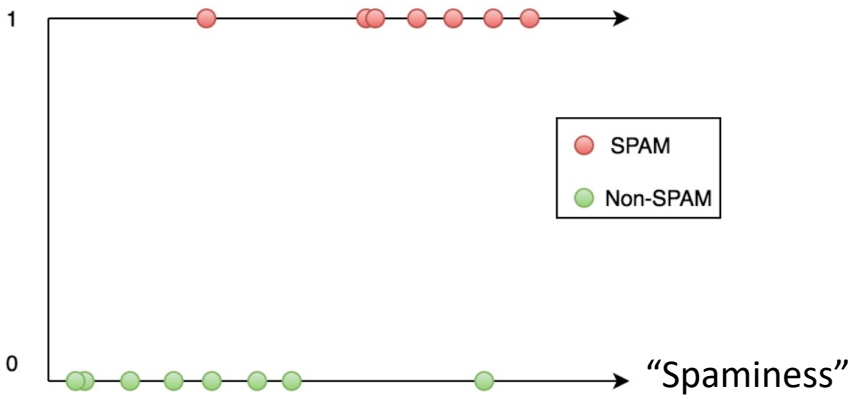


Logistic Regression (1)

- V': $\mathbf{x} = (x_1, x_2, \dots, x_n) \mapsto y \in \{0,1\}$
- Design V'
  - Step 1: Combine $x_1, x_2, \dots, x_n$ to get a "spaminess" value
  - Step 2: Convert the "spaminess" value into a probability P(Spam)
  - Step 3: Make predictions on y based on P(Spam)
    - e.g., $y = 1$ when P(Spam) > 0.5

Logistic Regression (2)

- Step 1: Combine $x_1, x_2, \ldots, x_n$ to get a "spaminess" value
  - A weight vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$
  - Spaminess: $\boldsymbol{\theta}^{\mathbf{T}} \cdot \mathbf{x}$

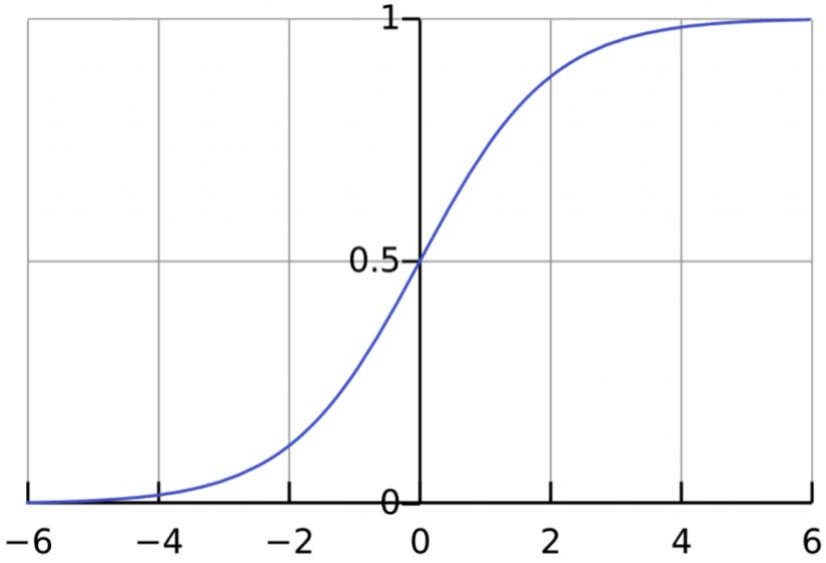


Logistic Regression (3)

- Step 2: Convert the "spaminess" value into a probability P(Spam)
  - Logistic function

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

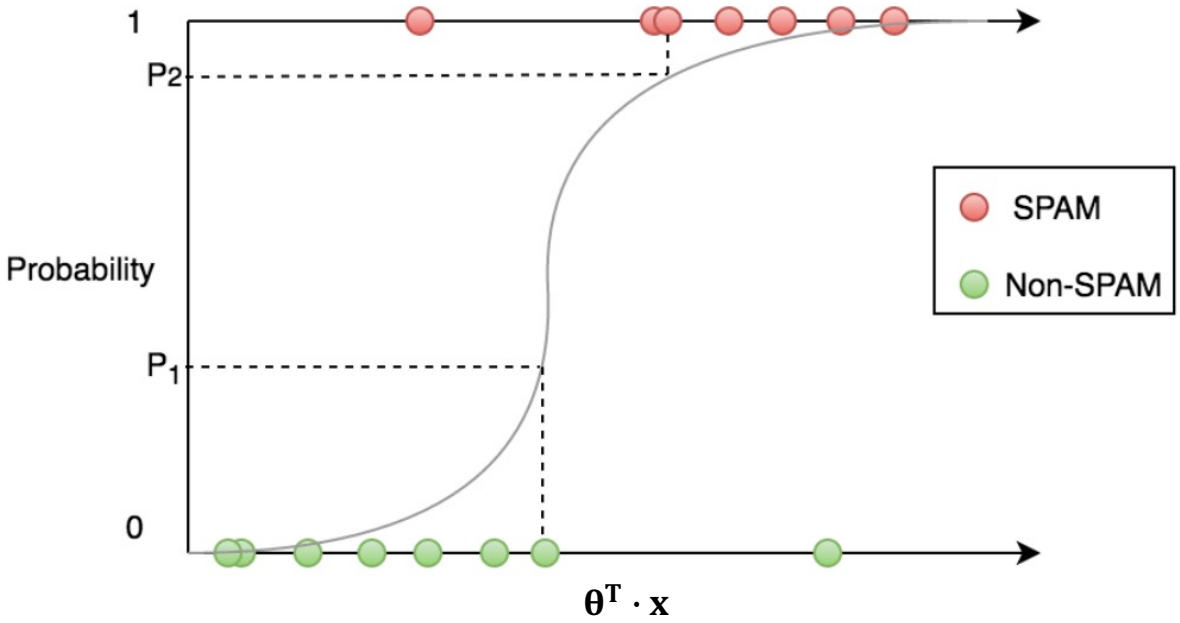
$$h_\theta(x) = g(\theta^T X)$$



Plot of logistic function g

Logistic Regression (4)

- Step 3: Make predictions on y based on P(Spam)



Logistic Regression: Training

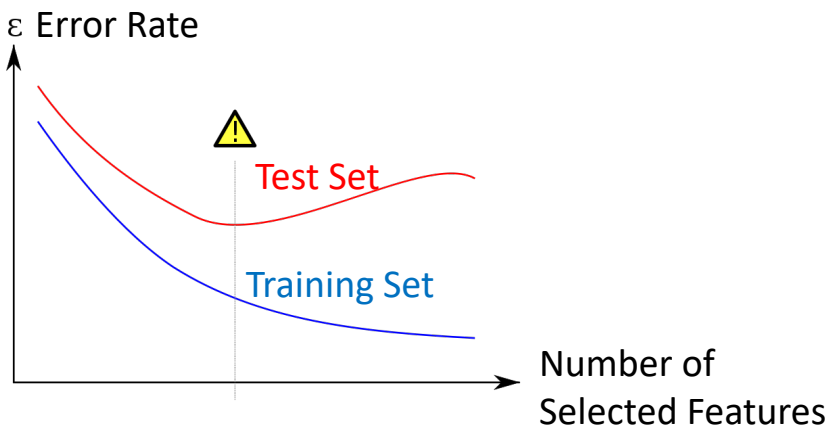
- How do we determine the value of $\boldsymbol{\theta}$?
  - Suppose we already have a $\boldsymbol{\theta}$ and some labeled examples, how do we know whether $\boldsymbol{\theta}$ is good enough?
- Define a loss function (e.g., logistic loss)
  - Wrong predictions -> large loss
  - Correct predictions -> small loss
- Run optimization algorithms to find $\boldsymbol{\theta}$, minimize the loss
  - e.g., Stochastic Gradient Descent (SGD)

Step 5: Evaluation

- Ground Truth
  - V: Email contents → {0,1}
- Hold out Method
  - Randomly partitioned data into two independent sets: a **test set,** a **training set**
  - Use **test set** instead of **training set** when assessing accuracy
- Cross-validation (k-fold)
  - Randomly partition the data into *k* **mutually exclusive** subsets, each approximately equal size
  - At *i*-th iteration, use $\mathbf{D_i}$ **as test set** and **others as training set**

Step 5: Evaluation

- Overfitting:



- Confusion Matrix:

	Predicted Spam	Predicted Ham
Spam	True Positive (TP)	False Negative (FN)
Ham	False Positive (FP)	True Negative (TN)

Summary: Designing an ML System

- Step 1: Choosing the Training Experience
- Step 2: Choosing the Target Function
- Step 3: Choosing a Representation for the Target Function
- Step 4: Choosing a Function Approximation Algorithm
- Step 5: Evaluation

DeepLog: Anomaly Detection through Deep Learning

- Anomaly Detection from System Logs
  - Identify **abnormal system behavior** from **large volume of system logs**
- Challenges
  - Large volume of data
  - Sequential data
  - Unstructured data
- Why deep learning?
  - Widely used for natural language processing (NLP)
  - *Log can be viewed as **a structured language**!*

Step 1: Choosing the Training Experience (1)

- What data do we have?
  - Large volume of log entries from normal system execution path
  - A few log entries of known attacks

- What data should we use?
  - Training?
  - Testing?

Step 1: Choosing the Training Experience (2)

- What data should we use?
  - Training: normal logs
  - Testing: normal logs and attack logs
- Advantages:
  - Prevent overfitting
  - Test the system's behavior on unseen attacks
- Disadvantages:
  - May classify any unseen behaviors as attacks (i.e., false positives)

Step 2: Choosing the Target Function

- Outputs: normal (-) v.s. abnormal (+)
- Inputs: Log entries from OpenStack VM deletion task (unstructured)
  - *t1* Deletion of file1 complete
  - *t2* Took 0.61 seconds to deallocate network …
  - *t3* VM Stopped (Lifecycle Event)
- Structured representation:
  - Log key
  - Parameter value (e.g., t1, file1)

Step 3: Choosing the Representation for the Target Function (1)

- The total number of distinct log keys is constant.
  - Log keys: $K = \{k_1, k_2, \ldots, k_n\}$
  - Parameter value vectors: (time interval, other parameter values)

log message (log key underlined)	log key	parameter value vector
t_1 Deletion of **file1** complete	k_1	$[t_1 - t_0, \text{file1Id}]$
t_2 Took **0.61** seconds to deallocate network ...	k_2	$[t_2 - t_1, 0.61]$
t_3 VM Stopped (Lifecycle Event)	k_3	$[t_3 - t_2]$
...	...	...

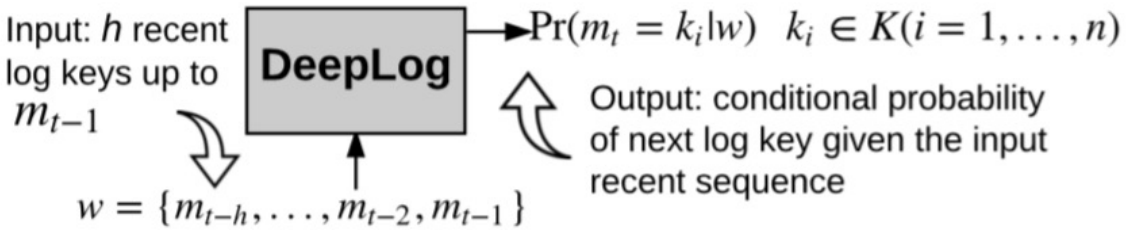
Table 1: Log entries from OpenStack VM deletion task.

Step 3: Choosing the Representation for the Target Function (2)

- Representation of Inputs:
  - **Log Keys:** structured, sequential, nominal
  - **Parameter Values:** structured, sequential, numerical (e.g., time, duration) or nominal (e.g., process id)
  - Different log keys have **different structures** for parameter values
- How to combine the inputs of different structures?
  - Train multiple models

Step 3: Choosing the Representation for the Target Function (3)

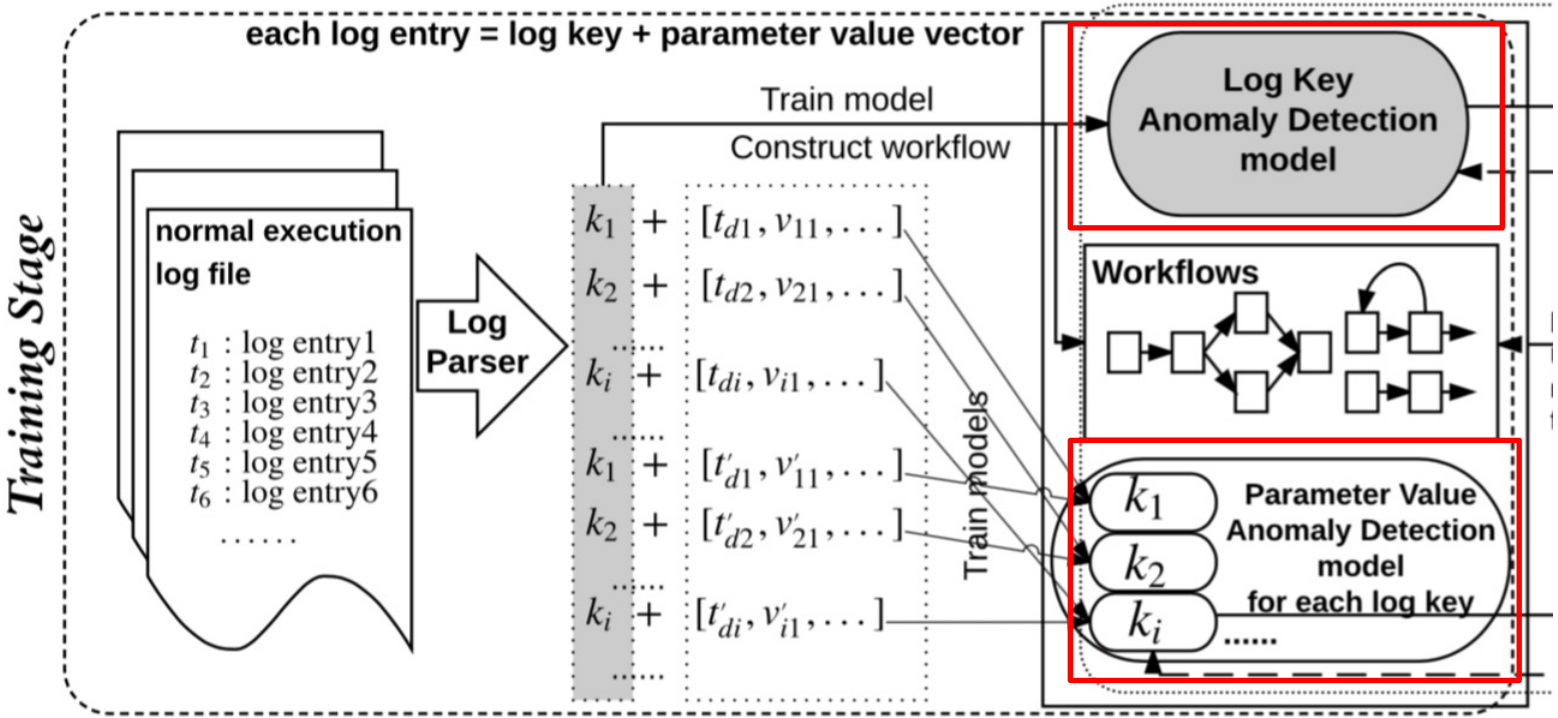
- Log key anomaly detection model
  - Log keys: $K = \{k_1, k_2, \dots, k_n\}$
  - Input: A window w of the h most recent log keys $w = \{m_{t-h}, \dots, m_{t-2}, m_{t-1}\}$, where $m_i \in K$
  - Output: $\Pr[m_t = k_i \mid w]$ for each $k_i \in K$, $(i = 1, \dots, n)$



Step 3: Choosing the Representation for the Target Function (4)

- Parameter value anomaly detection models
  - View each parameter value vector sequence (for a log key) as a separate time series
  - Train a separate model for each distinct log key value to predict the next parameter value
- Two steps of detecting anomaly
  - Predict the the next log key and parameter values
  - Compare the prediction against the observed log entry
    - Mark as anomaly if the probability for the observed log entry is low (not in the top g candidates)

Step 3: Choosing the Representation for the Target Function (5)



Step 4: Choosing a Function Approximation Algorithm

- Long Short-Term Memory (LSTM) Network
  - Has the capability of remembering previous inputs
  - Suitable for sequential data
  - A gentle walk through on LSTM networks (**optional**, 25 minutes): https://www.youtube.com/watch?v=WCUNPb-5EYI

Step 5: Evaluation – Log Key Model (1)

- Hadoop-Distributed File System (HDFS) Dataset
  - System logs generated by map-reduce jobs on more than 200 Amazon's EC2 nodes
  - Labeled by domain experts
  - Log entries are grouped into sessions

Log data set	Number of sessions		*n*: Number of log keys
	Training data (if needed)	Test data	
HDFS	4,855 normal; 1,638 abnormal	553,366 normal; 15,200 abnormal	29

- DeepLog does not use the abnormal training data

Step 5: Evaluation – Log Key Model (2)

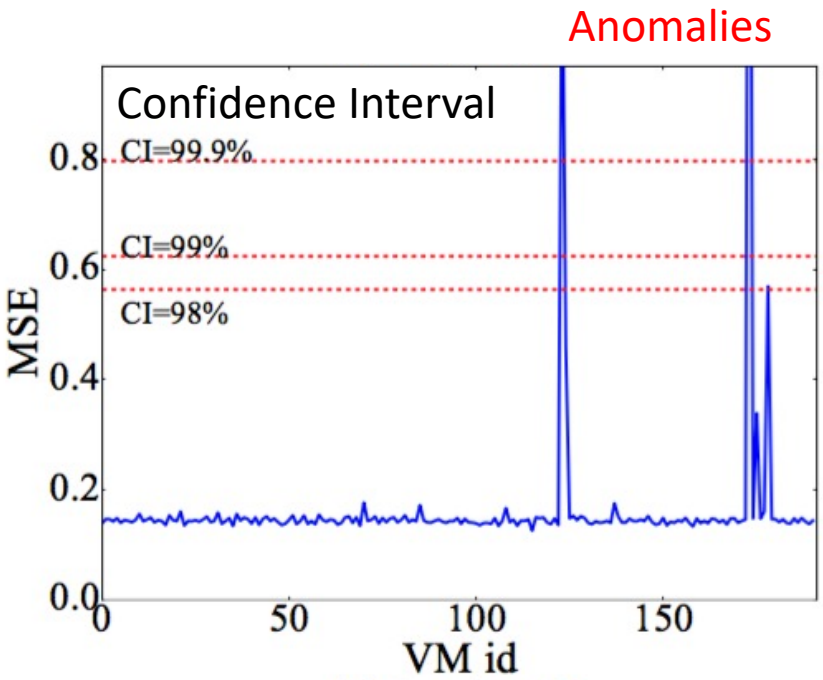
	Predicted as Normal	Predicted as Abnormal
Normal	552,533 (True Negative)	833 (False Positive)
Abnormal	619 (False Negative)	14581 (True Positive)

- Precision = True Positive / (True Positive + False Positive) = 94.60%
- Recall = True Positive / (True Positive + False Negative) = 95.93%

Step 5: Evaluation – Parameter Value Model

- **OpenStack Log Dataset**
  - Run VM-related tasks
  - Inject anomalies at different execution points

- **Mean-squared error (MSE)** between the parameter value vector and the prediction output vector from DeepLog



Discussion Questions

- How can you attack the spam filtering model we discussed?
  - Can you get around the filtering and send a spam to a user's inbox?
  - Can you trick the algorithm to filter a ham email?

- Do you think ML will replace human analysts in detecting security threats? Why or why not?

Reading

- [1] Androutsopoulos, Ion, et al. "An evaluation of naive bayesian anti-spam filtering." *arXiv preprint cs/0006013* (2000).

- [2] Du, Min, et al. "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning." *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017.

- [3] Sommer, Robin, and Vern Paxson. "Outside the closed world: On using machine learning for network intrusion detection." *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010.

Challenges for Machine Learning in Security

- Outlier Detection
- High Cost of Errors
- Semantic Gap
- Diversity with Data
- Difficulties with Evaluations

Sommer, Robin, and Vern Paxson. "Outside the closed world: On using machine learning for network intrusion detection." *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010.

Outlier Detection

- A different ML Problem
- ML needs large number of representatives for **each** class
  - What happens when $P(Spam)$ is very small?
- Not good at finding previously unknown malicious activities

High Cost of Errors

- **Example:** suppose a system generates
  - 1,000,000 audit records per day;
  - 10 audit records per intrusion;
  - Two intrusions per day.
- Intrusion: I, Alarm: A
- Detection rate: $P(A|I) = 99.9\%$
- False alarm rate: $P(A|\neg I) = 0.02\%$
- Given a detected record, what's the probability that the record represents a true intrusion?

$$P(I|A) = \frac{P(A|I)P(I)}{P(A|I)P(I) + P(A|\neg I)(1 - P(I))} = 9\%$$

Semantic Gaps

- Difficult to transfer results into **actionable** report for the network operator
- Difficult to find the difference between “abnormal activity” and attacks
- Unclear what the system learned
  - What do false positives and false negatives mean?
  - What features are used to produce correct results?

Diversity with Data and Concept Drift

- Large variability in network traffic over short time intervals

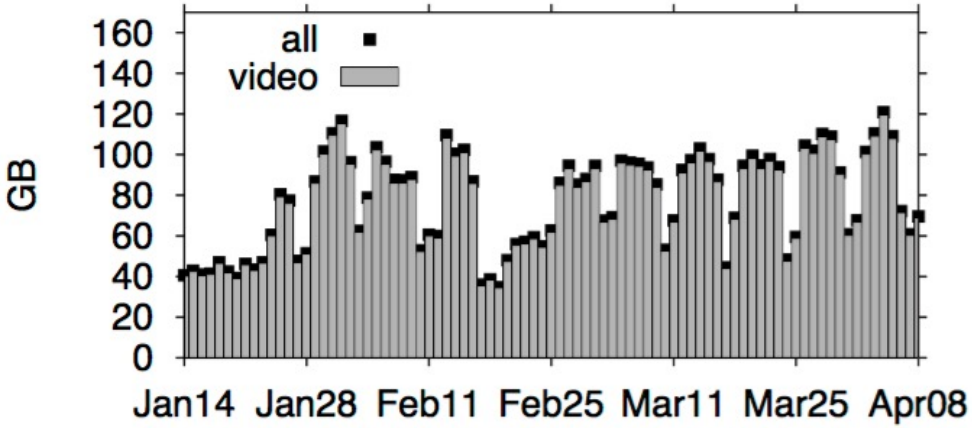


Figure 5: Bytes Per Day

Difficulties with Evaluations

- Lack of “ground truth”
- Outdated datasets
- Highly sensitive information (e.g., network traffic can include personal communications and business secrets)
- Difficulties with simulation and anonymization