University of Idaho
Department of Computer Science

CS 487/587 Adversarial Machine Learning

Dr. Alex Vakanski

Lecture 10

AML in Cybersecurity – Part III: Spam Filtering, URL Detection, Cyber-Physical Systems

Lecture Outline

- Spam filtering
  - Spam statistics
  - Spam filtering datasets
  - Spam filtering techniques
  - Pre-processing text in email messages
    - Tokenization, feature extraction, word embedding
  - Adversarial attacks against ML-based spam filters
- URL detection
  - Phishing URL detection
    - Adversarial attacks against phishing URL classifiers
- Cyber-physical Systems (CPS)
  - Machine learning-based CPS
  - Adversarial attacks against CPS
- Presentation by Shubham Pandey
  - Erba (2020) Constrained Concealment Attacks against Reconstruction-based Anomaly Detectors in Industrial Control Systems

Spam Filtering

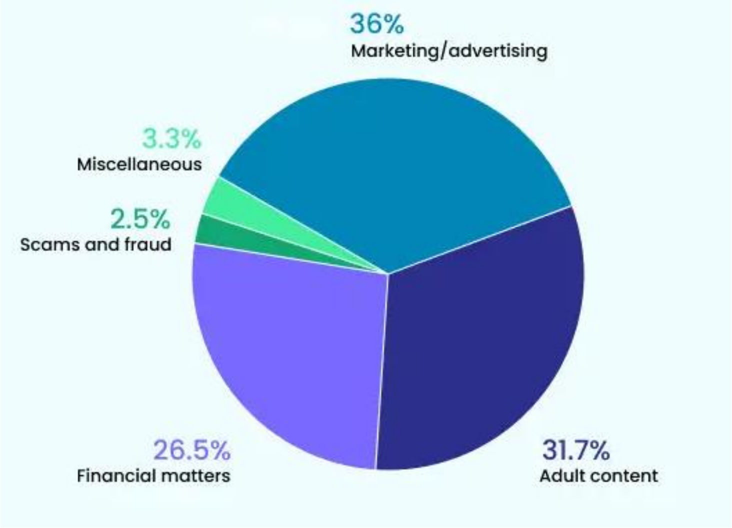
Spam Filtering

- The purpose of a ***spam filter*** is to determine whether an incoming message is legitimate (i.e., non-spam, ham) or unsolicited (i.e., spam)
- ML-based spam classifiers were among the first applications of machine learning in the cyber security domain (and in general, as well)
  - Subsequently, they were among the first to be attacked
  - Attackers' goal is to modify spam emails (without changing the nature of the message) to bypass spam filters
- Recent spam filters increasingly rely on machine learning and neural networks approaches for email classification
  - These approaches are being extensively used by email service providers like Gmail, Outlook, or Yahoo

Spam Statistics

Spam Statistics

- 49% of all sent emails are spam
  - 36% of all spam is some form of advertising
- Spam costs $20.5 billion yearly (reduced network bandwidth, storage capacity)
- About 162 billion spam emails are send every day
- Spammers receive on average 1 click for every 12 million emails sent
  - Even with this response, spammers earn millions of dollars yearly
- 80% of all spam is sent by 100 spammers
  - U.S. is home to 7 of the world's top 10 spammers

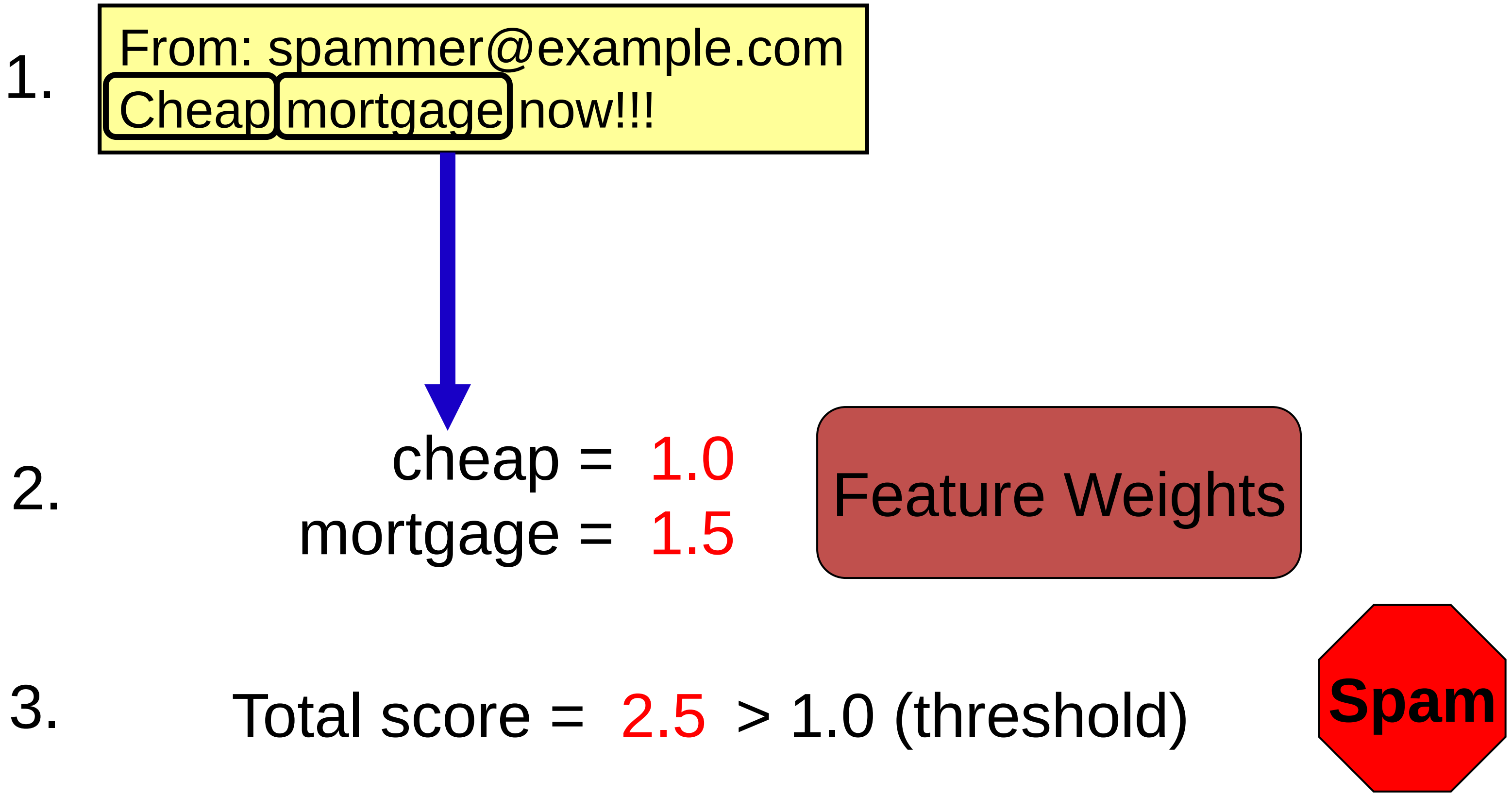


Info from: https://www.mailmodo.com/guides/email-spam-statistics/

Spam Filtering Adversarial Game

Spam Filtering

- In the following simple example, an email is classified as either a spam or a legitimate message based on cumulative weights assigned to words (or other features)



1. From: spammer@example.com
Cheap mortgage now!!!

2. cheap = 1.0
mortgage = 1.5

Feature Weights

3. Total score = 2.5 > 1.0 (threshold)

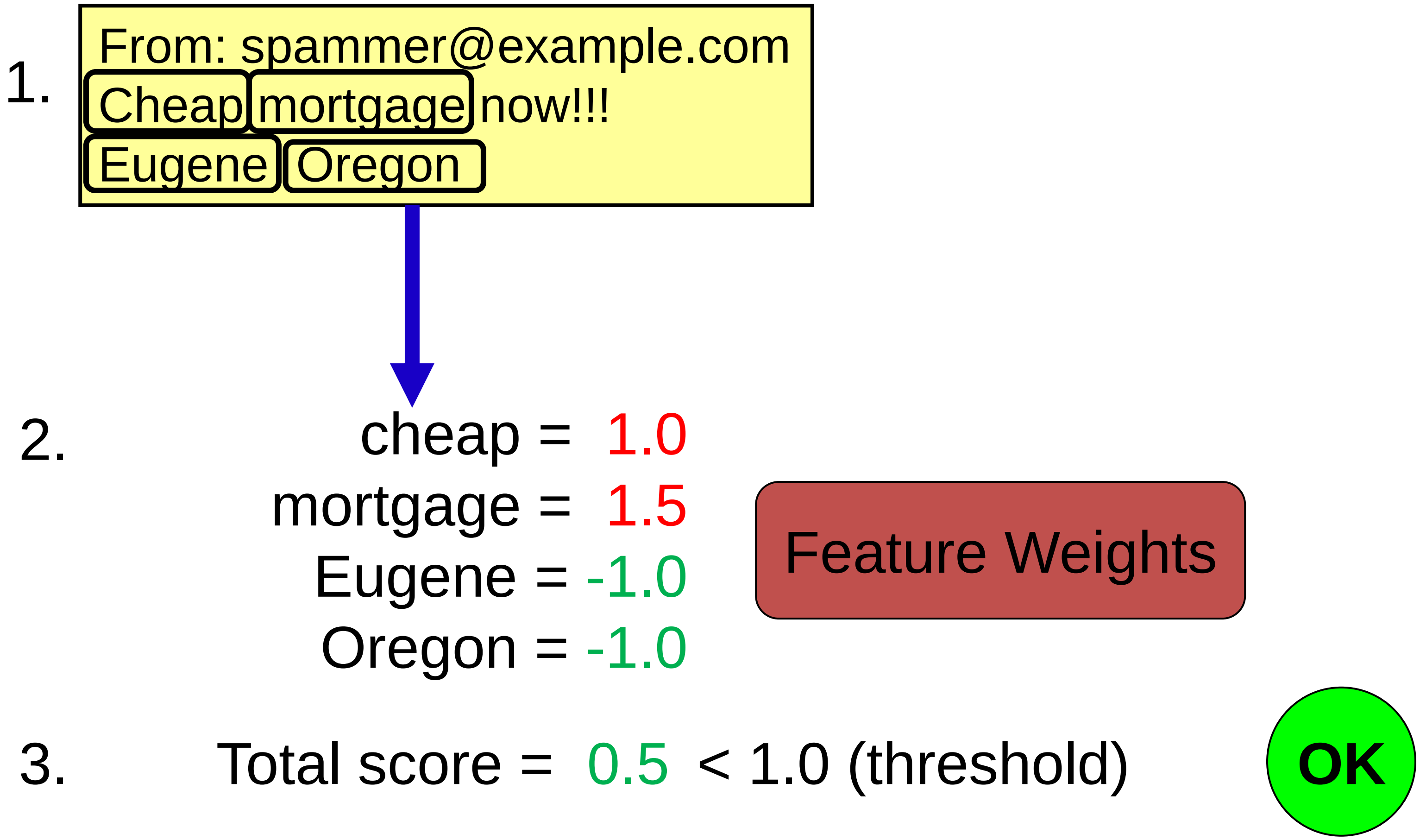
Spam

Slide credit: Daniel Lowd, Adversarial Machine Learning

Spam Filtering Adversarial Game

Spam Filtering

- The spammers adapt to evade the classifier, by adding regular words to reduce the overall score



1. From: spammer@example.com
Cheap mortgage now!!!
Eugene Oregon

2. cheap = 1.0
mortgage = 1.5
Eugene = -1.0
Oregon = -1.0

Feature Weights

3. Total score = 0.5 < 1.0 (threshold)

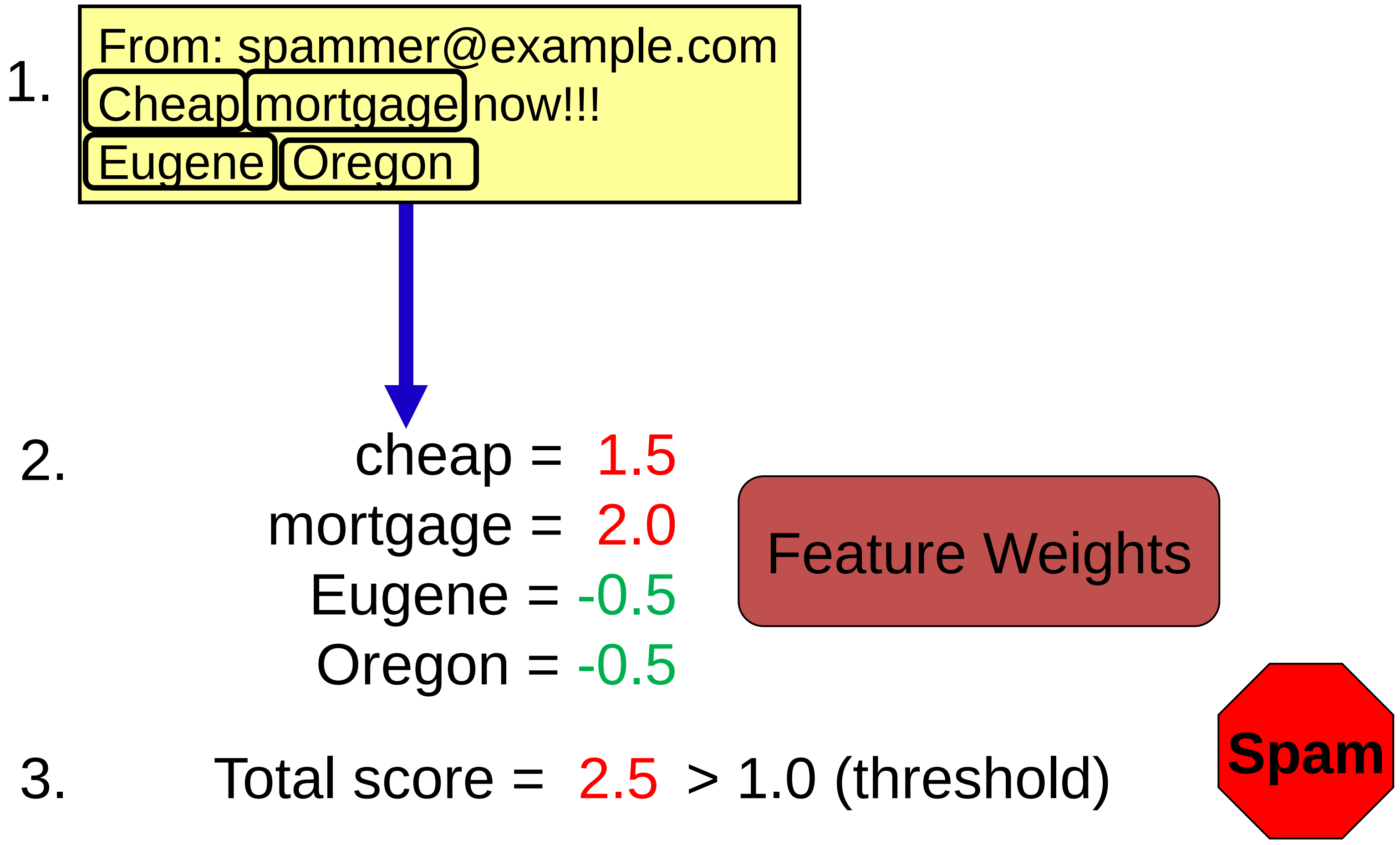
OK

Slide credit: Daniel Lowd, Adversarial Machine Learning

Spam Filtering Adversarial Game

Spam Filtering

- The spam classifier is updated by changing the feature weights



1\.

From: spammer@example.com
Cheap mortgage now!!!
Eugene Oregon

2\.

cheap = 1.5
mortgage = 2.0
Eugene = -0.5
Oregon = -0.5

Feature Weights

3. Total score = 2.5 > 1.0 (threshold)

Spam

Slide credit: Daniel Lowd, Adversarial Machine Learning

Spam Filtering Datasets

- There are many open datasets of spam and non-spam email messages
  - However, most datasets are of small size
- ***TREC 2007 Public Spam Corpus*** (link)
  - Also known as Trect07p, created in 2007
  - Consists of 50,200 spam emails (67%) and 25,200 ham (non-spam) emails (33%)
  - The format of the emails is raw data (HTML)
- ***Enron-Spam dataset*** (link)
  - Created in 2006
  - Includes 20,170 spam emails (55%) and 16,545 non-spam emails (45%)
  - Both raw messages and pre-processed messages are provided
- ***SPAMBASE*** (link)
  - Created in 1999
  - Includes 1,813 spam emails (39%) and 2,788 non-spam emails (61%)
  - Each email has 55 features, related to word frequency, character frequency, average length of uninterrupted sequence of capital letters, total number of capital letters, etc.

Spam Filtering Techniques

Spam Filtering Techniques

- Based on the used email filtering techniques, spam detectors can be generally classified into:
  - Content-based filtering techniques
  - Heuristic-based filtering techniques

Content-based Filtering

Spam Filtering Techniques

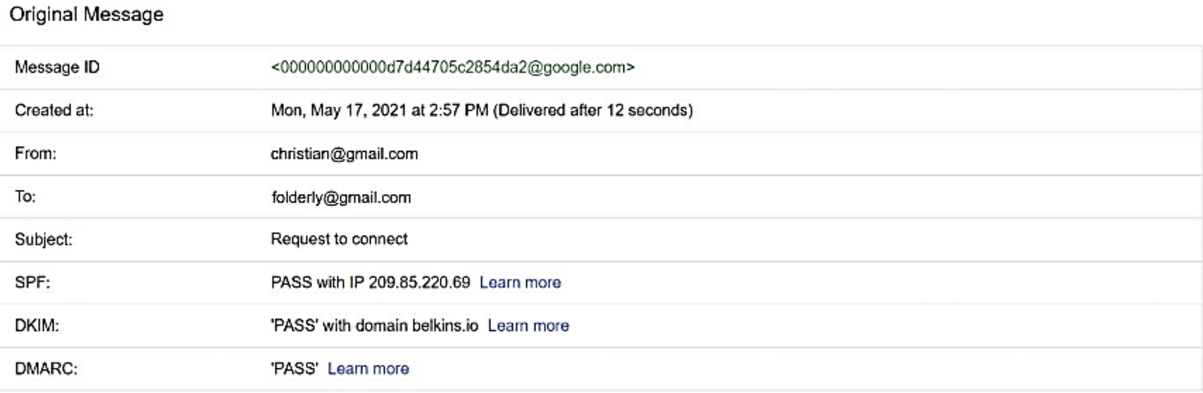
- ***Content-based filtering techniques***
  - The filter scans the content of incoming emails, looking for trigger keywords
    - E.g., keywords frequently used in spam emails, such as *free*, *buy*, *application*, *mortgage*
  - The content of the body and header of emails are scanned
- The frequency of occurrence and distribution of trigger words and phrases in the content of emails are used as features for training ML approaches, and afterward, for classifying new emails
  - Naïve Bayes classifiers were one of the early successful ML models for spam filtering
  - Other conventional ML approaches have been successfully applied, such as SVMs, *k*-nearest neighbors, decision trees, random forests
  - NNs and deep learning are commonly used nowadays for spam classification
- Almost all commercial spam filters use some form of content-based filtering
  - A limitation of this approach is that harmless emails containing spam trigger words can be blocked

Ganagavarapu (2020) – Applicability of Machine Learning in Spam and Phishing Email Filtering

Content-based Filtering

Spam Filtering Techniques

- Scanning the body of emails explores the what in the email
  - Scanning the header of emails explores the who sent the email
- Email headers display important information, such as:
  - Message ID – an identifier generated by the sender's email service
    - There can be no two identical message IDs, hence, it helps to detect forged email headers
  - Sender address – is used to consult black lists to check sender's domain reputation
  - DNS records – the DNS (Domain Name System) records of the sender allows to check the sender's SPF, DKIM, and DMARC policies regarding email authentication
    - SPF (Sender Policy Framework), DKIM (Domain Keys Identified Mail), DMARC (Domain-based Message Authentication, Reporting and Conformance)
- An example of a Gmail header



Original Message

Message ID	<000000000000d7d44705c2854da2@google.com>
Created at:	Mon, May 17, 2021 at 2:57 PM (Delivered after 12 seconds)
From:	christian@gmail.com
To:	folderly@gmail.com
Subject:	Request to connect
SPF:	PASS with IP 209.85.220.69 Learn more
DKIM:	'PASS' with domain belkins.io Learn more
DMARC:	'PASS' Learn more

Slide credit: Why Spam Filters Hate You

Heuristic-based Filtering

Spam Filtering Techniques

- ***Heuristic-based filtering techniques***
  - These approaches apply heuristic rules to discover similar patterns in a large number of spam and non-spam emails
  - Scores are assigned to each rule, and the scores are weighted based on the importance of the rule
  - Repeating patterns in a message increase the total score of being a spam
  - If the total score surpasses a predefined threshold, the message is labeled as spam
- Rules could be created based on:
  - Words and phrases, lots of uppercase characters, exclamation points, unusual Subject lines, special characters, web links, HTML messages, background colors, etc.
- Content-based filtering techniques can be considered a sub-category of the heuristic-based techniques
- A limitation of this approach is that it requires constant updating of the rules, to be able to cope with the continually adapting strategies by spammers

Ganagavarapu (2020) – Applicability of Machine Learning in Spam and Phishing Email Filtering

Black Lists and White Lists

Spam Filtering Techniques

- ***Black lists*** contain information of known spammers, collected by several sites
  - Senders of incoming emails are compared to the blacklist, to filter known spammers
- ***White lists*** are complementary to black lists, and contain addresses of trusted contacts
- Black lists and white lists are used for the first level of spam filtering
  - E.g., before applying content checks or heuristic rules
  - That is, the lists are used as a complementary tool, and not as the only tool for email classification

Preprocessing Text Data

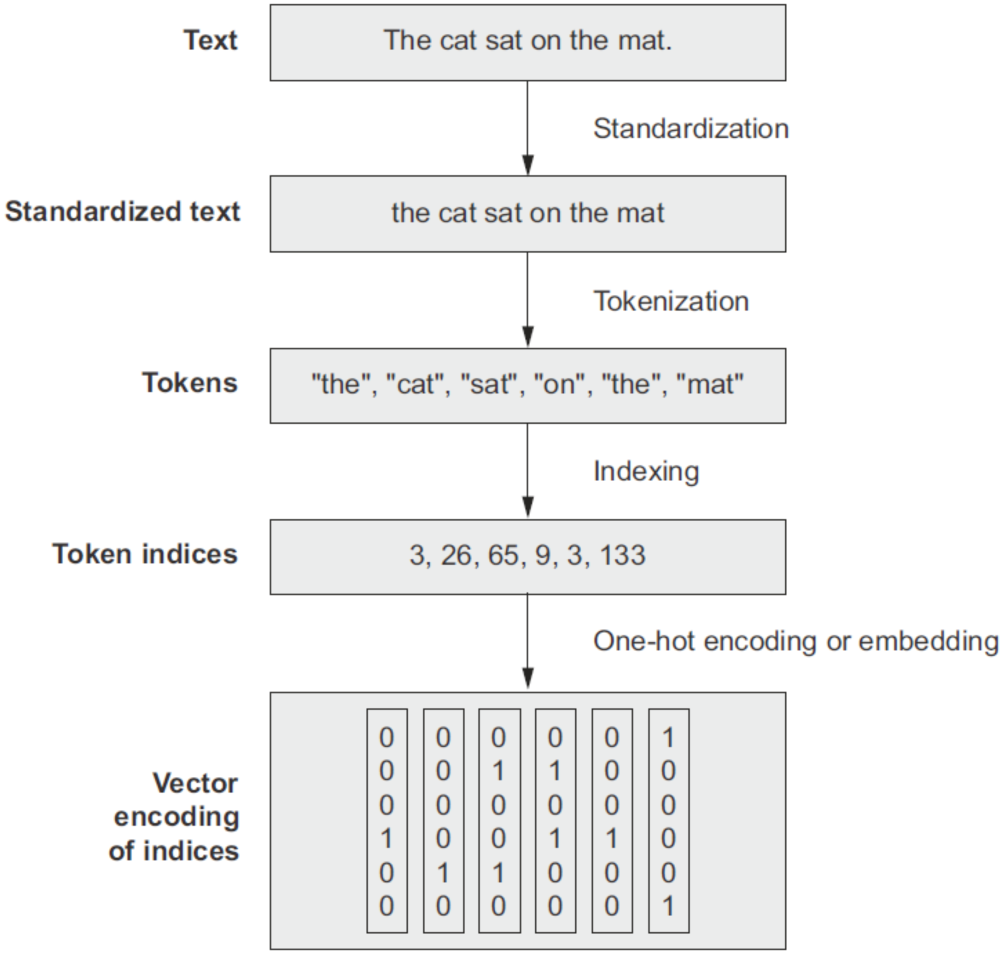
Pre-processing Text in Email Messages

- Preprocessing text data in emails for use by ML models typically involves:
  - Tokenization
  - Feature selection/extraction
- ***Tokenization***
  - Tokenization breaks up the text of an email into a sequence of representative symbols (***tokens***)
    - Tokens can be the individual words in the text, several consecutive words (e.g., *n*-grams), sub-words, or the individual characters in words (this is less common)
- Tokenization typically includes:
  - Remove punctuation signs (comma, period) or non-alphabetic characters (@, #, {,])
  - Optional step: remove stop words, such as *for*, *the*, *is*, *to*, *some*
    - These words appear in both spam and non-spam emails, and are not relevant for filtering
  - Correct spelling errors or abbreviations
  - Change all words to lower-case letters
    - I.e., the model should consider *Text* and *text* as the same word
  - Stemming and lemmatization - means transforming words to their base form
    - E.g., the words *buy-bought* or *grill-grilled* have a common root
  - Indexing – assign a numerical index to each token in the vocabulary

Preprocessing Text Data

Pre-processing Text in Email Messages

- Example of preprocessing text data



Text: The cat sat on the mat.

Standardization

Standardized text: the cat sat on the mat

Tokenization

Tokens: "the", "cat", "sat", "on", "the", "mat"

Indexing

Token indices: 3, 26, 65, 9, 3, 133

One-hot encoding or embedding

Vector encoding of indices

0	0	0	0	0	1
0	0	1	1	0	0
0	0	0	0	0	0
1	0	0	1	1	0
0	1	1	0	0	0
0	0	0	0	0	1

Figure form: Chollet (2021) Deep Learning with Python

n-Grams

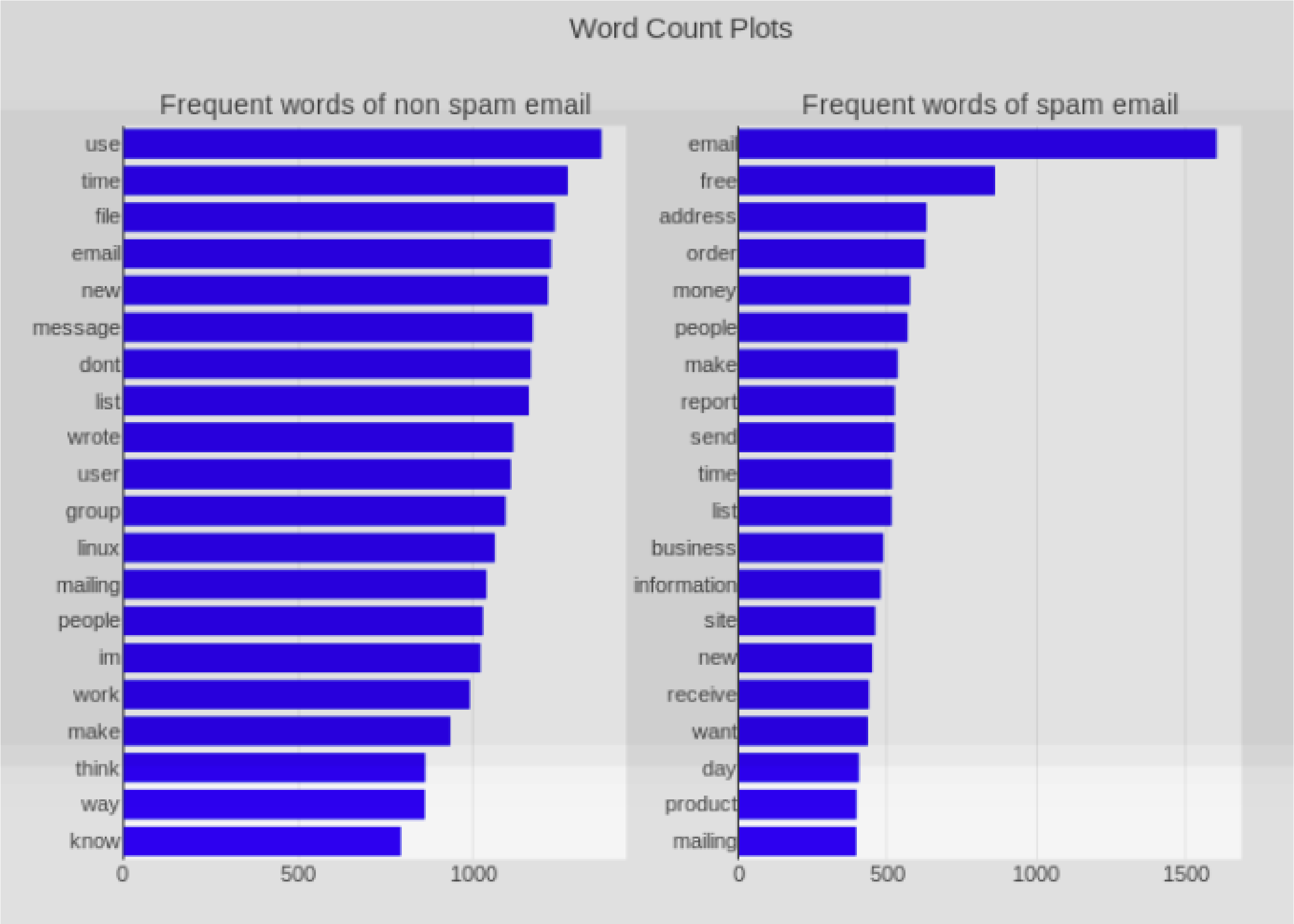
Pre-processing Text in Email Messages

- Instead of using single words or subwords as tokens, it is also possible to use *n* consecutive words as tokens, referred to as ***n-grams***
  - Combining several consecutive words together creates more specialized tokens
    - This type of tokenization is still popular for spam filtering and other NLP tasks
  - E.g., the word *play* is considered a neutral word in an email message, but the two-words phrase *play lotto* is less neutral
    - Such *n*-grams consisting of two adjacent pairs of words are called bigrams
    - *n*-grams consisting of single words are called unigrams
- The *n*-grams approach captures the words order and it can potentially provide more information for classifying spam messages

n-Grams

Pre-processing Text in Email Messages

- 1-gram model example of non-spam and spam emails



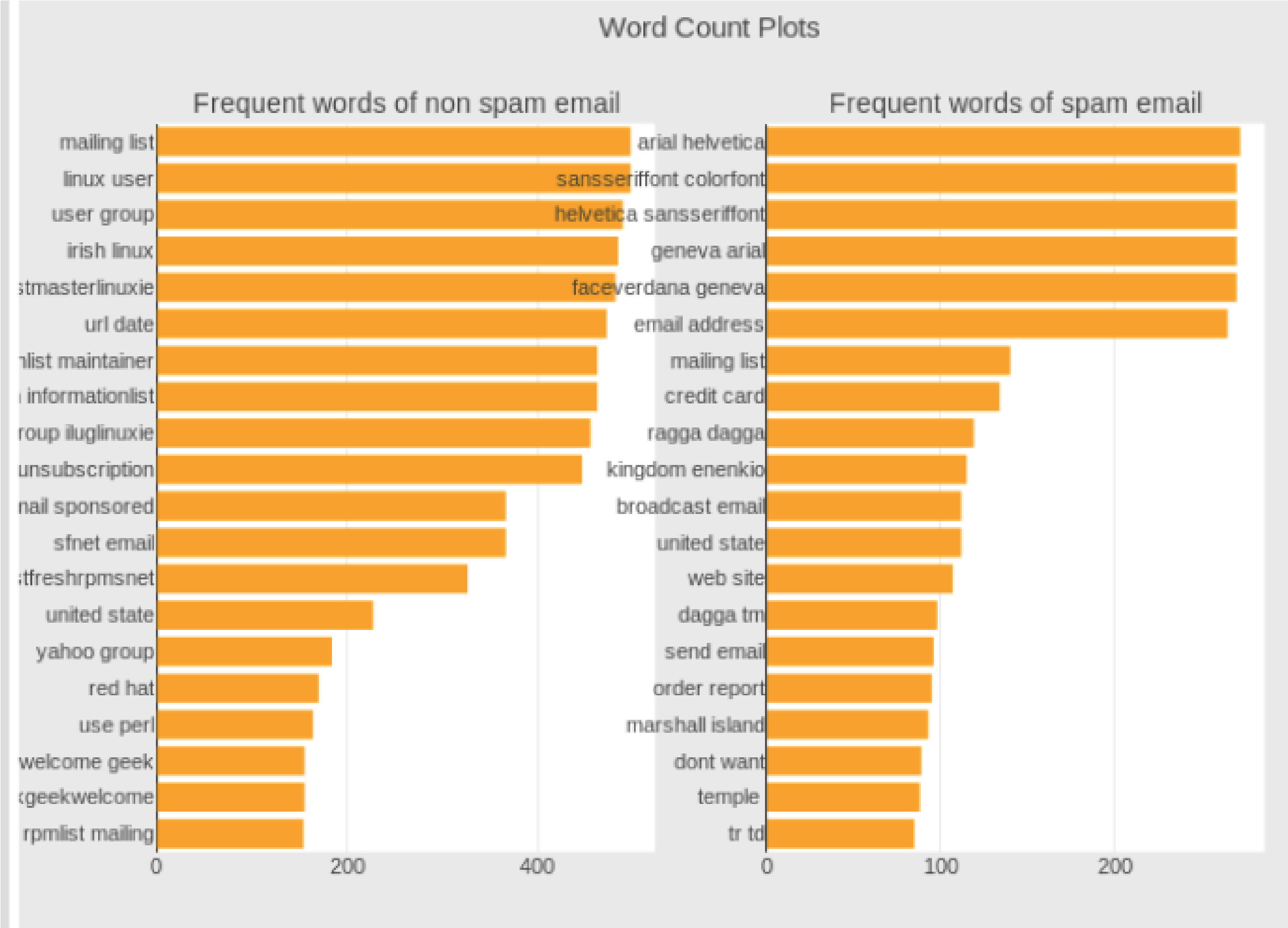
Bar chart visualization of 1-gram model

Figure from: How To Design A Spam Filtering System with Machine Learning Algorithm (link)

n-Grams

Pre-processing Text in Email Messages

- 2-gram model example of non-spam and spam emails



Bar chart visualization of 2-gram model

Figure from: How To Design A Spam Filtering System with Machine Learning Algorithm (link)

Representation of Groups of Words

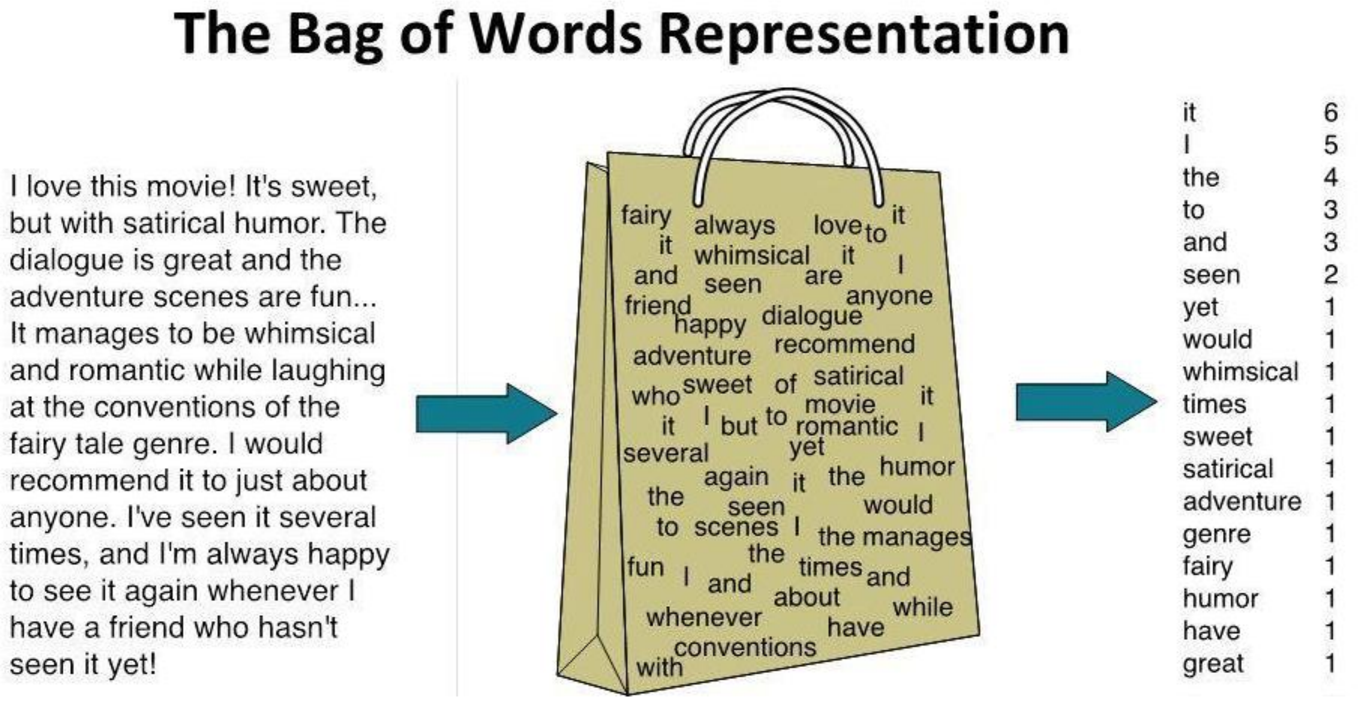
Pre-processing Text in Email Messages

- The representation of groups of words in text data can be divided into two categories of approaches:
  - ***Set models*** approach, where the text is represented as unordered collection of words
    - The order of the words in the text is not preserved
    - Representatives of this group is the bag-of-words model
  - ***Sequence models*** approach, where the text is represented as ordered sequences of words
    - These methods preserve the order of the words in the text
    - Representatives of this group are Recurrent Neural Networks and Transformer Networks
- In general, the order of words in natural language is not necessarily fixed, and sentences with different orders of the words can have the same meaning
  - However, in many cases the word order can be very important and a difference in the word order can significantly change the meaning of the text
  - Recent ML models for NLP employ sequence models where the order of the words is preserved

Bag-of-Words Approach

Bag-of-Words Approach

- ***Bag-of-words approach***
  - The tokenized words in text are represented as a bag (i.e., set) of words
  - The term *bag* implies that the order of the words and the structure of the text is lost
    - A numerical value is assigned to each token (can be either individual words or *n*-grams)
    - The frequency of occurrence of each word is typically used as a feature for training a ML classifier



The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1

Figure from: Implementation Of Bag Of Words Using Python (link)

Bag-of-Words Approach

- Bag-of-words example (based on the frequency of each word in the text)
  - Text: John likes to watch movies. Mary likes movies too.
  - Bag-of-words listing the words and the frequency of each word:

    {"John":1, "likes":2, "to":1, "watch":1, "movies":2, "Mary":1, "too":1}

- Approach:
  - Tokenize all spam and non-spam emails in a dataset
  - Create a vocabulary (token database) from the unique words (tokens) collected from all processed emails
  - Count the frequency of occurrence of tokens in spam and non-spam emails
  - Create two bags-of-words, pertaining to all spam and non-spam emails
    - E.g., the spam bag will contain trigger keywords (cheep, buy, stock) more frequently
  - A spam filter classifies an incoming email based on the probability of belonging to the spam or non-spam bag-of-words

Bag-of-Words Approach

Bag-of-Words Approach

- Bags-of-words examples for spam (left) and non-spam messages (right)

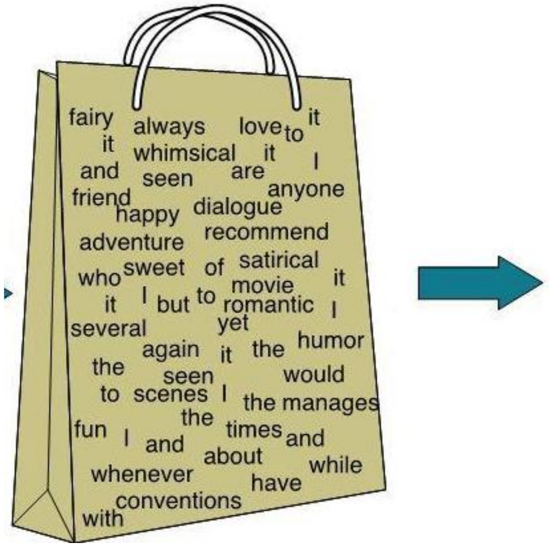
holiday
message
text
send
call
to now
ringtone
have
won
free
po box
club
statement
pic
new
or prize
line claim
every week
back
identifier code
claim
call
from
to receive
now
day
cash
find out
txt
chat
co uk
mobile
please call
customer service
stop
to
tone
to claim
prize
guaranteed call
to contact
call now
prize guaranteed
reply
will be
the latest
selected to
call identifier
150p msg
go to
from landline
tomorrow
need
wat
yes
right
people
feel
send
please
friend
always
ll
lt
gt
day
leave
happy
next
love
co
give
lol
good
night
tonight
call
msg
thing
know
call me
fine
amp
work
hi
ask
hey
go
said
going to
babe
got
say
home
ok
need to
take
life
let
still
ur
going
da
wa
already
number
come
dont
told
want
to
think
keep
place
anything
one
will
today
sent
time
ha
meet
back
tell
later
message
dear
hope
lunch
something
well

Figure from: How to Build a Spam Classifier in 10 Steps (link)

Bag-of-Words Approach

Bag-of-Words Approach

- In the shown example, we can notice that the words "it", "I", "the", and "to" have the highest number of occurrences
  - However, these words are not very indicative of the meaning of the text in the messages
- To account for that, the frequency of occurrence of words is often represented by ***TF-IDF (Term Frequency – Inverse Document Frequency)***
  - TF is calculated as the number of times a specific word (i.e., term) appears in a message, divided by the total number of words in the message
  - IDF is calculated as the logarithm of the total number of messages (i.e., documents) in the training set, divided by the number of messages in which the specific word appears in
  - A TF-IDF score for a specific word is obtained by multiplying TF and IDF
- In other words, TF-IDF assign weights to the words so that the words that appear in most messages will be considered less important for the spam classifier



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1

TF-IDF Example

Bag-of-Words Approach

- Example 1: word “the”
  - Let’s assume that the word “the” appears 4 times in a message: $TF = 4/20 = 0.2$
  - Also, assume that there are 100 message in the training set and the word “the” appears in all of them: $IDF = \log(100/100) = \log(1) = 0$
  - Therefore, TF-IDF will be $TF \cdot IDF = 0.2 \cdot 0 = 0$
- Example 2: the word “adventure”
  - If the word adventure appears once in this message: $TF = 1/20 = 0.05$
  - If it appears in 2 of the 100 messages in the training set: $IDF = \log(100/2) = \log(50) = 1.69$
  - Therefore, TF-IDF will be $TF \cdot IDF = 0.05 * 1.69 = 0.0845$
- TF-IDF will apply low weight for the word “the” since it appears in all messages, and hither weight for the word “adventure” since it appears only in a few messages

Feature Extraction

Feature Extraction

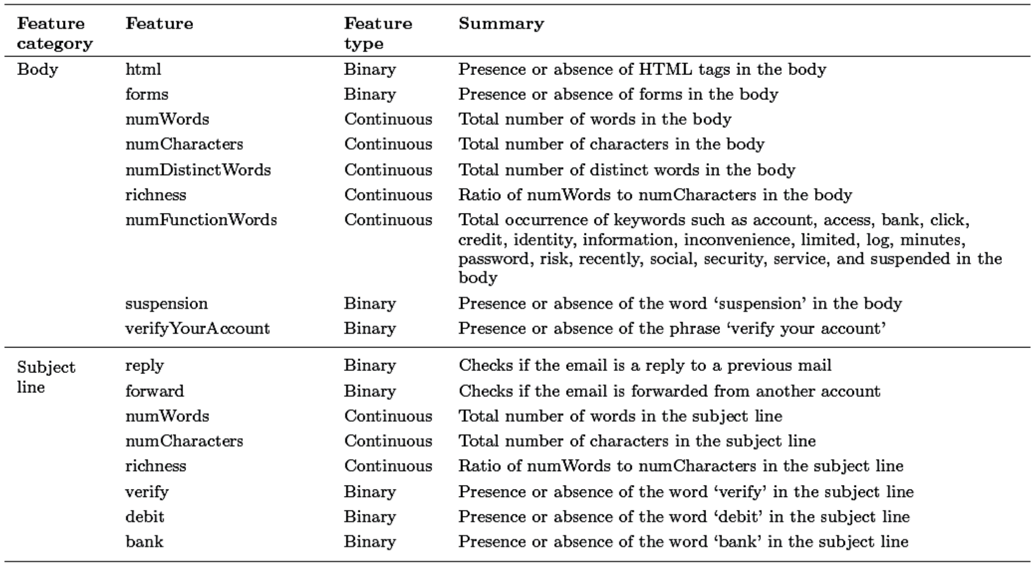
- Besides using TF-IDF for the individual tokens or words as input features for training ML spam classifiers, other approaches are based on extracting a custom set of features for training an ML model
- The extracted features from emails can include:
  - Body-based features: features extracted from the email message content
  - Subject line-based features: features extracted from the subject line of the email
  - Sender address-based features: features extracted from the information about the email address of the sender
  - URL-based features: features extracted from the anchor tags of HTML emails
  - Script-based features: features extracted from the information concerning the presence or absence of scripts in the email and the impact of such scripts

Ganagavarapu (2020) – Applicability of Machine Learning in Spam and Phishing Email Filtering

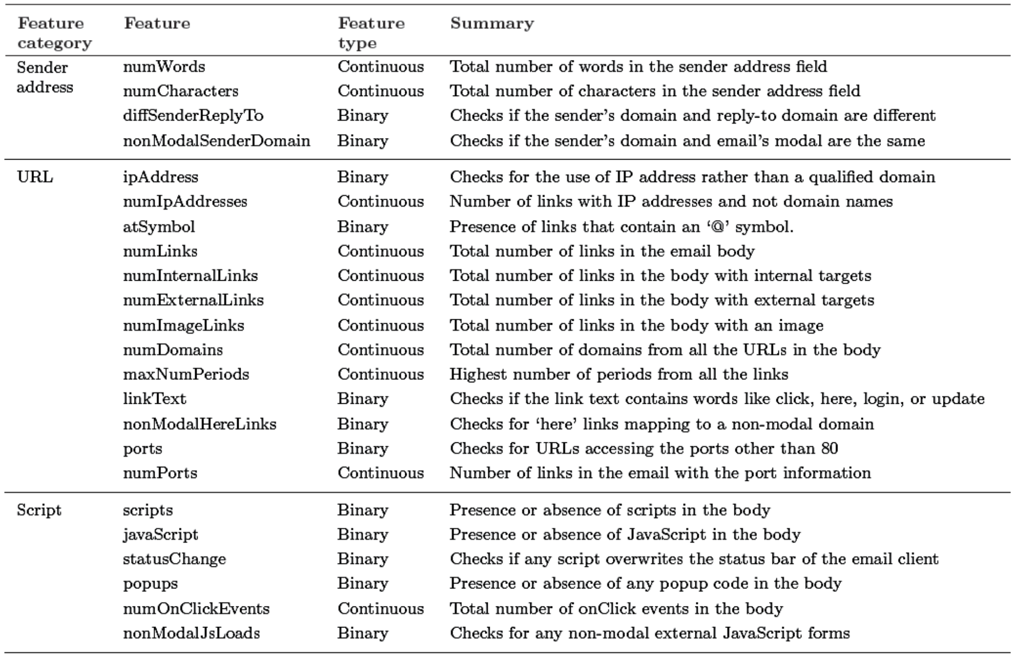
Feature Extraction

Feature Extraction

- Example: 40 features extracted from emails, categorized based on the information from the previous page
  - The features can be used to train a Naïve Bayes model or another ML model for classification of spam messages



Feature category	Feature	Feature type	Summary
Body	html	Binary	Presence or absence of HTML tags in the body
	forms	Binary	Presence or absence of forms in the body
	numWords	Continuous	Total number of words in the body
	numCharacters	Continuous	Total number of characters in the body
	numDistinctWords	Continuous	Total number of distinct words in the body
	richness	Continuous	Ratio of numWords to numCharacters in the body
	numFunctionWords	Continuous	Total occurrence of keywords such as account, access, bank, click, credit, identity, information, inconvenience, limited, log, minutes, password, risk, recently, social, security, service, and suspended in the body
	suspension	Binary	Presence or absence of the word 'suspension' in the body
	verifyYourAccount	Binary	Presence or absence of the phrase 'verify your account'
Subject line	reply	Binary	Checks if the email is a reply to a previous mail
	forward	Binary	Checks if the email is forwarded from another account
	numWords	Continuous	Total number of words in the subject line
	numCharacters	Continuous	Total number of characters in the subject line
	richness	Continuous	Ratio of numWords to numCharacters in the subject line
	verify	Binary	Presence or absence of the word 'verify' in the subject line
	debit	Binary	Presence or absence of the word 'debit' in the subject line
	bank	Binary	Presence or absence of the word 'bank' in the subject line



Feature category	Feature	Feature type	Summary
Sender address	numWords	Continuous	Total number of words in the sender address field
	numCharacters	Continuous	Total number of characters in the sender address field
	diffSenderReplyTo	Binary	Checks if the sender's domain and reply-to domain are different
	nonModalSenderDomain	Binary	Checks if the sender's domain and email's modal are the same
URL	ipAddress	Binary	Checks for the use of IP address rather than a qualified domain
	numIpAddresses	Continuous	Number of links with IP addresses and not domain names
	atSymbol	Binary	Presence of links that contain an '@' symbol.
	numLinks	Continuous	Total number of links in the email body
	numInternalLinks	Continuous	Total number of links in the body with internal targets
	numExternalLinks	Continuous	Total number of links in the body with external targets
	numImageLinks	Continuous	Total number of links in the body with an image
	numDomains	Continuous	Total number of domains from all the URLs in the body
	maxNumPeriods	Continuous	Highest number of periods from all the links
	linkText	Binary	Checks if the link text contains words like click, here, login, or update
	nonModalHereLinks	Binary	Checks for 'here' links mapping to a non-modal domain
	ports	Binary	Checks for URLs accessing the ports other than 80
	numPorts	Continuous	Number of links in the email with the port information
Script	scripts	Binary	Presence or absence of scripts in the body
	javaScript	Binary	Presence or absence of JavaScript in the body
	statusChange	Binary	Checks if any script overwrites the status bar of the email client
	popups	Binary	Presence or absence of any popup code in the body
	numOnClickEvents	Continuous	Total number of onClick events in the body
	nonModalJsLoads	Binary	Checks for any non-modal external JavaScript forms

Ganagavarapu (2020) – Applicability of Machine Learning in Spam and Phishing Email Filtering

Spam Filtering with Naive Bayes

Spam Filtering Techniques

- ***Naive Bayes classifier*** has been one of the most popular ML models for spam filtering
  - It is easy to implement, has low computational complexity, and provides statistical measure of the probability that a message is spam or non-spam
- From Bayes' theorem, the probability that an email message represented with a vector $\mathbf{x} = [x_1, x_2, \ldots]$ belongs to the spam category c_s is

$$p(c_s|\mathbf{x}) = \frac{p(c_s) \cdot p(\mathbf{x}|c_s)}{p(c_s) \cdot p(\mathbf{x}|c_s) + p(c_h) \cdot p(\mathbf{x}|c_h)}$$

  - In the equation c_h is the ham category
  - The prior probabilities $p(c_s)$ and $p(c_h)$ are typically estimated by dividing the number of training emails in each category by the total number of training emails
  - The probabilities $p(\mathbf{x}|c_s)$ and $p(\mathbf{x}|c_h)$ are calculated as a product of the probability that each feature belongs to the spam of ham bag-of-words, i.e., $p(\mathbf{x}|c_s) = \prod p(x_i|\, c_s)$
- If $p(c_s|\mathbf{x}) >$ threshold, the email message is classified as a spam

Spam Filtering Block Diagram

Spam Filtering Techniques

- A typical data flow in spam filtering

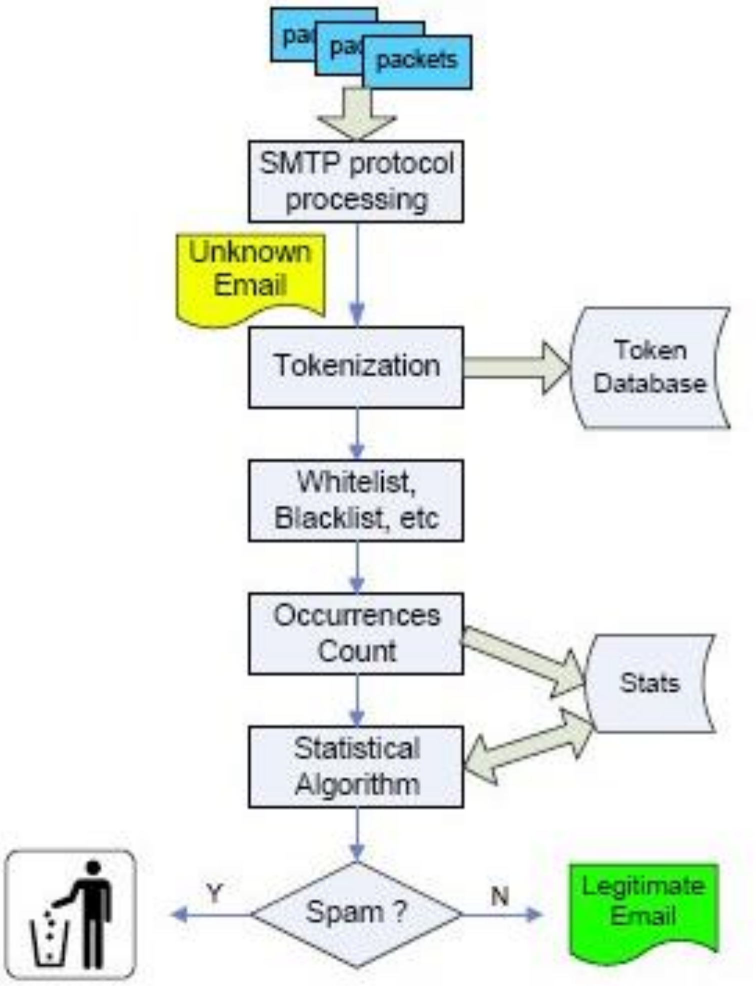


Figure from: Yan (2010) – Workload Characterization of Spam Email Filtering Systems

Sequence Model Approach

- *Sequence models* preserve the order of words in the input text
- As mentioned, commonly used models are Recurrent Neural Networks and Transformer Networks
  - Transformers have replaced RNNs in recent applications
- The application of sequence models typically involves:
  1. Tokenization to represent the words in text data with integer indices
  2. Mapping the integers to vector representations (embeddings)
  3. Pad the sequences in the text to have the same length
  4. Use the padded sequences as inputs to train a machine learning model
- The trained models take into account the ordering of words embeddings in the original text

Word Embedding

Sequence Model Approach

- *Word embedding* is converting words to a vector format, where the vectors represent the position of words in a higher-dimensional space
  - Words that have similar meanings should have close spatial positions of their vector representations in the embedding space
- Typical vectors for representing word embeddings have between 256 to 1,024 dimensions
  - E.g., embedding vector for the word ‘work’

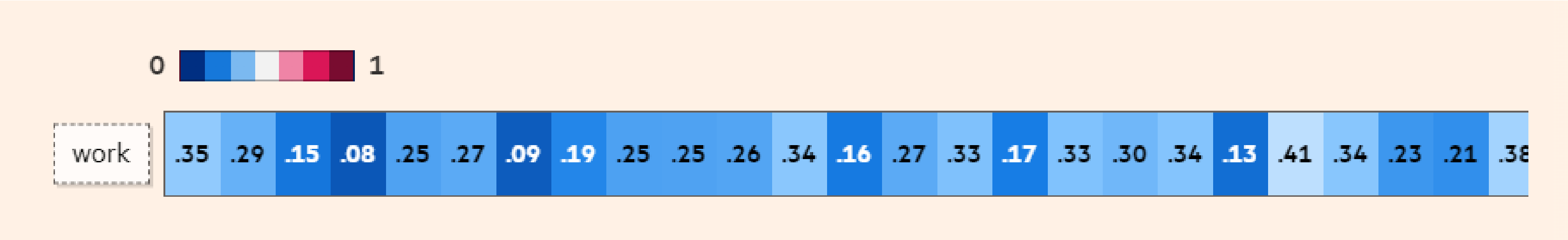


Figure from: Generative AI exists because of the transformer (link)

Word Embedding

Sequence Model Approach

- The figure shows an example of word embeddings space
  - The embedding vectors of words that have similar meanings are also similar
- Typically, the cosine distance between the vectors in the embedding space is used a distance metric
  - For given embedding vectors **u** and **v**, cosine similarity is $\cos\theta = \frac{\mathbf{u}\cdot\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$

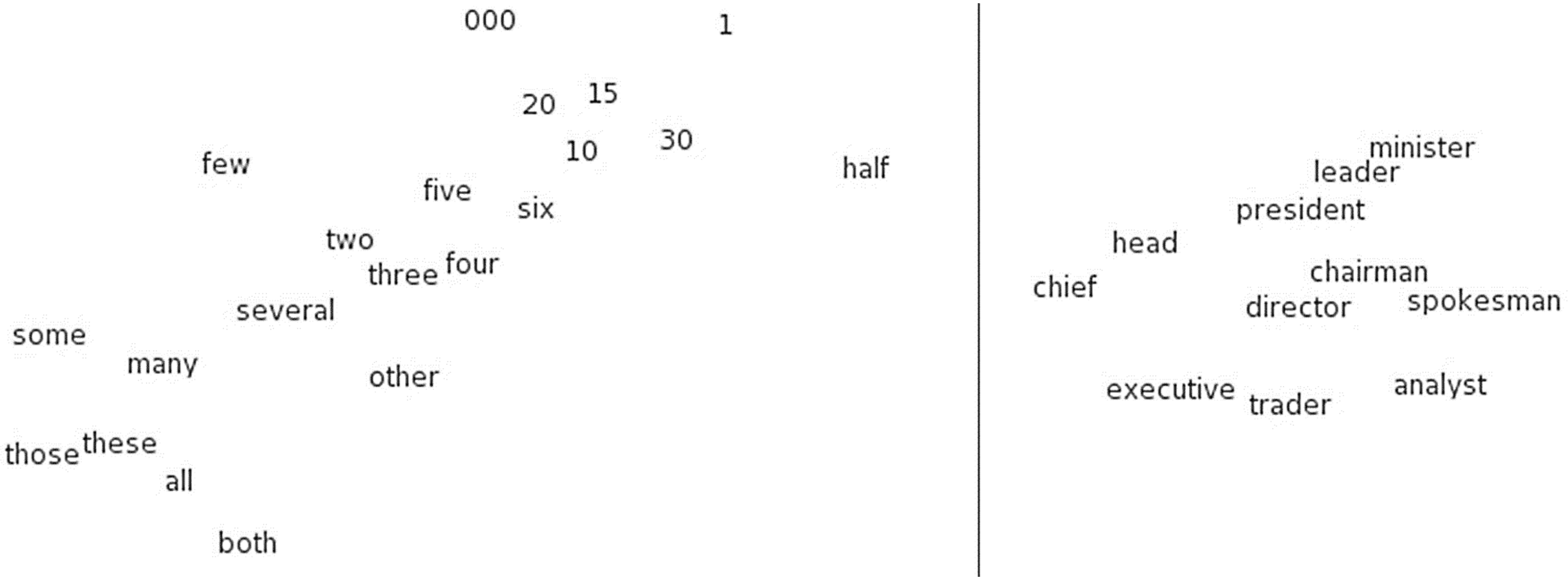
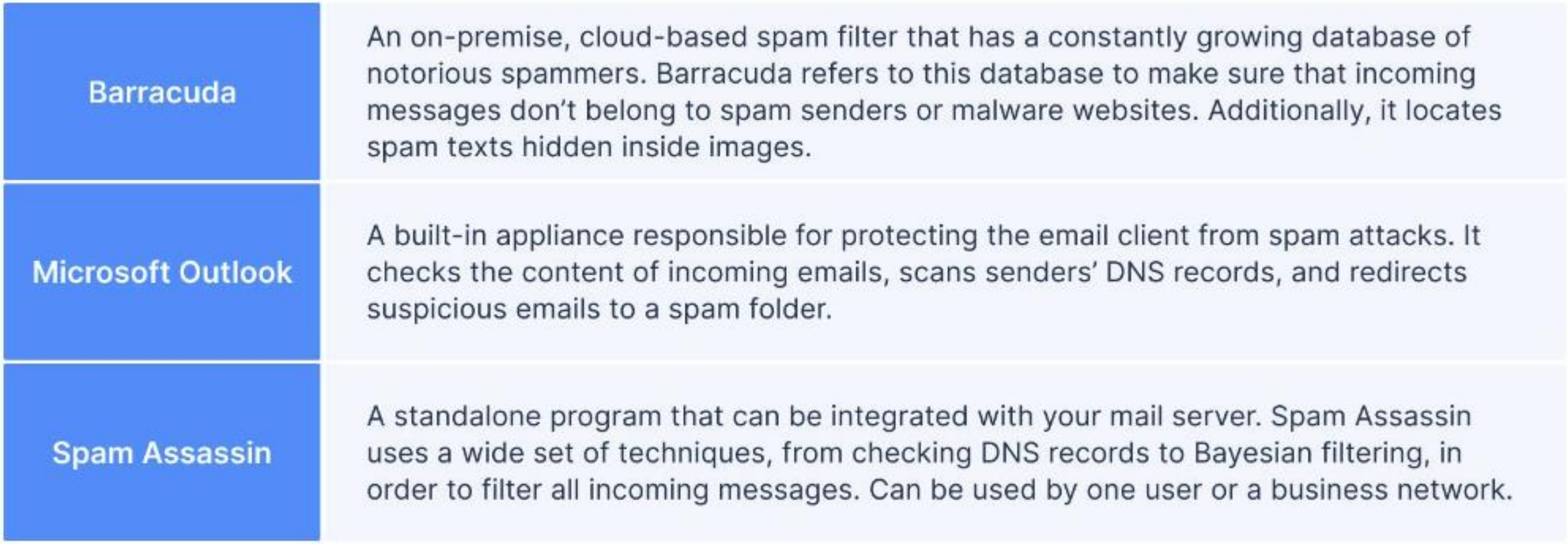


Figure from: How To Design A Spam Filtering System with Machine Learning Algorithm (link)

Commercial Spam Filters

Spam Filtering Techniques

- Examples of three different spam filters solutions
  - Gmail and Outlook use these spam filters



Barracuda	An on-premise, cloud-based spam filter that has a constantly growing database of notorious spammers. Barracuda refers to this database to make sure that incoming messages don't belong to spam senders or malware websites. Additionally, it locates spam texts hidden inside images.
Microsoft Outlook	A built-in appliance responsible for protecting the email client from spam attacks. It checks the content of incoming emails, scans senders' DNS records, and redirects suspicious emails to a spam folder.
Spam Assassin	A standalone program that can be integrated with your mail server. Spam Assassin uses a wide set of techniques, from checking DNS records to Bayesian filtering, in order to filter all incoming messages. Can be used by one user or a business network.

Slide credit: Why Spam Filters Hate You

AML against Spam Filters

Adversarial Attacks against ML-based Spam Filters

- Common approaches for creating adversarial attacks against ML-based spam filters include:
  - ***Bad words obfuscation*** – replace typical words in spam messages with synonyms or misspelled words
  - ***Good words insertion*** – insert into spam messages words that appear in legitimate messages
- ***Huang et al. (2011) Adversarial Machine Learning*** (link)
  - This work introduced an availability attack
    - I.e., the attack makes the spam filter unavailable for regular use
  - Attacked is a model called SpamBayes, which uses Naïve Bayes for spam filtering based on words occurrences in the content of email messages
  - Attack approach:
    - Send spam email messages that contain a very large set of words to the spam filter
    - The spam filter algorithm will recognize the emails as spam, and it will assign a higher spam score to every word in the received messages
    - As a result, future legitimate emails are more likely to be marked as spam

Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

AML against Spam Filters

Adversarial Attacks against ML-based Spam Filters

- ***Biggio et al. (2014) Security Evaluation of Pattern Classifiers under Attack*** (link)
  - White-box availability attack
  - Attacked are two spam classifiers: linear SVM and logistic regression classifier,
    - The classifiers used bag-of-words representation based on text content, where features are word occurrences
  - Attack approach:
    - Use an optimization approach to find most impactful non-spam words
    - Add $n_{\max}$ most impactful non-spam words to spam emails
    - This can cause the spam filter to increase the scores for the impactful good words, and as a result, the model will be more likely to classify non-spam emails as spam

Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

AML against Spam Filters

Adversarial Attacks against ML-based Spam Filters

- ***Sethi and Kantradzic (2018) Data Driven Exploratory Attacks on Black Box Classifiers in Adversarial Domains*** (link)
  - Black-box query-efficient evasion attacks
  - The authors trained 5 different spam classifiers using conventional ML methods: linear SVM, *k*-nearest neighbors, SVM with RBF kernel, decision tree, and random forest
  - SPAMBASE dataset was used for model training and evaluation
  - The authors introduced a framework called SEE (Seed-Explore-Exploit)
  - Attacks include:
    - Anchor Points (AP) attack: anchors are legitimate emails, that serve as a ground-truth for generating adversarial samples by applying perturbation to the legitimate emails
      - The spam filter is queried, and the procedure is repeated until the emails are classified as spam
    - Reverse Engineering (RE) attack: the goal is to discover the decision boundary for spam email classification, and ultimately learn a substitute model based on querying the target classifier
      - The generated samples against the substitute model are then transferred to the target classifier

Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

URL Detection

URL Detection

- ***URL (Uniform Resources Locator)*** is a web address that specifies the location of the webpage on a computer network
- A typical URL http://www.example.com/index.html consists of several components:
  - Protocol type = http
  - Domain name = www.example.com
  - File name = index.html
- The domain name is also referred to as FQDN (Fully Qualified Domain Name)
  - It identifies the server hosting the webpage
  - FQDN can be further divided into a prefix (subdomain) = www, and an RDN (Registered Domain Name) = example.com

Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

Phishing URL Detection

Phishing URL Detection

- ***Phishing*** refers to attacks where a victim is lured to a fake web, and is deceived into disclosing personal data or credentials
  - Most common phishing scam tactics are shown below
- *Phishing URLs* seem like legitimate URLs, and redirect the users to phishing web pages, which mimic the look and feel of their target websites
  - E.g., a fake bank website hopes that the user will enter personal data (password)

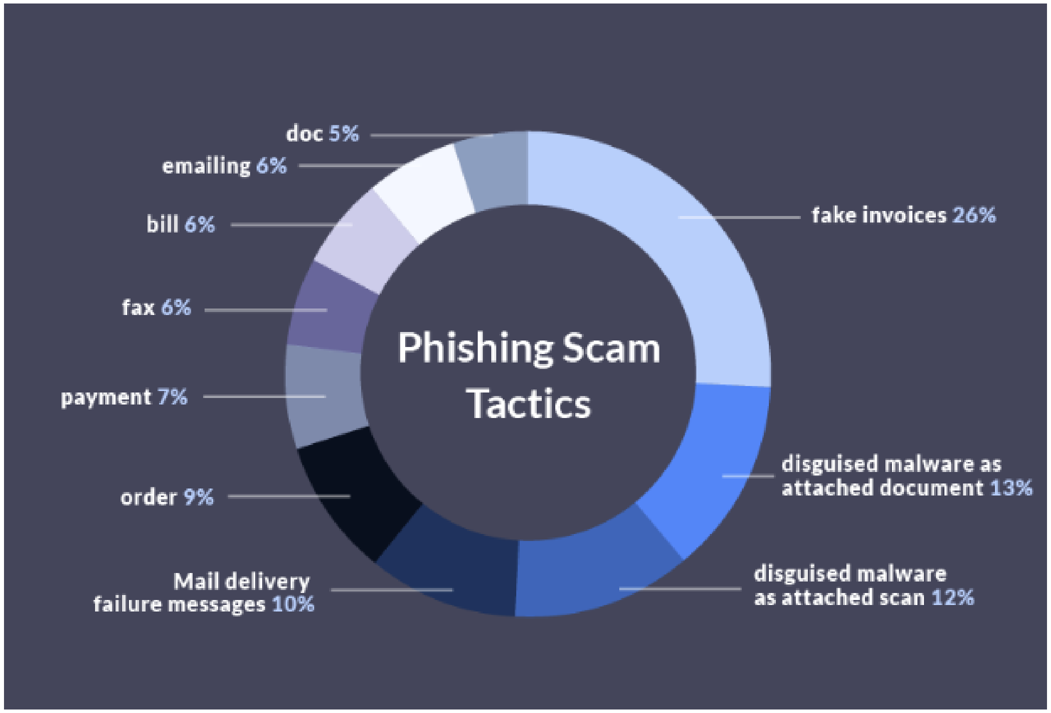


Figure from: https://www.propellercrm.com/blog/email-spam-statistics

Phishing Scam Statistics

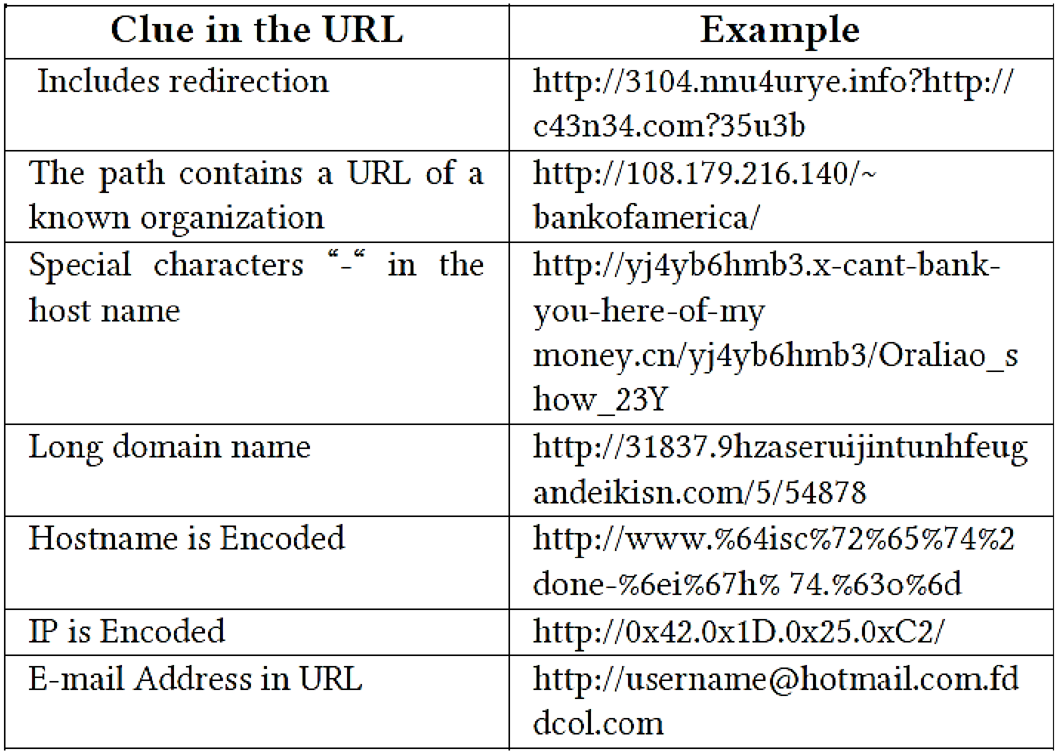
- Google blocks around 100 million phishing emails every day
- 90% of phishing attacks sent via messaging apps are sent through WhatsApp
- Fake invoices are used in 26% of phishing scams
- Top 5 phishing targets in 2022 were:
  - LinkedIn – 52% (i.e., URLs with links that mimic the LinkedIn website)
  - DHL – 14%
  - Google – 7%
  - Microsoft – 6%
  - FedEx – 6%

Slide credit: https://www.propellercrm.com/blog/email-spam-statistics

Phishing URL Detection

Phishing URL Detection

- Phishing emails are a more serious threat than spam emails, because they aim to steal users’ private information, such as bank accounts, passwords, SSNs
- Machine learning techniques are widely used to identify anomalous patterns in URLs as signs of possible phishing
  - Examples of such anomalous patters are shown in the table
- ML-based phishing detection models are usually embedded in web browsers as extensions, or into email spam filtering systems
  - Thus, they appear as a black-boxes to phishers, as it is difficult to identify which features or classification algorithms they use



Clue in the URL	Example
Includes redirection	http://3104.nnu4urye.info?http://c43n34.com?35u3b
The path contains a URL of a known organization	http://108.179.216.140/~bankofamerica/
Special characters “-“ in the host name	http://yj4yb6hmb3.x-cant-bank-you-here-of-my money.cn/yj4yb6hmb3/Oraliao_show_23Y
Long domain name	http://31837.9hzaseruijintunhfeugandeikisn.com/5/54878
Hostname is Encoded	http://www.%64isc%72%65%74%2done-%6ei%67h% 74.%63o%6d
IP is Encoded	http://0x42.0x1D.0x25.0xC2/
E-mail Address in URL	http://username@hotmail.com.fddcol.com

Table from: AlEroud and Karabatis (2020) Bypassing Detection of URL-based Phishing Attacks Using GAN

AML against Phishing URL Detectors

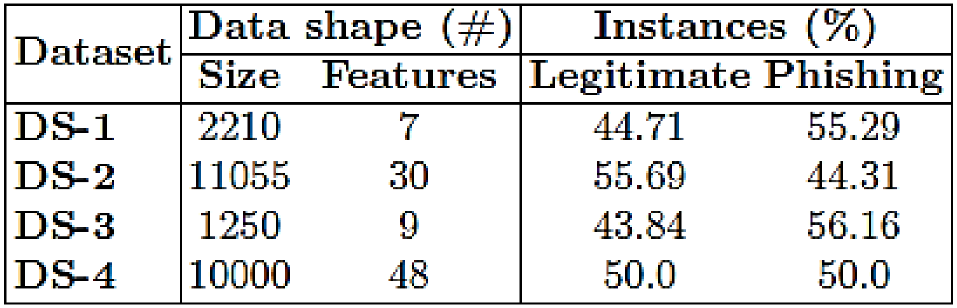
Adversarial Attacks against Phishing URL Classifiers

- ***Bahnsen et al. (2018) DeepPhish: Simulating Malicious AI*** (link)
  - White-box attack
  - Against a character-level LSTM-based phishing URL classifier
    - The classifier was trained using URLs from historical attacks
    - The LSTM model predicts the next character in the URL
  - The attack concatenates benign URLs to the phishing URLs to evade the classifier
    - The form of synthetic URLs is: http:// + compromised_domain + benign_URL
  - Limitation: concatenation od benign URLS can be signed, which makes the attack less effective for real URL detectors

AML against Phishing URL Detectors

Adversarial Attacks against Phishing URL Classifiers

- ***Shirazi et al. (2019) Adversarial Sampling Attacks Against Phishing Detection*** (link)
  - Gray-box attack, requires knowledge of the features used by the ML-based classifier
    - Such knowledge may not be accessible to the attacker, hence, this type of attacks may be less feasible in real-life scenarios
  - Eight features used: domain length, presence of non-alphabetic characters in domain name, ratio of hyperlinks referring to domain, presence of HTTPS protocol, matching domain name with copyright logo, and matching domain name with the page title
  - Try all possible combinations for the values of the features used by the classifier
    - An objective function minimizes: number of manipulated features + assigned feature values
    - The adversarial samples must be visually or functionally similar to the targeted websites
  - Characteristics of used datasets are shown in the table



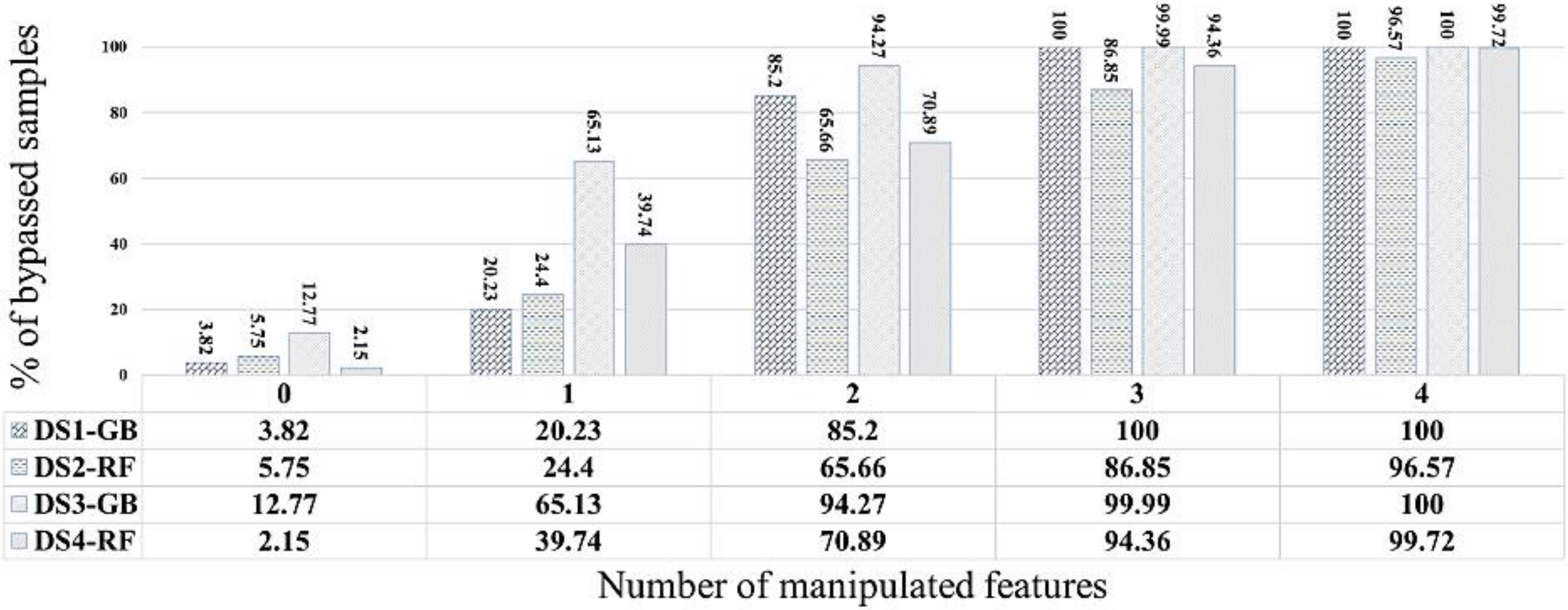
Dataset	Data shape (#)		Instances (%)	
	Size	Features	Legitimate	Phishing
DS-1	2210	7	44.71	55.29
DS-2	11055	30	55.69	44.31
DS-3	1250	9	43.84	56.16
DS-4	10000	48	50.0	50.0

Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

AML against Phishing URL Detectors

Adversarial Attacks against Phishing URL Classifiers

- Shirazi et al. (2019) cont'd
  - The percentage of evaded URL samples increased significantly with only one feature perturbed, and reached 100 when four features were manipulated

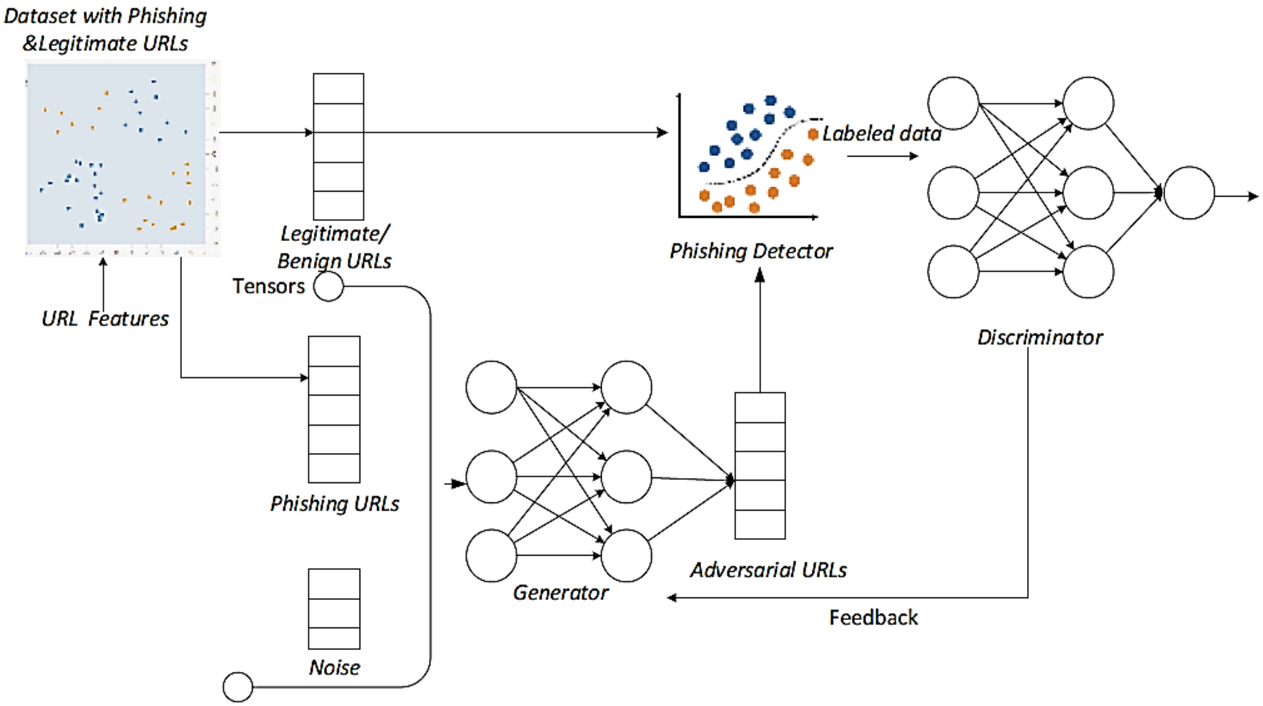


	0	1	2	3	4
▨ DS1-GB	3.82	20.23	85.2	100	100
▨ DS2-RF	5.75	24.4	65.66	86.85	96.57
□ DS3-GB	12.77	65.13	94.27	99.99	100
□ DS4-RF	2.15	39.74	70.89	94.36	99.72

AML against Phishing URL Detectors

Adversarial Attacks against Phishing URL Classifiers

- ***AlEroud and Karabatis (2020) Bypassing Detection of URL-based Phishing Attacks Using Generative Adversarial Deep Neural Networks*** (link)
  - Black-box evasion attack
  - Employs a GAN model to generate phishing URLs to evade ML phishing detectors
    - The generator used a perturbed version of phishing URLs and converted them to adversarial examples



Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

Cyber-physical Systems (CPS)

Cyber-physical Systems (CPS)

- ***Cyber-physical systems (CPSs)*** consist of hardware and software components that control and monitor physical processes
  - CPS are part of the ***critical infrastructure***, which includes the electric power grid, transportation networks, water supply networks, nuclear plants, telecommunications, etc.
- The increased use of ML-based models for controlling and monitoring CPS makes these systems vulnerable to adversarial attacks
  - Adversarial manipulation of sensory data (if undetected) can cause substantial physical and financial damage
    - This can range from major disruptions, power blackouts, to nuclear incidents
  - For instance, AML attacks on manufacturing production systems can cause:
    - Damage to the systems, processes, and equipment
    - Defective products
    - Safety threat to employees
    - Lost production time
    - Unscheduled maintenance (due to false alarms)

Industrial Control Systems (ICS)

Industrial Control Systems (ICS)

- ***Industrial Control Systems (ICS)*** are a subcategory of CPS
  - ICS commonly consist of sets of connected devices, such as PLCs, sensors, and actuators
- The recent trend of connected devices and cloud-based services in ICS exposes these systems to increased risk of cyber attacks
- Controllers make decisions for regulating process parameters based on readings from critical sensors
    - E.g., increase or decrease temperature to ensure it is within a target range
  - The ability of a malicious actor to manipulate sensory data in ICS can have catastrophic impacts on the control systems
- Stealthy attacks via injecting false sensory readings can cause the controller to place the system into an unsafe mode of operation
  - E.g., the attacker may compromise the computer network in a nuclear plant, and sends low temperature sensor readings, causing the controller to heat up the reactor above safe levels

Industrial Control Systems (ICS)

- The main components of ICS are depicted in the figure
  - Equipment – includes various field devices have sensors and actuators (motors, hydraulic or pneumatic cylinders), such as robots, machines, CNCs, etc.
  - PLCs (Programmable Logic Controllers) – industrial microcomputers that collect input data from local sensors and output control signals to actuators
  - SCADA (Supervisory Control And Data Acquisition) – is a central computer station that gathers information from multiple PLCs and manages the operation of the system
  - HMI (Human Machine Interface) – an interface for human operators to monitor and control the system

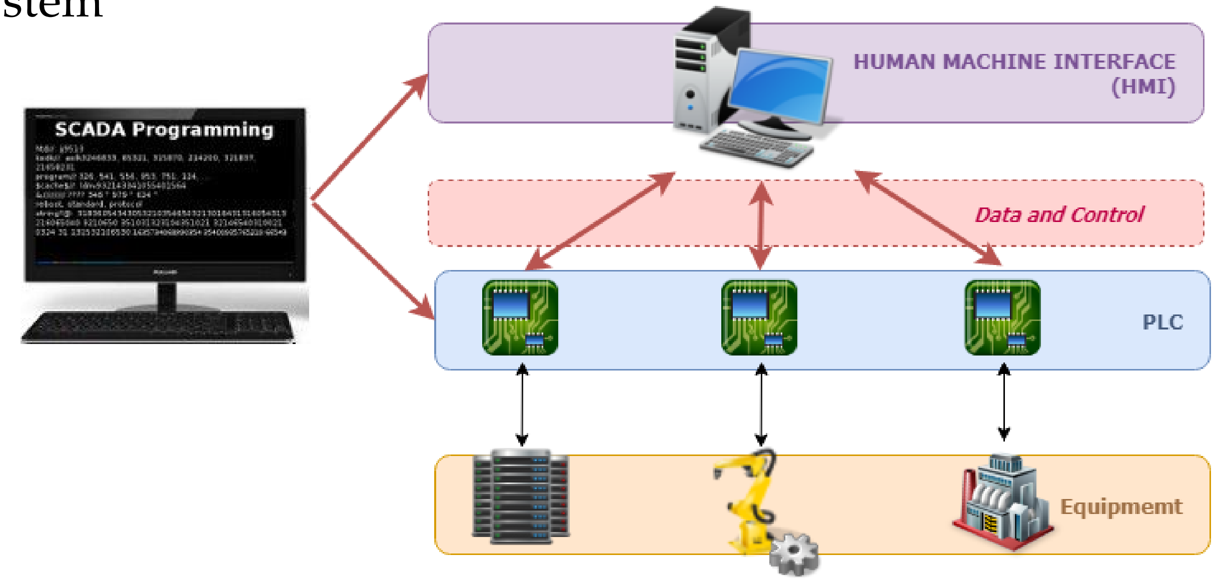


Figure from: Learn SCADA Software Programming for Remote Monitoring

Machine Learning-based CPS

Machine Learning-based CPS

- ***ML-based anomaly detection models*** are commonly used for monitoring the conditions in CPS, and for detecting abnormal conditions or system failures
  - E.g., in the figure, the condition monitoring system (CMS) will stop the production process if one of the modules fails
- In this case, CMS is implemented by training the ML model on historical process data from production modules, to learn the characteristics of normal and abnormal system conditions
- Adversarial attacks can manipulate the physical system without being detected by the CMS
  - Another objective of the attacker may be to trigger false alarms (to temporary stop the production)

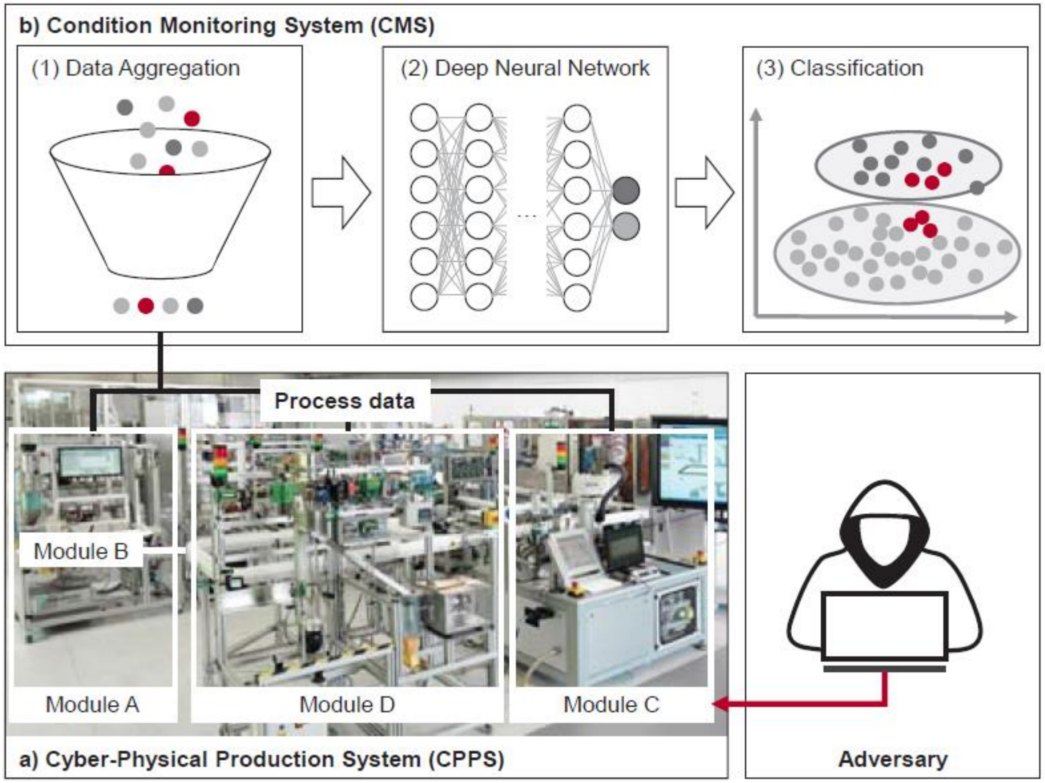


Figure from: Specht et al. (2021) – Generation of Adversarial Examples to Prevent Misclassification in Cyber-Physical Production Systems

Major CPS Attacks

- List of the most notorious attacks on CPS
  - **Stuxnet (2010)** – a malware attack on ICS around the world, e.g., it disrupted the uranium centrifuges in an Iranian nuclear plant
  - **New York dam attack (2013)** – a group of Iranian hackers accessed the Bowman Dam, but they didn't do any damage
  - **German steel mill (2014)** – the attackers caused extensive damage to the steel mill, by preventing the blast furnace from shutting down
  - **Ukraine power grid (2015 and 2016)** – a Russian-based cyber attack remotely disabled power stations and left about 250,000 customers without electricity for 6 hours (in 2015) and 3 hours (in 2016)
  - **Unknown water plant (referred to as Kemuri) (2016)** – attack on PLCs for controlling the valves used for water treatment chemical processing
  - **Water plant in Florida (2021)** – an attempted attack to poison the water by increasing the level of one chemical hundredfold, it was discovered immediately and corrected
  - **JBS Meat Processor (2021)** – a ransomware attack on the world's largest meat processor forced shutdown on 9 plants in US, JBS paid the requested $11M
  - **Change Healthcare (2024)** – a ransomware attack on software for prescriptions, the hackers were paid $22 M
  - **Colonial Pipeline (2021)** – details on the next page

Slide credit: 14 Major SCADA Attacks and What You Can Learn From Them

Colonial Pipeline Ransomware Attack

Major CPS Attacks

- An example of a major disruption by a CPS cyberattack is the ransomware attack on Colonial Pipeline on May 7, 2021
  - This was the largest cyberattack on an oil infrastructure in the U.S. history
  - This is not an AML attack, since there was no ML-based systems involved
- Colonial Pipeline carries gasoline, diesel, and jet fuel in Southeastern U.S.
  - About 45% of all fuel consumed on the East Coast arrives via their pipeline system
- The hackers attacked the billing system of the company
  - The attackers also stole 100 GB of data, and threatened to release it on the internet
    - The attackers cracked the password to the company's computer network
    - A key mistake: the company didn't use two-factor authentication
- The caused disruption resulted in a 6 days shutdown of the pipeline, leading to fuel shortage at gas stations, canceled flights
- Colonial Pipeline paid the requested ransom of $4.4 million
  - The hackers then sent the company a software application to restore their network
  - Fortunately, FBI was able to recover $2.3 million from the ransom payment

AML against CPS

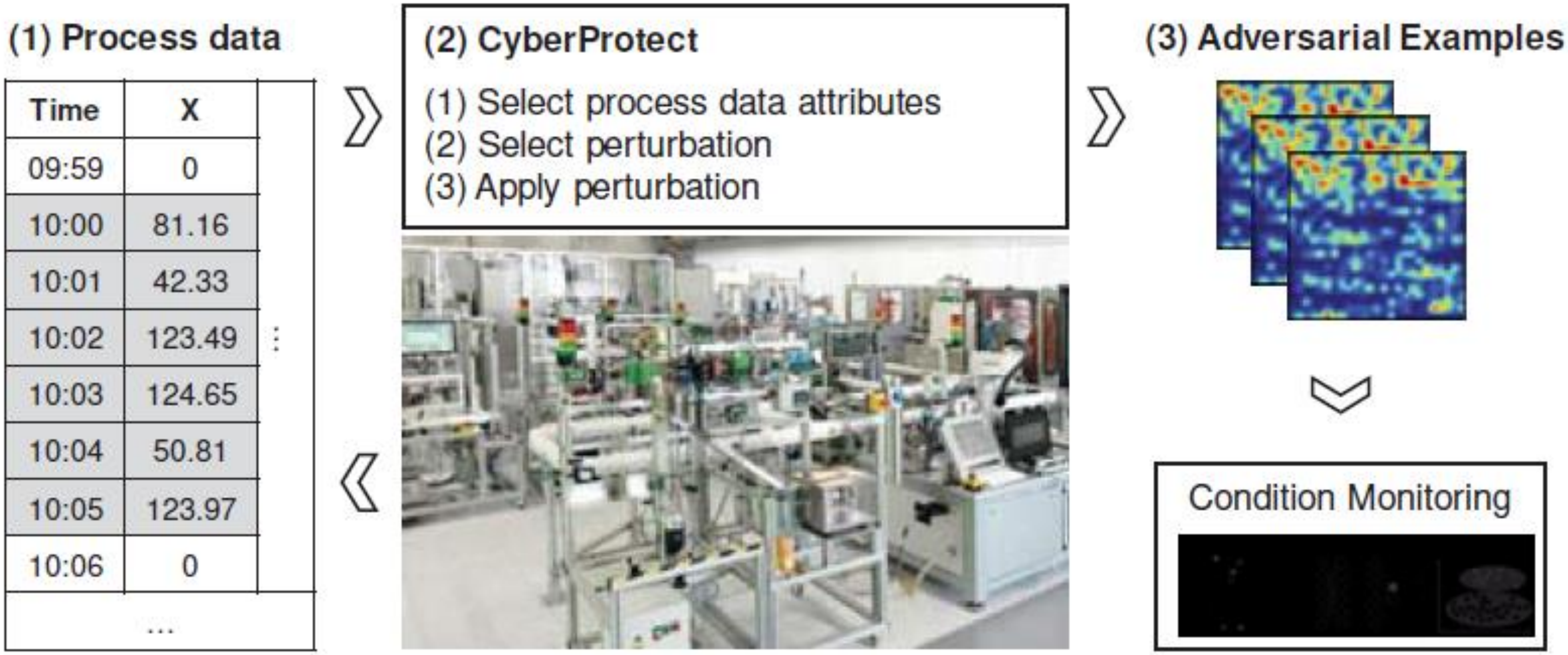
Adversarial Attacks against Cyber-physical Systems

- ***Specht et al. (2018) Generation of Adversarial Examples to Prevent Misclassification of Deep Neural Network based Condition Monitoring Systems for Cyber-Physical Production Systems*** (link)
  - Application: monitoring a process for manufacturing semi-conductors
  - White-box evasion attack
  - Dataset: SECOM, recorded from a semi-conductor manufacturing process
    - Each data instance contains 590 features collected from the manufacturing sensors
    - The dataset contains 1,567 samples, labeled as either normal or anomalous production cycle
  - Attacked model: a deep NN consisting of fully-connected layers
    - The model is used for anomaly detection, i.e., it detects anomalous conditions in the collected sensory data
  - FGSM attack was used to generate adversarial samples, that were classified by the ML model as normal sensory data

AML against CPS

Adversarial Attacks against Cyber-physical Systems

- Specht et al. (2018) cont'd
  - The work also introduces a defense approach called CyberProtect
    - It uses the generated adversarial samples to retrain the DNN model (with both clean samples and adversarial samples)
    - This defense approach increased the classification accuracy of the DNN model from 20% to 82%



(1) Process data

Time	X	
09:59	0	
10:00	81.16	
10:01	42.33	
10:02	123.49	⋮
10:03	124.65	
10:04	50.81	
10:05	123.97	
10:06	0	
...		

(2) CyberProtect

(1) Select process data attributes
(2) Select perturbation
(3) Apply perturbation

AML against CPS

Adversarial Attacks against Cyber-physical Systems

- ***Ghafouri et al. (2018) Adversarial Regression for Detecting Attacks in Cyber-Physical Systems*** (link)
  - Application: controlling a process for manufacturing liquid products
  - Gray-box evasion attack
  - Attacked models: 3 ML models used for anomaly detection
    - These include: linear regression, NN, and an ensemble of LR and NN
  - Dataset: TE-PCS, containing sensory data from the production process
    - The data instances have 41 sensory measurements and 12 controlled outputs
  - Attack approach:
    - Mixed-integer linear programming (MILP) is used for generating adversarial examples
    - A challenge for this task is that there are safety constraints for the pressure and temperature readings of the sensors (the generated adversarial samples must obey these constraints)
      - Therefore, the attack was formulated as a constrained optimization problem via MILP

Rosenberg et al. (2021) – AML Attacks and Defense Methods in the Cyber Security Domain

AML against CPS

Adversarial Attacks against Cyber-physical Systems

- ***Feng et al. (2017) A Deep Learning-based Framework for Conducting Stealthy Attacks in Industrial Control Systems*** (link)
  - Applications: monitoring and controlling a small lab-scale gas pipeline, and a water treatment plant
  - Gray-box evasion attack
  - Attacked model: an LSTM-based anomaly detector
  - Datasets:
    - Gas pipeline data: 68,803 time series with 11 sensory measurement data
    - Water treatment plant dataset: 496,800 signals having 51 sensors measurements
  - Attack:
    - A GAN model is used to generate malicious sensor measurements to bypass the anomaly detector
    - Attacks on both sensor and control channels of the PLC were designed
    - The success rate can reach up to 90%, depending on the number of manipulated sensor measurements

Constrained Concealment Attacks against Reconstruction-based Anomaly Detectors in Industrial Control Systems

A Erba, R Taormina, S Galelli, M Pogliani, M Carminati, S Zanero, NO Tippenhauer

Proceedings of the 36th Annual Computer Security Applications Conference, 2020

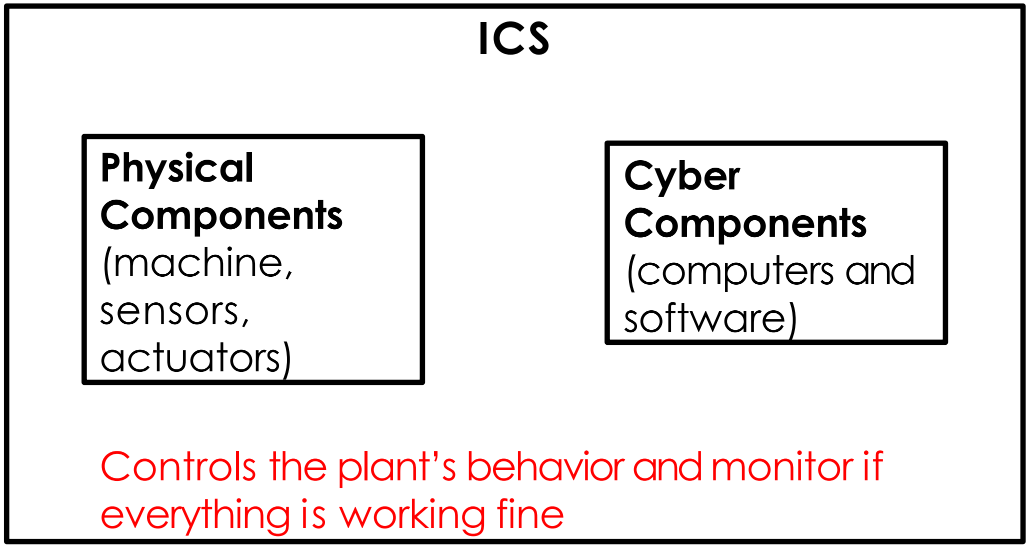
Presented By

Shubham Pandey

01st April

BACKGROUND

Industrial Control Systems (ICS) are systems used to control and monitor industrial processes — like water treatment, power grids, or manufacturing.



ICS

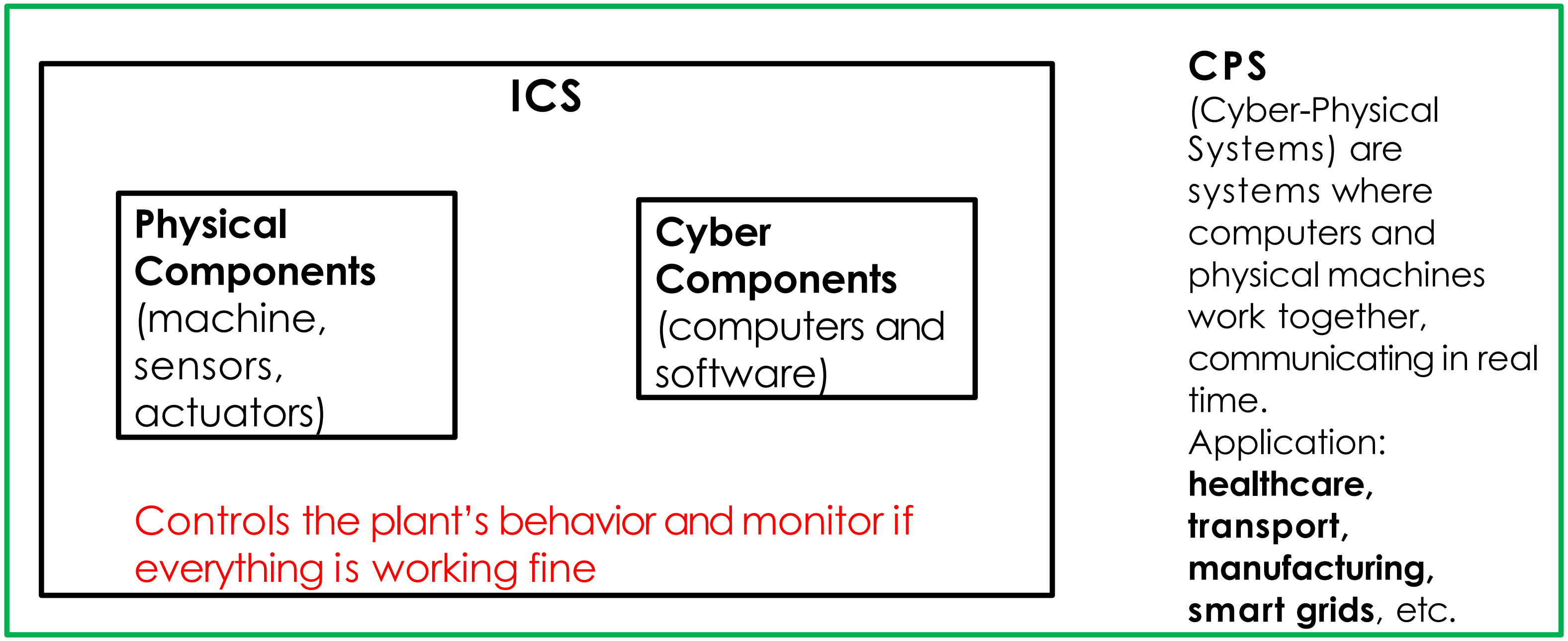
Physical Components (machine, sensors, actuators)

Cyber Components (computers and software)

Controls the plant’s behavior and monitor if everything is working fine

Contd.

Industrial Control Systems (ICS) are systems used to control and monitor industrial processes — like water treatment, power grids, or manufacturing.



ICS

Physical Components (machine, sensors, actuators)

Cyber Components (computers and software)

Controls the plant's behavior and monitor if everything is working fine

CPS (Cyber-Physical Systems) are systems where computers and physical machines work together, communicating in real time.

Application: **healthcare**, **transport**, **manufacturing**, **smart grids**, etc.

Contd.

Industrial Control Systems (ICS) are systems used to control and monitor industrial processes — like water treatment, power grids, or manufacturing.

- **PLCs (Programmable Logic Controllers)** read sensor data and decide what the actuators should do.
- **SCADA (Supervisory Control and Data Acquisition)** collects all this data, shows it to human operators, and sometimes stores it.

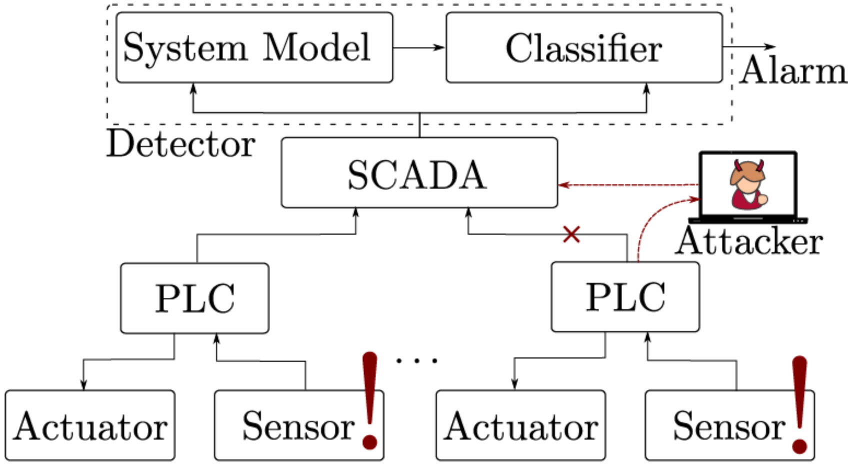
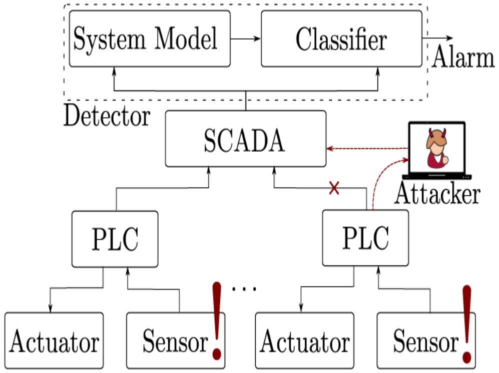


Figure 1: High level system and attacker model. The PLCs report sensor data about the anomalous process to the SCADA. The attacker can eavesdrop and manipulate a subset of the data provided to the SCADA. The reconstruction-based detector attempts to detect attacks based on a learned model of the system's benign operations.



Stanford | Center for International Security and Cooperation
Freeman Spogli Institute

Stuxnet was the name given to a highly complex digital malware that targeted, and physically damaged, Iran’s clandestine nuclear program from 2007 until its cover was blown in 2010 by computer security researchers. The malware targeted the computer systems controlling physical infrastructure such as centrifuges and gas valves.

Picture Courtesy: Space Imaging/Inta SpaceTurk shows the once-secret Natanz nuclear complex in Natanz, Iran

Evasion Attacks

An evasion attack is when an attacker creates special inputs (adversarial examples) that fool a trained machine learning model into making wrong predictions.

Levels of attacker knowledge

- **White-box attack** (Perfect knowledge): Has full access to (**D, X, f, w**)
- **Gray-box attack** (Limited knowledge): Has partial knowledge, like (D, **X, f,** w)
- **Black-box attack** (Zero knowledge): Has only estimates or guesses: (D, X, f, w)

where, training dataset D, the feature set X, the learning algorithm f, and the trained parameters w.

Contd.

An evasion attack is when an attacker creates special inputs (adversarial examples) that fool a trained machine learning model into making wrong predictions.

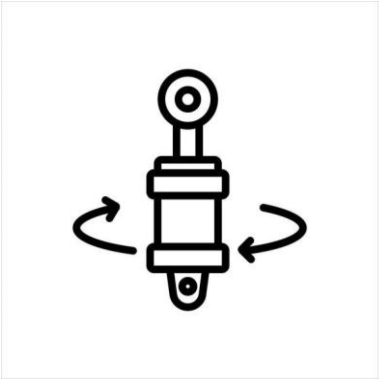
How the attack works?

By solving an optimization problem

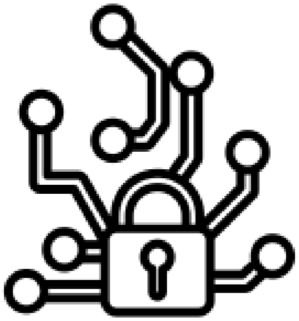
- The attacker changes the input just enough so the model gets fooled, but the input still looks normal
- This is done by minimizing distance measures (like L_0, L_2, or L_∞ norms) between original and adversarial input

Content of this Paper

System Model

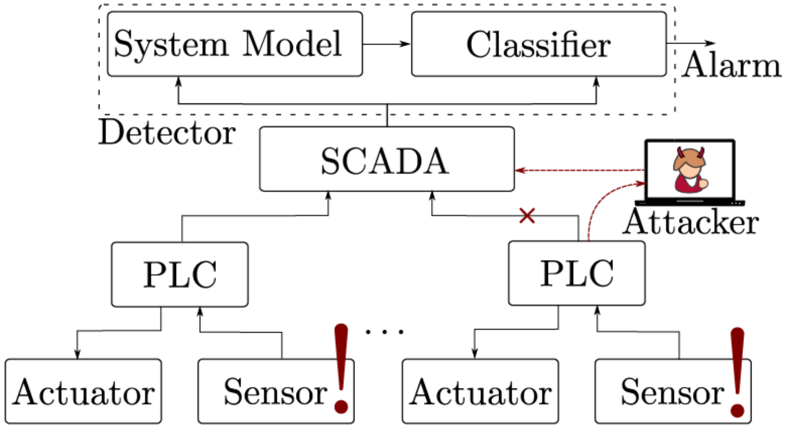


Attacker Model



General problem statement for constrained concealment attacks

System Model



- The realistic industrial system is made up of:
  - ➢ Sensors + Actuators
  - ➢ PLCs
  - ➢ SCADA System

Working (reference to water distribution (WADI) dataset)

1. All Sensors and Actuators are connected to PLC
2. All PLC deliver information to SCADA that is similar to control center that invigilates everything and decides whether the system is behaving normal or abnormal
3. Equipped with reconstruction-based classifier that predicts if it’s an attack or normal process at each time step

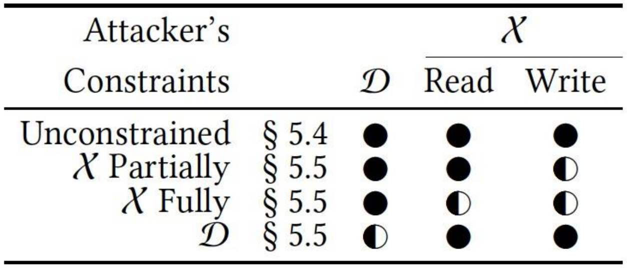
Attacker Model

1. **Attacker Goal and Capabilities**

The attacker subtly modifies sensor data in real time to hide anomalies from the detector, making the ICS appear normal even during the attack.

Goal: *Hide anomalies, trick the system*

Capabilities: *Access to ICS network, eavesdrop on PLC-SCADA communication, perform attack on real time*

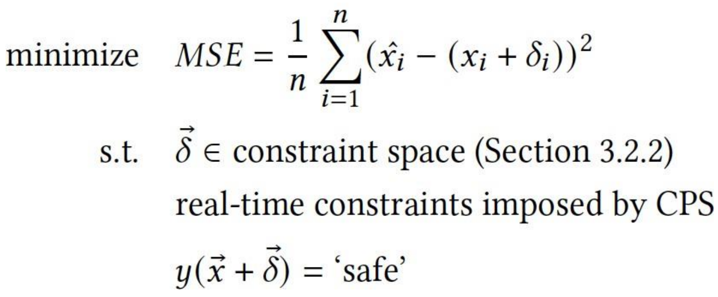


Attacker's			$\mathcal{X}$	
Constraints		$\mathcal{D}$	Read	Write
Unconstrained	§ 5.4	●	●	●
$\mathcal{X}$ Partially	§ 5.5	●	●	◐
$\mathcal{X}$ Fully	§ 5.5	●	◐	◐
$\mathcal{D}$	§ 5.5	◐	●	●

Working:

anomalous vector, x

Normally, the detector classifies, y(x) = "under attack"



$$\begin{aligned} \text{minimize} \quad & MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - (x_i + \delta_i))^2 \\ \text{s.t.} \quad & \vec{\delta} \in \text{constraint space (Section 3.2.2)} \\ & \text{real-time constraints imposed by CPS} \\ & y(\vec{x} + \vec{\delta}) = \text{‘safe’} \end{aligned}$$

(even though the system is under attack)

Attacker Model

2. **Attacker Knowledge**

Data tuple (D, X)

1. Unconstrained (D, X)
2. Feature Partially Constrained (D, **X**)
3. Feature Fully Constrained (D, **X**)
4. Data Constrained (**D**, X)

Defense tuple (f, w)

1. White box (**f, w**)
2. Black box (f, w)

Framework for Attack Computation

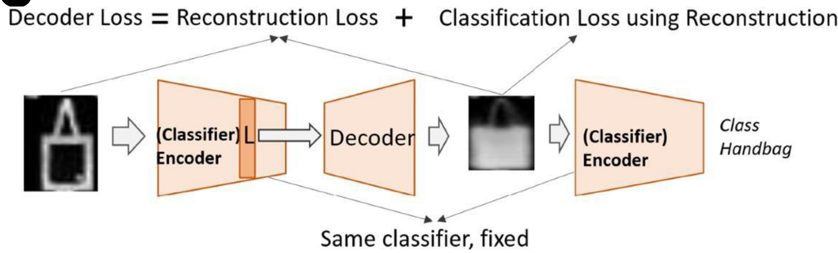
White-Box: Iterative Attack

- ➢ Attacker knows the model (has access to the anomaly detector).
- ➢ Uses an iterative algorithm to:
  - Query the detector (oracle) repeatedly.
  - Decide which sensor values to change and how much.
- ➢ Goal: make input look safe to the anomaly detector.
- ➢ Tunable: attacker can balance between speed and accuracy.
- ➢ Operates in real-time, more precise, but computationally heavier. (example PGD)

Black-Box: Learning-Based Attack

- ➢ Attacker doesn't know the model.
- ➢ Trains a Deep Neural Network to learn what "normal" looks like.
- ➢ At attack time:
  - The neural network modifies anomalous data to resemble normal.
  - Sends this data to SCADA.
- ➢ No queries to the model needed → fully black-box.
- ➢ Fast and real-time, but may be less accurate.

Design of Concealment Attacks



Reconstruction-based Attack detector

A **Deep Learning Autoencoder Model:** The model is trained on normal operating sensor readings of an ICS. The training objective is to optimize the Mean Squared Error (MSE) loss between the input (original sensor data) and the output (reconstructed sensor data)

$$\min_{\phi} \mathbb{E}_{(X,\vec{s}_t)\sim\mathcal{D}} \left[||DA(\phi,|X), \vec{s}_t||_2^2 \right]$$

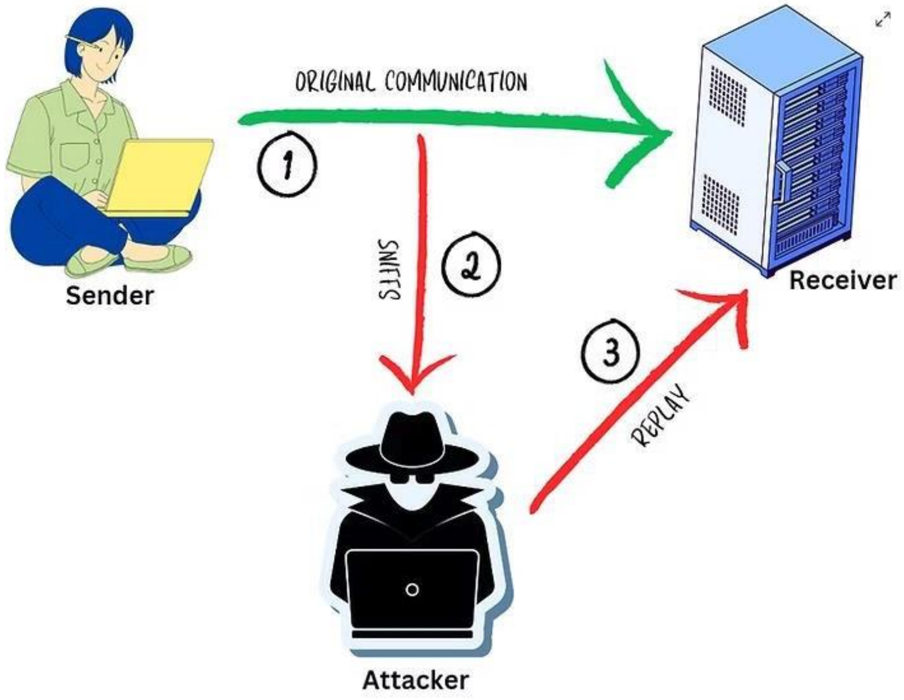
A **Classifier Function:** This function determines if the system is under attack based on the reconstruction error produced by the autoencoder

$$\varepsilon(\vec{e}) = \frac{1}{n}\sum_{i=1}^{n} {d_i}^2, \qquad y(X) = \begin{cases} \text{'under attack' if } \varepsilon(\vec{e}) > \theta \\ \text{'safe' otherwise} \end{cases}$$

Contd.

Replay Attack (Baseline)

- At this stage, the attacker does not possess knowledge of how the anomaly detection is performed.
- To avoid being detected, the attacker's strategy is to replay sensor readings that were recorded during a period when the system was not experiencing any anomalies.



Contd.

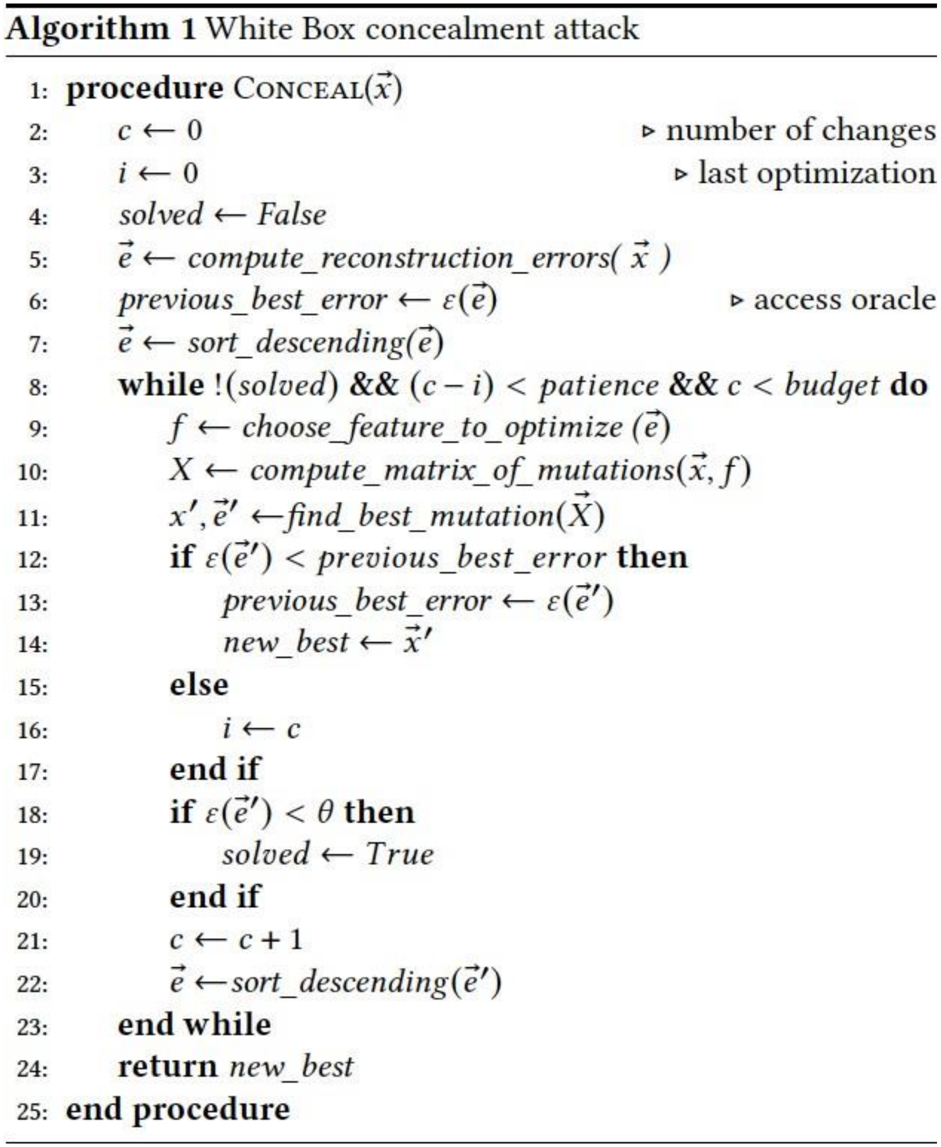
Iterative Attack

Key aspects of White box attack:

1. Oracle Access
2. Iterative Optimization
3. Mean Squared Error (MSE) Minimization

   $\varepsilon (e') < \theta$ (i.e. **$y(x + \delta)$ = 'safe'**

   (coordinate descent algorithm)
4. Constraint Awareness
5. Stopping Criteria
6. Heuristic for Modification
7. Real-time Applicability



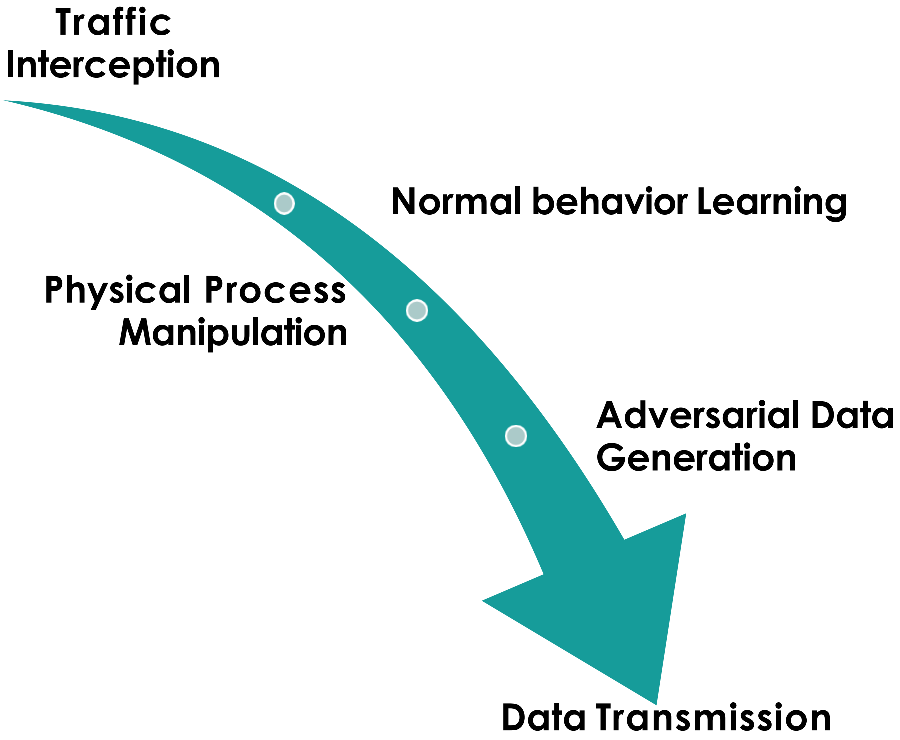
Algorithm 1 White Box concealment attack

```
procedure CONCEAL($\vec{x}$)
    $c \leftarrow 0$                                        ▷ number of changes
    $i \leftarrow 0$                                        ▷ last optimization
    $solved \leftarrow False$
    $\vec{e} \leftarrow compute\_reconstruction\_errors( \vec{x} )$
    $previous\_best\_error \leftarrow \varepsilon(\vec{e})$              ▷ access oracle
    $\vec{e} \leftarrow sort\_descending(\vec{e})$
    while $!(solved)$ && $(c - i) < patience$ && $c < budget$ do
        $f \leftarrow choose\_feature\_to\_optimize\ (\vec{e})$
        $X \leftarrow compute\_matrix\_of\_mutations(\vec{x}, f)$
        $x', \vec{e}' \leftarrow find\_best\_mutation(\vec{X})$
        if $\varepsilon(\vec{e}') < previous\_best\_error$ then
            $previous\_best\_error \leftarrow \varepsilon(\vec{e}')$
            $new\_best \leftarrow \vec{x}'$
        else
            $i \leftarrow c$
        end if
        if $\varepsilon(\vec{e}') < \theta$ then
            $solved \leftarrow True$
        end if
        $c \leftarrow c + 1$
        $\vec{e} \leftarrow sort\_descending(\vec{e}')$
    end while
    return $new\_best$
end procedure
```

Contd.

Learning Based attack

- ➢ Attacker has no knowledge of the detection model, only that it's reconstruction-based.
- ➢ Can only intercept and manipulate data between PLCs and SCADA.
- ➢ Assumes the detector learns physical behavior of the CPS during training.



Traffic Interception

Normal behavior Learning

Physical Process Manipulation

Adversarial Data Generation

Data Transmission

Implementation and Evaluation

Features	BATADAL Dataset	WADI Dataset
Generation Method	Simulated using epanetCPA (Matlab toolbox)	Data from a real-world ICS testbed
Source	BATADAL competition (2016-2017)	Singapore University of Technology and Design
Normal Data Duration	365 days of simulated normal operations	14 days of normal operations data
Attack Data	14 attacks simulated (across two datasets of 7 attacks each)	15 attacks on physical processes over two days of operations
Number of Sensors	43 sensors (tank levels, pressures, flows, status of valves/pumps)	82 sensors (from processes P1 and P2) (Total 103 in the testbed)
Sampling Frequency	Every 15 minutes (in the researchers' re-created version)	Every second
Variable Types	Both continuous (levels, pressures, flows) and binary (status)	Continuous (as the specific types are not detailed, but implied from sensor readings)

Resource and Evaluation Setup

Evaluation Metric

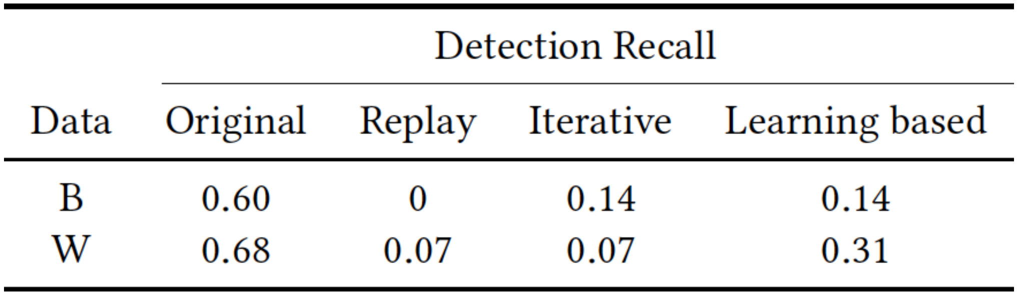
$$Recall = \frac{TP}{TP + FN}$$

- Recall measures the rate at which the anomaly detector correctly identifies anomalous instances
- A higher Recall indicates better anomaly detection performance
- A Recall closer to 0 signifies a higher number of misclassified anomalous instances as 'safe'

Computation Resources

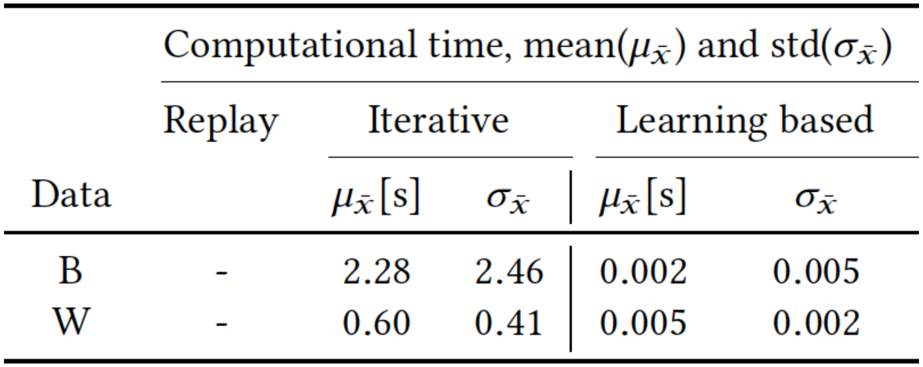
- Both the iterative and learning-based attacks were implemented using Python 3.7.1
- Neural networks used in the learning- based attacks were implemented and trained using Keras 2.3.1 with a TensorFlow 1.11.0 backend
- The experiments were conducted using a laptop

Performance of Unconstrained Concealment Attack



	Detection Recall			
Data	Original	Replay	Iterative	Learning based
B	0.60	0	0.14	0.14
W	0.68	0.07	0.07	0.31

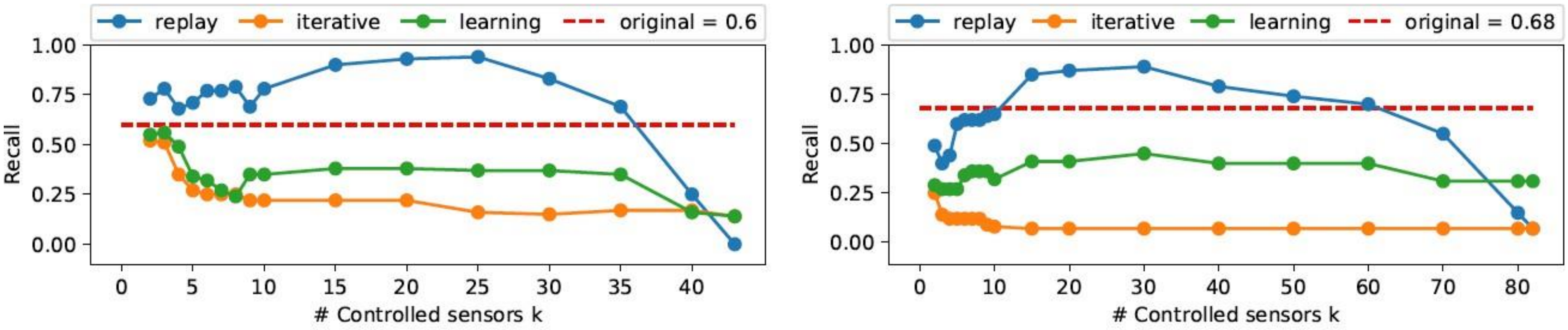
Detector Recall (BATADAL (B) and WADI (W)datasets), before and after unconstrained concealment attacks. The column ‘Original’ refers to the detection Recall over the data without concealment; ‘Replay’, reports the Recall after replay attack, while ‘Iterative’ and ‘Learning based’ columns report the Recall after our proposed adversarial concealment attacks.



	Computational time, mean($\mu_{\bar{x}}$) and std($\sigma_{\bar{x}}$)				
	Replay	Iterative		Learning based	
Data		$\mu_{\bar{x}}$[s]	$\sigma_{\bar{x}}$	$\mu_{\bar{x}}$[s]	$\sigma_{\bar{x}}$
B	-	2.28	2.46	0.002	0.005
W	-	0.60	0.41	0.005	0.002

Average required time (in seconds) to manipulate sensor readings. ‘Replay’ column is empty as replay attacks do not require computation. 'Iterative' and ‘Learning based’ columns report the mean and std deviation required to compute the manipulation sensor readings at a given time step.

Performance of Constrained Concealment Attack

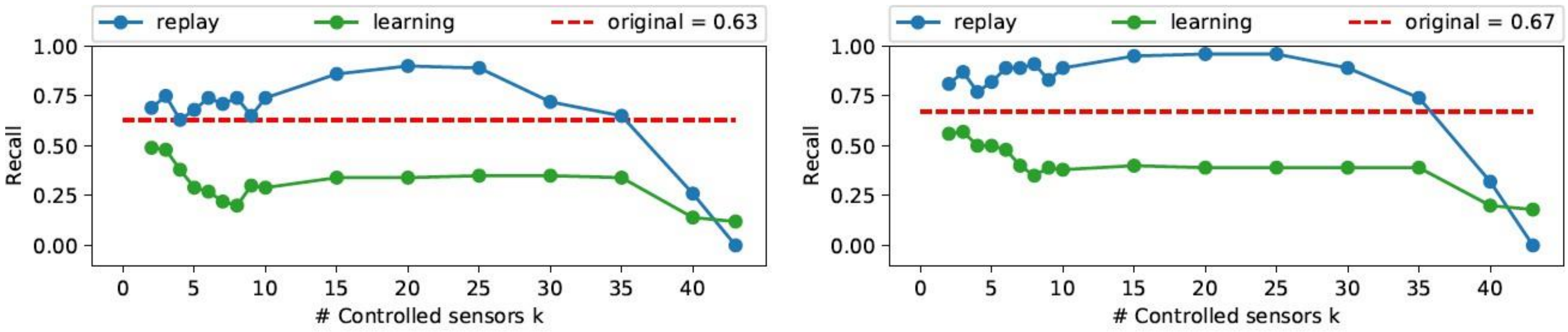


(a) BATADAL

(b) WADI

Impact of Partially Constrained attacker (best-case scenario), comparison between replay attack and our proposed concealment attacks. In the constrained scenario, we notice that replay attack performs bad increasing detector's Recall and raising more alarms than the original data without concealment. This is due to contextual anomalies introduced as part of large-scale replay. Both learning based and iterative approaches outperform the replay attack as they do not introduce contextual anomalies and reduce detector's Recall manipulating few features.

Contd.



(a) LSTM

(b) CNN

Generizability of our proposed Learning based attack compared with replay attack. Attack to LSTM (a) and CNN (b) based defenses on BATADAL dataset.

Conclusion

- Introduced the first real-time concealment attacks using AML principles on reconstruction-based anomaly detectors used in ICS.
- Tackled four major challenges using:
  - Iterative (white-box) attacks with oracle access
  - Learning-based (black-box) attacks using autoencoders
- Tested on two water distribution datasets (e.g., BATADAL, WADI).
- Autoencoder attack needs no model or physical system knowledge
  - Fast execution: ~5ms per input (real-time)
  - Works on-the-fly (every 10 seconds)
- Proved that reconstruction-based detectors are vulnerable, even under realistic constraints.

Critique of the Work

Strength

1. Addressed Novel Problems
2. Detailed Attacker Model
3. Evaluation of Real-World Data
4. Consideration of Real-Time Constraints

Critique of the Work

Limitations

1. Dependence on Datasets
2. More Assumptions
3. Witness behind picking best case
4. Details of Deep Learning Architecture and Constraints on Perturbations

Additional References

1. Rosenberg et al. (2021) – Adversarial Machine Learning Attacks and Defense Methods in the Cyber Security Domain (link)
2. Ganagavarapu (2020) – Applicability of Machine Learning in Spam and Phishing Email Filtering: Review and Approaches (link)
3. Blog Post by Vladislav Podolyako – Why Spam Filters Hate You (link)
4. Daniel Lowd – Adversarial Machine Learning
5. Blog Post by Emily Bauer – 15 Outrageous Email Spam Statistics that Still Ring True in 2018 (link)
6. Blog Post by Sie Huai Gan – How To Design A Spam Filtering System with Machine Learning Algorithm (link)