



University of Idaho

Department of Computer Science

CS 404/504
Special Topics:
Adversarial
Machine Learning

Dr. Alex Vakanski



Lecture 11

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
- Weilin Chen presentation
 - Kuleshov (2018) Adversarial Examples for Natural Language Classification Problems



Spam Filtering

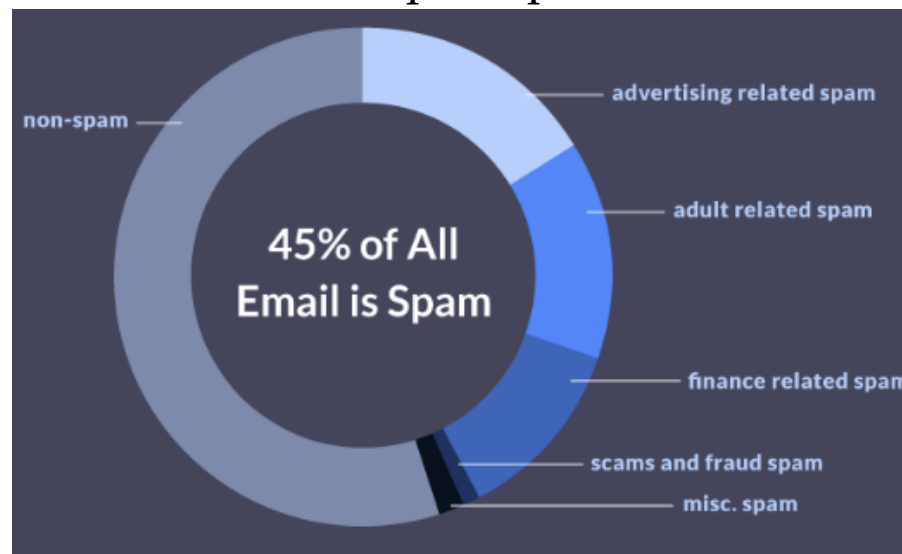
Spam Filtering

- 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

- 45% of all sent emails are spam (information from 2018)
 - 36% of all spam is some form of advertising
- Spam costs \$20.5 billion yearly (reduced network bandwidth, storage capacity)
- About 14.5 billion spam emails are sent 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

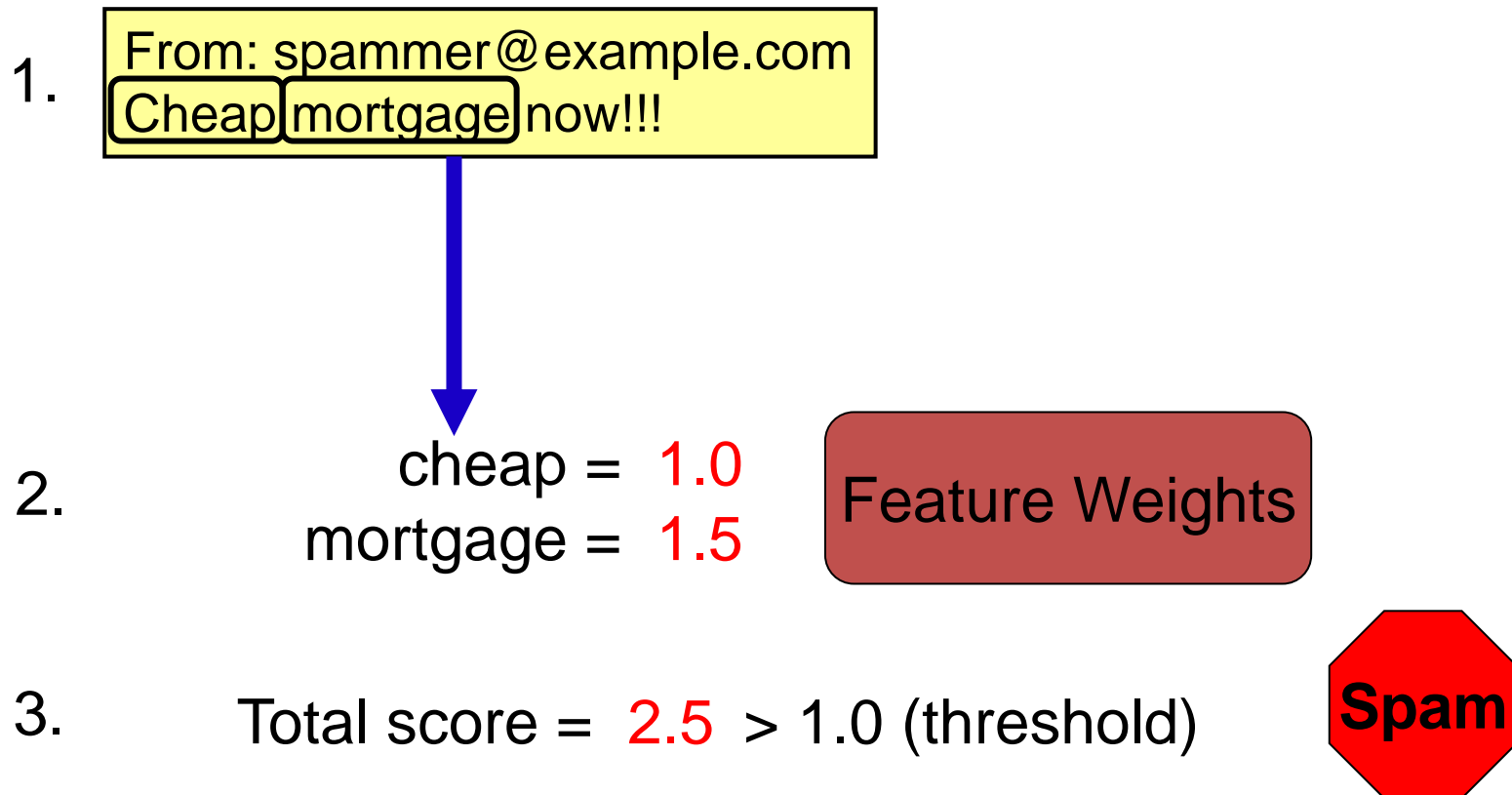




Spam Filtering Adversarial Game

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)

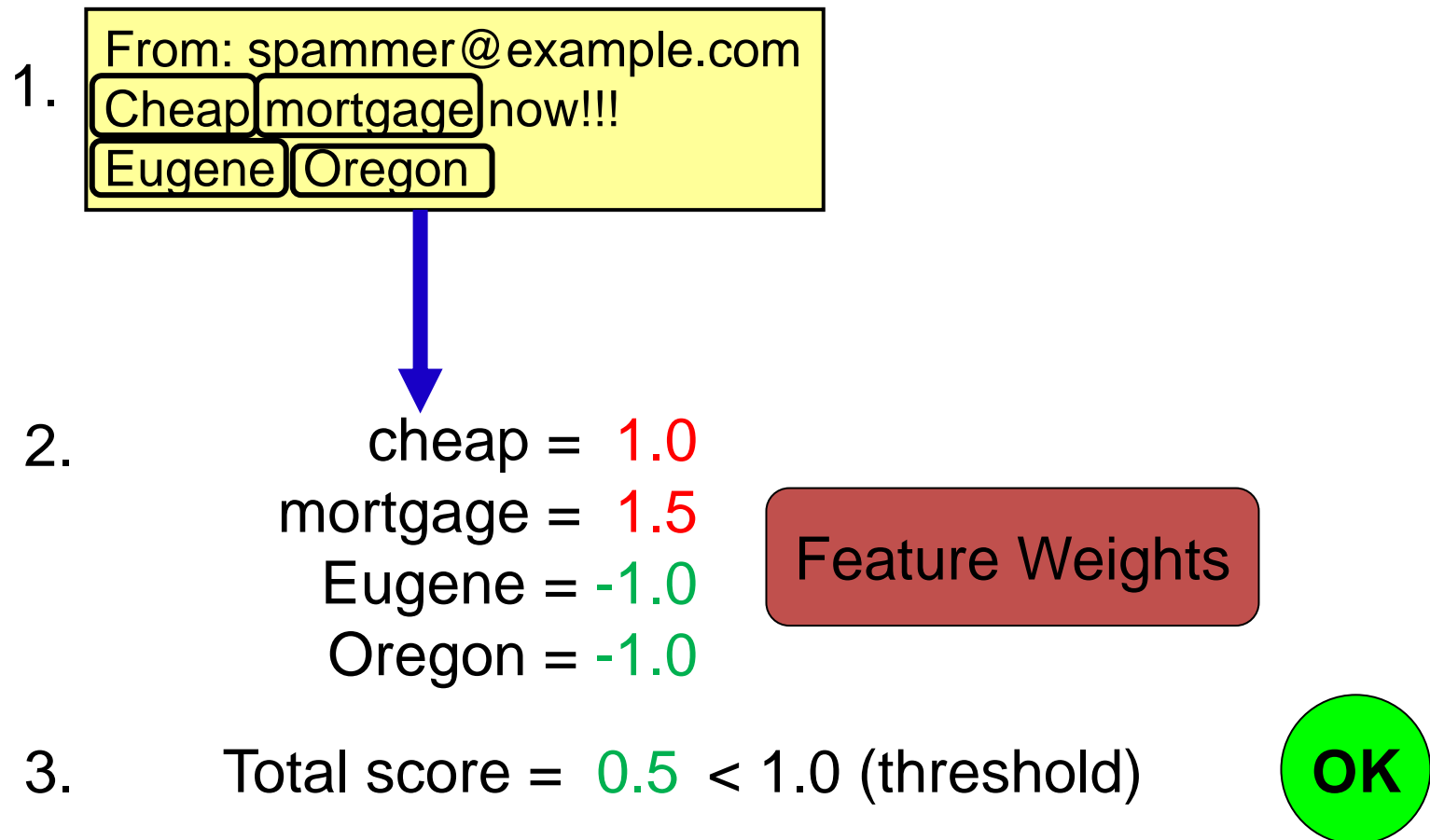




Spam Filtering Adversarial Game

Spam Filtering

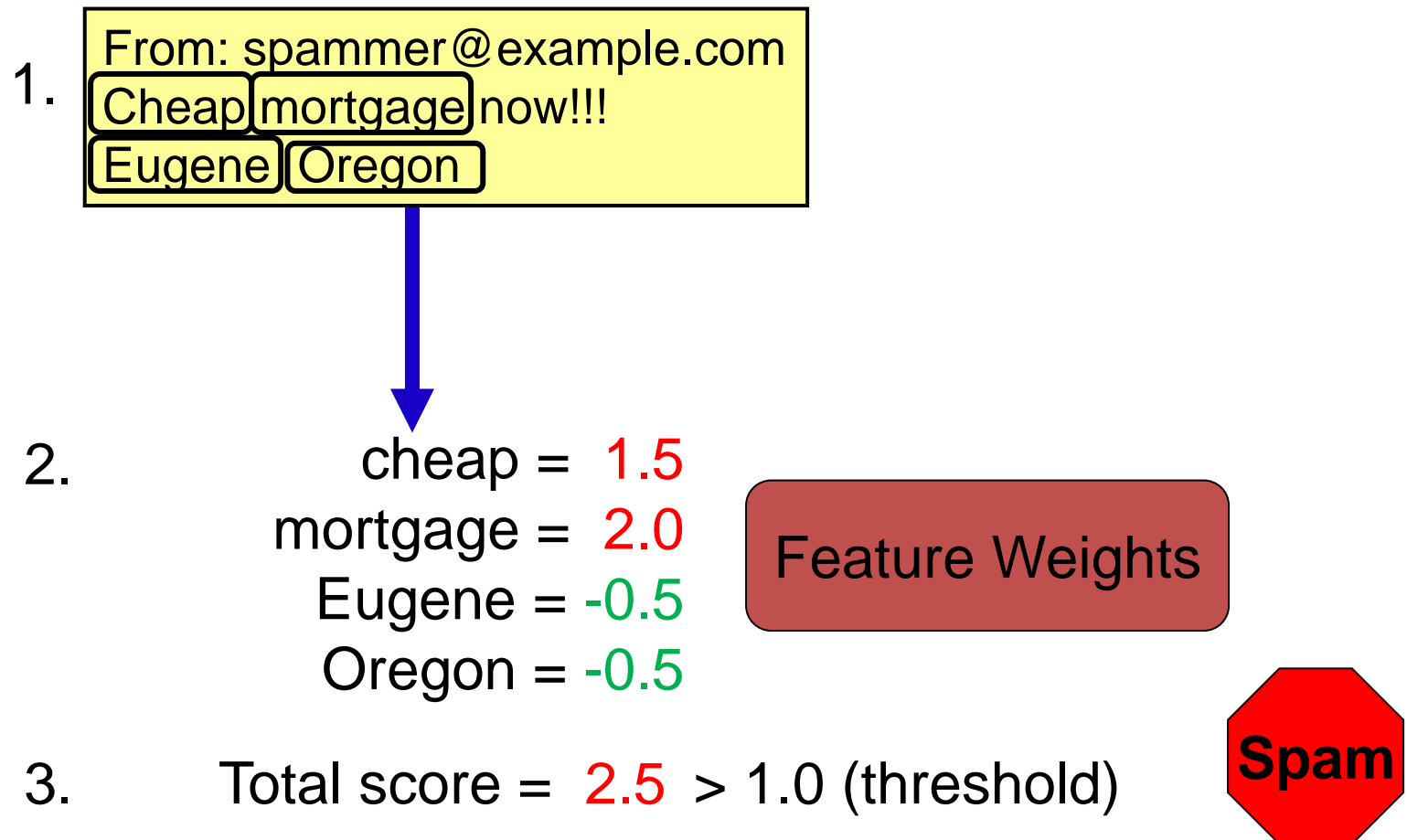
- The spammers adapt to evade the classifier, by adding regular words to reduce the overall score



Spam Filtering Adversarial Game

Spam Filtering

- The spam classifier is updated by changing the feature weights



Spam Filtering Datasets

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



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



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



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

Tokenization

Pre-processing Text in Email Messages

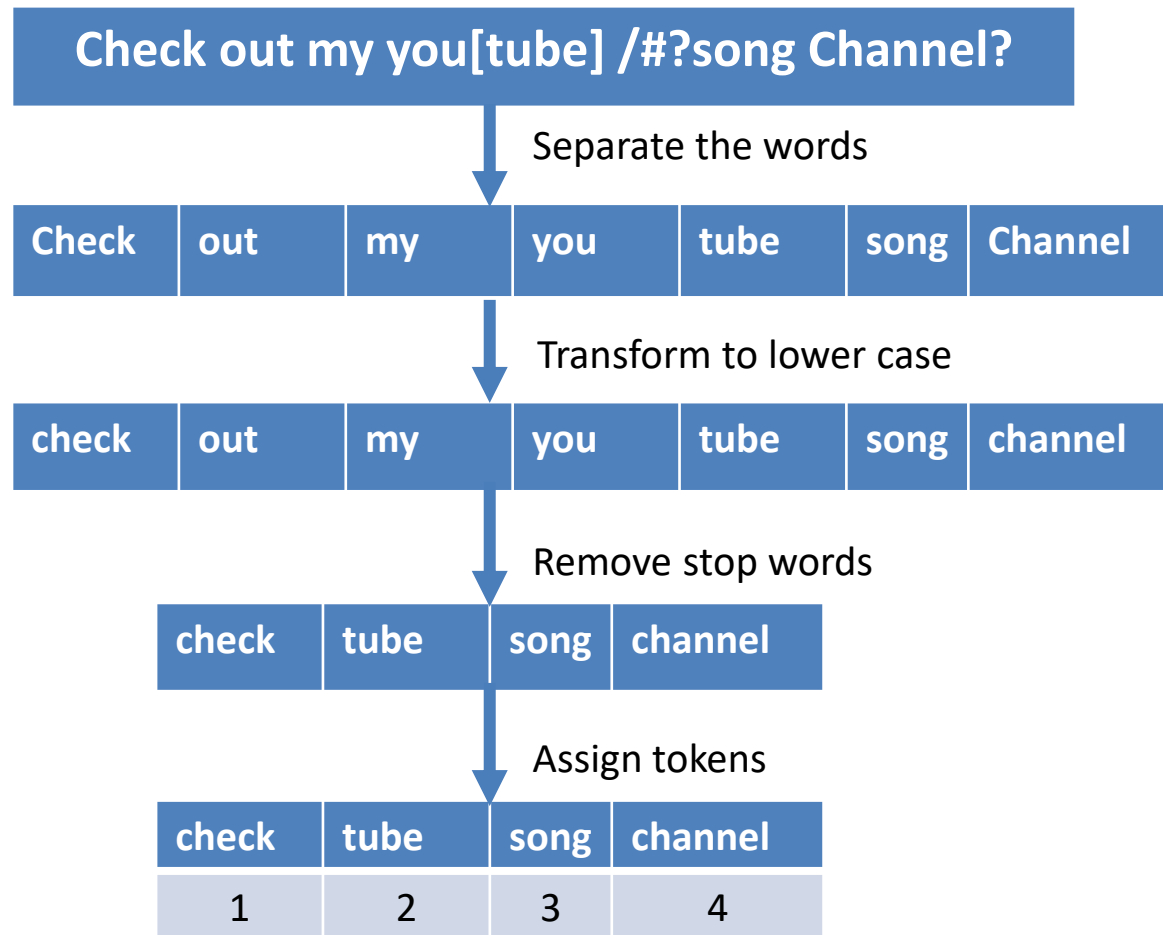
- Preprocessing text data in emails for use by ML models typically involves:
 - Tokenization
 - Feature selection/extraction
- **Tokenization**
 - Tokenization transforms an email into a sequence of representative symbols (*tokens*)
 - Tokens are often the individual words in the text, but they can also be several consecutive words (e.g., *n*-grams) or the individual characters in words (this is less common)
- Tokenization typically includes:
 - Remove punctuation signs (comma, period) or non-alphabetic characters (@, #, {, })
 - 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



Tokenization

Pre-processing Text in Email Messages

- Example of tokenization





n -Grams

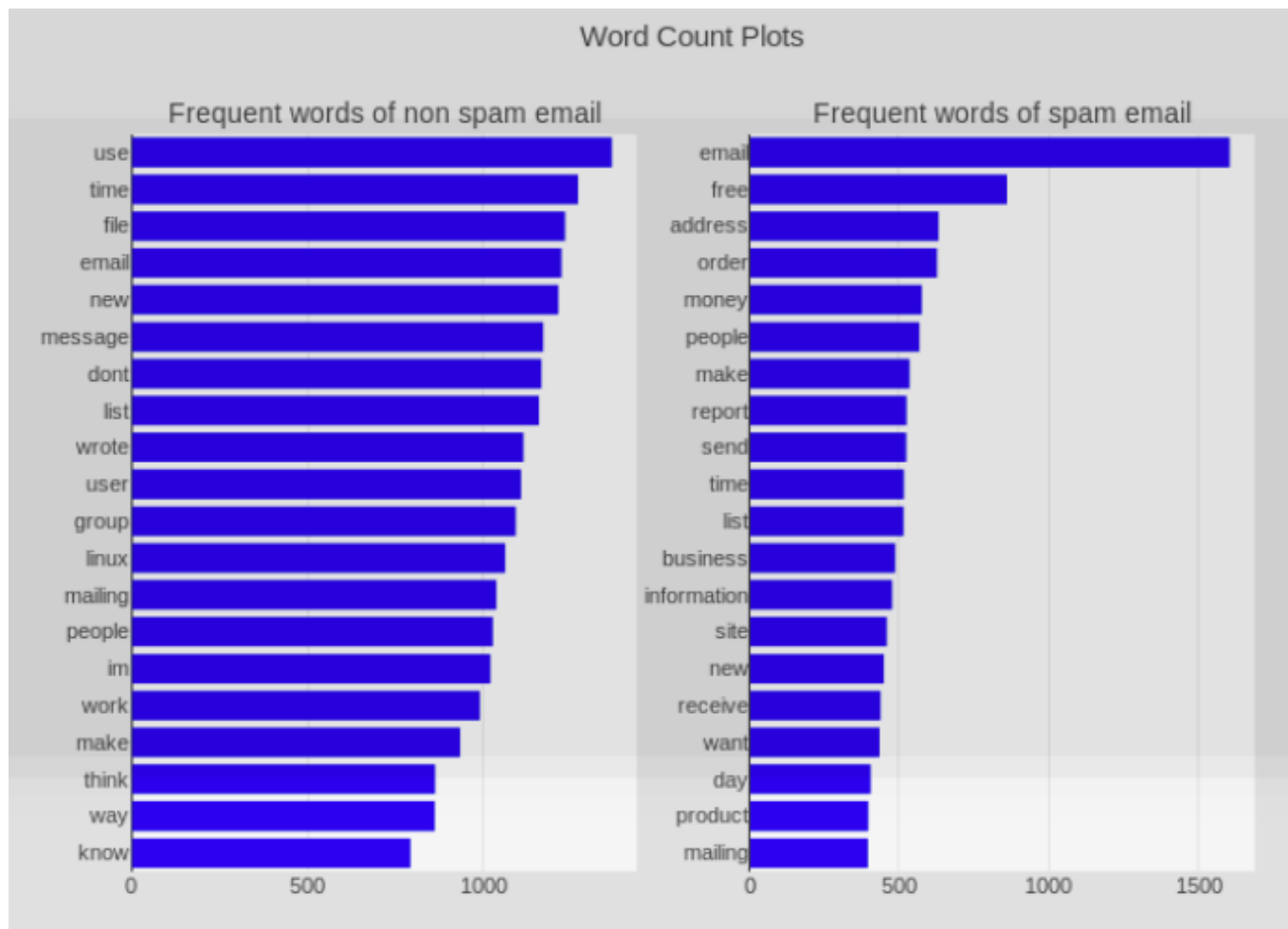
Pre-processing Text in Email Messages

- Instead of using single words 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
 - E.g., the word *play* is considered a neutral word in a 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



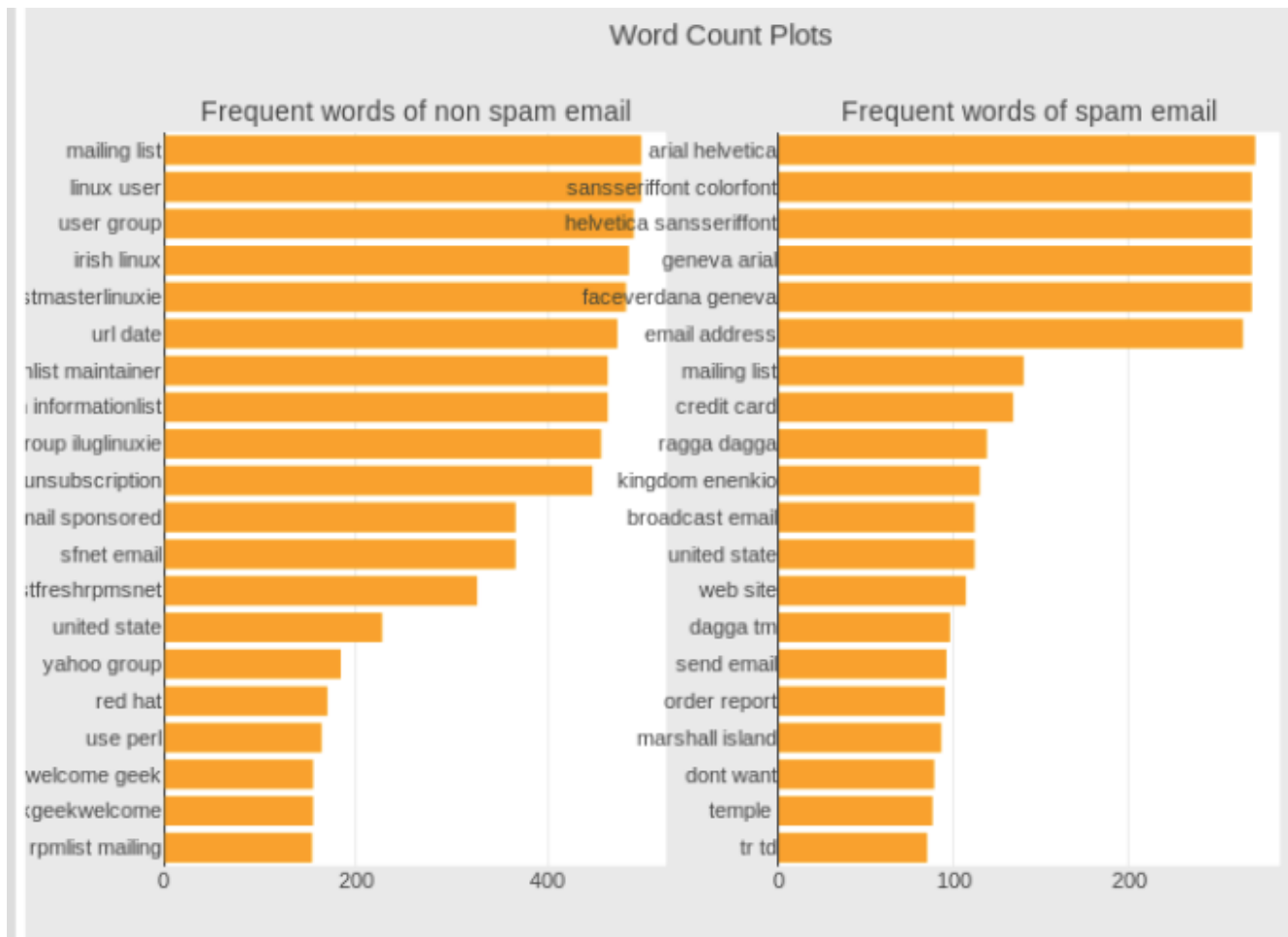
Bar chart visualization of 1-gram model



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



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
 - 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 some cases the word order can be important, therefore understanding the importance of the order of words in text is not straightforward

Bag-of-Words Approach

Bag-of-Words Approach

- *Bag-of-words approach*

- The tokenized words in emails 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)
 - Typically the frequency of occurrence of each word is used as a feature for training a 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



Bag-of-Words Approach

Bag-of-Words Approach

- Bag-of-words example
 - 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

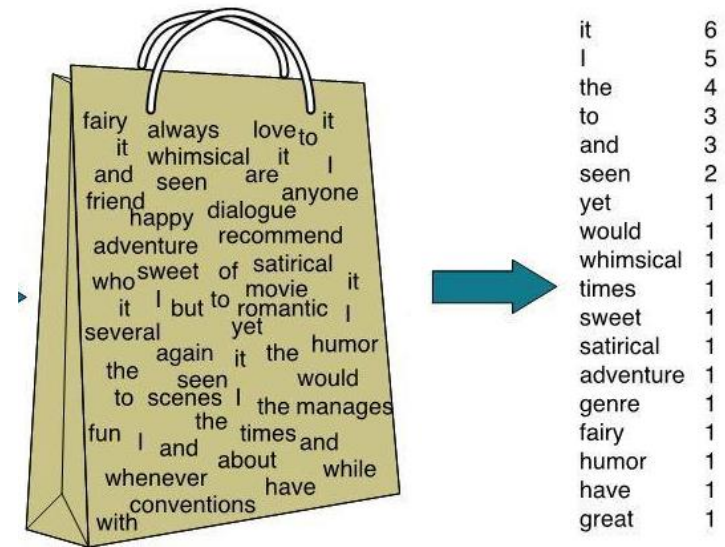
- Bags-of-words examples for spam (left) and non-spam messages (right)



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) 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





Spam Filtering with Naive Bayes

Spam Filtering Techniques

- **Naive Bayes classifier** was 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 $\mathbf{x} = [x_1, x_2, \dots]$ belongs to the spam category c_s is

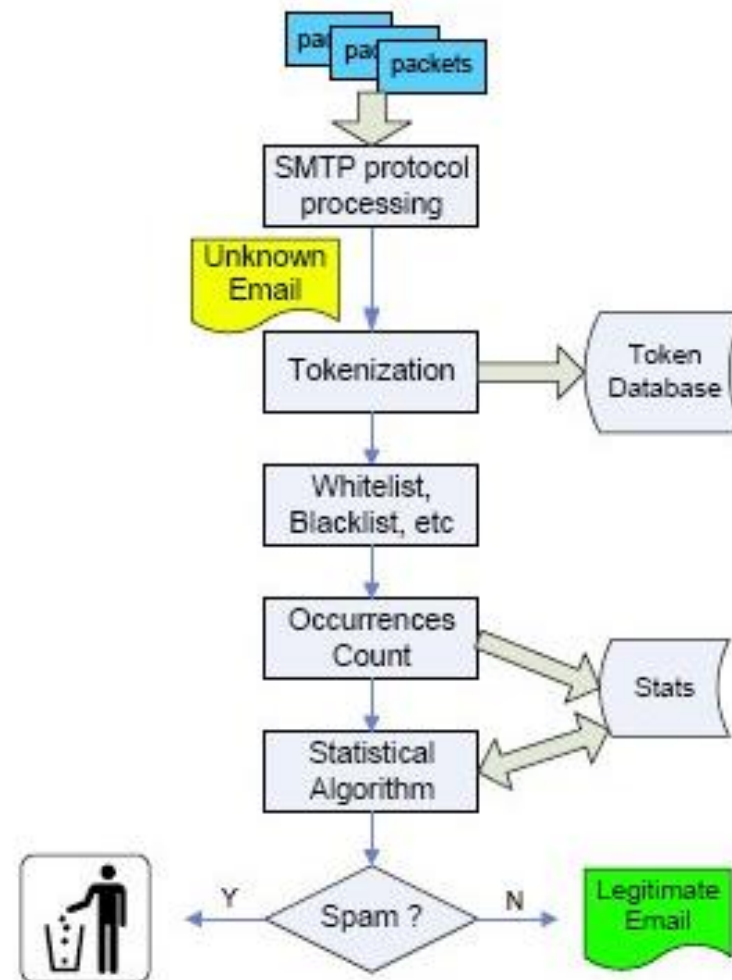
$$p(c_s|\mathbf{x}) = \frac{p(c_s) \cdot p(\mathbf{x}|c_s)}{p(c_s) \cdot p(\mathbf{x}|c_s) + p(c_h) \cdot p(\mathbf{x}|c_h)}$$

- 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 or ham bag-of-words, i.e., $p(\mathbf{x}|c_s) = \prod p(x_i|c_s)$
- If $p(c_s|\mathbf{x}) > \text{threshold}$, the email message is classified as a spam

Spam Filtering Block Diagram

Spam Filtering Techniques

- A typical data flow in spam filtering



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

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
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
Script	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
	scripts	Binary	Presence or absence of scripts in the body
Script	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



Sequence Model Approach

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. Use the embeddings 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

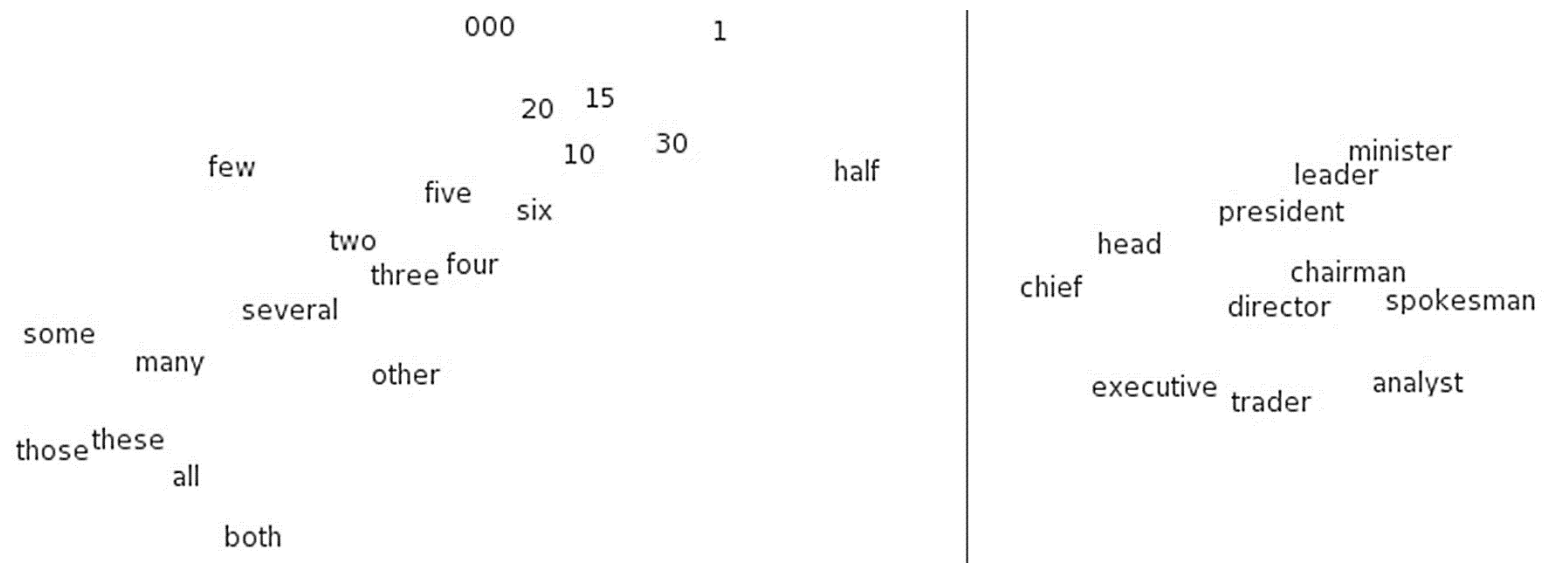
Sequence Model Approach

- **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**
- Typically, the cosine distance between the vectors in the embedding space is used as a distance metric
 - For given embedding vectors \mathbf{u} and \mathbf{v} , cosine similarity is $\cos\theta = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$

Word Embedding

Sequence Model Approach

- A very popular technique for generating word embeddings is **word2vec**
 - The word2vec algorithm uses an autoencoder to learn word embeddings from a large corpus of text
- Recent language models rely on Transformer Networks for generating word embeddings
 - E.g., word embeddings by BERT are commonly used in many NLP tasks
 - Most language models, like GPT, generate their word embeddings from training data
- Examples of word embeddings





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.



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



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 evasion 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



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

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

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)





Phishing Scam Statistics

Phishing URL Detection

- 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%



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 patterns 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



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 of 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

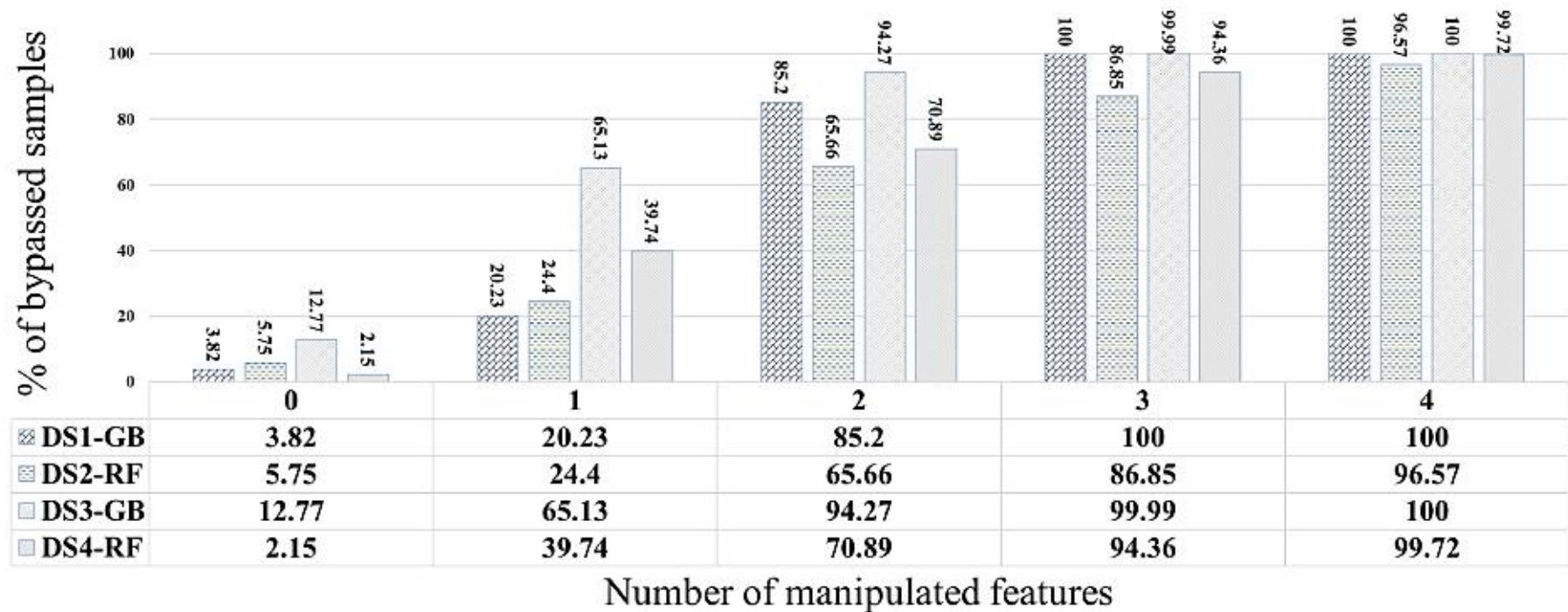
Dataset	Data shape (#)		Instances (%)	
	Size	Features	Legitimate	Phishing
DS-1	2210	7	44.71	55.29
DS-2	11055	30	55.69	44.31
DS-3	1250	9	43.84	56.16
DS-4	10000	48	50.0	50.0



AML against Phishing URL Detectors

Adversarial Attacks against Phishing URL Classifiers

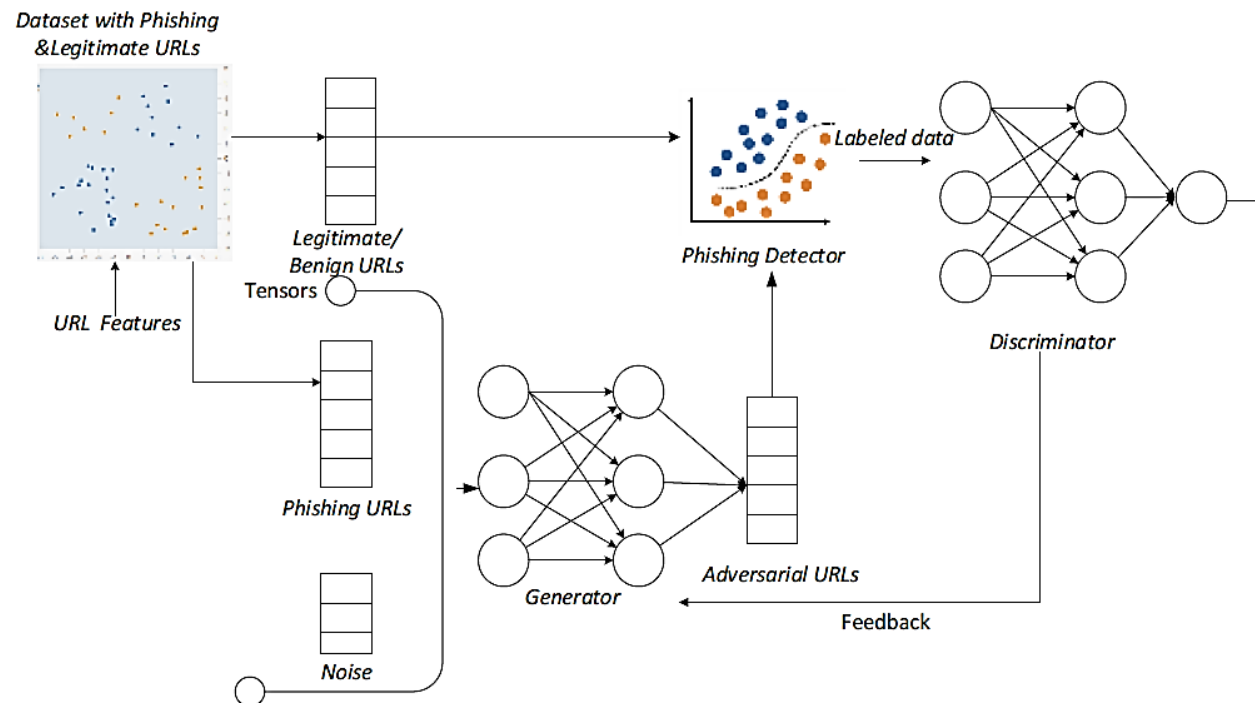
- Shirazi et al. (2019) cont'd
 - The percentage of evaded URL samples increased significantly with only one feature perturbed, and reached 100 when four features were manipulated



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





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)

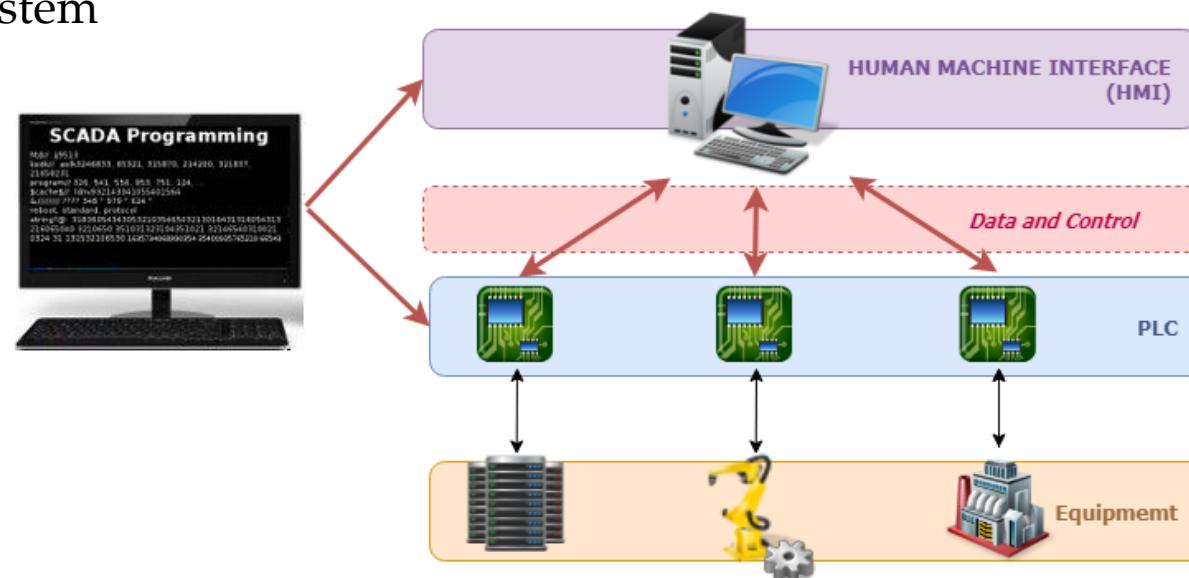
Industrial Control Systems (ICS)

- **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)

Industrial Control Systems (ICS)

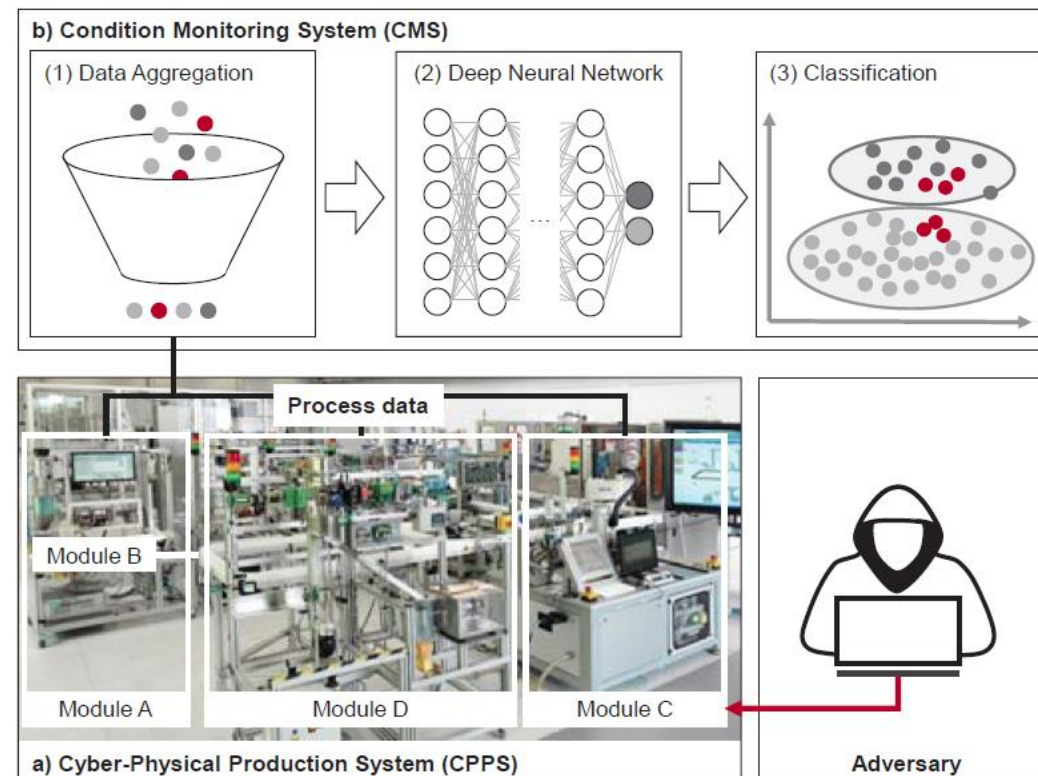
- The main components of ICS are depicted in the figure
 - **Equipment** – includes various field devices have sensors and actuators (motors, hydraulic or pneumatic cylinders), such as robots, machines, CNCs, etc.
 - **PLCs (Programmable Logic Controllers)** – industrial microcomputers that collect input 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



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 temporarily stop the production)





Major CPS Attacks

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
 - **Colonial Pipeline (2021)** – details on the next page



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
...	

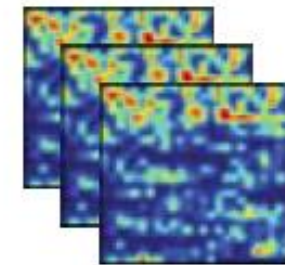


(2) CyberProtect

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



(3) Adversarial Examples



Condition Monitoring





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

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



AML against CPS

Adversarial Attacks against Cyber-physical Systems

- *Erba et al. (2020) Constrained Concealment Attacks against Reconstruction-based Anomaly Detectors in Industrial Control Systems* ([link](#))
 - Application: monitoring a water treatment plant
 - White-box and black-box evasion attacks
 - Attacked model: an autoencoder-based anomaly detection system
 - Datasets:
 - BATDAL dataset: contains readings from 43 sensors (e.g., tank water levels, inlet pressure, outlet pressure, and flow for valves and pumps, etc.)
 - WADI dataset: contain 82 sensor measurements
 - Attack approach:
 - Enumerate all possible operations for every sensor, and use the coordinate descent algorithm (iterative FGSM) for generating adversarial samples



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](#))

READING: ADVERSARIAL EXAMPLES FOR NATURAL LANGUAGE CLASSIFICATION PROBLEMS

CS504 - Adversarial Machine Learning

Weilin Chen
chen3536@vandals.uidaho.edu

University of Idaho

April 12, 2023

Table of Contents

- 1 Introduction
- 2 Background
- 3 Adversarial Examples For Natural Language Classification
- 4 Experiments
- 5 Discussion
- 6 Conclusion

Table of Contents

- 1 Introduction
- 2 Background
- 3 Adversarial Examples For Natural Language Classification
- 4 Experiments
- 5 Discussion
- 6 Conclusion

Introduction

Research Purpose

Modern machine learning algorithms are often susceptible to adversarial examples — maliciously crafted inputs that are undetectable by humans but that fool the algorithm into producing undesirable behavior. Adversarial examples arise in:

- **Image classification**
- **Speech recognition**
- **Reinforcement learning**
- **Other domains**

The existence of adversarial inputs has obvious security implications and also reveals important shortcomings in our understanding of modern machine learning algorithms.

Introduction

Adversarial Examples

Task: Spam filtering. **Classifier:** LSTM. **Original label:** 100% Spam. **New label:** 89% Non-Spam.
Text: your application petition has been accepted recognized thank you for your loan borrower request petition , which we recieved yesterday , your refinance subprime application petition has been accepted recognized good credit or not , we are ready to give you a \$ oov loan , after further review , our lenders have established the lowest monthly payments . approval process will take only 1 minute . please visit the confirmation link below and fill-out our short 30 second secure web-form . http : oov

Task: Sentiment analysis. **Classifier:** CNN. **Original label:** 81% Positive. **New label:** 100% Negative.
Text: i went moved to wing wednesday which is all-you-can-eat wings for \$ oov even though they raise the prices it 's still ever really great deal . you can eat as many wings you want to get all the different flavors tastes and have a good time enjoying the atmosphere . the girls are smoking hot ! all the types of sauces dressings are awesome ! and i had at least 25 wings in one sitting . i would definitely certainly go again just simply not every wednesday friday maybe once a month .

Task: Fake news detection. **Classifier:** Naive Bayes. **Original label:** 97% Fake. **New label:** 100% Real
Text: trump supporter whose brutal ferocious beating by black mob gangsta was caught on video tape asks demands : " what happened to america ? " [video] , " david oov , a 49 year old former chicago rochester man who was brutally beaten by a mob lowlife of black democrats asks demands , " what happened to america ? " here is his very sad disappointing story

Figure: Adversarial examples for three natural language classification tasks.



Table of Contents

- 1 Introduction
- 2 Background**
- 3 Adversarial Examples For Natural Language Classification
- 4 Experiments
- 5 Discussion
- 6 Conclusion

Background

classification problems



The goal of **classification problems** is to learn a mapping $f : X \rightarrow Y$ from an input $x \in X$ to a target label $y \in Y$, which lies in some finite set of K classes $Y = y_1, y_2, \dots, y_K$. The classifier f associates a score $f_{y_k}(x)$ to each class y_k and outputs the class with the highest score. In this paper, f was parametrized by a deep neural network or a linear model.



Background

Adversarial Examples and Image Classification

In the context of image classification, given a classifier f , x_0 is an adversarial perturbation of x targeting class y_0 (distinct from the true class y of x) if:

$$f(x_0) = y_0 \text{ and } \|x - x_0\| \leq \epsilon$$

where ϵ is a small constant. The norm $\|\cdot\|$ captures the notion of an imperceptible perturbation; popular choices include the ℓ_2 or the ℓ_1 norms.

For simplicity, x_0 is an adversarial example for f .

Adversarial examples can be obtained by solving an optimization problem of the form:

$$\max_{x_0} J(x_0) \text{ s.t. } \|x - x_0\| \leq \epsilon,$$

Here $J(x_0)$ measures the extent to which x_0 is adversarial and may be a function of a target class $y_0 \neq y$, e.g., $J(x_0) = f_{y_0}(x_0) - f_y(x_0)$. Algorithms for solving the above objective include the FGSM or iterative methods based on constrained gradient descent.

Background

Classifying Natural Language Utterances

Text classification problems arise in various domains, including **biomedical** (Aggarwal Zhai, 2012), **spam filtering** (Androutsopoulos et al., 2000), and **financial** (Schumaker Chen, 2009).

Linear classifiers with n -gram features often perform surprisingly well on text classification benchmarks (Wang Manning, 2012). In recent years, variants of recurrent networks — especially classifiers based on long short-term memory (Hochreiter Schmidhuber, 1997)—have helped improve state-of-the-art accuracy; most recently, convolutional neural networks have been shown to be competitive with recurrent methods (Kim, 2014; Zhang et al., 2015)



Background

Differences Between NLCP Image Classification

These differences pose natural constraints on the notion of an adversarial example, which their work explores

- *discrete nature* : the inputs x consist of discrete symbols such as characters or words
- *higher dimensionality*: which is typically proportional to vocabulary size
- *higher-level*: natural language representations are in a sense “higher-level” than image pixels, since they raw words encode significantly more meaning than raw pixel values.



Table of Contents

- 1 Introduction
- 2 Background
- 3 Adversarial Examples For Natural Language Classification**
- 4 Experiments
- 5 Discussion
- 6 Conclusion

Adversarial Examples For Natural Language Classification

Types Of Adversarial Examples

Altered Adversarial Examples:

- A large class of adversarial inputs are formed by adding imperceptible perturbations to ordinary dataset samples. The author propose to refer to this general type of adversarial attack as altered adversarial examples.

Obfuscated Examples:

- In which the input appears as white noise but triggers unwanted behavior (e.g., audio that turns on a smartphone)

Concatenative Examples:

- In which the input is combined with a distracting sequence that contains irrelevant information.

Altered examples encompass the original notion of adversarial perturbation, and apply in arguably more common settings.



Adversarial Examples For Natural Language Classification

A Specially-crafted Constraint Function

the goal of this function is to ensure that both utterances share the same meaning and retain common syntactic properties (e.g. the style of writing should remain similar). Specifically, the function c is comprised on two constraints that capture sentence similarity on two levels.

- **Semantic similarity.** A thought vector can be seen as a mapping from sentences to a vector space, in which sentences with similar meanings are close to each other. In this context, constraint is defined as:

$$\|\mathbf{v} - \mathbf{v}_0\|_2 < \gamma_1$$

where \mathbf{v} and \mathbf{v}_0 are thought vectors associated with x and x_0 , respectively, and 1 is a hyper-parameter.



Adversarial Examples For Natural Language Classification

A Specially-crafted Constraint Function

- **Syntactic similarity.** To ensure that adversarial sentences are well-formed, a syntactic constraint is introduced, which relies on a language model $P : X \rightarrow [0, 1]$. Specifically, the language model probability is required to be similar between the perturbed and the original example:

$$|\log P(x_0) - \log P(x)| < \gamma_2$$



Adversarial Examples For Natural Language Classification

Greedy Construction Of Altered Adversarial Examples

In brief, an iterative procedure is proposed that considers at each step all valid one-word changes to a sentence and chooses the one that improves the objective the most. This procedure effectively replaces individual words with their synonyms, resulting in a new sentence of the same length that approximately preserves the original meaning.

Algorithm 1: Greedy Optimization Strategy for Finding Adversarial Examples

Data: Datapoint x , termination threshold τ , neighborhood size N , parameters $\gamma_1, \gamma_2, \delta$.

We initialize the algorithm at the uncorrupted data point: $x' \leftarrow x$;

while *Objective is below the threshold* $J(x') < \tau$ *and fraction of words replaced is less than* δ **do**

 Create a working set $W = \emptyset$;

for each word w *in* x **do**

for each word \bar{w} *among the* N *closest to* w *and different from* w **do**

 substitute w' with \bar{w} to get \bar{x} and if \bar{x} satisfies Equ. (5), then $W \leftarrow W \cup \{x'\}$;

 Choose highest scoring word replacement $x' \leftarrow \arg \max_{\bar{x} \in W} J(\bar{x})$ or if $W = \emptyset$, then **break**;

return x' ;



Table of Contents

- 1 Introduction
- 2 Background
- 3 Adversarial Examples For Natural Language Classification
- 4 Experiments**
- 5 Discussion
- 6 Conclusion

Experiments

Tasks

There are adversarial examples on three natural language classification tasks, summarized in Table 1. 10% of the training set for validation is held out; all adversarial examples are generated and evaluated on the test set.

Dataset	Task	#Train	#Test
Trec07p	Spam filtering	67.9k	7.5k
Yelp	Sentiment analysis	560k	38k
News	Fake news detection	5.3k	1.0k

Table 1: Summary of datasets and tasks

Experiments

Tasks

- **Spam filtering.** The TREC 2007 Public Spam Corpus (Trec07p) contains 50,199 spam emails and 25,220 ham (non-spam) emails.
- **Sentiment analysis.** The Yelp Review Polarity dataset consists of almost 600,000 customer reviews from Yelp, covering primarily restaurant reviews. Each review is labeled as either positive or negative.
- **Fake news detection.** The News dataset contains 6,336 articles scraped from online sources, and includes both real and fake news. Each article contains a headline and body text and is associated with a binary label.



Experiments

Models

Adversarial example on a range of models that are widely used for classification are studied, including both linear classifiers and state-of-the-art deep learning algorithms.

- **Naive Bayes.** This linear model has a long history in text classification and it is still popular for its simplicity.
- **Long short-term memory.** Long-short term memory (LSTM) is widely used in sequence modeling.
- **Shallow word-level convolutional networks.** A CNN with an embedding layer, a temporal convolutional layer, followed by max-pooling over time, and a fully connected layer for classification is trained.
- **Deep character-level convolutional networks.** Which includes 4 stages. Each stage has 2 convolutional layers with batchnormalization and 1 max-pooling layer.



Experiments

Main Experiment

- All models are susceptible to adversarial examples to a certain degree
- All methods are equally robust to random perturbations

Data		NB	LSTM	WCNN	VDCNN
Trec07p	CLN	97.1%	99.1%	99.7%	
	RND	97.7%	98.6%	99.6%	
	ADV	15.1%	39.8%	64.5%	
Yelp	CLN	87.9%	95.3%	94.9%	95.1%
	RND	86.8%	94.5%	94.7%	93.1%
	ADV	9.0%	24.0%	39.0%	53.0%
News	CLN	91.0%	93.0%	96.0%	93.4%
	RND	84.0%	94.6%	93.3%	92.7%
	ADV	9.0%	37.0%	71.0%	11.0%

Table 2: Classifier accuracy on each dataset. CLN, RND, and ADV stand for clean, randomly corrupted, and adversarially corrupted inputs.

Experiments

Human Evaluation

- The quality and the coherence of examples are verified via human experiments
- Assign labels (e.g. positive or negative review) to both the original and adversarially altered versions
- Human evaluators achieved similar accuracies on both types of inputs
- adversarial alterations preserved key semantics sufficiently well to be recognized by a human

Input	Trec07p	Yelp	News
Original	87%	93%	64%
Adversarial	93%	87%	58%

Table 3: Human classification accuracy on adversarial examples for the LSTM model.

Experiments

Human Evaluation

- Human annotators rated the “writing quality” of the same set of examples on a scale of one to five
- Five being the highest possible quality and likely generated by a human, and one being the lowest quality, likely generated by a machine
- Humans tend to assign similar scores to both sets of samples.
- Adversarial examples were of comparable quality to the original examples

Input	Trec07p	Yelp	News
Original	2.64	2.37	2.72
Adversarial	2.75	2.38	2.47

Table 4: Human classification accuracy on adversarial examples for the LSTM model.

Experiments

Transferability

- An intriguing property of image classification models is that adversarial examples generated for one classifier are likely to be misclassified by other classifiers.
- Whether adversarial texts transfer between the four models are examined, focusing on the Yelp dataset.

	NB	LSTM	WCNN	VDCNN
NB	20%	77%	75%	88%
LSTM	67%	17%	64%	83%
WCNN	63%	64%	17%	84%
VDCNN	77%	85%	87%	23%


Table 6: Transferability of adversarial examples. Row i and column j show the accuracy of adversarial samples generated for model i evaluated on  University of Idaho

Table of Contents

- 1 Introduction
- 2 Background
- 3 Adversarial Examples For Natural Language Classification
- 4 Experiments
- 5 Discussion**
- 6 Conclusion

Discussion

Applications Of Language-Based Adversarial Examples

- This work demonstrates the existence of adversarial examples in state-of-the-art models for spam and sentiment classification
- Their observations lend further evidence to the prevalence of adversarial attacks in the natural language domain.
- Adversarial inputs can also improve algorithms via adversarial training by serving as extra data and thus increasing performance and robustness to adversarial attacks.
- The existing search procedure naturally generalizes to beam search, and could modify phrases rather than words.



Table of Contents

- 1 Introduction
- 2 Background
- 3 Adversarial Examples For Natural Language Classification
- 4 Experiments
- 5 Discussion
- 6 Conclusion**

Conclusion

Attribution and Future Study

- Generalize the concept of adversarial examples to natural language classification by proposing a simple yet effective similarity metric for text.
- Evaluate their approach on several classification tasks and show that a simple greedy algorithm is effective at finding adversarial examples in each setting.
- The presence of adversarial examples for text classification poses threat to real-world machine learning systems.
- Further study of adversarial examples for text classification with help defend these systems and improve the accuracy of classification algorithms via adversarial training.





Thank You for Your Attention!