Deep learning in Cybersecurity

Subject: Computer Security

Professor: Dr. Anand Nayyar
Team members: Nguyen Thi Hong Yen - 2320120895
Tran Thanh Danh - 2321121844
Class: CS 376 D

FOREWORD

'Cybersecurity' can be visualized as an assortment of technologies, techniques, processes, and policies to protect the Confidentiality, Integrity, and Availability (CIA) triad of the compute, network, software and data resources. The expeditious technological advancements in each dimension of computing and the popularity of the Internet have resulted in a scenario wherein the people's day-to-day activities heavily rely on the information systems. Since these information systems are open to the Internet, there has been an increasing awareness of the socio-economic impacts associated with the ever-advancing cyberattacks. Many business and academic organizations realize the need for the use of advanced security tools and techniques to protect their information system and networks from the adverse impacts of malicious activities. On the other end, cybersecurity researchers and professionals put up more efforts to design powerful and adaptive algorithms to enhance the efficiency of the security tools and techniques in terms of high detection rate and less false alarm rate. Over the past few decades, cybersecurity research areas like intrusion detection, spam detection, malware analysis, etc. have been exploiting the complete benefits of machine learning and deep learning techniques for the design of a set of efficient, accurate and robust defense-in-depth security models. Machine learning approaches outperform the traditional rule-based security solutions, in turn, the better generalization ability of deep learning approaches enables them to provide significant performance improvements over the machine learning approaches. However, limited domain experts and difficulty in identifying the malicious activities due to impersonation, polymorphism, obfuscation, and compression poses a significant challenge in the field of cybersecurity irrespective of its applications in various domains like social networks, Internet of Things, smart grids, etc. This book aims to bridge the gap between cybersecurity and machine learning & deep learning techniques by focusing on new, automated, and effective security solutions to replace traditional cybersecurity mechanisms. Each chapter of this book makes the readers walk through the recent advancements in machine learning and deep learning algorithms and their applications towards the development of intelligent and future-proof security solutions.

TABLE OF CONTENTS

PLEDGED

The project is performed by my group and research some documents. We do confirm that we don't copy from any document and existing project before.

Performed Students
Nguyen Thi Hong Yen
Tran Thanh Danh

ACKNOWLEDGMENT

We would like to express our sincere thanks to Dr. Anand Nayyar - Professor of Duy Tan University devotedly guided during the group project. He created many favorable conditions and gave valuable advice to help us complete the group project. Sincerely thank you so much to you have dedicated to teaching, imparting on us valuable knowledge, and facilitating help during the learning process during the course, and help us have a solid theoretical basis to study this project " Deep learning in Cybersecurity"
Thank you!

Performed Students
Nguyen Thi Hong Yen
Tran Thanh Danh

A. OVERVIEW

I. Introduction

Today is the era of Internet where everything become digitalized including purchasing items in mall, bank transactions, ticket reservations, online shopping, top secrets in government organization especially in military and defense. On one hand we are enjoying the privilege of digitalization which results on accumulating data in terabytes, but on the other hand the accumulated data need to be converted to information and the information need to be mined to get knowledge to enhance user experience in usage of Internet.
While the users are enjoying the advantage of Internet, the most concerning thing about Internet is the safety our data. The technique of protecting computers, networks, programs and data from unauthorized access or attacks that are aimed for exploitation is called Cyber Security. The digital world is replacing the physical, the more realistic view is that the two worlds are coming together. Organizations are not really prepared to face the challenges which is the result of the amalgamation of digital and physical world. In the case of cybercrime, it states that it isn't displacing physical acts of crime, they are occurring in concert. Criminals who might once have used explosives to cripple critical infrastructure, such as transportation, power grids or water systems, for example, may be now able to achieve their goals remotely by attacking the computers that operate those systems, incurring less risk in the process. Cyber Security is broadly categorized into four major areas, they are:

a) Application Security
b) Information Security
c) Disaster Recovery
d) Network Security

II. Objectives

In this project, we focus on researching and deeply understanding in the deep learning and cybersecurity. The roles of deep learning in common and also in particular cybersecurity. Some limitations and problem of cybersecurity in Viet Nam. Give some sub-methods or descriptive technology in deep learning, and also in particular cybersecurity methods. We represent the major algorithm and specific technique in sub-methods. We introduce some application of deep learning in cybersecurity. We give an example for each part by diagram, explaining and conclude what we actually understand.

III. The content of research

Content 1: Research and analyze the topic. Focusing on 2 field such as Deep learning, and cybersecurity
Content 2: Analyze and give conclusion for question: "Is deep learning the future of cybersecurity"
Content 3: Research the types of cyber security attacks
Content 4: Research the influence of cyber security attack on a specific topic
Content 5: Research the role of deep learning in cybersecurity
Content 6: Research and understand about security methods
Content 7: Research how to apply deep learning in cybersecurity
Content 8: Research some methods of deep learning in cybersecurity
Content 9: Research and write descriptive technologies algorithm methods of deep learning in cybersecurity.

Content 10: Case study: Research the application of deep learning of Cybersecurity in Website in VietNam. The problem and the solution
Content 11: Write document

IV. Boundary

Research about all key words in the topic.
Research and introduce about technology
Research descriptive technology such as algorithm and example

V. Layouts

1. Overview of the topic
2. Introduction of technology
3. Descriptive technology
4. Case study
5. Conclusion and development in the future

B. INTRODUCTION

I. An overview of Cybersecurity

1. The brief history and the evolution of Cybersecurity

a) The history

- March 16, 1971 – Discovery of the Creeper Virus. Believe it or not, the idea of a computer virus preceded computer networks. Mathematician John von Neumann predicted the idea in the late 1940s, but it wasn't until 30 years later before someone created one. During the age of ARPANET (the internet at its earliest form) in 1971, the few users of the network were surprised when their screens displayed the phrase: "I'm the creeper, catch me if you can." At the time, users had no idea who or what it could be. Creeper was a worm, a type of computer virus that replicates itself and spreads to other systems; it was created by Bolt, Beranek and Newman. Unlike today's malicious viruses, all Creeper did was display messages. Sept. 20, 1983 – The First U.S. Patent for Cybersecurity. As computers began to evolve, inventors and technology experts around the world were rushing to make history and claim patents for new computer systems. The first U.S. patent for cybersecurity came in September of 1983 when MIT was granted U.S. Patent 4,405,829 for a "cryptographic communications system and method." The patent introduced the RSA (Rivest-Shamir-Adleman) algorithm, which was one of the first public-key cryptosystems. Cryptography is the bedrock of modern cybersecurity.
- June 9, 1993 – The First DEF CON Conference DEF CON is one of the world's most popular cybersecurity technical conferences. Started in June of 1993 by Jeff Moss, it opened in Las Vegas with roughly 100 people. Today the conference is attended by over 20,000 cybersecurity professionals from around the world.
- February 1995 – The Birth of Secure Sockets Layer (SSL) 2.0. The security protocol that allows people to do simple things like purchase items online securely was made possible by the Secure Sockets Layer (SSL) internet protocol. Netscape began developing the SSL protocol not long after the National Center for Supercomputing

Applications released the first web browser. In February 1995, Netscape released SSL 2.0, which became the core of the language for securely using the web, called Hypertext Transfer Protocol Secure. Today, when you see "HTTPS" in a website address, you know its communications with your browser are encrypted.

- Oct. 1, 2003 – Anonymous is Born. Anonymous was the first universally known hacker group. The group has no leader and represents many online and offline community users. Together, they exist as an anarchic, digitized global brain. Wearing the mask of Guy Fawkes, the group gained national attention when the group hacked the Church of Scientology website with distributed DDoS attacks. Anonymous continues being linked to numerous high-profile incidents; its main cause is protecting citizens' privacy. Jan. 12, 2010 Operation Aurora Reveals a Nation-as-Hacker
- Before 2010, disclosures of security breaches were considered highly unusual. On Jan. 12 of that year, Google shocked the world when it announced "Operation Aurora," a major breach of its infrastructure in China. Google initially thought the attackers' goal was to access Gmail accounts of Chinese human rights activists. Analysts discovered the true intent was identifying Chinese intelligence operatives in the U.S. who may have been on watch lists for American law enforcement agencies. The attacks also hit more than 50 companies in the internet, finance, technology, media, and chemical sectors. Recent Exploits, Countermeasures and Looking Forward
- In recent years, massive breaches have hit name brands like Target, Anthem, Home Depot, Equifax, Yahoo, Marriott, and more – compromising data for the companies and billions of consumers. In reaction, stringent regulations to protect citizen privacy like the EU General Data Protection Regulation (GDPR) and the new California Consumer Privacy Act are raising the bar for compliance. And cyberspace has become a digital battleground for nation-states and hacktivists. To keep up, the cybersecurity industry is constantly innovating and using advanced machine learning and AI-driven approaches, for example, to analyze network behavior and prevent adversaries from winning. It's an exciting time for the market, and looking back only helps us predict where it's going.

b) The evolution

Cybersecurity in practice, is reactionary. CyberSafety however, is not a reactionary response to a threat, but rather the proactive anticipatory preparedness of AI/SOAR technology the basis for the anticipatory cyber safety approach evolved from combining software defined networking (SDN) methods with security automation and automated intelligence. Traditional networking is being progressively replaced by SDN capability that enables dynamic programmable networks. It is the new promising approach to designing, building and managing more secure networks. Although SDN promises more flexible network management (Vizv´ary and Vykopal, 2014), the real answer lies within today's SOAR orchestration, automated intelligence and deep learning risk aware security. Thus, the combination of SDN and SOAR AI deep learning will outpace the conflict between cyber-attacks and cyber defensive systems.

Embedded SOAR has been a buzzword in the security space for some time. Recently, the U.S. Patent Office issued several new patents that completely embrace SOAR systems, methods and architecture. One patent, No. US 10326777 B21 covers the Internet technology used to identify threats, orchestrate security, automate security, and apply AI-based proactive response. The three key elements of SOAR technologies employ:

1) Orchestration methodologies that interface multiple cybersecurity technologies to prevent an attack

2) Automated technology that automatically writes a new-rule and inserts the rule into the exact area of the code to prevent a new never-before-seen (polymorphic) attack within milliseconds, without human intervention.

In short, from the inspection mode to the analytic mode to the action mode, i.e., SOAR technology automatically inserts proactive security code to block a cyber-attack and alert (react) the security technology of the attempted breach and stops it at the first packet handshake. The unique factor of SOAR technology is that its "R" or response attribute, literally represents three proactive risk attributes, i.e., risk aware, risk response, and risk reporting.

SOAR can not only prevent all known forms of cyber-attack, but also can "learn" to anticipate any future attack from new "unseen" mutated attack threat. The automation and orchestration features of SOAR have reached a level of sophistication where it can be integrated into an existing security framework without relying on human assistance. Therefore, the motivation behind this paper is to provide a perspective on AI-based SOAR technology and how it effectively prevents attacks.

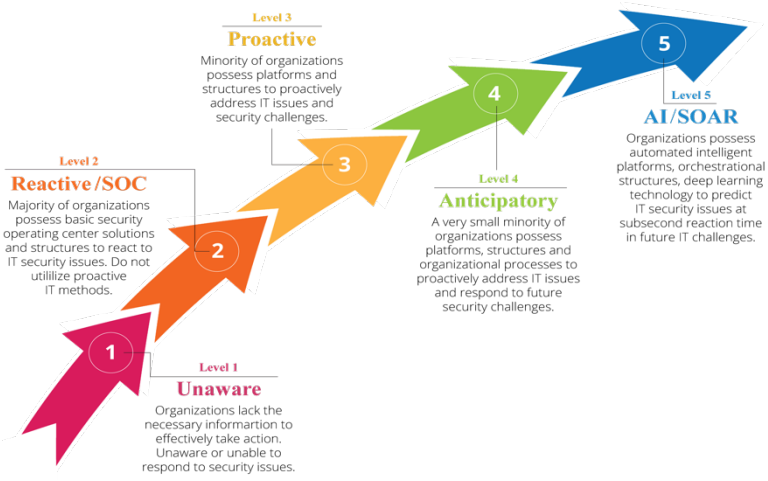


Figure 1: The evolution of cybersecurity

2. What is the definition of Cybersecurity?

Computer security, cybersecurity, or information technology security (IT security) is the protection of computer systems and networks from the theft of or damage to their hardware, software, or electronic data, as well as from the disruption or misdirection of the services they provide.

The field is becoming more important due to increased reliance on computer systems, the Internet, and wireless network standards such as Bluetooth and Wi-Fi, and due to the growth of "smart" devices, including smartphones, televisions, and the various devices that constitute the "Internet of things". Owing to its complexity, both in terms of politics and technology, cybersecurity is also one of the major challenges in the contemporary world.

3. The purpose of Cybersecurity

- Confidentiality is roughly equivalent to privacy and avoids the unauthorized disclosure of information. It involves the protection of data, providing access for those who are allowed to see it while disallowing others from learning anything about its content. It

prevents essential information from reaching the wrong people while making sure that the right people can get it. Data encryption is a good example to ensure confidentiality.

- Integrity refers to the methods for ensuring that data is real, accurate and safeguarded from unauthorized user modification. It is the property that information has not be altered in an unauthorized way, and that source of the information is genuine.
- Availability is the property in which information is accessible and modifiable in a timely fashion by those authorized to do so. It is the guarantee of reliable and constant access to our sensitive data by authorized people.

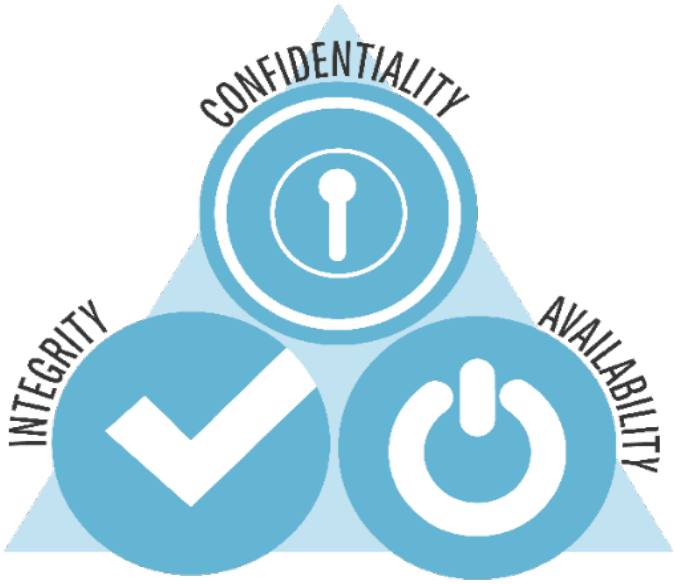


Figure 2: The purpose of cybersecurity

4. Cyber-attack and influences

4.1. The victim of cyber attack

According to McAfee's Economic Impact of Cyber Crime (February 2018), cyber criminals adapt at a fast pace. The scale of malicious activity across the internet is quite astounding. The figures are frightening on a monthly or yearly scale, let alone daily! Cyber criminals are constantly finding new technologies to target victims. With the introduction of Bitcoin, payment, and transfers to/from cyber criminals is untraceable.

4.2. The purpose of cyber attack

A cyberattack may steal, alter, or destroy a specified target by hacking into a susceptible system. Cyberattacks can range from installing spyware on a personal computer to attempting to destroy the infrastructure of entire nations. Legal experts are seeking to limit the use of the term to incidents causing physical damage, distinguishing it from the more routine data breaches and broader hacking activities.

4.3. The influence of Cyberattack

4.3.1. The influence of Cyberattack in business:

A successful cyber-attack can lead to the loss of your company's critical data including customers' personal information. Attackers can use this data to demand ransom from you or harass your customers.
Loss of customers' data to hackers has led to the fall of many companies including billion-dollar firms in the financial and healthcare industry. When hackers steal your customer data, you risk falling into endless lawsuits and eventually declaring bankruptcy.
At least 60 percent of companies that experience cyberattacks fail within six months. As mentioned earlier, a cyberattack that involves customers' data loss can result in endless lawsuits hence driving businesses into bankruptcy.

4.3.2. The influence of Cyberattack in different industries.

Healthcare/Medical

This information-intensive industry is a frequent target for its stores of data. Health care and medical organizations access and store electronic healthcare records, which contain large amounts of personal information as well as financial details. The WannaCry ransomware attack, for instance, devastated operations at Britain's National Health Service (NHS) and negatively impacted patient care.
This industry's strict compliance standards aim to detect any exploitable vulnerabilities, but healthcare entities need to have their networks and systems locked down to facilitate HIPAA compliance and protect electronic protected health information (ePHI).

Banking/Credit/Financial

This industry is a prime target for obvious reasons. After all, these organizations deal with what attackers want most — money and personal information.
In a 2016 survey, Accenture found that 78% of financial institutions were confident in their cybersecurity strategies, yet 1 of every 3 is successfully attacked (at an average of 85 breach attempts per year).
The FDIC requires penetration testing for financial institution compliance. Banks, credit unions, and other financial institutions must ensure security and confidentiality of customer information, put controls in place to prevent illicit access of information, and make sure customer and consumer information are properly disposed of.

Government/Military

Government and military security breaches tend to be high-profile, so this industry's presence in this blog is unlikely to surprise. This sector is targeted by:

- Foreign powers trying to spy upon or negatively impact a global competitor
- Hacktivists looking to make a political statement
- Cybercriminals seeking to monetize the abundant personal information in federal, state, and local databases.

The Department of Defense in September 2018 released its cyber strategy addressing the need to "ensure the U.S. military's ability to fight and win wars in any domain, including

cyberspace," as well as "preempt, defeat or deter malicious cyber activity targeting U.S. critical infrastructure that could cause a significant cyber incident."

Energy/Utilities

The energy and utility sector face its own particular concerns. Though highly regulated and subject to tough compliance laws (such as NERC), there is great potential for hacktivism and cyberterrorism. They usually have equipment separated by miles of empty space, and motivated hackers can cause widespread power outages undermining critical defense infrastructure and risking the health and safety of millions of citizens.
After all, it is the energy grid and utilities that power, literally, our economy and everyday lives. A known national security priority, this area is also at risk of malware infections of the many mobile connections (web, mobile, and network security are critical). At the same time, backup restoration services are important too.

Education

Educational institutions are targeted for several reasons:

- Valuable intellectual property from campus research
- Student and employee personal information
- Computer processing power

Additionally, higher education institutions have a great turnover in their population, which can result in poor password protection and susceptibility to social engineering.
Between 2005 and 2015, higher education was one of the highest hits with a total of 539 breaches involving nearly 13 million records. Then, in a later Gemalto report, the number of lost, stolen, or compromised data records was up 164 percent in the first six months of 2017 compared to the last half of 2016.

5. The vulnerability and the common type of Cybersecurity attacks

Vulnerability factor exploits how vulnerable an organization or government establishment is to cyber-attacks. Organizations without maintenance systems might be running on old servers which are more vulnerable than updated systems. An organization can be vulnerable to a denial of service attack and a government establishment can be defaced on a web page. A computer network attack disrupts the integrity or authenticity of data, usually through malicious code that alters program logic that controls data, leading to errors in output.

Types of Cyber Attacks

A cyber-attack is an exploitation of computer systems and networks. It uses malicious code to alter computer code, logic or data and lead to cybercrimes, such as information and identity theft.
We are living in a digital era. Now a day, most of the people use computer and internet. Due to the dependency on digital things, the illegal computer activity is growing and changing like any type of crime.
Cyber-attacks can be classified into the following categories:

a. *Web-based attacks:* These are the attacks which occur on a website or web applications. Some of the important web-based attacks are as follows

b. *Injection attacks:* It is the attack in which some data will be injected into a web application to manipulate the application and fetch the required information. Example: SQL Injection, code Injection, log Injection, XML Injection etc.
c. *DNS Spoofing:* is a type of computer security hacking. Whereby a data is introduced into a DNS resolver's cache causing the name server to return an incorrect IP address, diverting traffic to the attacker's computer or any other computer. The DNS spoofing attacks can go on for a long period of time without being detected and can cause serious security issues.
d. *Session Hijacking:* It is a security attack on a user session over a protected network. Web applications create cookies to store the state and user sessions. By stealing the cookies, an attacker can have access to all of the user data.
e. *Phishing:* is a type of attack which attempts to steal sensitive information like user login credentials and credit card number. It occurs when an attacker is masquerading as a trustworthy entity in electronic communication.
f. *Brute force:* It is a type of attack which uses a trial and error method. This attack generates a large number of guesses and validates them to obtain actual data like user password and personal identification number. This attack may be used by criminals to crack encrypted data, or by security, analysts to test an organization's network security.
g. *Denial of Service*: It is an attack which meant to make a server or network resource unavailable to the users. It accomplishes this by flooding the target with traffic or sending it information that triggers a crash. It uses the single system and single internet connection to attack a server. It can be classified into the following
h. *Volume-based attacks*: Its goal is to saturate the bandwidth of the attacked site, and is measured in bit per second.
i. *Protocol attacks:* It consumes actual server resources, and is measured in a packet.
j. *Application layer attacks:* Its goal is to crash the web server and is measured in request per second.
k. *Dictionary attacks:* This type of attack stored the list of a commonly used password and validated them to get the original password.
l. *URL Interpretation:* It is a type of attack where we can change the certain parts of a URL, and one can make a web server to deliver web pages for which he is not authorized to browse.
m. *File Inclusion attacks:* It is a type of attack that allows an attacker to access unauthorized or essential files which is available on the web server or to execute malicious files on the web server by making use of the include functionality.
n. *Man in the middle attacks:* It is a type of attack that allows an attacker to intercepts the connection between client and server and acts as a bridge between them. Due to this, an attacker will be able to read, insert and modify the data in the intercepted connection.
o. *System-based attacks,* these are the attacks which are intended to compromise a computer or a computer network. Some of the important system-based attacks are as follows:
    - *Virus:* It is a type of malicious software program that spread throughout the computer files without the knowledge of a user. It is a self-replicating malicious computer program that replicates by inserting copies of itself into other computer programs when executed. It can also execute instructions that cause harm to the system
    - *Worm:* It is a type of malware whose primary function is to replicate itself to spread to uninfected computers. It works same as the computer virus. Worms often originate from email attachments that appear to be from trusted senders.

- *Trojan horse:* It is a malicious program that occurs unexpected changes to computer setting and unusual activity, even when the computer should be idle. It misleads the user of its true intent. It appears to be a normal application but when opened/executed some malicious code will run in the background.
- *Backdoors:* It is a method that bypasses the normal authentication process. A developer may create a backdoor so that an application or operating system can be accessed for troubleshooting or other purposes.
- *Bots:* A bot (short for "robot") is an automated process that interacts with other network services. Some bots program run automatically, while others only execute commands when they receive specific input. Common examples of bots' program are the crawler, chatroom bots, and malicious bots.

6. Why is Cybersecurity important?

Cybersecurity is important because it encompasses everything that pertains to protecting our sensitive data, personally identifiable information (PII), protected health information (PHI), personal information, intellectual property, data, and governmental and industry information systems from theft and damage attempted by criminals and adversaries.

7. The types of Cybersecurity.

Critical infrastructure security:
Critical infrastructure security consists of the cyber-physical systems that modern societies rely on. Having the infrastructure of an electricity grid on the internet makes it vulnerable to cyber-attacks.
Organizations with responsibility for any critical infrastructures should perform due diligence to understand the vulnerabilities and protect their business against them. The security and resilience of this critical infrastructure is vital to our society's safety and well-being.

Application security:
You should choose application security as one of the several must-have security measures adopted to protect your systems. Application security uses software and hardware methods to tackle external threats that can arise in the development stage of an application.
Applications are much more accessible over networks, causing the adoption of security measures during the development phase to be an imperative phase of the project.

Network security:
As cybersecurity is concerned with outside threats, network security guards against unauthorized intrusion of your internal networks due to malicious intent.
Network security ensures that internal networks are secure by protecting the infrastructure and inhibiting access to it.
To help better manage network security monitoring, security teams are now using machine learning to flag abnormal traffic and alert to threats in real-time. Network administrators continue to implement policies and procedures to prevent unauthorized access, modification, and exploitation of the network.

Cloud security:
Improved cybersecurity is one of the main reasons why the cloud is taking over.

Cloud security is a software-based security tool that protects and monitors the data in your cloud resources. Cloud providers are constantly creating and implementing new security tools to help enterprise users better secure their data.
The myth flying around cloud computing is that it's less secure than traditional approaches. People tend to believe that your data is more secure when stored on physical servers and systems you own and control. However, it has been proven through cloud security that control does not mean security and accessibility matters more than physical location of your data.

Internet of things (IoT) security
IoT refers to a wide variety of critical and non-critical cyber-physical systems, like appliances, sensors, televisions, wifi routers, printers, and security cameras.
IoT's data center, analytics, consumer devices, networks, legacy embedded systems, and connectors are the core technology of the IoT market.
IoT devices are frequently sent in a vulnerable state and offer little to no security patching. This poses unique security challenges for all users.
This calls for vendors to invest in learning more about security challenges to suggest and implement more strategic solutions. In the meantime, IoT devices are near impossible to avoid and finding an IT provider that can manage your security is your best option.

8. Cybersecurity Tasks

a. Monitor all operations and infrastructure. This could be something you do by yourself, or you could be leading a team — either way, your daily bread and butter involves going through alerts and logs (the computer security equivalent of video surveillance) in order to keep an eye on your organization's digital security footprint.

b. Maintain all security tools and technology. This could be a shared responsibility or the sole responsibility of the IT security manager and their team.

c. Monitor internal and external policy compliance. You want to ensure that both your vendors and employees understand your cybersecurity risk management policies and that they operate within that framework. The IT security manager is the living embodiment of policy, and while you may not always be in charge of enforcement, you are responsible for making sure things are in line internally.

d. Monitor regulation compliance. This is particularly important if you're in a heavily regulated industry and are dealing with things like credit card information, health care data, or other personally identifiable information.

e. Work with different departments in the organization to reduce risk. From technical controls to policies (and everything in between), you'll likely be tasked with working across the aisle of departments in your organization to get everyone on the same page.

f. Implement new technology. If your organization is looking at a new technology, you must evaluate it and help implement any controls that might mitigate the risk of its operation.

g. Audit policies and controls continuously. Cybersecurity is a circular process, and as a manager, you must drive that process forward. This means regularly auditing the policies and controls you put into place. These audits will tell you if there's anything you need to improve, remediate, or quickly fix.

h. Ensure cybersecurity stays on the organizational radar. Does it seem as though the organization you're with isn't being proactive about cybersecurity? As the IT security manager, your job is to make the benefits clearly visible and champion all efforts going forward.

i. Detail out the security incident response program. Every organization should have a well-defined and documented plan of action to put into place if a security incident does occur.

9. Some Cybersecurity tools

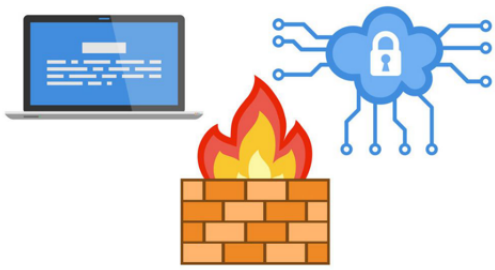


Figure 3: Network Firewall

As hacking and cyber-criminals become more sophisticated and defenses become stronger, you might assume that a firewall is obsolete. And while a firewall is arguably the most core of security tools, it remains one of the most important. Its job is to block any unauthorized access to your system.
A firewall monitors network traffic as well as connection attempts, deciding on whether or not these should be able to pass freely onto your network or computer. Of course, while they are useful, they do have limitations. Skilled hackers have learned how to create data and programs that trick firewalls into believing that they are trusted – this means that the program can pass through the firewall without any problems. Despite these limitations, firewalls are still very effective in detecting the large majority of less sophisticated malicious attacks on your business.

Antivirus Software
For individuals, firewalls and antivirus software constitute the bare minimum of security. At an enterprise level, though, two security layers aren't always enough. "Our clients have had those and still have been hacked," Busch said "If a firewall is the door to your house", Nayak said, "antivirus software might be the door to your bedroom that protects you against threats already in your system by scanning existing files". This souped-up software checks file signatures for signs of malignancy, but also monitors behavior. "A good EDR system can detect suspicious activity running on an endpoint," said Nicol — whether that endpoint is a PC, a Mac, or a server.
EDR is especially important, Busch explained, when a hacker has entered a system. For the hack to have a serious impact, the hacker must be able to siphon information out of your network. But EDR software can essentially quarantine compromised devices, so no new intel can be sent or received. That cuts off hacks at the knees. Even in less serious situations, EDR monitoring makes unusual activity visible to system administrators. That can be essential to flagging moles and much more. It's pricey, though, so EDR is typically only used by major companies.

10. Example of Cybersecurity issues

b. Leaving the door open

Being targeted from an outside attack is scary. Having someone on the inside cutting corners that could leave you wide open to attack can be just as bad. During onboarding a newly contracted client we discovered that their remote desktop services (RDS) were open to the public, leaving their server compromised and vulnerable to even a basic hack.
The compromise wasn't done maliciously, but out of laziness. The previous administrator wanted a convenient way to remote into the network in order to do his admin duties. Instead of using VPN and logging into RDS each time, he left the system open. The problem was easily fixed by our team who updated the core configuration capabilities on the RDS server, therefore closing the door.
It's an all-too-common occurrence when a vulnerability is exposed due to in-house IT staff. Whether you have a team of technicians or one administrator, it's in your best interest to investigate and close the gaps in your network. This can be done by performing routine assessments.

b. You get what you pay for

Free deals, especially when it comes to security software, are incredibly limited in their protection capabilities. The unfortunate situation of a client who invested their security posture in free software, and got exactly what they paid for.
The client was hit with a virus that acted as an DHCP server, propagating on the network by directing computers needing an IP address to grab it from the nearest infected system. Once it got an address from the infected system, it would be redirected to a website outside the network that contained malicious content. There, it would initiate a download of even more viruses and malware payloads, therefore accelerating the spread on the network.
It took our team three days to figure out and contain all the viruses which had spread throughout the network. Fortunately, ransomware didn't exist at the time, but it still cost the client to undergo this thorough clean-up job.
This is a reminder that, not only do you get a more robust security package with the paid version, it also comes with support from the security partner. It was that support that enabled our team to isolate and quarantine all the viruses on the network. We also recommend using multiple layers of protection, in addition to different technologies, to counter any cybersecurity breach.

c. Gone Spear Phishing

Hackers often use a company's vulnerabilities to set up sophisticated phishing scams, known as spear phishing. Unsuspectingly, they convince employees to give up valuable information and even large sums of money before they realize something isn't right.
This is the story of a client who was targeted for this kind of attack and how extensive the damage can be. One of the users in the accounting department received a phishing email, which was obviously fake to security experts, but to the average person seemed legitimate. The person clicked on a link which downloaded a file onto their computer, enabling the attackers to watch the employee's inbox and develop a profile for the company. Once they developed their profile and had a good understanding of who signed the checks, they went to work, requesting money to be wired to a foreign bank.

Fortunately, while the money was being prepared to be wired, we were able to discern what was going on and stop the transfer from happening. However, the hackers were able to breach several of the company's clients who also were compromised. Attacks like this don't stop at one victim, they catalog as many victims as they can, working down the client chain.

d. The weakest link

One layer of protection, and sometimes the easiest to penetrate, is employee behavior and understanding. In another email incident a small business was left exposed and it cost them. Acting as the owner, hackers were able to send an email displaying the owner's name, but from a generic mailbox, asking for a six-figure amount to be wired to a bank in China. It wasn't uncommon for the owner to make such a request, so the employee set up the transfer without verifying with the owner.
They found out within 48 hours that the request was fraudulent, but it was too late and the money was gone. In incidents like this, no amount of technical solutions can remedy the situation. This is about a breakdown in business processes. We recommend training your team on how to deal with these kinds of requests. From there, we can do penetration testing to ensure that employees are adhering to these processes.

11. The challenges of Cybersecurity

b. Expanded Attack Opportunities for Hackers

Another challenge of cybersecurity is dealing with the increasing overlap between the physical and virtual worlds of information exchange. As driverless cars and other self-regulated devices become the norm, the Internet of Things (IoT) and BYOD business policies give criminals more access to cyber-physical systems. That includes cars, factories, the smart fridge and toaster in your kitchen, to even one's medical pacemaker. In the future, infiltrating one of these systems may mean infiltrating them all.

b. Complicated Regulation

The regulatory environment is also complicating cybersecurity, especially the political discussions around consumer privacy. The European Union recently implemented the General Data Protection Regulation (GDPR) framework, creating more hurdles for companies to ensure they can do business without incurring hefty fines. The security mandates of regulatory agreements like the GDPR require all companies to be held to a higher standard, which can translate into more complications for SMBs and startups in the short term. In the long term, the virtual environment would likely be safer for everyone involved. However, there is a balance that must be achieved between protecting the consumer and offering that same consumer the choice of new business.

c. Lack of IT Talent

A critical challenge of cybersecurity is the lack of qualified professionals to do the job. There are many people on the low end of the cybersecurity spectrum with generic skills. Security Experts who know how to protect companies from sophisticated hackers are rare. Those who know how to get things done understand how in-demand they are. When they work, they charge fees that most smaller enterprises cannot afford. Only the biggest and richest companies in the world can afford these elite-level services, another hurdle that SMBs have to overcome to compete online.

12. Cybersecurity in the future

The future of cybersecurity will be largely determined by what hackers will be after. It will primarily be in two areas: Health and Wealth. More specifically, this means private healthcare-related data and sensitive payment information. Private health records can be worth a fortune. Credit card data can be used for fraud or sold on the black market.
These are the two categories that currently get hacked the most, and hackers are only going to target them more and more in the future. But as a response, expect governments and regulators to step up and try to mitigate the threats. The EU's General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA) are just the beginning.
Hackers will continue to be more sophisticated, using new methods and tools to gain access to private information. At the same time, technology will continue to evolve, providing hackers with an even larger attack surface and more vulnerabilities to exploit. Even unsophisticated hackers will automate their strategies in their efforts to infiltrate vulnerable companies. Hackers will also take advantage of the fact that companies increasingly rely on their supply chains and will target popular third-party tools, suppliers, and companies as a way to breach as many targets as possible.
To defend against such cyberattacks, companies will need to use more effective security solutions with innovative approaches. For instance, companies will assess their cybersecurity as seen from the hacker's point of view. The goal will be to not only increase cyber resilience internally within their specific company, but also across the company's supply chain.

II. An overview of Deep learning

1. Brief history and the evolution of deep learning

b. Brief history

- In 1989, Yann LeCun et al. applied the standard backpropagation algorithm, which had been around as the reverse mode of automatic differentiation since 1970, to a deep neural network with the purpose of recognizing handwritten ZIP codes on mail. While the algorithm worked, training required 3 days.
- By 1991 such systems were used for recognizing isolated 2D hand-written digits, while recognizing 3D objects was done by matching 2D images with a handcrafted 3D object model.
- In 1994, André de Carvalho, together with Mike Fairhurst and David Bisset, published experimental results of a multi-layer boolean neural network, also known as a weightless neural network, composed of a 3-layers self-organizing feature extraction neural network module (SOFT) followed by a multi-layer classification neural network module (GSN), which were independently trained.
- In 1995, Brendan Frey demonstrated that it was possible to train (over two days) a network containing six fully connected layers and several hundred hidden units using the wake-sleep algorithm, co-developed with Peter Dayan and Hinton.
- In 2006, publications by Geoff Hinton, Ruslan Salakhutdinov, Osindero and I showed how a many-layered feedforward neural network could be effectively pre-trained one layer at a time, treating each layer in turn as an unsupervised restricted Boltzmann machine, then fine-tuning it using supervised backpropagation.

- The 2009 NIPS Workshop on Deep Learning for Speech Recognition was motivated by the limitations of deep generative models of speech, and the possibility that given more capable hardware and large-scale data sets that deep neural nets (DNN) might become practical.
- In 2010, researchers extended deep learning from TIMIT to large vocabulary speech recognition, by adopting large output layers of the DNN based on context-dependent HMM states constructed by decision trees.
- In 2012, a team led by George E. Dahl won the "Merck Molecular Activity Challenge" using multi-task deep neural networks to predict the bio-molecular target of one drug.
- In March 2019, Yoshua Bengio, Geoffrey Hinton, and Yann LeCun were awarded the Turing Award for conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing.

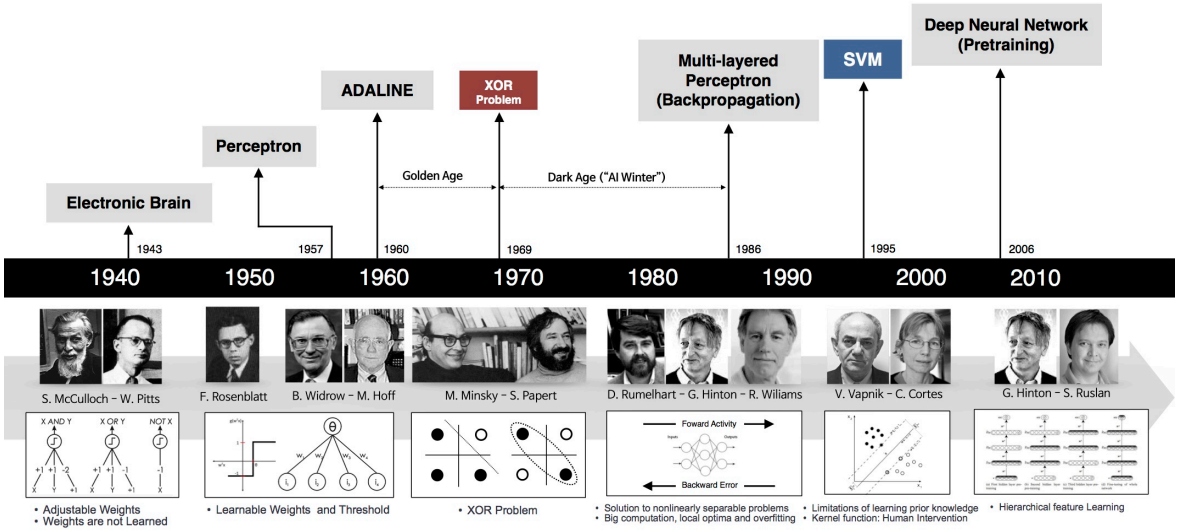


Figure 4: This history of deep learning

b. The evolution

Most of us know Deep Learning to be a 21st Century invention, but believe it or not, it has been around since the 1940s. The reason most of us are unaware about Deep Learning advancements/researches of 20th century is because the approaches used back then, were relatively unpopular due to their various shortcomings and the fact that it has had a couple of re-brandings since then.

New original research in any field requires an understanding of the history, evolution and major breakthroughs that led to the popularization of said field. Deep Learning is no exception. A broader look at the history of Deep Learning reveals 3 major waves of advancements:

Cybernetics — During 1940–1960
Connectionism — During 1980–1990
Deep Learning — Since 2006

Cybernetics

Is the earliest predecessor of modern Deep Learning, based on the idea of biological learning — how human brains learn. Advancements done in the name of cybernetics were based on the goal to replicate the working of a human/animal brain in a simpler

computational model, which would help build systems that would start learning like actual brains and provide conclusions given some input. Further research under this mindset continues separately till now under Computational Neuroscience.
Cybernetics was kicked off by the development of the McCulloch-Pitts Neuron. It was an attempt to mimic the biological neuron. It was based on a linear model that would take various inputs [X1, X2 …. Xn], for each input the model had some weights [W1, W2 … Wn] and the output f (x, w) = X1W1 + X2W2 + …. + XnWn. This model could only output True/False based on the inputs and weights.

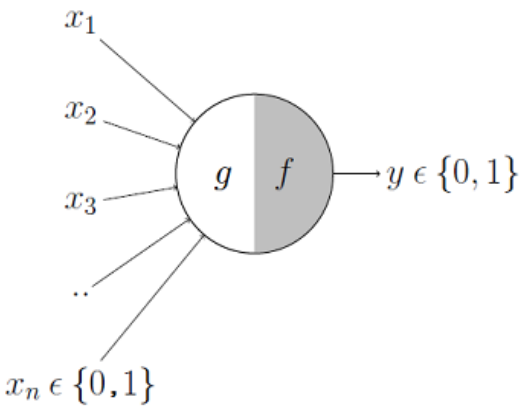


Figure 5: The cybernetics

The weights needed to be set properly, and were input by a human manually.
Later in the 1950s, Perceptron was developed by Frank Rosenblatt, an American psychologist, that would learn the weights automatically. Perceptron was initially designed to be an electrical machine rather than a program/software. Frank built Perceptron for image recognition, it contained photocells(receivers) connected to multiple neurons which would classify the inputs captured by the photocells. Though Perceptron was a remarkable machine for that time, it made bold claims which couldn't be fulfilled at the time.
ADALINE —Developed by Bernard Widrow, known as adaptive linear element, which was developed around the same time as Perceptron, could also adapt to the weights based on the weighted sum of the inputs during the learning phase.

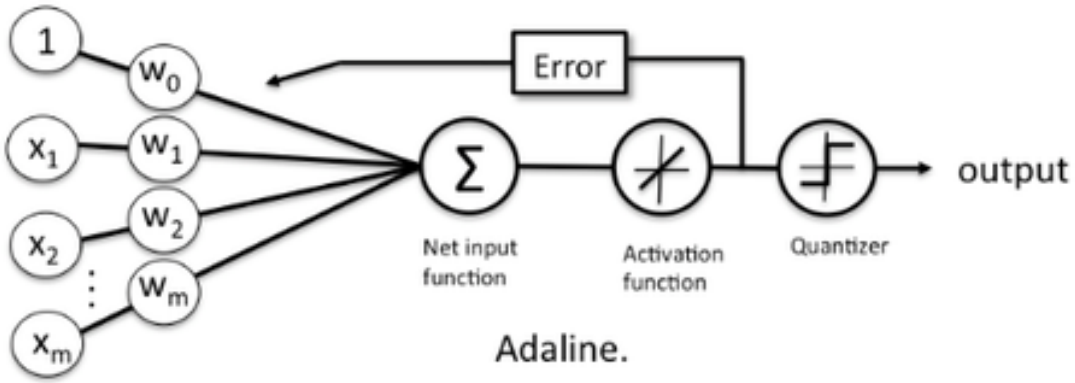


Figure 6: The Adaline works

The learning function for ADALINE is similar to stochastic gradient descent used in Linear Regression today.

These Linear models had various limitations and the critics who saw these limitations caused major dip in their popularity and stagnate the research for a while. One major limitation was that these linear models could not train for XOR functions.
Because these models were inspired by neuro-scientific research, the dip in their popularity also inspired exploration of models apart from neuroscientific basis.
Connectionism or Parallel Distributed Processing became popular in 1980s. This approach was inspired by cognitive sciences. Connectionism showed promise compared to various symbolic reasoning approaches scientists were exploring in the 1980s known as Classicists. Even though if we look at the brain in a more abstract view, Symbolic Reasoning approach fits well but are hard to implement explicitly using classical programmatic models. So practical connectionists viewed their work as leveraging Neural Nets to achieve similar effect as Symbolic Reasoning. But Radical Connectionist simply discarded the idea of Symbolic Reasoning stating that it could not explain various complex features of our brain anyway and was an incorrect perception of the human brain.
Concept of Artificial Neural Network (ANNs) was introduced during this wave. The main idea behind ANNs was to develop a network of individual units that can be programmed to achieve intelligent behavior. This was the first time the concept of hidden layers was introduced.
A network of artificial neurons connected with each other allowed parallel signal processing distributed along various branches of the network. The connections b/w the "neuron" units contained weights to control strength of the effect a neuron has on another.

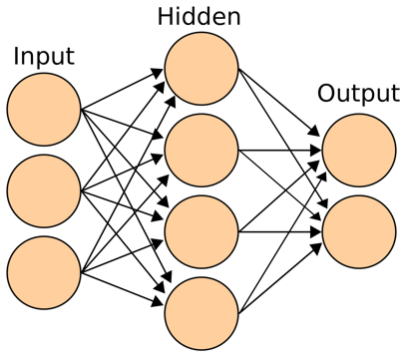


Figure 7: The concept of Artificial Neural Networks (ANNs)

This approach was perceived to be quite similar to what happens inside our nervous system and this caused some hipe among the researchers about the effectiveness of these models.
During this wave of Connectionism, various models like LSTM, distributed representation and processing, back-propagation to train deep neural nets were developed and continue to remain key components of various advanced applications of Deep learning to this date.
But in the mid-1990s the startups based on AI started to make unrealistic claims and could never deliver that level of sophistication from these models due to the lack of computational resources. Investors pulled back and this led to dip in this second wave of Deep Learning.
The second wave never died but was diminished. Research went on in various labs but applications were very few until early 2000s.
Deep Learning: After two dips, the third wave emerged in 2006 with a breakthrough. Geoffrey Hinton used Greedy Layer-wise Training to train Deep Belief Networks.
In the simplest of forms the DBN's are a composition of multiple hidden layers and each layer containing various latent variables. Connections exist b/w layers but not between the variables inside each layer. A very simple implementation of DBN can also be called restricted Boltzmann machines.

The advancements by Geoffrey Hinton were used by other researchers to train different types of Deep Networks. This enabled researchers around the world to trainer deeper and deeper neural networks and led to the popularization of the term Deep Learning.
While it might seem that Geoffrey Hinton lead to emergence of Deep Learning, you can't ignore the increase in computational powers and availability of large datasets. Same algorithms developed during Connectionism started to give better results when trained on larger and larger datasets.
The difference b/w then and now, is, with more and more people using online services we have a lot more data and a lot of superior computational resources to work with that data, hence increasing the accuracy for various models.
A more interesting and complex applications of Deep Learning are surfacing but are at early stages of practical use. For example, Deep Learning has been used to develop 3D map of the brain (Connectome) to help neuroscientists and cognitive scientists study the brain. Pharmaceutical companies are beginning to use Deep Learning to predict how different molecules will react and help accelerate drug development.

2. What is the definition of deep learning?

Deep learning is a class of machine learning algorithms that uses multiple layers to progressively extract higher level features from the raw input.
For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

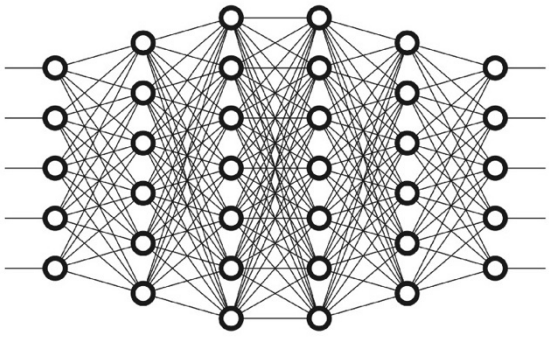


Figure 8: Deep learning networks

3. Deep learning methods

3.1. Deep learning for regression

Regression (or prediction) is simple. The knowledge about the existing data is utilized to have an idea of the new data. Take an example of house prices prediction. In cybersecurity, it can be applied to fraud detection. The features (e.g., the total amount of suspicious transaction, location, etc.) determine a probability of fraudulent actions.

Deep learning methods for regression
For regression tasks, the following deep learning models can be used:

- Artificial Neural Network (ANN)
- Recurrent Neural Network (RNN)
- Neural Turing Machines (NTM)

- Differentiable Neural Computer (DNC)

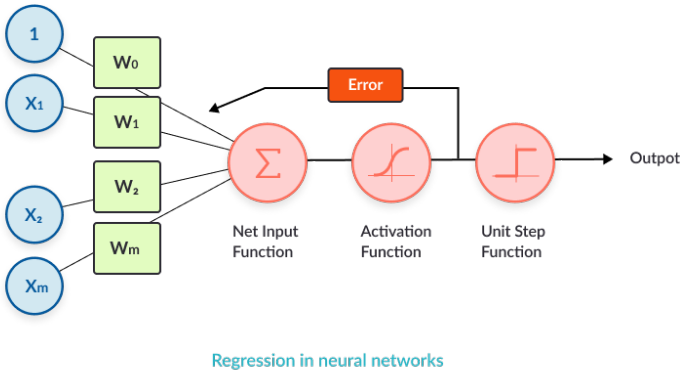


Figure 9: Deep learning methods for regression

3.2. Deep learning for classification

Classification is also straightforward. Imagine you have two piles of pictures classified by type (e.g., dogs and cats). In terms of cybersecurity, a spam filter separating spams from other messages can serve as an example. Spam filters are probably the first ML approach applied to Cybersecurity tasks.

Deep learning methods for classification

- Artificial Neural Network
- Convolutional Neural Networks

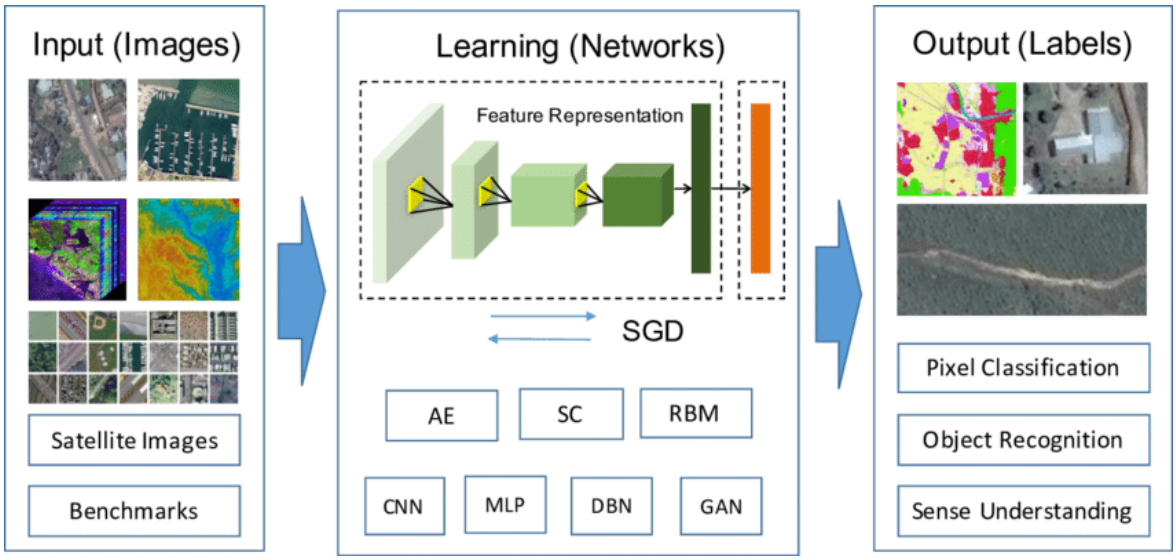


Figure 10: Deep learning methods for classification

3.3. Deep learning for clustering

Clustering is similar to classification with the only but major difference. The information about the classes of the data is unknown. There is no idea whether this data can be classified. This is unsupervised learning.
Supposedly, the best task for clustering is forensic analysis. The reasons, course, and consequences of an incident are obscure. It's required to classify all activities to find anomalies. Solutions to malware analysis (i.e., malware protection or secure email gateways) may implement it to separate legal files from outliers.
Another interesting area where clustering can be applied is user behavior analytics. In this instance, application users cluster together so that it is possible to see if they should belong to a particular group. Usually clustering is not applied to solving a particular task in cybersecurity as it is more like one of the subtasks in a pipeline (e.g., grouping users into separate groups to adjust risk values).

Deep learning for clustering

- Self-organized Maps (SOM) or Kohonen Networks

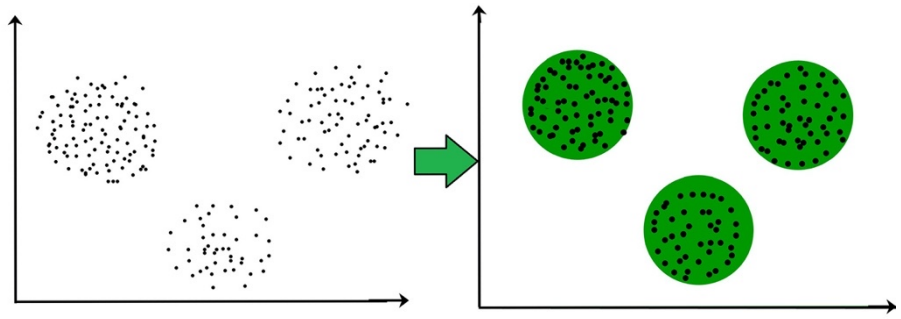


Figure 11: Deep learning for clustering

3.4. Deep learning for association rule learning

Netflix and SoundCloud recommend films or songs according to your movies or music preferences. In cybersecurity, this principle can be used primarily for incident response. If a company faces a wave of incidents and offers various types of responses, a system learns a type of response for a particular incident (e.g., mark it as a false positive, change a risk value, run the investigation). Risk management solutions can also have a benefit if they automatically assign risk values for new vulnerabilities or misconfigurations built on their description.

Deep learning for association rule learning

- Deep Restricted Boltzmann Machine (RBM)
- Deep Belief Network (DBN)

- Stacked Autoencoder

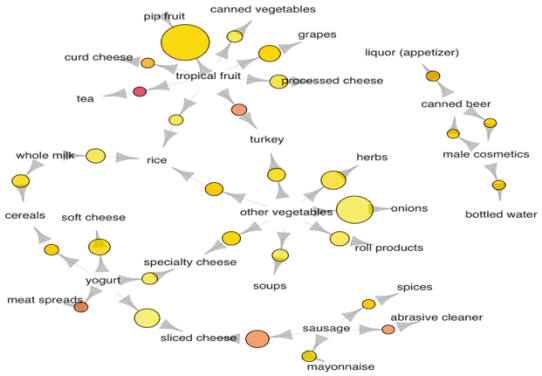


Figure 12: Deep learning for association rule learning

3.5. Deep learning generative models

The task of generative models differs from the above-mentioned ones. While those tasks deal with the existing information and associated decisions, generative models are designed to simulate the actual data (not decisions) based on the previous decisions.
The simple task of offensive cybersecurity is to generate a list of input parameters to test a particular application for Injection vulnerabilities.
Alternatively, you can have a vulnerability scanning tool for web applications. One of its modules is testing files for unauthorized access. These tests are able to mutate existing filenames to identify the new ones. For example, if a crawler detected a file called login.php, it's better to check the existence of any backup or test its copies by trying names like login_1.php, login_backup.php, login.php.2017. Generative models are good at this.

Deep learning generative models

- Variational Autoencoders
- Generative adversarial networks (GANs)
- Boltzmann Machines

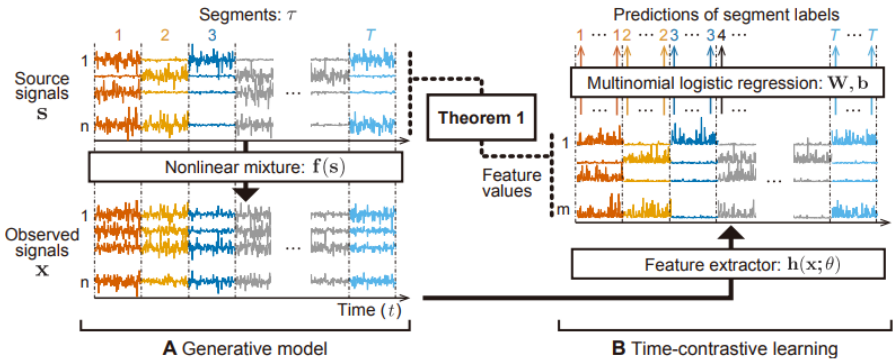


Figure 13: Deep learning generative models

4. How deep learning used in Cybersecurity?

- Regression (or prediction) — a task of predicting the next value based on the previous values.
- Classification — a task of separating things into different categories.
- Clustering — similar to classification but the classes are unknown, grouping things by their similarity.
- Association rule learning (or recommendation) — a task of recommending something based on the previous experience.
- Dimensionality reduction — or generalization, a task of searching common and most important features in multiple examples.
- Generative models — a task of creating something based on the previous knowledge of the distribution.

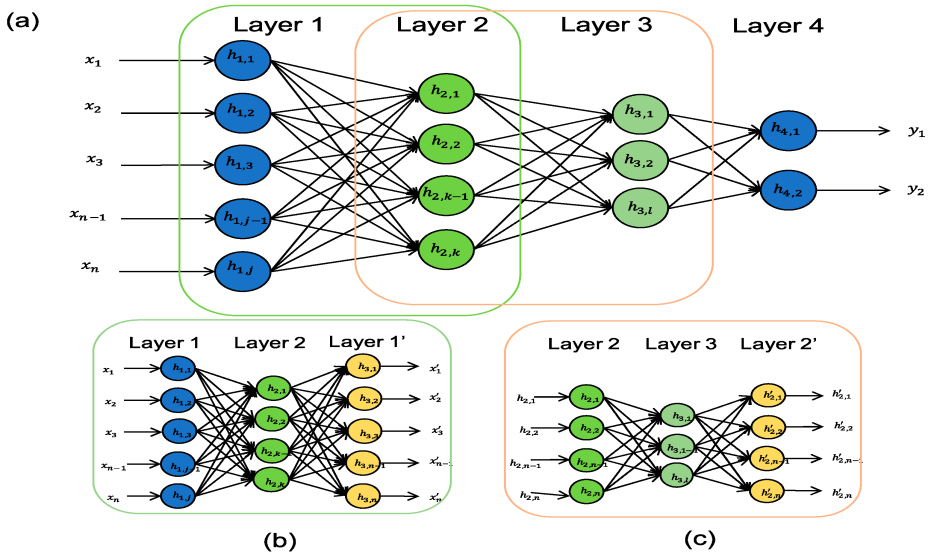


Figure 14: Deep learning in cybersecurity

5. Is Deep learning the Future of Cybersecurity?

While machine vision applications can easily sift through image data, or speech recognition can integrate human voice into conventional neural network architectures, cybersecurity is a different kettle of fish altogether—and that's because for a cybersecurity application to be effective, it must be able to test the entirety of any dataset for weaknesses or exploits.

"When you're using traditional machine learning, whether that's for vision, for speech, or for text, you must conduct feature engineering first," David says. "If you're using machine learning to detect a malicious executable file, you have to decide in advance what the important features to look out for are. In cybersecurity, we are working with different file formats, file sizes, and applications."

Machine learning techniques look for the key features within datasets. These are narrowly defined by human operators, meaning that much of the granularity of a dataset is easily missed out. This, David explains, is part of the reason that it is so easy for adversaries to evade detection. "Imagine you have a big executable file. By performing traditional machine learning, you're only looking at several hundred feature locations within the file, and you

can quite easily disregard most of the raw data. So, it's quite easy for the attacker to very slightly modify the raw data and evade detection," says David. "On the other hand, end-to-end deep learning applied to raw data does not require feature extraction. Traditional machine learning solutions tend to do much better than antivirus solutions, but in our case, we have an even bigger margin over traditional machine learning solutions."

6. The role of applying Deep learning in Cybersecurity

Deep Learning to combat malware and online fraud. The crime-as-a-service sector evolves fast, making increasingly innovative new developments available to cybercriminals so they can successfully reach their targets. These developments have become dynamic threats, able to adapt to the security measures deployed by users and organizations to combat cybercrime. In this context, one major challenge is unquestionably the classification of malware. The malware that seems to be "fashionable" today is already obsolete tomorrow and is replaced by another one with completely different or improved features. At the same time, the newest varieties of malware continue to coexist with older forms, which are still used by cyber criminals without the means to innovate. Therefore, classification in the cybercriminal ecosystem is very complex. Faced with a context such as this, the blacklists and indicators of compromise (IOCs) typical of Cyber Threat Intelligence are not enough to handle the threat, which transforms itself when it detects it has been identified.
The great strength of Deep Learning for cybersecurity is that it makes it possible to learn from this dynamism in real time and develop new classification criteria without human intervention. Thanks to this, detection and classification become much more efficient and proactive. At the same time, its applications are infinite. In the case of buguroo, for example, we also use it together with our development of identification based on biometric behavior. This allows us to rapidly recognize whether a person is interacting with their computer or it is a bot, or if there is a cybercriminal attempting a user Account Takeover or interacting with a user's account from anywhere in the world (Remote Access Trojan).

C. TECHNOLOGY DESCRIPTION

I. Deep learning in Cybersecurity

1. Deep learning for regression

1.1. Artificial Neural Network (ANN)

The Artificial Neural Network (ANN) consists of 3 main components: The input layer and the output layer consist of only 1 layer; the hidden layer can have 1 or more layers depending on the specific problem. ANN works in a way that describes how the nervous system works with neurons connected. In ANN, except for the input layer, all nodes belonging to other layers are fully-connected with nodes belonging to the previous layer. Each node in the hidden layer receives the input matrix from the previous layer and combines the weight to get the result. The overview of ANN
Logistic Regression has the function is sigmoid function

$$g(z) = \frac{1}{1 + e^{-z}}$$

Hypothesys function:

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

The diagram is:

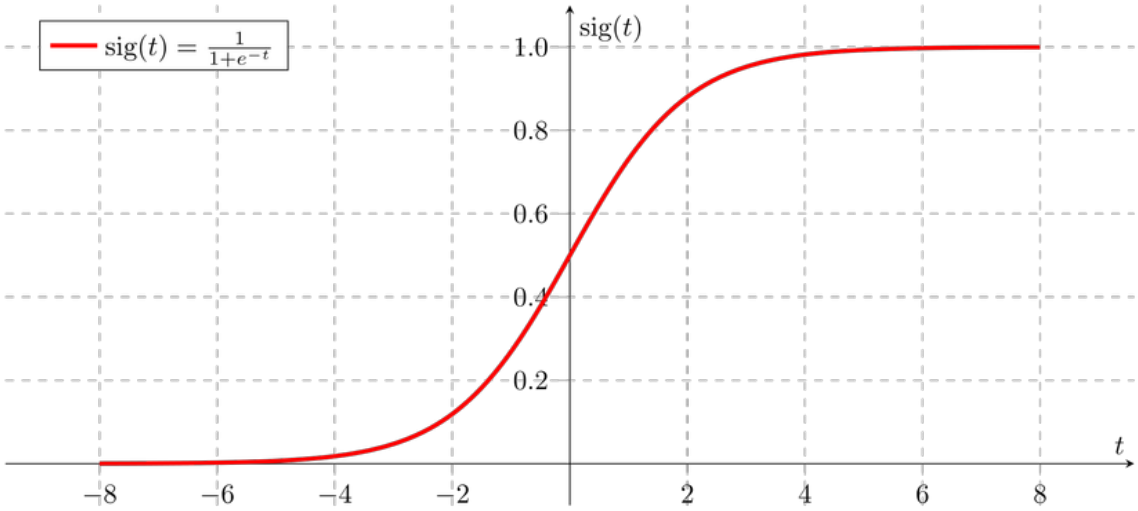


Diagram 1: The diagram for Hypothesys function

Cost function:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)})$$

With

$$Cost(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) \text{ nếu } y = 1 \\ -\log(1 - h_\theta(x)) \text{ nếu } y = 0 \end{cases}$$

cost function:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)]$$

Combine to Regurlarization

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

So, with ANN for every node belonging to another layer, the input layer is a Logistic Regression we have

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\theta(x^{(i)})_k) + (1 - y_k^{(i)}) \log\left(1 - h_\theta(x^{(i)})_k\right)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1} (\Theta_j^{(l)})^2$$

Methodology 1: Hypothesys function

Our job now is to find \ Theta Θ such that J (\ Theta) J (Θ) min.
To find the minimum of J (\ Theta) J (Θ) we apply the Gradient Descent algorithm.

Repeat {

$$\Theta_j := \Theta_j - \alpha \frac{\partial}{\partial \Theta_j} J(\Theta);$$

}

A is the learning rate.
In order to do this, it is necessary to calculate \ frac {∂} {∂ \ Theta_ {j}} J (\ Theta)
∂Θ
j ∂ J (Θ), to calculate this derivative is relatively difficult and we need to implement an algorithm called backpropagation to calculate.

Forward propagation
We have the neuron network:

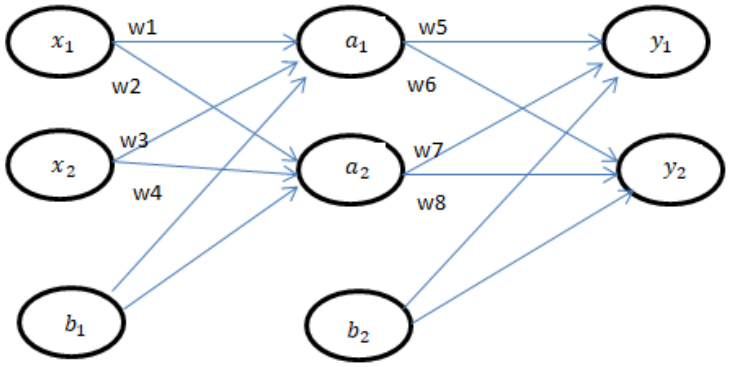


Figure 15: neuron networks

Note:
X1, x2 is features of input
Y1, y2 is output
B1, b2 is bias
W1, w2 is number
With the name of forward propagation, we carry out calculate a1,a2, ,y1,y2 from left to right

$$z_1 = x_1 w_1 + x_2 w_3 + b_1$$
$$a_1 = sigmoid(z_1) = \frac{1}{1+e^{x_1 w_1 + x_2 w_3 + b_1}}$$

The same:

$$z_2 = x_1 w_2 + x_2 w_4 + b_2$$
$$a_2 = \frac{1}{1+e^{x_1 w_2 + x_2 w_4 + b_2}}$$
$$z_3 = a_1 w_5 + a_2 w_7 + b_2$$
$$y_1 = \frac{1}{1+e^{a_1 w_5 + a_2 w_7 + b_2}}$$
$$z_4 = a_1 w_6 + a_2 w_8 + b_2$$
$$y_2 = \frac{1}{1+e^{a_1 w_6 + a_2 w_8 + b_2}}$$

Forward propagation is a process of calculating the value at each node to serve the calculation in Backpropagation.

Back propagation

$$\frac{\partial J(w)}{\partial y_1} = [-(T_1 * \log(y_1) + T_2 * \log(y_2))]' = -T_1 * \frac{1}{y_1 * \ln(10)}$$

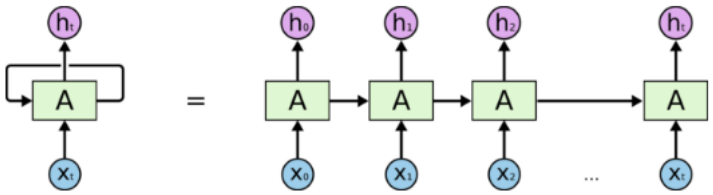
$$\frac{\partial y_1}{\partial z_3} = \left(\frac{1}{1+e^{-(z_3)}}\right)' = -\frac{\left(e^{-(z_3)}\right)'}{(1+e^{-(z_3)})^2} = -\frac{e^{-(z_3)}(-(z_3))'}{(1+e^{-(z_3)})^2} = \frac{e^{-(z_3)}}{(1+e^{-(z_3)})^2}$$

$$\frac{\partial z_3}{\partial w_5} = (a_1 w_5 + a_2 w_7 + b_2)' = a_1$$

Methodology 2: Hypothesys function

1.2. Recurrent Neural Network (RNN)

The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If you want to predict the next word in a sentence you better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations and you already know that they have a "memory" which captures information about what has been calculated so far.



An unrolled recurrent neural network.

Figure 16: Recurrent Neural Networks

"Whenever there is a sequence of data and that temporal dynamics that connects the data is more important than the spatial content of each individual frame."
– Lex Fridman (MIT)

Different types of RNN's: The core reason that recurrent nets are more exciting is that they allow us to operate over sequences of vectors: Sequences in the input, the output, or in the most general case both. A few examples may make this more concrete:

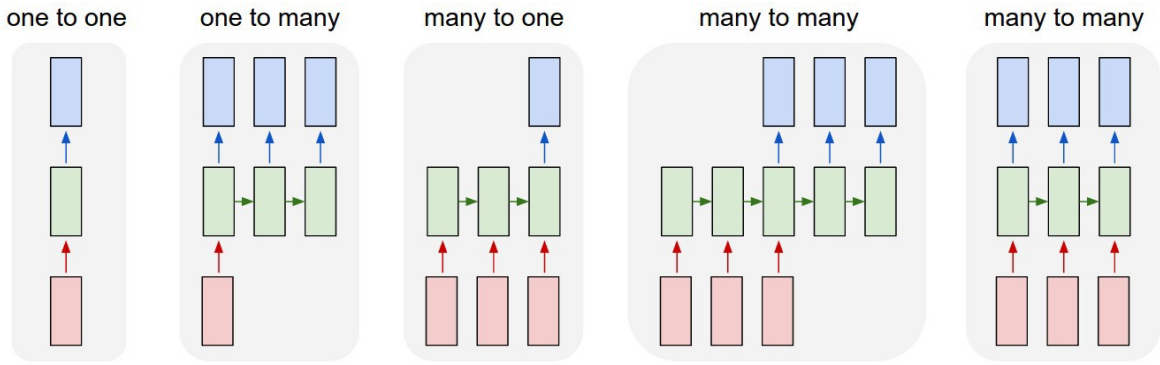


Figure 17: Recurrent Neural Networks

Different types of Recurrent Neural Networks. (2) Sequence output (e.g. image captioning takes an image and outputs a sentence of words). (3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). (4) Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). (5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like. Each rectangle in above image represent Vectors and Arrows represent functions. Input vectors are Red in color, output vectors are blue and green holds RNN's state.

One-to-one: This also called as Plain/Vaniall Neural networks. It deals with Fixed size of input to Fixed size of Output where they are independent of previous information/output. Example: Image classification.

One-to-Many: it deals with fixed size of information as input that gives sequence of data as output. Example: Image Captioning takes image as input and outputs a sentence of words.

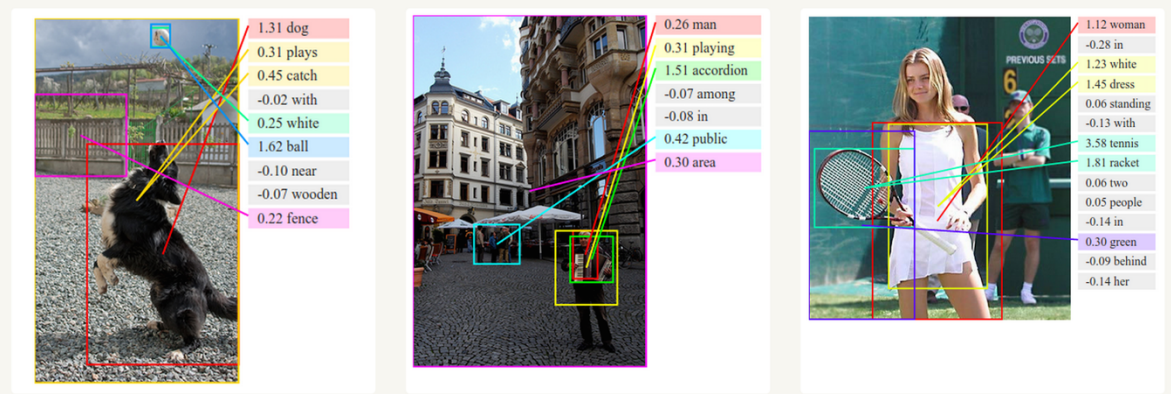


Figure 18: Image Captioning takes image as input and outputs a sentence of words.

Many-to-One: It takes Sequence of information as input and outputs a fixed size of output. Example: sentiment analysis where a given sentence is classified as expressing positive or negative sentiment.

Many-to-Many: It takes a Sequence of information as input and process it recurrently outputs a Sequence of data. Example: Machine Translation, where an RNN reads a sentence in English and then outputs a sentence in French.

Bidirectional Many-to-Many: Synced sequence input and output. Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like. Example: video classification where we wish to label each frame of the video.

Advantages of Recurrent Neural Network
The main advantage of RNN over ANN is that RNN can model sequence of data (i.e. time series) so that each sample can be assumed to be dependent on previous ones
Recurrent neural network is even used with convolutional layers to extend the effective pixel neighborhood.
Disadvantages of Recurrent Neural Network
Gradient vanishing and exploding problems.
Training an RNN is a very difficult task.
It cannot process very long sequences if using tanh or relu as an activation function

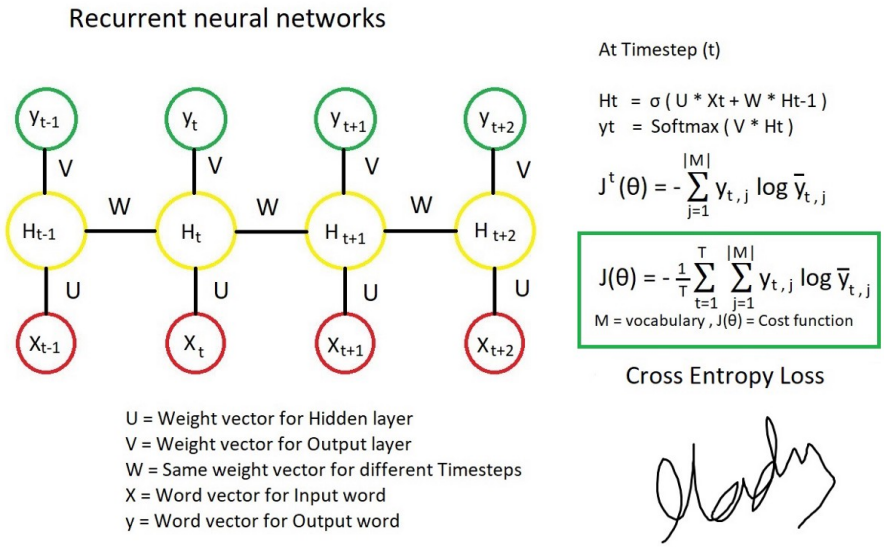


Figure 19: Recurrent neural networks

1.3. Neural Turing Machines (NTM)

A Neural Turing machine (NTMs) is a recurrent neural network model. The approach was published by Alex Graves et. al. in 2014. NTMs combine the fuzzy pattern matching capabilities of neural networks with the algorithmic power of programmable computers. An NTM has a neural network controller coupled to external memory resources, which it interacts with through attentional mechanisms. The memory interactions are differentiable end-to-end, making it possible to optimize them using gradient descent. An NTM with a long short-term memory (LSTM) network controller can infer simple algorithms such as copying, sorting, and associative recall from examples alone.

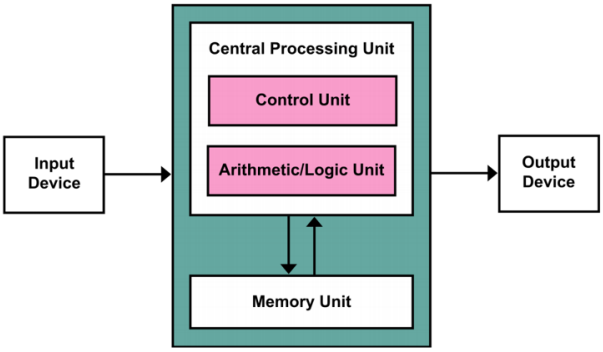


Figure 20: Neural Turning machines

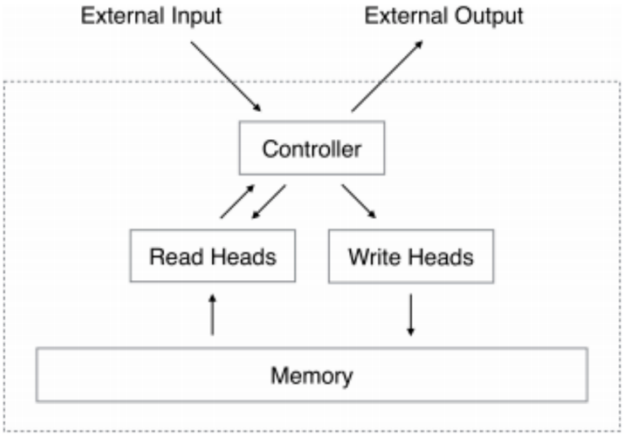
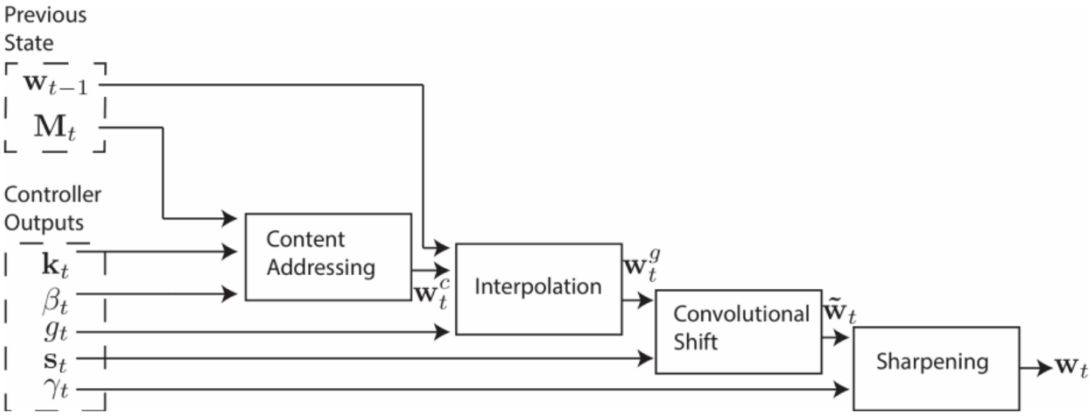


Figure 21: Neural Turning machines works

Algorithm:

1. Reading
2. Writing involves both erasing and adding
3. Addressing



Algorithm: Neural Turning machines works

4. Addressing

1. Focusing by Content

• Each head produces key vector kt of length M

• Generated a content-based weight wt c based on similarity measure, using 'key strength' βt

$$w_t^c(i) \leftarrow \frac{\exp\left(\beta_t K\left[\mathbf{k}_t, \mathbf{M}_t(i)\right]\right)}{\sum_j \exp\left(\beta_t K\left[\mathbf{k}_t, \mathbf{M}_t(j)\right]\right)}.$$

$$K\left[\mathbf{u}, \mathbf{v}\right] = \frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}|| \cdot ||\mathbf{v}||}.$$

Methodology 2: Addressing in NTMs

2. InterpolaBon
• Each head emits a scalar InterpolaBon gate gt

$$\mathbf{w}_t^g \longleftarrow g_t \mathbf{w}_t^c + (1 - g_t)\mathbf{w}_{t-1}.$$

3. ConvoluBonal shifs
• Each head emits a distribuBon over allowable integer shifs st

$$\tilde{w}_t(i) \longleftarrow \sum_{j=0}^{N-1} w_t^g(j)\, s_t(i-j)$$

Sharpening
• Each head emits a scalar sharpening parameter γt

$$w_t(i) \longleftarrow \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_j \tilde{w}_t(j)^{\gamma_t}}$$

This can operate in three complementary modes
• A weighting can be chosen by the content system without any modificaBon by the locaBon system
• A weighting produced by the content addressing system can be chosen and then shifed
• A weighting from the previous Bme step can be rotated without any input from the content-based addressing system

1.4. Differentiable Neural Computer (DNC)

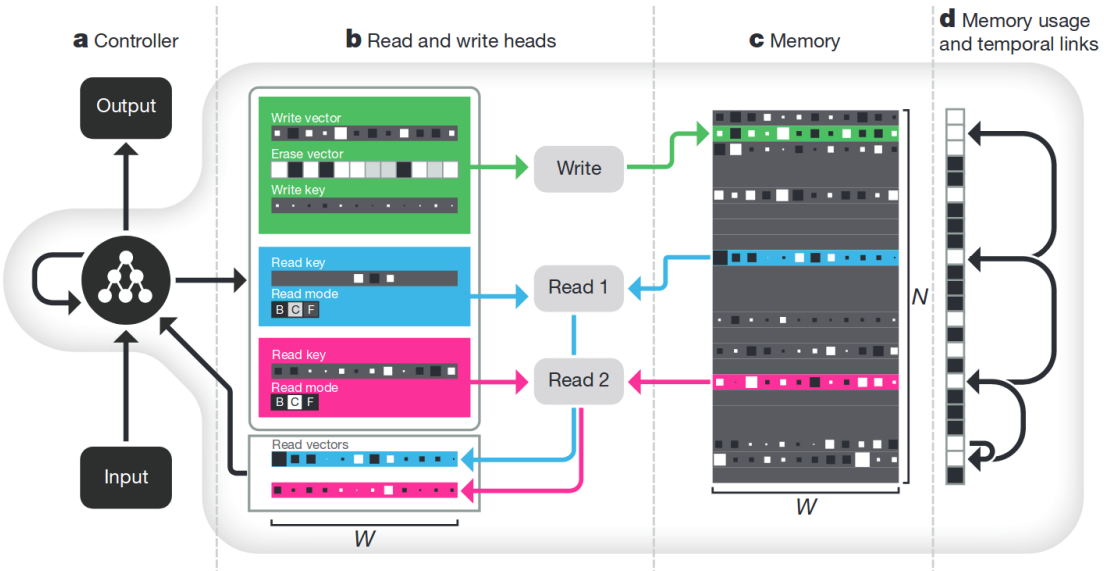


Figure 1.4-1: Overview of Differentiable Neural Computers

The Differentiable Neural Computer is a neural network that takes advantage of memory augmentation and, at the same time, the attention mechanism. The original paper is Hybrid computing using a neural network with dynamic external memory, by Alex Graves, Greg

Wayne. It was published in October 2016. Let's think of a DNC as a machine with a CPU and a RAM.
A neural network, the controller, will take the role of the CPU. A memory, which is just a matrix, will take the role of the RAM. Each row of the memory matrix is called location. The matrix will have N locations, which are W-dimensional vectors. Then, the memory is an N x W matrix. The controller will store and read information from the memory, and the network will learn how to do that by mean of some parameter vectors, but we will talk about that. The main point is that the memory of the network is external to the network itself.
Normally, the entire knowledge of a network is stored in its weights. Instead, a DNC is something called "Memory augmented neural network", and this peculiarity provides it with some nice features. The memory augmentation is not a novelty introduced by this paper; actually, it isn't a new thing at all. What is cool in the DNC is the system of vectors and operations mediating between controller and memory. This is called the attention mechanism, which is an essential concept of these past few years

Weightings and heads

Differentiability

Within the heads, a differentiable attention mechanism is applied in three different ways, as we will see. When I say differentiable, I'm saying that the functions producing the read and write weightings are differentiable, and then we can apply the gradient descent to the heads.

Intuition behind weightings

When I talk about weightings, I'm just referring to the following concept: the controller wants to do something which involves memory, and it doesn't just look at every location of the memory; instead, it focuses its attention on those locations which contain the information it is looking for. Who put the information in those locations? It was the controller itself to write it there before: it just wants to recover it now.
I said that every unit and operation in this structure is differentiable, that is we can learn everything by just iteratively applying gradient descent. So, the way in which the heads work is learned too!
To recap: the DNC can learn how to produce good weightings for each input, that is, it knows how to differently weight its memories on the base of their relative importance with regard to the given input. The weighting produced for an input is a distribution over the N locations for their relative importance in a particular process.

The Attention Noble Threefold Path.

There are three kinds of interactions between the controller and the memory, and they are mediated by the interface vector:
Content lookup: a particular set of values within the interface vector, which we will collect in something called key vector, is compared to the content of each location. This comparison is made by mean of a similarity measure (in this case, the cosine similarity).
Temporal memory linkage: the transitions between consecutively written locations are recorded in an N x N matrix, called temporal link matrix L. The sequence by which the controller writes in the memory is information by itself, and it's something we want to store.
Just think of it in this way: if I can't remember where I left my smartphone, I can try to remember where I was before arriving at my present position.

Dynamic memory allocation: each location has a usage level represented as a number from 0 to 1. A weighting that picks out an unused location is sent to the write head so that it knows where to store new information. The word "dynamic" refers to the ability of the controller to reallocate memory that is no longer required, erasing its content. Also, the allocation mechanism is independent of the size and contents of the memory, and that means that we can extend the memory without compromising or having to retrain the network.

2. Deep learning for classification

2.1. Artificial Neural Network

Artificial neural network (ANN), usually called Neural Network (NN), is an algorithm that was originally motivated by the goal of having machines that can mimic the brain. A neural network consists of an interconnected group of artificial neurons. They are physical cellular systems capable of obtaining, storing information, and using experiential knowledge. Like human brain, the ANN's knowledge comes from examples that they encounter. In human neural system, learning process includes the modifications to the synaptic connections between the neurons. In a similar way, ANNs adjust them structure based on output and input information that flows through the network during the learning phase. Data processing procedure in any typical neural network has two major steps: the learning and application step. At the first step, a training database or historical price data is needed to train the networks. This dataset includes an input vector and a known output vector. Each one of the inputs and outputs are representing a node or neuron. In addition, there are one or more hidden layers. The objective of the learning phase is to adjust the weights of the connections between different layers or nodes. After setting up the learning samples, in an iterative approach a sample will be fed into the network and the resulting outputs will be compared with the known outputs. If the result and the unknown output are not equal, changing the weights of the connections will be continued until the difference is minimized. After acquiring the desired convergence for the networks in the learning process, the validation dataset is applied to the network for the validating step (Shahkarami A. et al. 2014).
Learning Paradigms in ANNs: The ability to learn is a peculiar feature pertaining to intelligent systems, biological or otherwise. In artificial systems, learning (or training) is viewed as the process of updating the internal representation of the system in response to external stimuli so that it can perform a specific task. This includes modifying the network architecture, which involves adjusting the weights of the links, pruning or creating some connection links, and/ or changing the firing rules of the individual neurons. ANN approach learning has demonstrated their capability in financial modelling and prediction as the network is presented with training examples, similar to the way we learn from experience. In this paper, a three-layered feed-forward ANN model was structured to predict stock price index movement is given. This ANN model consists of an input layer, a hidden layer and an output layer, each of which is connected to the other. At least one neuron would be employed in each layer of the ANN model. Inputs for the network were twelve technical indicators which were represented by twelve neurons in the input layer. Each neuron (unit) in the network is able to receive input signals, to process them and to send an output signal. Each neuron is connected at least with one neuron, and each connection is evaluated by a real number, called the weight coefficient, that reflects the degree of importance of the given connection in the neural network (Daniel et al. 1997)
The error between the predicted output value and the actual value is backpropagated through the network for the updating of the weights. This method is proven highly successful in training of multi-layered neural networks. The network is not just given reinforcement for

how it is doing on a task. Information about errors is also filtered back through the system and is used to adjust the connections between the layers, thus improving performance.
This a supervised learning procedure that attempts to minimize the error between the desired and the predicted outputs. If the error of the validation patterns increases, the network tends to be over adapted and the training should be stopped.
The most typical activation function used in neural networks is the logistic sigmoid transfer function. This function converts an input value to an output ranging from 0 to 1.

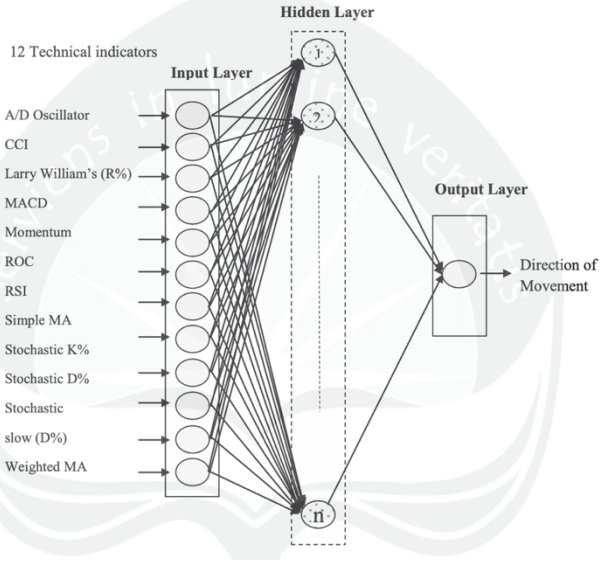


Figure 22: Neural network with 3-layer feed forward

The effect of the threshold weights is to shift the curve right or left, thereby making the output value higher or lower, depending on the sign of the threshold weight. The output values of the units are modulated by the connection weights, either magnified if the connection weight is positive and greater than 1.0, or being diminished if the connection weight is between 0.0 and 1.0. If the connection weight is negative or (value < 0) then tomorrow close price value < than today's price (loss). If (value > 0.5) then then tomorrow close price value > than today's price (profit). As shown in Fig. 2, the data flows from the input layer through zero, one, or more succeeding hidden layers and then to the output layer. The backpropagation (BP) algorithm is a generalization of the delta rule that works for networks with hidden layers. It is by far the most popular and most widely used learning algorithm by ANN researchers. Its popularity is due to its simplicity in design and implementation. The idea is to train a network by propagating the output errors backward through the layers. The errors serve to evaluate the derivatives of the error function with respect to the weights, which can then be adjusted. It involves a two-stage learning process using two passes: a forward pass and a backward pass. The basic back propagation algorithm consists of three steps (Fig. 2). Although, the most typical activation function used in neural networks is the logistic sigmoid transfer function. This function converts an input value to an output ranging from 0 to 1. The effect of the threshold weights is to shift the curve right or left, thereby making the output value higher or lower, depending on the sign of the threshold weight. The output values of the units are modulated by the connection weights, either magnified if the connection weight is positive and greater than 1.0, or being diminished if the connection weight is between 0.0 and 1.0. If the connection weight is negative or (value < 0) then tomorrow close price value < than today's price (loss). If (value

> 0.5) then then tomorrow close price value > than today's price (profit). As shown in Fig. 2, the data flows from the input layer through zero, one, or more succeeding hidden layers and then to the output layer. The backpropagation (BP) algorithm is a generalization of the delta rule that works for networks with hidden layers. It is by far the most popular and most widely used learning algorithm by ANN researchers. Its popularity is due to commercial back propagation tools provide the most impact on the neural network training time and performance. The output value for a unit is given by the following Equation

$$y = f(h_j)$$

$$= \left(\sum_{i=1}^{n} w_{ij}\, x_i, \theta_j\right)$$

$$= \begin{cases} 1 & w_{ij}\, x_i \geq \theta \\ 0 & w_{ij}\, x_i < \theta \end{cases} \qquad (i = 1,2,\ldots n) \qquad (1)$$

Where y the output value is computed from set of input patterns, Xi of ith unit in a previous layer, Wij is the weight on the connection from the neuron i th to j, j is the threshold value of the threshold function f, and n is the number of units in the previous layer. The function $f(x)$ is a sigmoid hyperbolic tangent function (Barndorff-Nielsen et al. 1993)

$$f(x) = \tanh(x) = \frac{1-e^{-z}}{1+e^{-z}} \qquad \text{Threshold: } f(x) = \begin{cases} 0 & x < 0,1 \\ 1 & if\ otherwise \end{cases} \qquad (2)$$

where $f(x)$ is the threshold function remains the most commonly applied in ANN models due to the activation function for time series prediction in back-propagation (Najeb Masoud, 2014):

$$y = f\left(\sum_{i-1}^{n} w_i\, x_i - u\right) = \left(\sum_{i=1}^{n} w_i\, x_i\right)$$

Once the output has been calculated, it can be passed to another neuron (or group of neurons) or sampled by the external environment. In terms of the weight change, Δwij, the formula equation is given as:

$$\Delta w_{ij} = \eta \delta_j x_i$$

where η is the learning rate (0<η<1), δj is the error at neuron j, xi is an input vector and with the weights vector. This rule of IDX can also be rewritten as:

$$\Delta w_i = -\eta(t_i - x_i w_i) x_i$$

Although a high learning rate, η, will speed up training (because of the large step) by changing the weight vector, w, significantly from one phase to another. According to Wythoff BJ. (1993) suggests that η, [0.1,1.0].

2.2. Convolutional Neural Networks

Convolutional Neural Networks are very similar to ordinary Neural Networks from the previous chapter: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply.

Architecture Overview

Recall: Regular Neural Nets. As we saw in the previous chapter, Neural Networks receive an input (a single vector), and transform it through a series of hidden layers. Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer, and where neurons in a single layer function completely independently and do not share any connections. The last fully-connected layer is called the "output layer" and in classification settings it represents the class scores.
Regular Neural Nets don't scale well to full images. In CIFAR-10, images are only of size 32x32x3 (32 wide, 32 high, 3 color channels), so a single fully-connected neuron in a first hidden layer of a regular Neural Network would have 32*32*3 = 3072 weights. This amount still seems manageable, but clearly this fully-connected structure does not scale to larger images. For example, an image of more respectable size, e.g. 200x200x3, would lead to neurons that have 200*200*3 = 120,000 weights. Moreover, we would almost certainly want to have several such neurons, so the parameters would add up quickly! Clearly, this full connectivity is wasteful and the huge number of parameters would quickly lead to overfitting.
3D volumes of neurons. Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. (Note that the word depth here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) For example, the input images in CIFAR-10 are an input volume of activations, and the volume has dimensions 32x32x3 (width, height, depth respectively). As we will soon see, the neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would for CIFAR-10 have dimensions 1x1x10, because by the end of the ConvNet architecture we will reduce the full image into a single vector of class scores, arranged along the depth dimension. Here is a visualization:

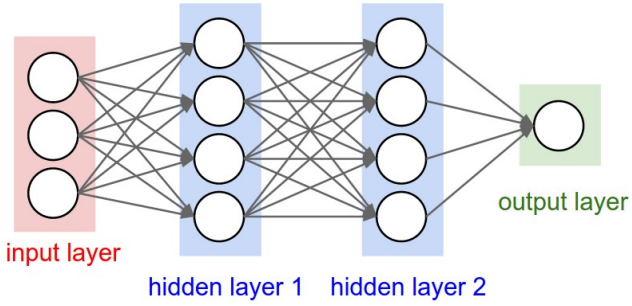


Figure 23: CNNs

Input: Acc_x, Acc_y, Acc_z, Gry_x, Gry_y, Gry_z, Mag_x, Mag_y, Mag_z, Pre

```
Sliding Window Process
sf ← Extract Shadow Features
Normalize sf by equation (2)
regularization feature data, size = 13×13
repeat:
  Forward Propagation:
  cd ← Convolution2D(sf);
  mp ← Max_pooling(cd);
  fc ← Fully_connected(mp);
  class label ← Soft_max(fc);
  Backward Propagation:
    conduct backward propagation with Adam;
Until w_i convergences;
 Use the trained network to predict the labels
```

Algorithm 1: CNNs

3. Deep learning for clustering

3.1. Self-organized Maps (SOM)

A self-organizing map (SOM) or self-organizing feature map (SOFM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map, and is therefore a method to do dimensionality reduction. Self-organizing maps differ from other artificial neural networks as they apply competitive learning as opposed to error-correction learning (such as backpropagation with gradient descent), and in the sense that they use a neighborhood function to preserve the topological properties of the input space.

Algorithm

- Randomize the node weight vectors in a map
- Randomly pick an input vector
- Traverse each node in the map
- Use the Euclidean distance formula to find the similarity between the input vector and the map's node's weight vector
- Track the node that produces the smallest distance (this node is the best matching unit, BMU)
- Update the weight vectors of the nodes in the neighborhood of the BMU (including the BMU itself) by pulling them closer to the input vector

$$1.\ W_v(s+1) = W_v(s) + \theta(u, v, s) \cdot \alpha(s) \cdot (D(t) - W_v(s))$$

A variant algorithm:

- Randomize the map's nodes' weight vectors

- Traverse each input vector in the input data set
- Traverse each node in the map
- Use the Euclidean distance formula to find the similarity between the input vector and the map's node's weight vector
- Track the node that produces the smallest distance (this node is the best matching unit, BMU)
- Update the nodes in the neighborhood of the BMU (including the BMU itself) by pulling them closer to the input vector

$$1.\ W_v(s+1) = W_v(s) + \theta(u, v, s) \cdot \alpha(s) \cdot (D(t) - W_v(s))$$

- Increase and repeat from step 2

3.2. Kohonen Networks

The objective of a Kohonen network is to map input vectors (patterns) of arbitrary dimension N onto a discrete map with 1 or 2 dimensions. Patterns close to one another in the input space should be close to one another in the map: they should be topologically ordered. A Kohonen network is composed of a grid of output units and N input units. The input pattern is fed to each output unit. The input lines to each output unit are weighted. These weights are initialised to small random numbers.
Learning in Kohonen Networks
The learning process is as roughly as follows: initialize the weights for each output unit
loop until weight changes are negligible, for each input pattern

- Present the input pattern
- Find the winning output unit
- Find all units in the neighborhood of the winner
- Update the weight vectors for all those units
- Reduce the size of neighbourboods if required

The winning output unit is simply the unit with the weight vector that has the smallest Euclidean distance to the input pattern. The neighbourhood of a unit is defined as all units within some distance of that unit on the map (not in weight space). In the demonstration below all the neighbourboods are square. If the size of the neighbourhood is 1 then all units no more than 1 either horizontally or vertically from any unit fall within its neighbourhood. The weights of every unit in the neighbourhood of the winning unit (including the winning unit itself) are updated using

$$\vec{w}_i = \vec{w}_i + \alpha(\vec{x}_i - \vec{w}_i) \tag{21}$$

This will move each unit in the neighbourhood closer to the input pattern. As time progresses the learning rate and the neighbourhood size are reduced. If the parameters are well chosen the final network should capture the natural clusters in the input data.

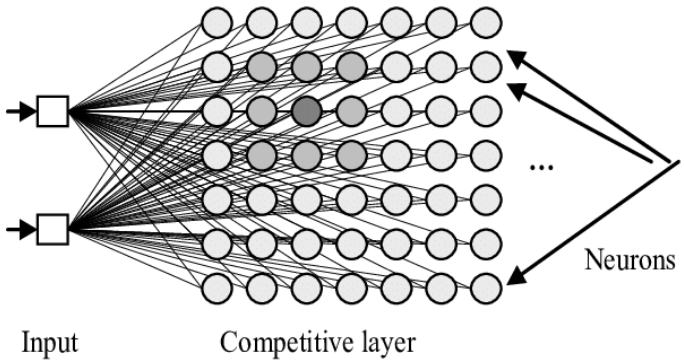


Figure 24: Kohonen Networks

4. Deep learning for association rule learning

4.1. Deep Restricted Boltzmann Machine (RBM)

Boltzmann Machines (BMs) are a particular form of log-linear Markov Random Field (MRF), i.e., for which the energy function is linear in its free parameters. To make them powerful enough to represent complicated distributions (i.e., go from the limited parametric setting to a non-parametric one), we consider that some of the variables are never observed (they are called hidden). By having more hidden variables (also called hidden units), we can increase the modeling capacity of the Boltzmann Machine (BM). Restricted Boltzmann Machines further restrict BMs to those without visible-visible and hidden-hidden connections. A graphical depiction of an RBM is shown below.

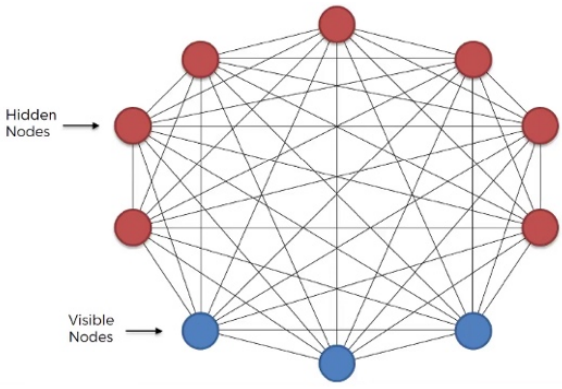


Figure 25: Boltzmann Machines (BMs)

Boltzmann Machine uses neural networks with neurons that are connected not only to other neurons in other layers but also to neurons within the same layer. Everything is connected to everything. Connections are bidirectional, visible neurons connected to each other and hidden neurons also connected to each other. Boltzmann Machine doesn't expect input data, it generates data. Neurons generate information regardless they are hidden or visible. For Boltzmann Machine all neurons are the same, it doesn't discriminate between hidden and visible neurons. For Boltzmann Machine whole things are system and its generating state of the system. The best way to think about it is through an example nuclear power plant.

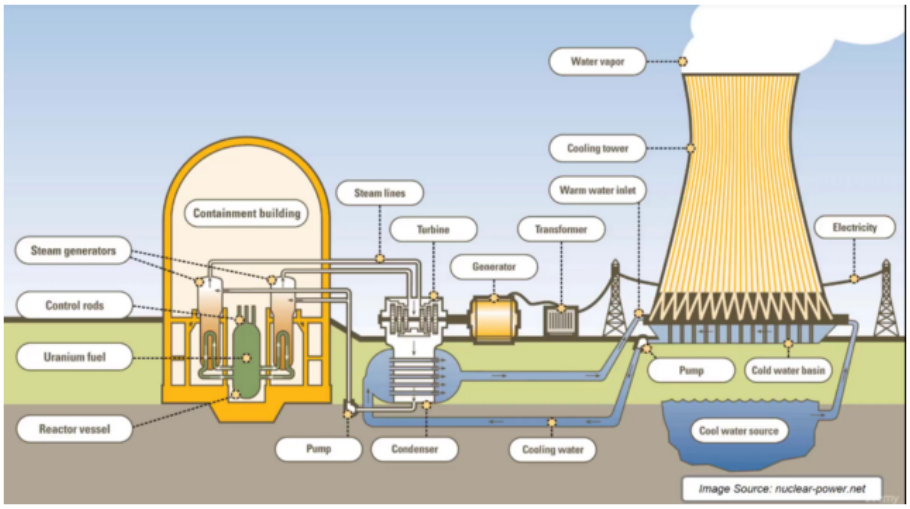


Figure 26: nuclear power plant.

In an RBM, we have a symmetric bipartite graph where no two units within the same group are connected. Multiple RBMs can also be stacked and can be fine-tuned through the process of gradient descent and back-propagation. Such a network is called a Deep Belief Network. Although RBMs are occasionally used, most people in the deep-learning community have started replacing their use with General Adversarial Networks or *Variational Autoencoders.*

RBM is a Stochastic Neural Network which means that each neuron will have some random behavior when activated. There are two other layers of bias units (hidden bias and visible bias) in an RBM. This is what makes RBMs different from autoencoders. The hidden bias RBM produces the activation on the forward pass and the visible bias helps RBM to reconstruct the input during a backward pass. The reconstructed input is always different from the actual input as there are no connections among the visible units and therefore, no way of transferring information among themselves.

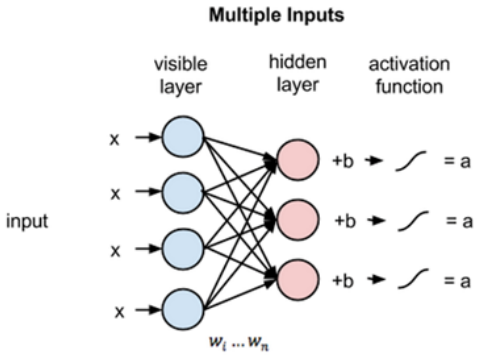


Figure 27: RBM

The above image shows the first step in training an RBM with multiple inputs. The inputs are multiplied by the weights and then added to the bias. The result is then passed through a sigmoid activation function and the output determines if the hidden state gets activated or not. Weights will be a matrix with the number of input nodes as the number of rows and the number of hidden nodes as the number of columns. The first hidden node will receive the vector multiplication of the inputs multiplied by the first column of weights before the corresponding bias term is added to it.

And if you are wondering what a sigmoid function is, here is the formula:

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}$$

So, the equation that we get in this step would be,

$$\mathbf{h}^{(1)} = S(\mathbf{v}^{(0)T}W + \mathbf{a})$$

where h (1) and v (0) are the corresponding vectors (column matrices) for the hidden and the visible layers with the superscript as the iteration v (0) means the input that we provide to the network) and a is the hidden layer bias vector.
(Note that we are dealing with vectors and matrices here and not one-dimensional values.)

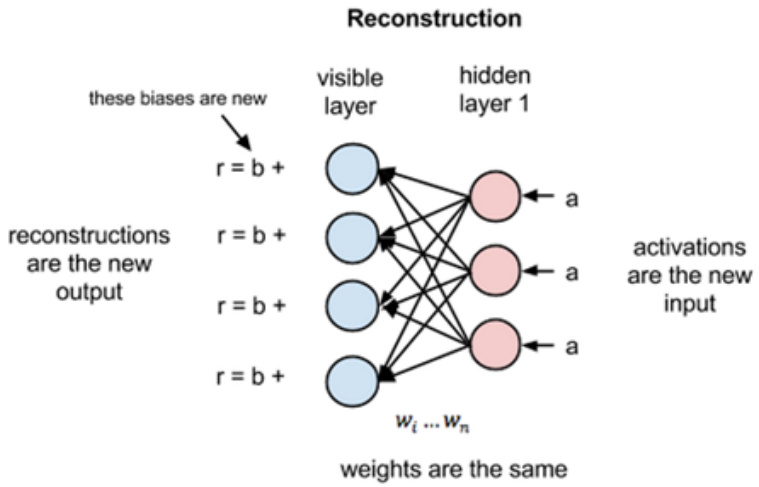


Figure 28: Reconstruction RBMs

Now this image shows the reverse phase or the reconstruction phase. It is similar to the first pass but in the opposite direction. The equation comes out to be:

$$\mathbf{v}^{(1)} = S(\mathbf{h}^{(1)}W^T + \mathbf{b})$$

where v (1) and h (1) are the corresponding vectors (column matrices) for the visible and the hidden layers with the superscript as the iteration and b is the visible layer bias vector.

4.2. Deep Belief Network (DBN)

Deep Belief Networks are a graphical representation which are essentially generative in nature i.e. it produces all possible values which can be generated for the case at hand. It is an amalgamation of probability and statistics with machine learning and neural networks. Deep Belief Networks consist of multiple layers with values, wherein there is a relation between the layers but not the values. The main aim is to help the system classify the data into different categories.

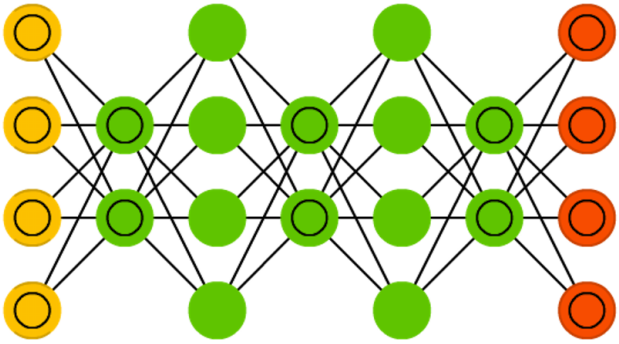


Figure 29: DBN

Training a Deep Belief Network

The first step is to train a layer of properties which can obtain the input signals from the pixels directly. The next step is to treat the values of this layer as pixels and learn the features of the previously obtained features in a second hidden layer. Every time another layer of properties or features is added to the belief network, there will be an improvement in the lower bound on the log probability of the training data set.
Example:

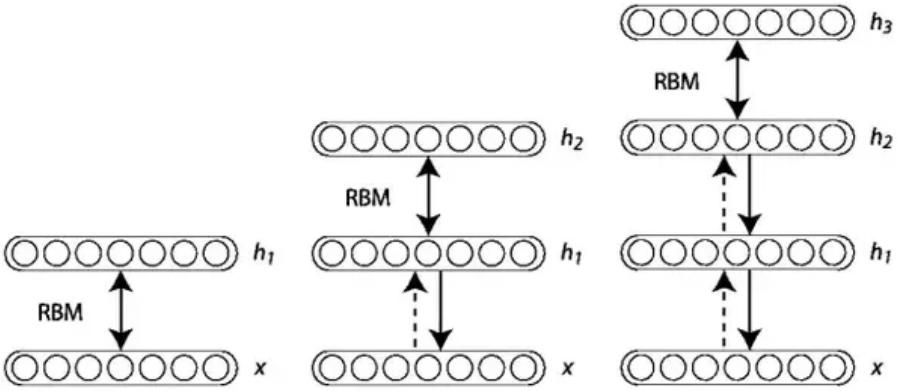


Figure 30: Training a Deep Belief Network

Implementation

MATLAB can easily represent visible layer, hidden layers and weights as matrices and execute algorithms efficiently. Hence, we choose MATLAB to implement DBN. These handwritten digits of MNIST9 are then used to perform calculations in order to compare the performance against other classifiers. The MNIST9 can be described as a database of handwritten digits. There are 60,000 training examples and 10,000 testing examples of digits. The handwritten digits are from 0 to 9 and are available in various shapes and positions for each and every image. Each of them is normalized and centered in 28x28 pixels and are labeled. The methods to decide how often these weights are updated are — mini batch, online and full-batch. Online learning takes the longest computation time because its updates weights after each training data instance. Full-batch goes through the training data and updates weights, however, it is not advisable to use it for big datasets. Mini-batch divides a dataset into smaller bits of data and performs the learning operation for every

chunk. This method takes less computation time. Hence, we use mini-batch learning for implementation.
An important thing to keep in mind is that implementing a Deep Belief Network demands training each layer of RBM. For this purpose, the units and parameters are first initialized. It is followed by two phases in Contrastive Divergence algorithm — positive and negative. In the positive phase, the binary states of the hidden layers can be obtained by calculating the probabilities of weights and visible units. Since it is increasing the probability of the training data set, it is called positive phase. The negative phase decreases the probability of samples generated by the model. The greedy learning algorithm is used to train the entire Deep Belief Network. The greedy learning algorithm trains one RBM at a time and until all the RBMs have been taught.

4.3. Stacked Autoencoder

An Autoencoder is a neural network which is an unsupervised learning algorithm which uses back propagation to generate output value which is almost close to the input value. Let's see now how an autoencoder actually works in detail. It takes input such as image or vector anything with a very high dimensionality and run through the neural network and tries to compress the data into a smaller representation with two principal components. The first one is the encoder which is simply a bunch of layers that are full connected layers or convolutional layers which are going to take the input and compress it to a smaller representation which has less dimensions then the input which is known as bottleneck. Now from this bottleneck it tries to reconstruct the input using full connected layers or convolutional layers.

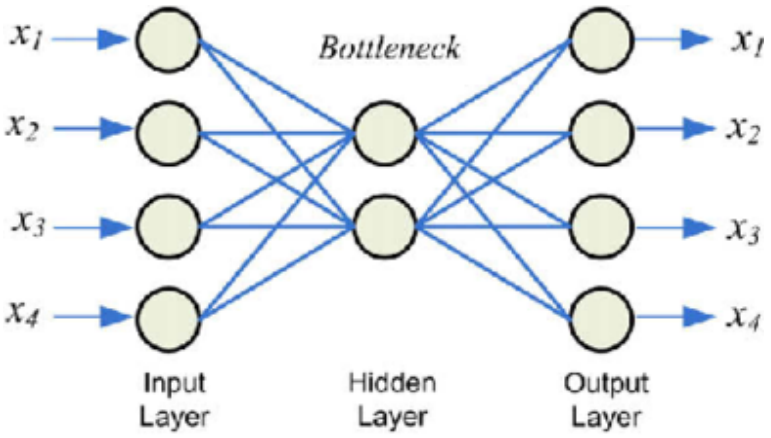


Figure 31: Autoencoder

A stacked autoencoder is a neural network consist several layers of sparse autoencoders where output of each hidden layer is connected to the input of the successive hidden layer.

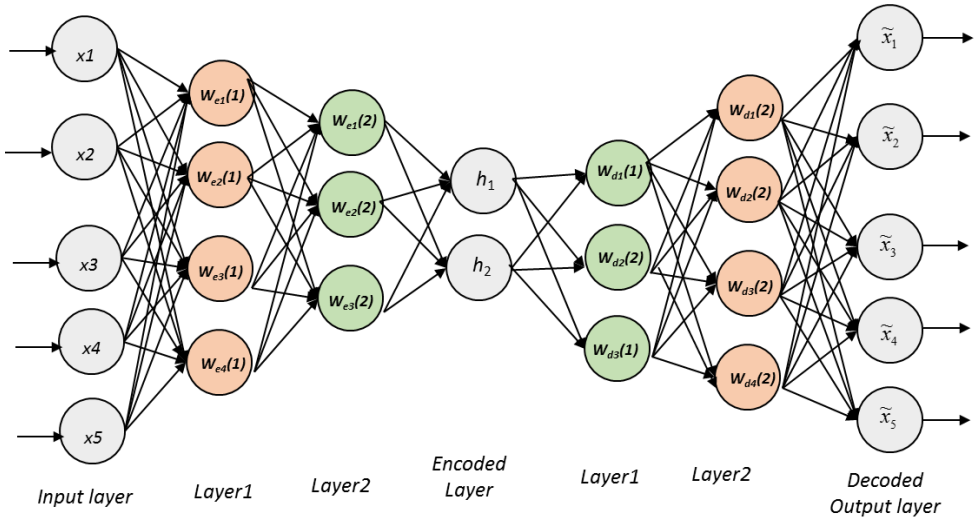


Figure 32: stacked autoencoder

As shown in Figure above the hidden layers are trained by an unsupervised algorithm and then fine-tuned by a supervised method. Stacked autoencoder mainly consists of three steps Train autoencoder using input data and acquire the learned data.
The learned data from the previous layer is used as an input for the next layer and this continues until the training is completed.
Once all the hidden layers are trained use the backpropagation algorithm to minimize the cost function and weights are updated with the training set to achieve fine tuning.
The recent advancements in Stacked Autoencoder is it provides a version of raw data with much detailed and promising feature information, which is used to train a classier with a specific context and find better accuracy than training with raw data.
Stacked autoencoder improving accuracy in deep learning with noisy autoencoders embedded in the layers
Stacked autoencoder are used for P300 Component Detection and Classification of 3D Spine Models in Adolescent Idiopathic Scoliosis in medical science. Classification of the rich and complex variability of spinal deformities is critical for comparisons between treatments and for long-term patient follow-ups.
As shown in Figure above the hidden layers are trained by an unsupervised algorithm and then fine-tuned by a supervised method. Stacked autoencoder mainly consists of three steps

- Train autoencoder using input data and acquire the learned data.
- The learned data from the previous layer is used as an input for the next layer and this continues until the training is completed.
- Once all the hidden layers are trained use the backpropagation algorithm to minimize the cost function and weights are updated with the training set to achieve fine tuning.

The recent advancements in Stacked Autoencoder is it provides a version of raw data with much detailed and promising feature information, which is used to train a classier with a specific context and find better accuracy than training with raw data.
Stacked autoencoder improving accuracy in deep learning with noisy autoencoders embedded in the layers
Stacked autoencoder are used for P300 Component Detection and Classification of 3D Spine Models in Adolescent Idiopathic Scoliosis in medical science. Classification of the rich and complex variability of spinal deformities is critical for comparisons between treatments and for long-term patient

5. Deep learning generative models

5.1. Variational Autoencoders

The basic idea behind a variational autoencoder is that instead of mapping an input to fixed vector, input is mapped to a distribution. The only difference between the autoencoder and variational autoencoder is that bottleneck vector is replaced with two different vectors one representing the mean of the distribution and the other representing the standard deviation of the distribution.
Loss function for variational autoencoder

$$li\ (\theta, \phi) = -Ez{\sim}q\theta(z|xi)\ [logp\phi(xi|z)] + KL\ (q\theta(z|xi)\ ||p(z))$$

The loss function in variational autoencoder consists of two terms. First, one represents the reconstruction loss and the second term is a regularizer and KL means Kullback-Leibler divergence between the encoder's distribution qθ (z|x) and p (z). This divergence measures how much information is lost when using q to represent p

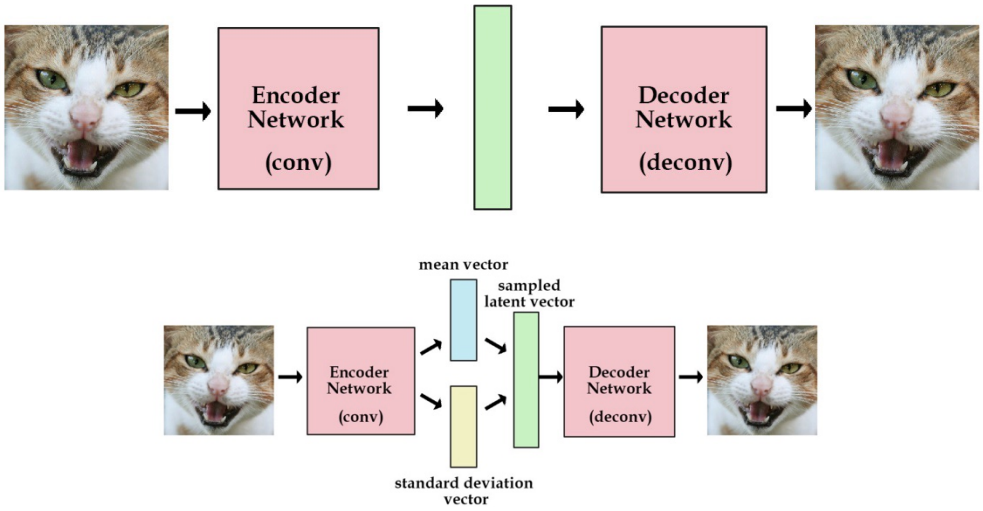


Figure 33: Example of variational autoencoder

Recent advancements in VAE as mentioned in which improves the quality of VAE samples by adding two more components. Firstly, a pre-trained classifier as extractor to input data which aligns the reproduced images. Secondly, a discriminator network for additional adversarial loss signals. Other significant improvement in VAE is Optimization of the Latent Dependency Structure by. In this VAE parameters, network parameters are optimized with a single objective. Interference is formed through sampling which produces expectations over latent variable structures and incorporates top-down and bottom-up reasoning over latent variable values.

Autoencoders Application

Today data denoising and dimensionality reduction for data visualization are the two major applications of autoencoders. With dimensionality and sparsity constraints, autoencoders can learn data projections which is better than PCA.

Previously Autoencoders are used for dimensionality reduction or feature learning. In recent developments with connection with the latent variable models have brought autoencoders to forefront of the generative modelling.
Autoencoders or its variants such as stacked, sparse or VAE are used for compact representation of data. For example, a 256x256 pixel image can be represented by 28x28 pixel. Google is using this type of network to reduce the amount band width you use it on your phone. If you download an image the full resolution of the image is downscaled and then sent to you via wireless internet and then in your phone a decoder that reconstructs the image to full resolution.
Autoencoders are used in Natural Language Processing, where NLP enclose some of the most difficult problems in computer science. With advancement in deep learning and indeed, autoencoders are been used to overcome some of these problems
Word Embedding: Words or phrases from a sentence or context of a word in a document are sorted in relation with other words.

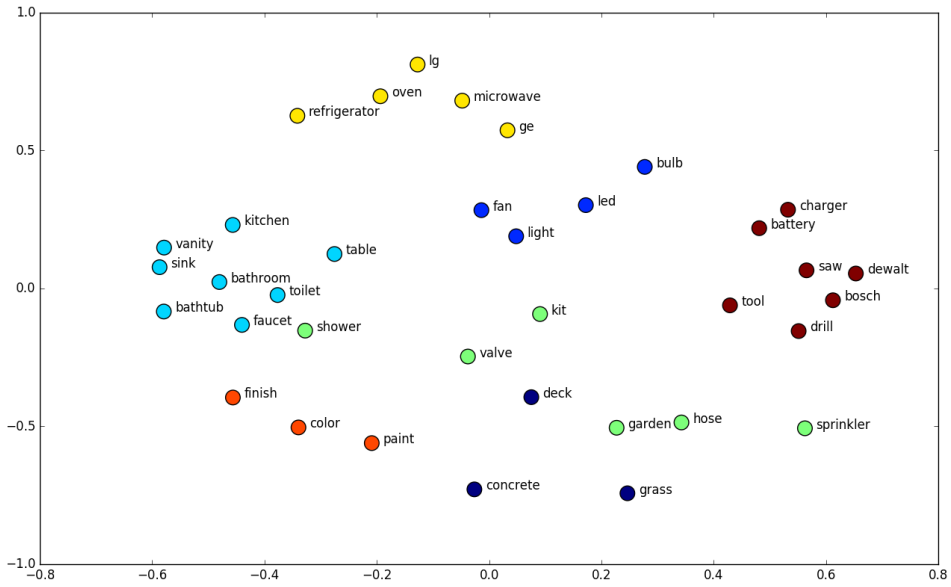


Figure 34: Autoencoders Application

Document Clustering: classification of documents such as blogs or news or any data into recommended categories. The challenge is to accurately cluster the documents into categories where there actually fit. Hinton used autoencoder to reduce the dimensionality vectors to represent the word probabilities in newswire stories. The Figure below shows the comparisons of Latent Semantic analysis and an autoencoder based on PCA and nonlinear dimensionality reduction algorithm proposed by Roweis where autoencoder outperformed LSA.

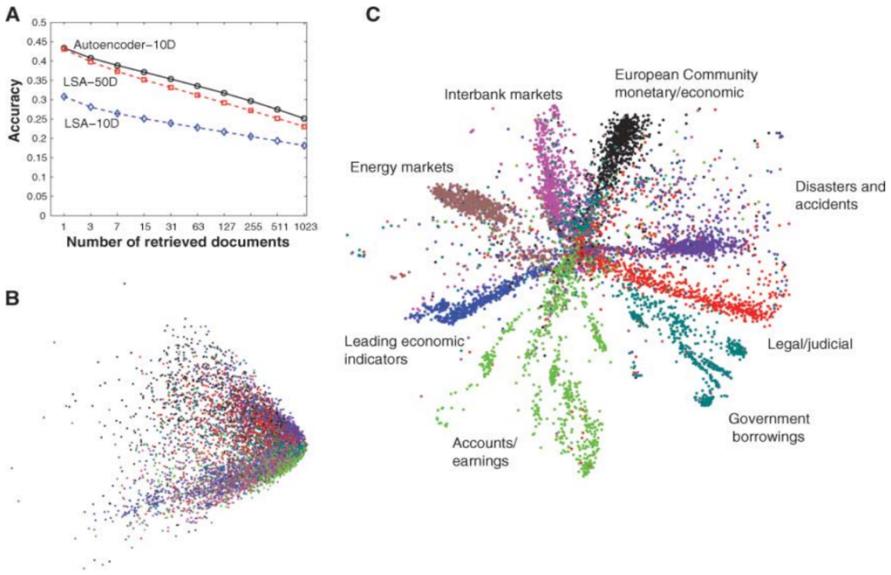


Figure 35: Autoencoders Application

Machine translation: it has been studied since late 1950s and an incredibly a difficult problem to translate text from one human language to another human language. With the use of autoencoders machine translation has taken a huge leap forward to accurately translate text from one language to another.

Autoencoders to extract speech: A deep generative model of spectrograms containing 256 frequency bins and 1,3,9 or 13 frames has been created by. This model has one visible layer and one hidden layer of 500 to 3000 binary latent variables.

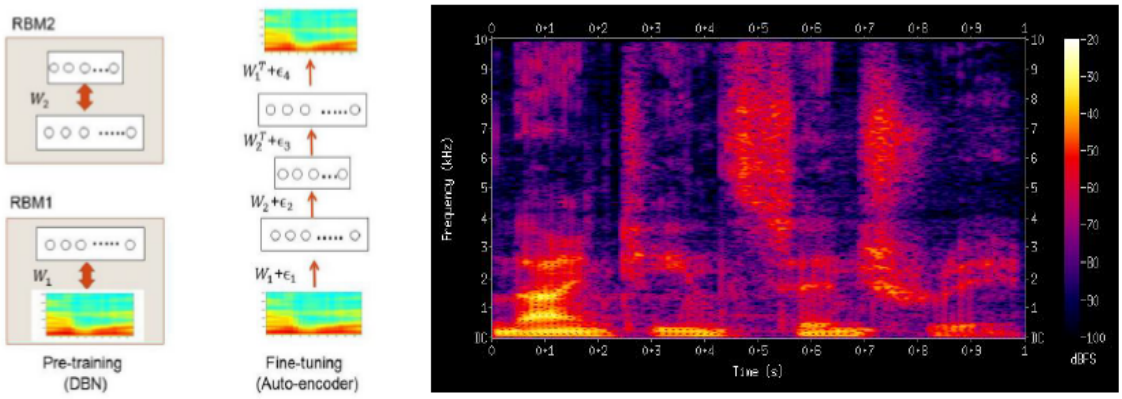


Figure 36: Machine translation

With Deep Denoising Autoencoders (DDAE) which has shown drastic improvement in performance has the capability to recognize the whispered speech which has been a problem for a long time in Automatic Speech Recognition (ASR). This has been implemented in various smart devices such as Amazon Alexa.
Reverberant speech recognition using deep learning in front end and back of a system. This model is built by Mimura, Sakai and Kawahara, 2015 where they adopted a deep autoencoder (DAE) for enhancing the speech at the front end and recognition of speech is performed by DNN-HMM acoustic models at the back end.
Denoising of speech using deep autoencoders: In actually conditions we experience speech signals are contaminated by noise and reverberation. If this speech is used by SR it may experience degradation in speech quality and in turn effect the performance.

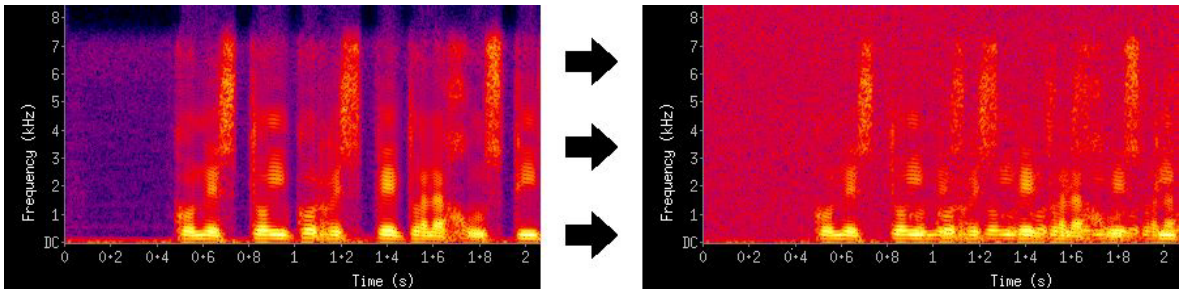


Figure 37: Deep Denoising Autoencoders (DDAE)

In order to improve the accuracy of the ASR system on noisy utterances, will be trained a collection of LSTM networks, which map features of a noisy utterance to a clean utterance. The figure below shows the model used by (Marvin Coto, John Goddard, Fabiola Martínez) 2016.

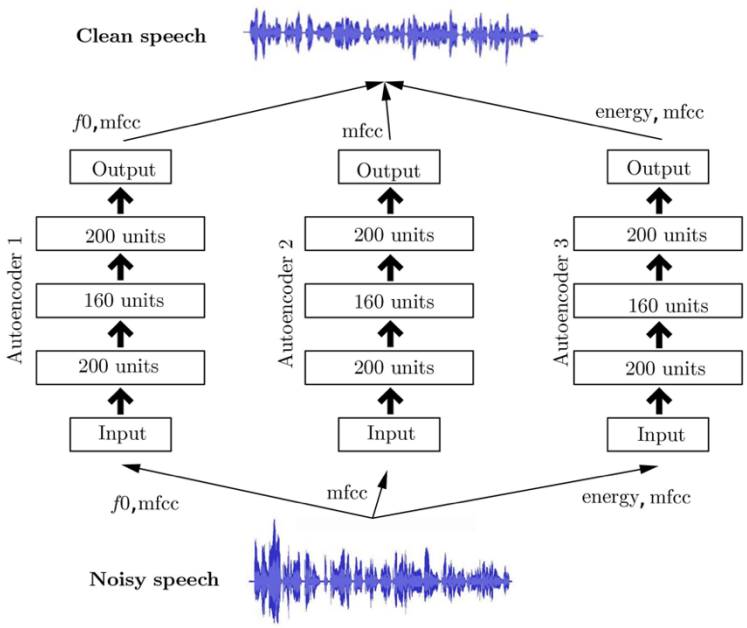


Figure 38: LSTM

Spatio-Temporal Autoencoder for Video Anomaly Detection:
Anomalous events detection in real-world video scenes is a challenging problem due to the complexity of "anomaly" as well as the cluttered backgrounds, objects and motions in the scenes. In (Zhao, Deng and Shen, 2018) they proposed model called Spatio-Temporal Autoencoder which utilizes deep neural networks to learn video representation automatically and extracts features from both spatial and temporal dimensions by performing 3-dimensional convolutions. They introduced a weight-decreasing prediction loss for generating future frames, which enhances the motion feature learning in videos. Since most anomaly detection datasets are restricted to appearance anomalies or unnatural motion anomalies. Figure below shows the architecture of the network. An encoder followed by two branches of decoder for reconstructing past frames and predicting the future frames.

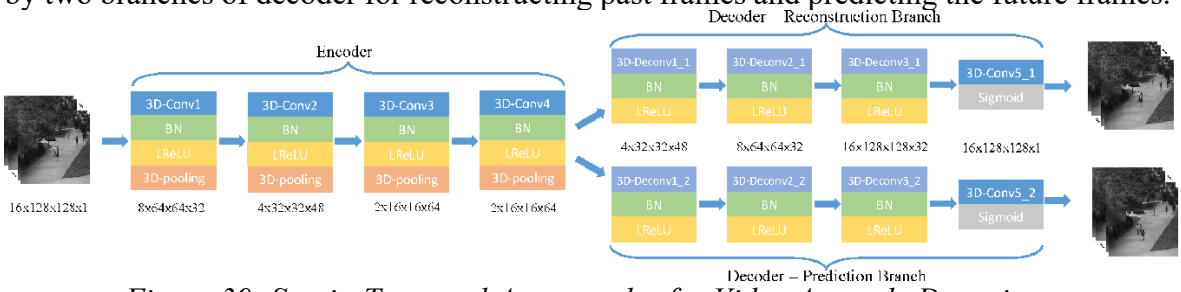


Figure 39: Spatio-Temporal Autoencoder for Video Anomaly Detection:

Reconstruction image using Convolutional Autoencoders: CAE are useful in reconstruction of image from missing parts. For this the model has to be trained with two different images as input and output. The input image can rather be a noisy version or an image with missing parts and with a clean output image. During training process, the model learns and fills the gaps in the input and output images. Here is an example below how CAE replace the missing part of the image.

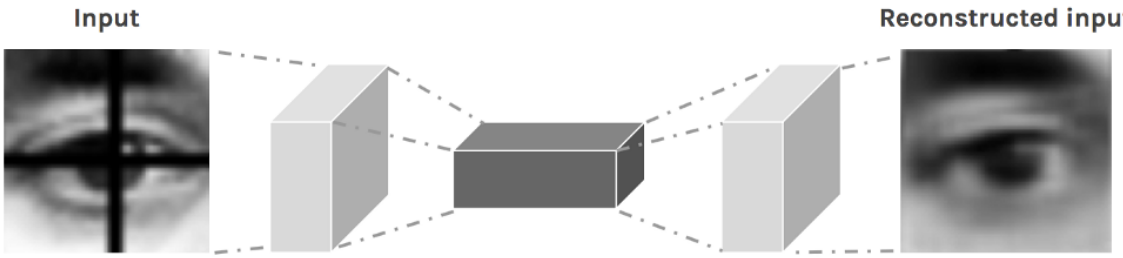


Figure 40: Reconstruction image

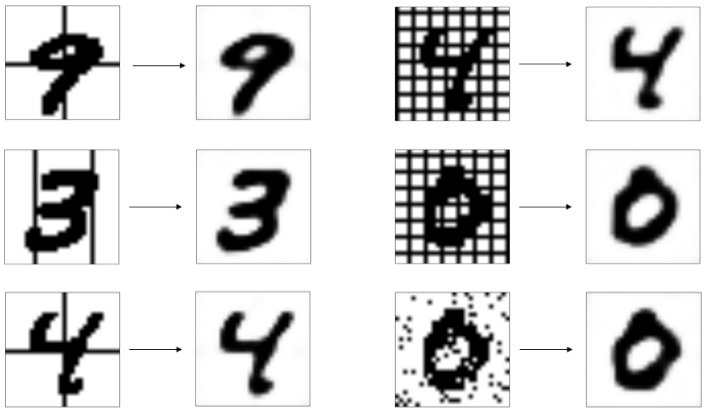


Figure 41: Reconstruction image

Paraphrase Detection: in many languages two phrases may look differently but when it comes to the meaning they both mean exactly same. Deep learning autoencoders allow us

to find such phrases accurately. Many other advanced applications include full image colorization, generating higher resolution images by using lower resolution as input.

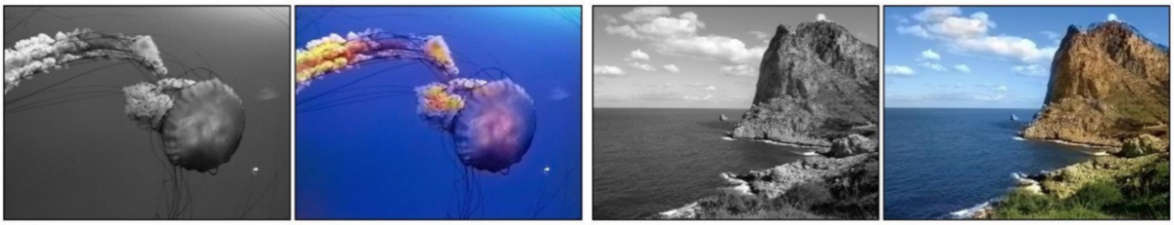


Figure 42: Paraphrase Detection:

5.2. Generative adversarial networks (GANs)

A GAN is a class of machine learning systems containing two deep neural networks, where they compete in a zero-sum game against one another. In this internal game, each network's gains and losses are ultimately balanced.

GANs are a type of generative models, which are an important subset of machine learning that require realistic and statistically accurate generation of target data.

Developed in 2014 by Ian Goodfellow and fellow researchers at the University of Montreal, and described by Yann LeCun, Facebook's VP and Chief AI Scientist, as "the coolest idea in machine learning in the last twenty years," GANs have incredible potential for innovative applications and use cases, particularly in the realms of art, music, and language. This is mainly due to a GAN's powerful ability to recreate a new, similar data set based on an original training data set.

In a GAN there are two running machine learning models, that compete to improve the functioning of their counterpart, the opposing model. Both models work by learning object rules and using the same two methods to characterize neuron sets, but for opposing goals.

GANs make use of generative algorithms — contrasted from discriminative algorithms which classify input data, or features, by predicting their labels, categories, or fields — to determine the distribution or probability of a data point being classified as a particular feature based on its label, category, or field. These two aforementioned algorithms are executed by the discriminator and generator, respectively, as illustrated in the image below:

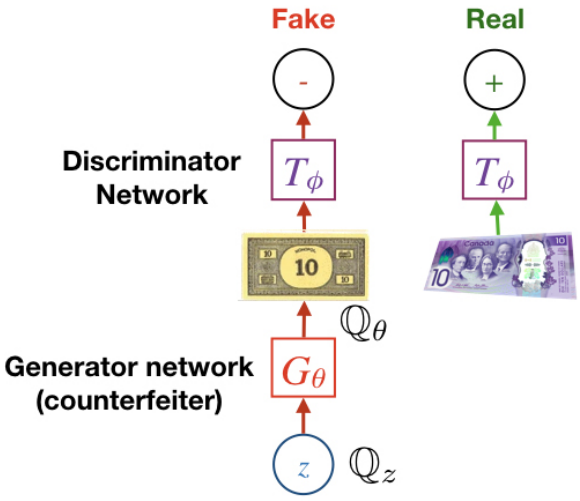


Figure 43: GANs

The GAN training algorithm involves training both the discriminator and the generator model in parallel. The algorithm is summarized in the figure below, taken from the original 2014 paper by Goodfellow, etc. titled Generative Adversarial Networks:

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

The generator creates false sample sets that are as close to 1 as possible, 1 being mathematically true. With the discriminator attempting to identify sets that are 1 and discarding those that are false sets, essentially not 1 or mathematically true. However, the generator, cannot simply create a false set out of nothing, there is training of the model required in which it learns from true sets to create its own true sets. Then optimizes these creations again the discriminator.

Diving a bit more deeply into this, the generator, the false set creator, learns how classes are distributed. While the discriminator network or machine learning model, learns how to identify true classes, by learning how to adjust its definition of a true class via adjusting how it classifies the data set it is given.

The generator creates new and similar data points that it gives to the discriminator in the hopes of it classifying the data point as authentic, despite its actually being fake. It is important to note that there is a double feedback loop taking place in the architecture of a GAN; the discriminator is in a feedback loop with the true training data set, and the generator is in a feedback loop with the discriminator, continuously sending it very similar, but unauthentic data: the discriminator must have a true training data set in order to confirm authenticity of other data points. The final result is a data set that has very similar, while not being exactly the same, characteristics to the training data set.

5.3. Boltzmann Machines

Boltzmann Machine is a generative unsupervised model, which involve learning a probability distribution from an original dataset and using it to make inferences about never before seen data.

Boltzmann Machine have an input layer (also referred to as the visible layer) and one or several hidden layers (also referred to as the hidden layer).

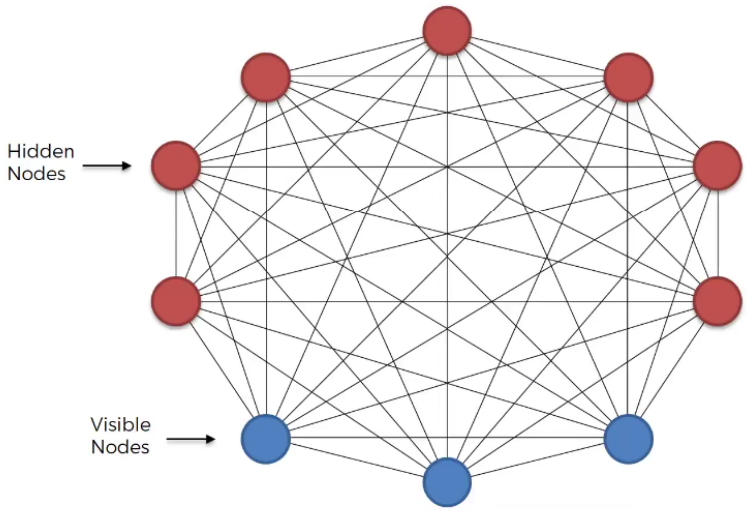


Figure 44: Boltzmann Machine

Boltzmann Machine use neural networks with neurons that are connected not only to other neurons in other layers but also to neurons within the same layer. Everything is connected to everything. Connections are bidirectional, visible neurons connected to each other and hidden neurons also connected to each other. Boltzmann Machine doesn't expect input data, it generates data. Neurons generate information regardless they are hidden or visible.
For Boltzmann Machine all neurons are same, it doesn't discriminate between hidden and visible neurons. For Boltzmann Machine whole things is system and its generating state of the system. Suppose for example we have a nuclear power station and there is certain thing we can measure in nuclear power plant like temperature of containment building, how quickly turbine is spinning, pressure inside the pump etc. There are lots of things we are not measuring like speed of wind, the moisture of the soil in this specific location, its sunny day or rainy day etc. All these parameters together form a system, they all work together. All these parameters are binary. So, we get a whole bunch of binary numbers that tell us something about the state of the power station. What we would like to do, is we want to notice that when it is going to in an unusual state. A state that is not like a normal state which we had seen before. And we don't want to use supervised learning for that. Because we don't want to have any examples of states that cause it to blowup. We would rather be able to detect that when it is going into such a state without even having seen such a state before. And we could do that by building a model of a normal state and noticing that this state is different from the normal states. That what Boltzmann Machine represent. The way this system work, we use our training data and feed into the Boltzmann Machine as input to help system adjust its weights. It resembles our system not any nuclear power station in the world. It learns from input, what are the possible connections between all these parameters, how do they influence each other and therefore it becomes a machine that represent our system. We can use this Boltzmann Machine to monitor our system
Boltzmann Machine learn how system work in its normal states through good example. Boltzmann Machine consist of a neural network with an input layer and one or several hidden layers. The neurons in the neural network make stochastic decisions about whether to turn on or off based on the data we feed during training and the cost function the Boltzmann Machine is trying to minimize.

By doing so, the Boltzmann Machine discovers interesting features about the data, which help model the complex underlying relationships and patterns present in the data.These Boltzmann Machine use neural networks with neurons that are connected not only to other neurons in other layers but also to neurons within the same layer. That makes training an unrestricted Boltzmann machine very inefficient and Boltzmann Machine had very little commercial success.

II. How deep learning used in Cyber security?

1. Deep learning methods in cybersecurity

1.1. Deep Belief Networks

1.1.1. Deep Autoencoders Algorithms in cybersecurity

Auto-encoders are neural networks comprising two symmetrical components, an encoder and a decoder, as shown in Figure 3. The encoder extracts features a.k.a latent representations from raw data, and the decoder reconstructs the input data from these latent representations. During training, the divergence between the input of the encoder and the output of the decoder is gradually reduced and hence, the latent representations coming out of encoder gradually tend to represent the essence in the original data while throwing away all the higher dimensional details. It is important to note that this entire process requires no supervised (class labels or regression outcome) information. Many famous auto-encoder variants exist, such as de-noising auto-encoders, variational auto-encoders and sparse auto-encoders.

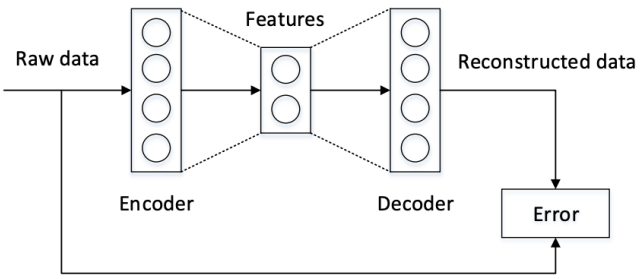


Figure 45: Auto-encoders

Auto-encoders are mainly used for feature space reduction and the latent representations are used further downstream in the workflow to train different types of models. Auto-encoders also do a really good job in capturing the complex multivariate distribution of the input feature space. Because of this characteristic, they are widely used in anomaly detection tasks.
Since this is a time-series problem, we use LSTM (long short-term memory) networks in our auto-encoder. They are a variant of RNNs (recurrent neural networks) aimed at retaining time dependent characteristics over long sequences. These networks require each sample in the shape (time steps, features) where time steps are a tunable dimension. Hence, input data is a 3-D array of shape (number of samples, time steps, features).

1.1.2. Restricted Boltzmann Machines Algorithms in cybersecurity

Restricted Boltzmann Machine is an undirected graphical model that plays a major role in Deep Learning Framework in recent times. It was initially introduced as Harmonium by Paul Smolensky in 1986 and it gained big popularity in recent years in the context of the Netflix Prize where Restricted Boltzmann Machines achieved state of the art performance in collaborative filtering and have beaten most of the competition. It is an algorithm which is useful for dimensionality reduction, classification, regression, collaborative filtering, feature learning, and topic modeling. Now let's see how Restricted Boltzmann machine differs from other Autoencoder.

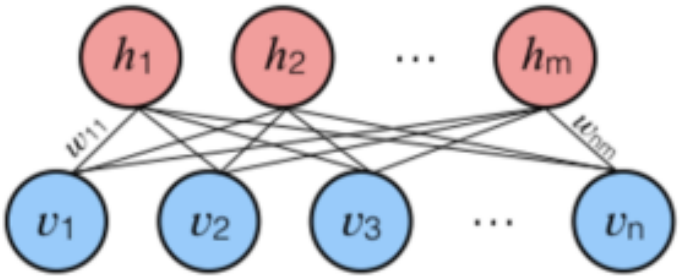


Figure 46: Restricted Boltzmann machine

Restricted Boltzmann Machines are shallow, two-layer neural nets that constitute the building blocks of deep-belief networks. The first layer of the RBM is called the visible, or input layer, and the second is the hidden layer. Each circle represents a neuron-like unit called a node. The nodes are connected to each other across layers, but no two nodes of the same layer are linked

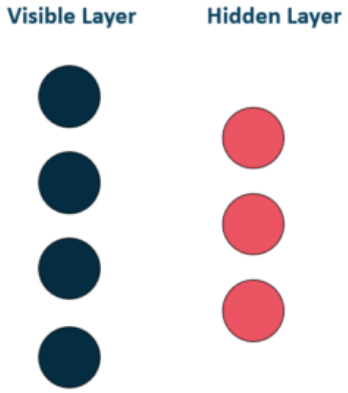


Figure 47: Node

The restriction in a Restricted Boltzmann Machine is that there is no intra-layer communication. Each node is a locus of computation that processes input and begins by making stochastic decisions about whether to transmit that input or not. Working of Restricted Boltzmann Machine

Each visible node takes a low-level feature from an item in the dataset to be learned. At node 1 of the hidden layer, x is multiplied by a weight and added to a bias. The result of those two operations is fed into an activation function, which produces the node's output, or the strength of the signal passing through it, given input x.

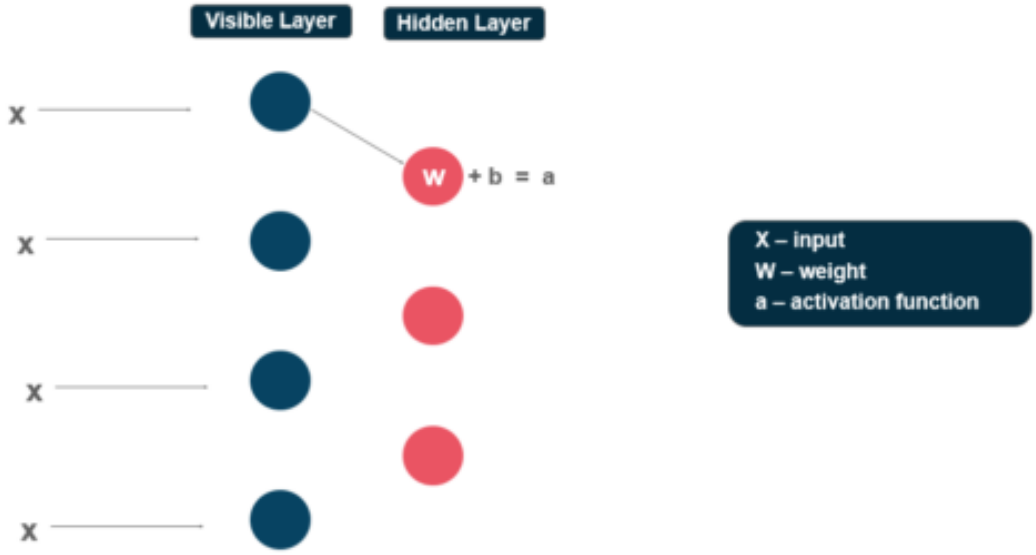


Figure 48: Node

Next, let's look at how several inputs would combine at one hidden node. Each x is multiplied by a separate weight, the products are summed, added to a bias, and again the result is passed through an activation function to produce the node's output

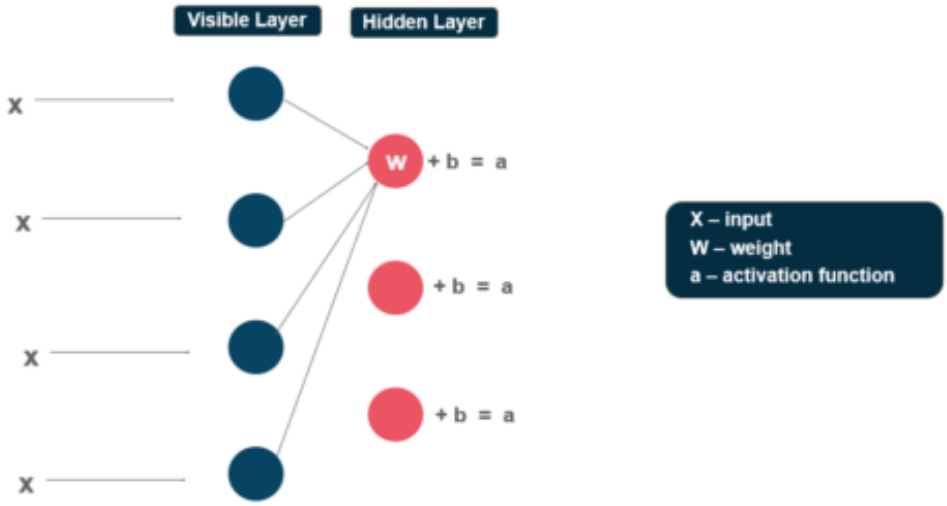


Figure 49: Node

At each hidden node, each input x is multiplied by its respective weight w. That is, a single input x would have three weights here, making 12 weights altogether (4 input nodes x 3 hidden nodes). The weights between the two layers will always form a matrix where the rows are equal to the input nodes, and the columns are equal to the output nodes.

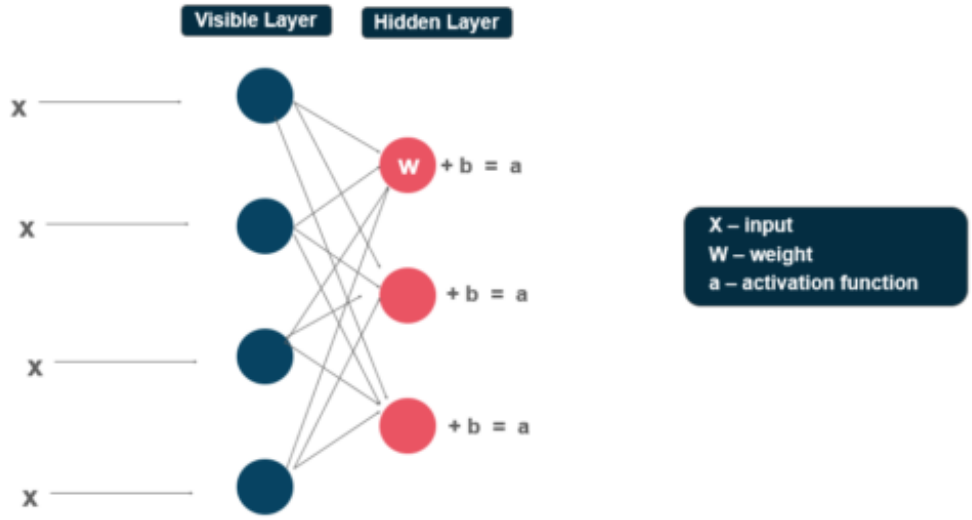


Figure 50: Node

Each hidden node receives the four inputs multiplied by their respective weights. The sum of those products is again added to a bias (which forces at least some activations to happen), and the result is passed through the activation algorithm producing one output for each hidden node.

Now that you have an idea about how Restricted Boltzmann Machine works, let's continue our Restricted Boltzmann Machine Tutorial and have a look at the steps involved in the training of RBM.

Training of Restricted Boltzmann Machine

The training of the Restricted Boltzmann Machine differs from the training of regular neural networks via stochastic gradient descent.

The Two main Training steps are:

Gibbs Sampling

The first part of the training is called Gibbs Sampling. Given an input vector v we use p(h|v) for prediction of the hidden values h. Knowing the hidden values we use p(v|h):

$$p(v_i = 1 \mid h) = \frac{1}{1+e^{(-(a_i + w_i h_j))}} = \tilde{O}\,(a_i + \sum h_i w_{ij})$$

for prediction of new input values v. This process is repeated k times. After k iterations, we obtain another input vector v_k which was recreated from original input values v_0.

$$p(h_i = 1 \mid v) = \frac{1}{1+e^{(-(b_i + W_j v_i))}} = \sigma\,(b_j + \sum_i v_i w_{ij})$$

Contrastive Divergence step

The update of the weight matrix happens during the Contrastive Divergence step. Vectors v_0 and v_k is used to calculate the activation probabilities for hidden values h_0 and h_k:

$$p(v_i = 1 \mid h) = \frac{1}{1+e^{(-(a_i + w_i h_j))}} = \tilde{O}\,(a_i + \sum h_i w_{ij})$$

The difference between the outer products of those probabilities with input vectors v_0 and v_k results in the updated matrix:

$$\Delta W = v_0) \otimes P(h_0 \mid v_0) - v_k \otimes P(h_k \mid v_k) - v_k)$$

Using the update matrix, the new weights can be calculated with gradient ascent, given by:

$$W_{new} = W_{old} + \triangle W$$

Now that you have an idea of what are Restricted Boltzmann Machines and the layers of RBM, let's move on with our Restricted Boltzmann Machine Tutorial and understand their working with the help of an example.

Collaborative Filtering

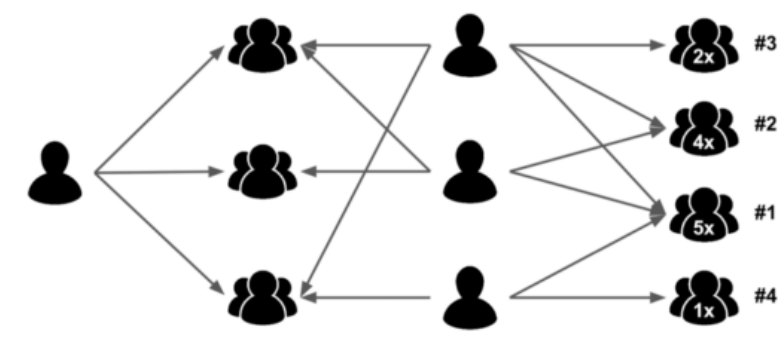


Figure 51: Collaborative Filtering

RBMs have found applications in dimensionality reduction, classification, collaborative filtering and many more. They can be trained in either supervised or unsupervised ways, depending on the task.

Recognizing Latent factors in the Data

Let us assume that some people were asked to rate a set of movies in the scale of 1–5 and each movie could be explained in terms of a set of latent factors such as drama, fantasy, action and many more. Restricted Boltzmann Machines are used to analyze and find out these underlying factors.

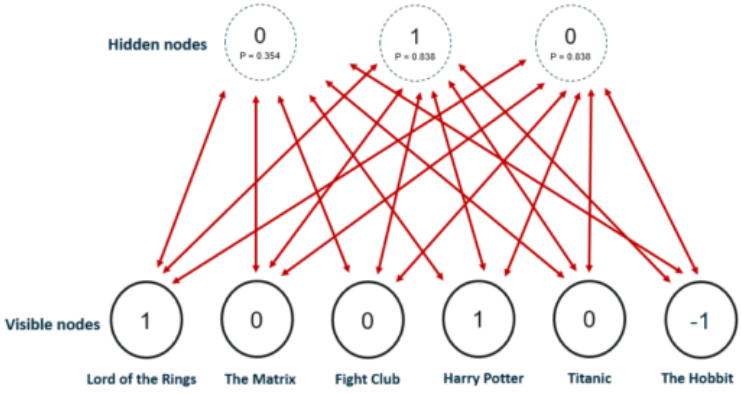


Figure 52: RBMs

The analysis of hidden factors is performed in a binary way, i.e, the user only tells if they liked (rating 1) a specific movie or not (rating 0) and it represents the inputs for the input/visible layer. Given the inputs, the RMB then tries to discover latent factors in the data that can explain the movie choices and each hidden neuron represents one of the latent factors.

Let us consider the following example where a user likes Lord of the Rings and Harry Potter but does not like The Matrix, Fight Club and Titanic. The Hobbit has not been seen yet so it gets a -1 rating. Given these inputs, the Boltzmann Machine may identify three hidden factors Drama, Fantasy and Science Fiction which correspond to the movie genres.

Using Latent Factors for Prediction

After the training phase, the goal is to predict a binary rating for the movies that had not been seen yet. Given the training data of a specific user, the network is able to identify the latent factors based on the user's preference and sample from Bernoulli distribution can be used to find out which of the visible neurons now become active.

Figure 53: Recognizing Latent factors in the Data

The image shows the new ratings after using the hidden neuron values for the inference. The network identified Fantasy as the preferred movie genre and rated The Hobbit as a movie the user would like.

The process from training to the prediction phase goes as follows:

- Train the network on the data of all users
- During inference-time, take the training data of a specific user
- Use this data to obtain the activations of hidden neurons
- Use the hidden neuron values to get the activations of input neurons
- The new values of input neurons show the rating the user would give yet unseen movies

1.1.3. DBNs or RBMs or Deep Autoencoders Coupled with Classification Layers Algorithms in cybersecurity

Both RBMs and autoencoders can be combined with a classification layer to perform a classification using a fully connected layer or layers. The layers trained by applying unsupervised learning are used as feature extractors, and they constitute inputs into the fully connected layers that are trained using back propagation. Unlike the RBM or autoencoder layers, these layers require labels to train. These types of networks have shown success in a number of applications, including acoustic modeling, speech recognition, and image recognition.

1.2. Recurrent Neural Networks Algorithms in cybersecurity

The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If you want to predict the next word in a sentence you better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations and you already know that they have a "memory" which captures information about what has been calculated so far.

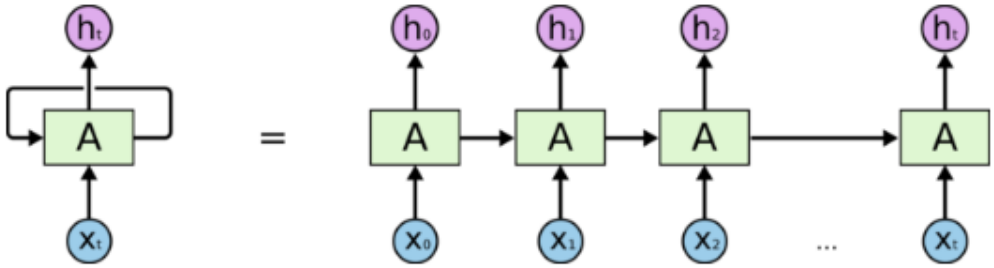


Figure 54: RNN

"Whenever there is a sequence of data and that temporal dynamics that connects the data is more important than the spatial content of each individual frame."
– Lex Fridman (MIT)

Different types of RNN's

The core reason that recurrent nets are more exciting is that they allow us to operate over sequences of vectors: Sequences in the input, the output, or in the most general case both. A few examples may make this more concrete:

Figure 15 for RNN

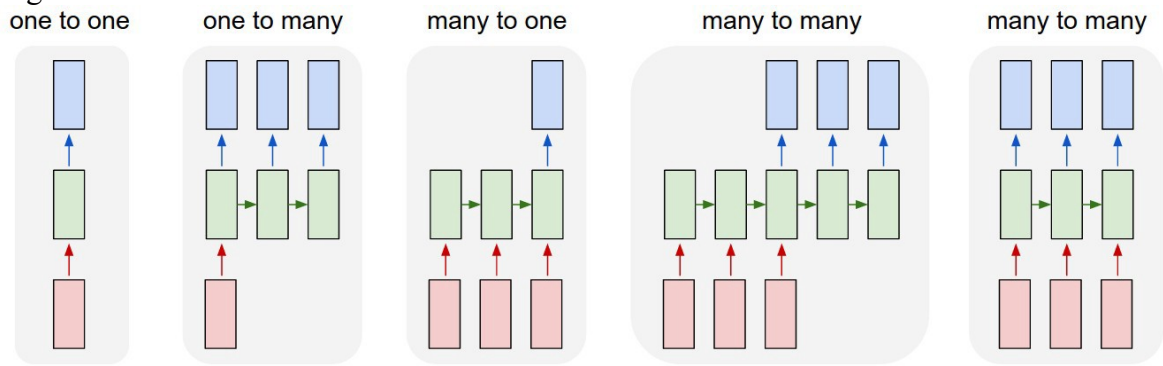


Figure 55: Different types of RNN's

Different types of Recurrent Neural Networks. (2) Sequence output (e.g. image captioning takes an image and outputs a sentence of words). (3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). (4) Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). (5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like.

Each rectangle in above image represent Vectors and Arrows represent functions. Input vectors are Red in color, output vectors are blue and green holds RNN's state.

One-to-one: This also called as Plain/Vaniall Neural networks. It deals with Fixed size of input to Fixed size of Output where they are independent of previous information/output.
Ex: Image classification.
One-to-Many: it deals with fixed size of information as input that gives sequence of data as output.
Example: Image Captioning takes image as input and outputs a sentence of words.

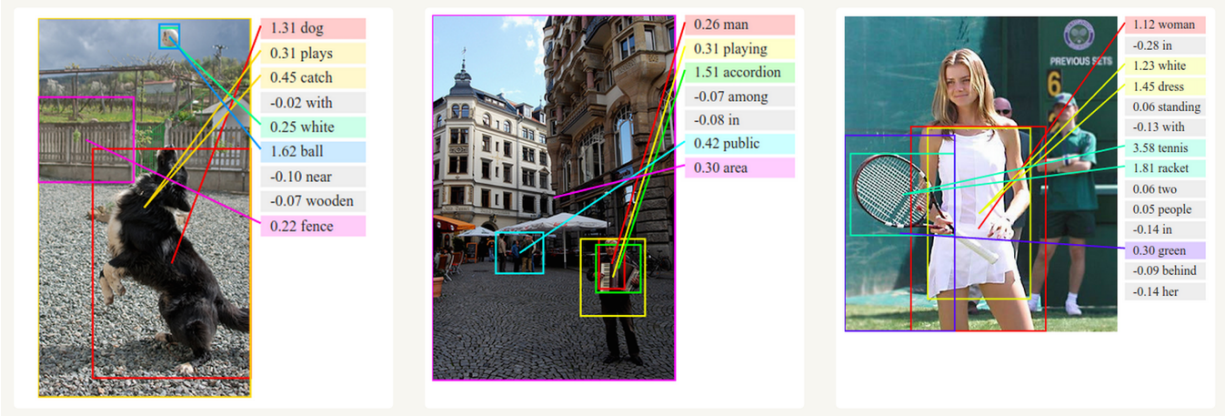


Figure 56: RNN

Many-to-One: It takes Sequence of information as input and outputs a fixed size of output.
Ex: sentiment analysis where a given sentence is classified as expressing positive or negative sentiment. Many-to-Many: It takes a Sequence of information as input and process it recurrently outputs a Sequence of data. Example: Machine Translation, where an RNN reads a sentence in English and then outputs a sentence in French.
Bidirectional Many-to-Many: Synced sequence input and output. Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like. Example video classification where we wish to label each frame of the video.
Advantages of Recurrent Neural Network
The main advantage of RNN over ANN is that RNN can model sequence of data (i.e. time series) so that each sample can be assumed to be dependent on previous ones
Recurrent neural network is even used with convolutional layers to extend the effective pixel neighborhood.
Disadvantages of Recurrent Neural Network
Gradient vanishing and exploding problems.
Training an RNN is a very difficult task.

It cannot process very long sequences if using tanh or relu as an activation function

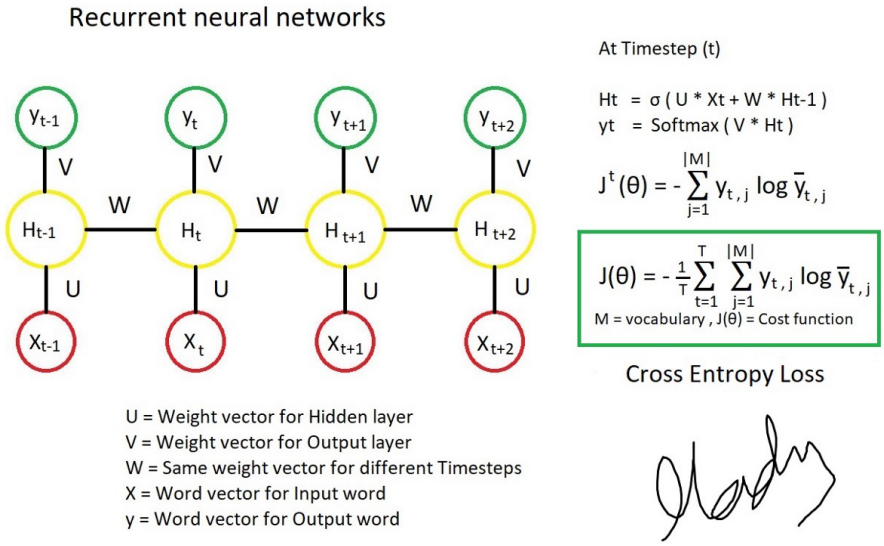


Figure 57: RNN algorithm

1.3. Convolutional Neural Networks Algorithms in cybersecurity

Convolutional Neural Networks are very similar to ordinary Neural Networks from the previous chapter: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply.

Architecture Overview

Recall: Regular Neural Nets. As we saw in the previous chapter, Neural Networks receive an input (a single vector), and transform it through a series of hidden layers. Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer, and where neurons in a single layer function completely independently and do not share any connections. The last fully-connected layer is called the “output layer” and in classification settings it represents the class scores.

Regular Neural Nets don’t scale well to full images. In CIFAR-10, images are only of size 32x32x3 (32 wide, 32 high, 3 color channels), so a single fully-connected neuron in a first hidden layer of a regular Neural Network would have 32*32*3 = 3072 weights. This amount still seems manageable, but clearly this fully-connected structure does not scale to larger images. For example, an image of more respectable size, e.g. 200x200x3, would lead to neurons that have 200*200*3 = 120,000 weights. Moreover, we would almost certainly want to have several such neurons, so the parameters would add up quickly! Clearly, this full connectivity is wasteful and the huge number of parameters would quickly lead to overfitting.

3D volumes of neurons. Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. (Note that the word depth here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) For example, the input images in CIFAR-10 are an input volume of activations, and the volume has dimensions 32x32x3 (width, height, depth respectively). As we will soon see, the neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would for CIFAR-10 have dimensions 1x1x10, because by the end of the ConvNet architecture we will reduce the full image into a single vector of class scores, arranged along the depth dimension.

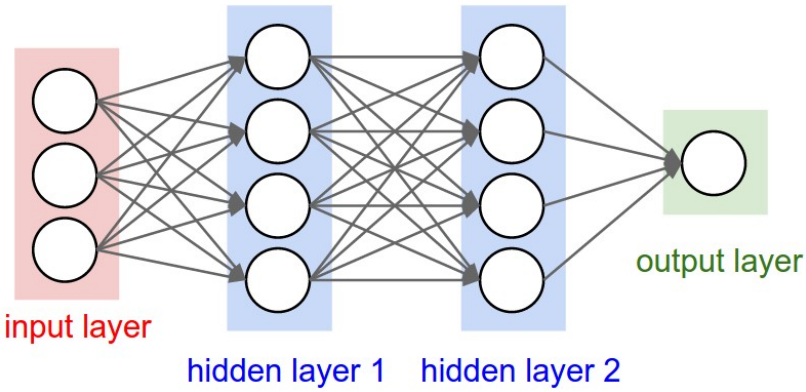


Figure 58: CNNs

Input: Acc_x, Acc_y, Acc_z, Gry_x, Gry_y, Gry_z, Mag_x, Mag_y, Mag_z, Pre

```
Sliding Window Process
sf ← Extract Shadow Features
Normalize sf by equation (2)
regularization feature data, size = 13×13
repeat:
  Forward Propagation:
  cd ← Convolution2D(sf);
  mp ← Max_pooling(cd);
  fc ← Fully_connected(mp);
  class label ← Soft_max(fc);
  Backward Propagation:
    conduct backward propagation with Adam;
Until w_i convergences;
  Use the trained network to predict the labels
```

Algorithm3: CNN

Deep learning for clustering and self-organized Maps (SOM)

A self-organizing map (SOM) or self-organizing feature map (SOFM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-

dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map, and is therefore a method to do dimensionality reduction. Self-organizing maps differ from other artificial neural networks as they apply competitive learning as opposed to error-correction learning (such as backpropagation with gradient descent), and in the sense that they use a neighborhood function to preserve the topological properties of the input space.

Algorithm

- Randomize the node weight vectors in a map
- Randomly pick an input vector
- Traverse each node in the map
- Use the Euclidean distance formula to find the similarity between the input vector and the map's node's weight vector
- Track the node that produces the smallest distance (this node is the best matching unit, BMU)
- Update the weight vectors of the nodes in the neighborhood of the BMU (including the BMU itself) by pulling them closer to the input vector

1. $W_v(s+1) = W_v(s) + \theta(u, v, s) \cdot \alpha(s) \cdot (D(t) - W_v(s))$

- Increase and repeat from step 2 while

A variant algorithm:

- Randomize the map's nodes' weight vectors
- Traverse each input vector in the input data set
- Traverse each node in the map
- Use the Euclidean distance formula to find the similarity between the input vector and the map's node's weight vector
- Track the node that produces the smallest distance (this node is the best matching unit, BMU)
- Update the nodes in the neighborhood of the BMU (including the BMU itself) by pulling them closer to the input vector

$$1.\ W_v(s+1) = W_v(s) + \theta(u, v, s) \cdot \alpha(s) \cdot (D(t) - W_v(s))$$

- Increase and repeat from step 2

Kohonen Networks

The objective of a Kohonen network is to map input vectors (patterns) of arbitrary dimension N onto a discrete map with 1 or 2 dimensions. Patterns close to one another in the input space should be close to one another in the map: they should be topologically ordered. A Kohonen network is composed of a grid of output units and N input units. The input pattern is fed to each output unit. The input lines to each output unit are weighted. These weights are initialised to small random numbers.

Learning in Kohonen Networks

The learning process is as roughly as follows: initialize the weights for each output unit loop until weight changes are negligible for each input pattern, present the input pattern, find the winning output unit, find all units in the neighborhoods of the winner, update the weight vectors for all those units, reduce the size of neighbourboods if required

The winning output unit is simply the unit with the weight vector that has the smallest Euclidean distance to the input pattern. The neighbourhood of a unit is defined as all units within some distance of that unit on the map (not in weight space). In the demonstration

below all the neighbourboods are square. If the size of the neighbourhood is 1 then all units no more than 1 either horizontally or vertically from any unit fall within its neighbourhood. The weights of every unit in the neighbourhood of the winning unit (including the winning unit itself) are updated using

$$\vec{w}_i = \vec{w}_i + \alpha(\vec{x}_i - \vec{w}_i) \tag{21}$$

This will move each unit in the neighbourhood closer to the input pattern. As time progresses the learning rate and the neighbourhood size are reduced. If the parameters are well chosen the final network should capture the natural clusters in the input data.

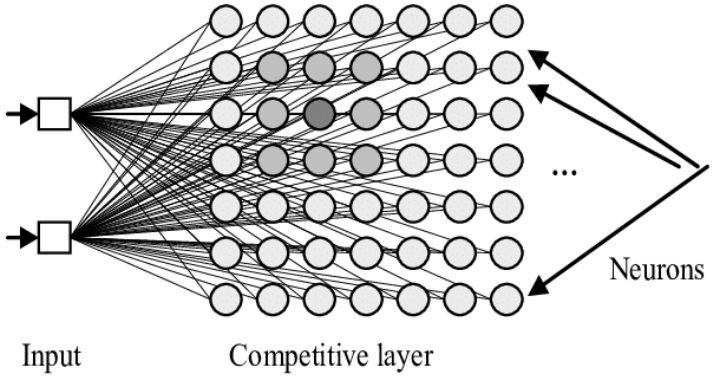


Figure 59: Learning in Kohonen Networks

1.4. Generative Adversarial Networks Algorithms in cybersecurity

A GAN is a class of machine learning systems containing two deep neural networks, where they compete in a zero-sum game against one another. In this internal game, each network's gains and losses are ultimately balanced.

GANs are a type of generative models, which are an important subset of machine learning that require realistic and statistically accurate generation of target data.

Developed in 2014 by Ian Goodfellow and fellow researchers at the University of Montreal, and described by Yann LeCun, Facebook's VP and Chief AI Scientist, as "the coolest idea in machine learning in the last twenty years," GANs have incredible potential for innovative applications and use cases, particularly in the realms of art, music, and language. This is mainly due to a GAN's powerful ability to recreate a new, similar data set based on an original training data set.

In a GAN there are two running machine learning models, that compete to improve the functioning of their counterpart, the opposing model. Both models work by learning object rules and using the same two methods to characterize neuron sets, but for opposing goals.

GANs make use of generative algorithms — contrasted from discriminative algorithms which classify input data, or features, by predicting their labels, categories, or fields — to determine the distribution or probability of a data point being classified as a particular feature based on its label, category, or field. These two aforementioned algorithms are executed by the discriminator and generator. The GAN training algorithm involves training both the discriminator and the generator model in parallel. The algorithm is

summarized in the figure below, taken from the original 2014 paper by Goodfellow, et al. titled Generative Adversarial Networks:

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

The generator creates false sample sets that are as close to 1 as possible, 1 being mathematically true. With the discriminator attempting to identify sets that are 1 and discarding those that are false sets, essentially not 1 or mathematically true. However, the generator, cannot simply create a false set out of nothing, there is training of the model required in which it learns from true sets to create its own true sets. Then optimizes these creations again the discriminator.

Diving a bit more deeply into this, the generator, the false set creator, learns how classes are distributed. While the discriminator network or machine learning model, learns how to identify true classes, by learning how to adjust its definition of a true class via adjusting how it classifies the data set it is given.

The generator creates new and similar data points that it gives to the discriminator in the hopes of it classifying the data point as authentic, despite its actually being fake. It is important to note that there is a double feedback loop taking place in the architecture of a GAN; the discriminator is in a feedback loop with the true training data set, and the generator is in a feedback loop with the discriminator, continuously sending it very similar, but unauthentic data: the discriminator must have a true training data set in order to confirm authenticity of other data points. The final result is a data set that has very similar, while not being exactly the same, characteristics to the training data set.

Recursive Neural Networks Algorithms in cybersecurity

Restricted Boltzmann Machine is an undirected graphical model that plays a major role in Deep Learning Framework in recent times. It was initially introduced as Harmonium by Paul Smolensky in 1986 and it gained big popularity in recent years in the context of the Netflix Prize where Restricted Boltzmann Machines achieved state of the art performance in collaborative filtering and have beaten most of the competition.

It is an algorithm which is useful for dimensionality reduction, classification, regression, collaborative filtering, feature learning, and topic modeling.

Now let's see how Restricted Boltzmann machine differs from other Autoencoder.

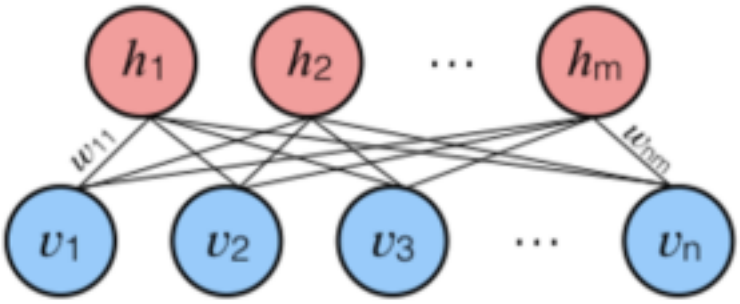


Figure 60: Restricted Boltzmann machine

Restricted Boltzmann Machines are shallow, two-layer neural nets that constitute the building blocks of deep-belief networks. The first layer of the RBM is called the visible, or input layer, and the second is the hidden layer. Each circle represents a neuron-like unit called a node. The nodes are connected to each other across layers, but no two nodes of the same layer are linked

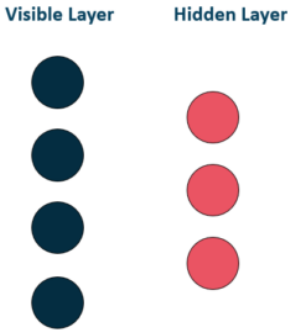


Figure 61: Restricted Boltzmann Machines

The restriction in a Restricted Boltzmann Machine is that there is no intra-layer communication. Each node is a locus of computation that processes input and begins by making stochastic decisions about whether to transmit that input or not.

Working of Restricted Boltzmann Machine

Each visible node takes a low-level feature from an item in the dataset to be learned. At node 1 of the hidden layer, x is multiplied by a weight and added to a bias. The result of those two operations is fed into an activation function, which produces the node's output, or the strength of the signal passing through it, given input x.

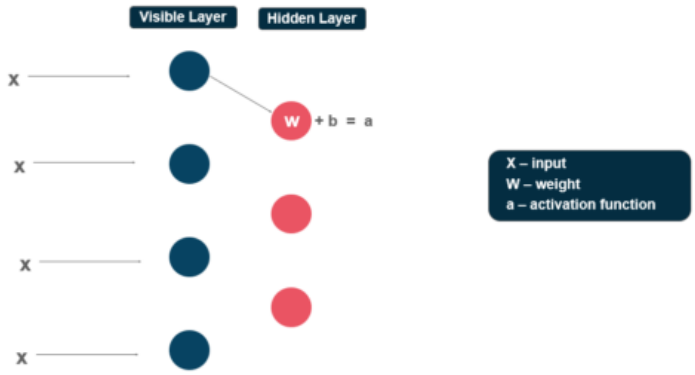


Figure 62 Restricted Boltzmann Machine

Next, let's look at how several inputs would combine at one hidden node. Each x is multiplied by a separate weight, the products are summed, added to a bias, and again the result is passed through an activation function to produce the node's output

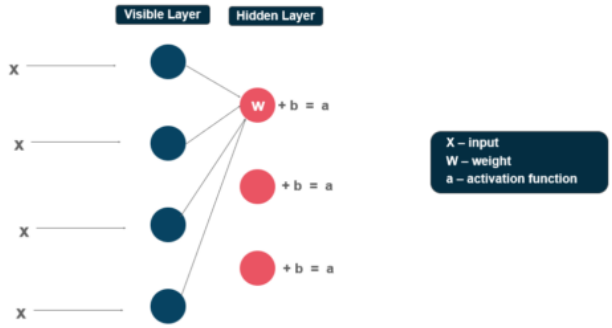


Figure 63: Restricted Boltzmann Machine

At each hidden node, each input x is multiplied by its respective weight w. That is, a single input x would have three weights here, making 12 weights altogether (4 input nodes x 3 hidden nodes). The weights between the two layers will always form a matrix where the rows are equal to the input nodes, and the columns are equal to the output nodes.

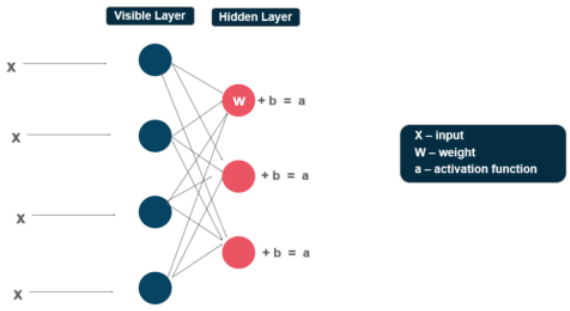


Figure 64: Restricted Boltzmann Machine

Each hidden node receives the four inputs multiplied by their respective weights. The sum of those products is again added to a bias (which forces at least some activations to happen), and the result is passed through the activation algorithm producing one output for each hidden node.

Now that you have an idea about how Restricted Boltzmann Machine works, let's continue our Restricted Boltzmann Machine Tutorial and have a look at the steps involved in the training of RBM.

Training of Restricted Boltzmann Machine: The training of the Restricted Boltzmann Machine differs from the training of regular neural networks via stochastic gradient descent. The Two main Training steps are:

Gibbs Sampling

The first part of the training is called Gibbs Sampling. Given an input vector v we use p(h|v) for prediction of the hidden values h. Knowing the hidden values we use p(v|h):

$$p(v_i = 1 \mid h) = \frac{1}{1+e(-(a_i+w_i h_j))} = \tilde{O}\,(a_i + \sum h_j w_{ij})$$

for prediction of new input values v. This process is repeated k times. After k iterations, we obtain another input vector v_k which was recreated from original input values v_0.

$$p(h_i = 1 \mid v) = \frac{1}{1+e(-(b_j+W_j v_i))} = \sigma\,(b_j + \sum_i v_i w_{ij})$$

Contrastive Divergence step

The update of the weight matrix happens during the Contrastive Divergence step. Vectors v_0 and v_k is used to calculate the activation probabilities for hidden values h_0 and h_k:

$$p(v_i = 1 \mid h) = \frac{1}{1+e(-(a_i+w_i h_j))} = \tilde{O}\,(a_i + \sum h_j w_{ij})$$

The difference between the outer products of those probabilities with input vectors v_0 and v_k results in the updated matrix:

$$\Delta W = v_O) \otimes P(h_O \mid v_O) - v_k \otimes P(h_k \mid v_k) - v_k)$$

Using the update matrix, the new weights can be calculated with gradient ascent, given by:

$$W_{new} = W_{old} + \triangle W$$

Now that you have an idea of what are Restricted Boltzmann Machines and the layers of RBM, let's move on with our Restricted Boltzmann Machine Tutorial and understand their working with the help of an example.

Collaborative Filtering

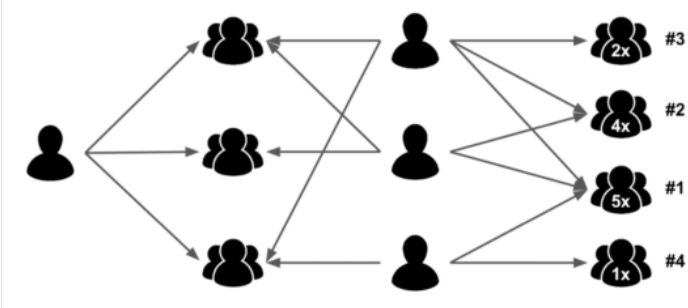


Figure 65: Collaborative Filtering

RBMs have found applications in dimensionality reduction, classification, collaborative filtering and many more. They can be trained in either supervised or unsupervised ways, depending on the task.

Recognizing Latent factors in the Data. Let us assume that some people were asked to rate a set of movies in the scale of 1–5 and each movie could be explained in terms of a set of latent factors such as drama, fantasy, action and many more. Restricted Boltzmann Machines are used to analyze and find out these underlying factors.

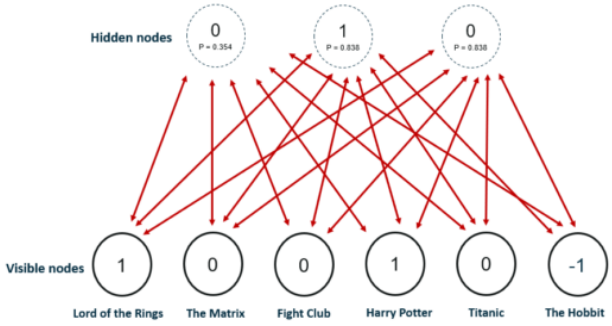


Figure 66: RBMs

The analysis of hidden factors is performed in a binary way, the user only tells if they liked a specific movie or not (rating 0) and it represents the inputs for the input/visible layer. Given the inputs, the RMB then tries to discover latent factors in the data that can explain the movie choices and each hidden neuron represents one of the latent factors. Let us consider the following example where a user likes Lord of the Rings and Harry Potter but does not like The Matrix, Fight Club and Titanic. The Hobbit has not been seen yet so it gets a -1 rating. Given these inputs, the Boltzmann Machine may identify three hidden factors Drama, Fantasy and Science Fiction which correspond to the movie genres.

Using Latent Factors for Prediction

After the training phase, the goal is to predict a binary rating for the movies that had not been seen yet. Given the training data of a specific user, the network is able to identify the latent factors based on the user's preference and sample from Bernoulli distribution can be used to find out which of the visible neurons now become active.

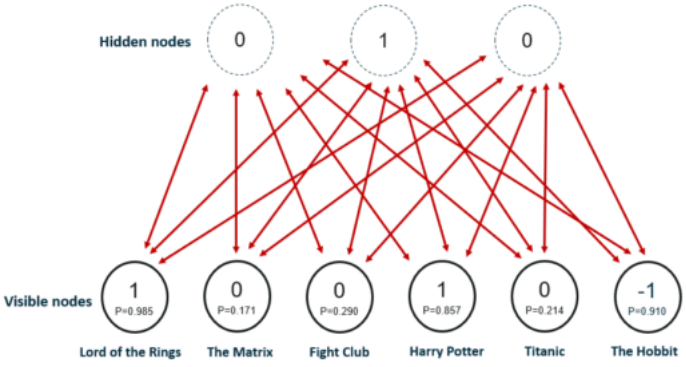


Figure 67: Using Latent Factors for Prediction

The image shows the new ratings after using the hidden neuron values for the inference. The network identified Fantasy as the preferred movie genre and rated The Hobbit as a movie the user would like.

The process from training to the prediction phase goes as follows:

- Train the network on the data of all users
- During inference-time, take the training data of a specific user

- Use this data to obtain the activations of hidden neurons
- Use the hidden neuron values to get the activations of input neurons
- The new values of input neurons show the rating the user would give yet unseen movies

DBNs or RBMs or Deep Autoencoders Coupled with Classification Layers Algorithms in cybersecurity

Both RBMs and autoencoders can be combined with a classification layer (Figure 5) to perform a classification using a fully connected layer or layers. The layers trained by applying unsupervised learning are used as feature extractors, and they constitute inputs into the fully connected layers that are trained using back propagation. Unlike the RBM or autoencoder layers, these layers require labels to train. These types of networks have shown success in a number of applications, including acoustic modeling, speech recognition, and image recognition.

1.5. Recursive Neural Networks Algorithms in cybersecurity

The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If you want to predict the next word in a sentence you better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations and you already know that they have a "memory" which captures information about what has been calculated so far "Whenever there is a sequence of data and that temporal dynamics that connects the data is more important than the spatial content of each individual frame."– Lex Fridman (MIT)

Different types of RNN's

The core reason that recurrent nets are more exciting is that they allow us to operate over sequences of vectors: Sequences in the input, the output, or in the most general case both. A few examples may make this more concrete:

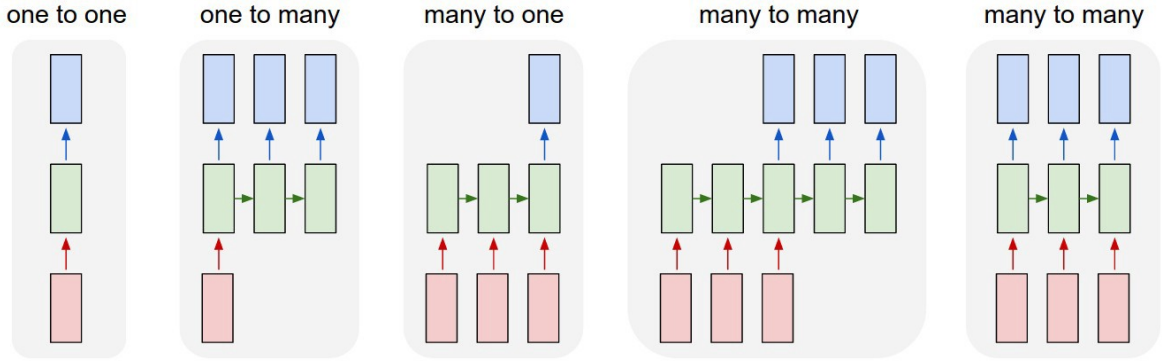


Figure 68 RNNs

Different types of Recurrent Neural Networks. (2) Sequence output (e.g. image captioning takes an image and outputs a sentence of words). (3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). (4) Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). (5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like. Each rectangle

in above image represent Vectors and Arrows represent functions. Input vectors are Red in color, output vectors are blue and green holds RNN's state.
One-to-one: This also called as Plain/Vaniall Neural networks. It deals with Fixed size of input to Fixed size of Output where they are independent of previous information/output.
Ex: Image classification.
One-to-Many: it deals with fixed size of information as input that gives sequence of data as output. For example: Image Captioning takes image as input and outputs a sentence of words.
Many-to-One: It takes Sequence of information as input and outputs a fixed size of output.
Example: sentiment analysis where a given sentence is classified as expressing positive or negative sentiment.
Many-to-Many: It takes a Sequence of information as input and process it recurrently outputs a Sequence of data. Example: Machine Translation, where an RNN reads a sentence in English and then outputs a sentence in French.
Bidirectional Many-to-Many: Synced sequence input and output. Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like.
Ex: video classification where we wish to label each frame of the video.
Advantages of Recurrent Neural Network
The main advantage of RNN over ANN is that RNN can model sequence of data (i.e. time series) so that each sample can be assumed to be dependent on previous ones
Recurrent neural network is even used with convolutional layers to extend the effective pixel neighborhood.
Disadvantages of Recurrent Neural Network
Gradient vanishing and exploding problems.
Training an RNN is a very difficult task.
It cannot process very long sequences if using tanh or relu as an activation function

III. Cyber Applications of Deep Learning Methods

Machine Learning can perform efficiently in small scale data and lower configured systems. This emerges the Deep Learning which is a subfield of the Machine learning approach. Deep Learning can perform well only in a tremendous amount of data and high configured systems. The main advantage of Deep Learning on cybersecurity is the deep neural network process. This process deeply examines the data and for this study, the deep learning algorithm requires a vast amount of data. In industry, government sector, peoples and organization generating a large amount of data. For handling cyber security for this much amount of data the deep learning is the precise answer. The big companies like Google, Amazon, Microsoft, and salesforceman.com to follow deep learning technology now cybersecurity institute takes and follow them in this algorithm but deep learning is a challenging field for full helpfulness. Deep Learning algorithm is categorized into supervised deep learning and unsupervised deep learning. Supervised deep learning approach will predict only the targeted values from a set of data but in unsupervised approach, there are no such target values and it simply predicts all the possible values from the dataset. The usage of these two categories depends on the implementation of cybersecurity needs. The approach of deep learning to the field of cybersecurity ranges from the intrusion detection system for sensor networks, transport layer, Malicious Code Detection, Hybrid malware classification, Behaviors detection of Botnet, traffic identification and anomaly detection. Benefits deep learning for cybersecurity protects the

plenty of data and the data are hybrid categories. It only focuses on the multivariate categorical distributions.

Deep Learning in Network Intrusion Detection System

Network Intrusion Detection System is used to detect the security compromises in any type of organization. It is the first line of defensive mechanism which can be very useful for the detection of the intrusion in the system. Here, it will also produce false results due to the time consuming and low latency of the audition of files. But it can be improved with the help of the deep learning approach which produces better results than that of the normal approach. The authors (Niyaz 2014) stated that the Network Intrusion Detection System (NIDS) can be improved with the Self-taught Learning (STL), a deep learning-based method to automate the detection system with high precise results. STL contains two stages which are used as classifiers. The first method is used to the unlabeled data which are learned the good representation of the data. Another stage is to present the data in a related form where the learning is done on the previous stage. These stages are applied to the NIDS which easily provides better results and also helps to do things well in future too. The STL classification achieves an accuracy of up to 98% than the all other classifications. In the future, this can also be used for the observation of the raw network traffic of the system which is very much advanced than that of the NIDS.

Deep Learning in Sensor Network Security

In sensor networks, a vast number of cyberattacks are possible for the malfunction of the sensor devices. Here, the intrusion or threats are also screened and monitored with the help of the IDS. (Wang 2015) proposed a deep learning-based approach to these problems. SCDNN which the combination of both Spectral Clustering and Deep Neural Network algorithms that are the clustering of data and ink subsets and the distances are calculated with the help of deep neural approach. These algorithms are applied to both the training data and the testing dataset. The sensed data are classified using DNN classifier after the clustering the data. The results of this approach not only better than that of the traditional approach but also advances the Support Vector Machine and Bayesian algorithm which are all the machine learning approaches. It also provides the facility to work for a large amount of dataset which is considerably larger than the machine learning. So, Deep learning approach is widely used for the vast dataset or project which yields best results than that of the smaller project work.

Deep Learning Approach in Detecting Threats to the IoT Environment

Internet of Things is the communication between devices to devices with human intervention. Here, a large number of attacks is possible at the gateway of the network. Most common attacks in the IoT environment are Denial of Service attacks and Denial of Sleep attacks. Denial of Service is simply the host devices are denied with flooding of requests from the attackers which restricts the host to connect with the intended devices. In Denial of Sleep attack which mostly drains the power of the sensor nodes where the sensor device lost its ability to connect with the IoT environment. Using the traffic of this environment the attackers can also read the packets and send false responses to the devices. Here, the authors used the deep random neural networks (Deep RNN) which help to understand the network attacks more precisely. The processing of this approach comparatively helps to reduce the disabling of nodes in the network with an early prediction of the attack happening in the environment. In the future, this approach can also be applicable to the Zigbee and Bluetooth devices to reduce the network attack of the ad-hoc network. Here, the attacks and intrusions are prevented with the Deep RNN technique which provides a unique solution to the core problem present in real time. The results are far best than that of the traditional machine learning approach where the Support Vector Machine is very difficult to be used in the IOT environment which is normally a large system which provides vast data with high amount

of security issues. Instead of SVM, the Deep RNN approach is used to generate the efficiency of the system.

Deep Learning in Android Malware Detection

In the Android platform, large numbers of applications are developed and deployed everyday which also leads a large number of malware application with causes serious troubles to the Android OS and corrupts the data of our smartphone devices. Each malware effects are different from each other. The detection and prevention of this malware are a tedious process in cyber forensics. It can be easily done with the help of a deep learning approach. Here the researchers (Yuan 2014) proposed a model for this malware detection for Android environment which is Deep Belief Network (DBN). It consists of two phases namely Pre-training phase and propagation phase. In the pre-training phase, the learning space is given to the DBN whereas in the propagation phase the trained pre-training phases are again finely improved with supervision. The dataset which contains both normal applications as well as the malware containing applications which are fed into this DBN framework to detect the malware software's present in it. It provides a result of up to 96.4% better than that of the machine learning approach which includes SVM, Naive Bayesian algorithm and LR. So, deep learning plays a major role in now a day's challenges to eradicating the intrusion or malware in the system.

Hybrid Malicious Code Detection Using Deep Learning

A hybrid malicious code detection using deep learning algorithm based on AutoEncoder and DBN. The autoencoder deep learning method is applied which is responsible for to reduce the dimensionality of data with nonlinear mapping, and extracting the main features of the data. The authors (Li 2013) used a Deep Belief Networks learning methods to detect malicious code. This deep belief network is consisting of multilayer restricted Boltzmann machines and a layer of BP neural network. This algorithm is based on both supervised and unsupervised learning. First, every layer of RBM is learned under unsupervised training then the output vector of the last layer of RBM is given as an input vector of BP neural network which undergoes supervised training. The outcome shows accurate detection performance and reduces the time complexity. The autoencoder consists of two parts encoder and decoder and three layers (input, output, hidden layers) here the data dimensionality reduction is defined when the hidden layer neurons are less than the number of input layer neurons and output layer neurons. The autoencoder takes place by three steps which are pretraining, unrolling and fine-tuning. The pretraining process is nothing but the output vector of each RBM hidden layer neuron is given to the input for next RBM. Connecting these independent RBM into a multilayered autoencoder is done in the unrolling process. After the pretraining process to get optimal weights the adjustment is done on initial weights further is called a Fine-tuning process. The hybrid detection algorithm will take care of digitizing and normalizing the input data, reducing the dimensionality of the data, feature mapping, DBM classifier, through the RBM learning rules the network layer is trained layer by layer, and finally the input test samples into the trained classifier to detect the malicious code and the normal code. The result of this algorithm shows greater accuracy of detecting hybrid malicious code by using the combination of AutoEncoder and DBM and this ensure the best time complexity for this model.

Learning Network Based Malware Detection

Malware are increasing day by day parallel along with the evolution of technology. The detection of malware is a complex process which requires a large amount of time and works in the modern period whereas a new approach this problem is Stacked Auto-Encoders (SAEs) framework. It is used for the intelligent detection of the malware present in the

network. The authors (Hardy 2015) proposed this method which contains two phases which are having both unsupervised and supervised learning a feature of SAE framework. This model is specially designed to detect the presence of malware in Windows API calls. By performing this type of detection, helps to improve the systems which are free from all the general attacks that will affect the sensitive data and also improve the performance of the system. Staked Auto-Encoders provides a wide range of classification with deep learning impact leads to the three layers in the input layer, Hidden layer, and Output layer. Each layer helps to extract the features from the data which is useful for the future application of classifier. This method also provides better results than that of the other approaches to cybersecurity.

Deep Learning Approach for Flash Malware Detection

Adobe Flash is a platform-independent software where all the multimedia is supported by them. It can also subject to a large number of attacks in recent years due to the development of technology. This software is widely used by all the users who operate on the computer. So, this can be an easy target by the attacks who all try to hack the system of the users. Recently, Casper malware which affects a large number of host systems leads to a large amount of data loss and also gives the opportunity to the hacker to earn money through this mode of attacks. To avoid this type of malware the use deep learning approach SWF is used. This method is better than that of machine learning because of the complex nature of the malware data present in today's environment. The data are first extracted as static and dynamic analysis and then the SWF classifier is applied to the system. This classifier is proposed by the authors to detect this malware in the present-day flash. This approach can also be used to detect the intrusions in the other independent platforms also. It is the additional advantage of using deep learning approach where the same classifier is used to detect the problems.

Applications for Deep Learning

The most of deep learning application apply to market, sales and finance field. Deep learning technique protects sensitive data and product from malicious code and hacker attack.

Learning-Based Detection

JavaScript code analysis is done for generating detection model. These models to be consist of deep learning techniques.

Evaluation

The malicious Java code applicable for a larger dataset which is consists of 2700 samples.
Zero-Day Attacks Detection: A zero-day attack is an ability to identify malicious code and analysis of unknown attacks.
Malware, Zero Day, and APT Detection: Deep learning is classified as malicious code without any rules. The neural network learns to identify the malicious code once identify the unknown files. In real time apply to in this process need for high accuracy. They are trained in malware detection neural network must analyze the millions of accurate malicious code classification. These are trained neural network deploy to the endpoint of the device. For example, a trained neural network would be deployed in mobile device endpoint. It is access to without network connection. So, in which that detect the malicious attacks.
Advanced Intrusion Detection and Network Anomaly Detection
Now intrusion detection system work through the grouping of anomaly detection and misuse detection techniques. These techniques are deploying to particular system network and host level. System guesses of intrusion based on deep learning techniques. Deep learning

techniques analyze and filter to unknown codes. Packet analysis techniques include deep learning. These are preventing distributed denial of service, ransomware, spam, and phishing and botnet attack. It can find unusual behavior for network communication. In this method find the network communication likelihood. Network communication also selects as low likelihood.

Phishing Detection

Phishing attack already discussed previously. It is sensitive to personal data gathering techniques. Cyberattack mostly injected to these types of method. Machine learning and deep learning is performed to pattern recognition so attack to be detected. Deep insight is a graph analysis algorithm. Deep insight text used to analyze email pattern identification and harmless mail. These are the best tool for identifying phishing attack.

Identification of Insider Threats

The pattern could be gritty to the device, user individual and network let use to deep learning techniques. Those threats in progress to be indicated are correlated data let's identify subtitles. They are modelled for application, key business system and, user behavior. The incorporate data to be a loss to deploy the file drop box and any other cloud service. APT identification is the main added benefit for the system. It is intrusion detection, firewall, and server log and information security management event. Its is could not just insider. Training material needed for the neural network could be providing an important resource. Traffic Detection Using Deep Learning Cyber-attack is mostly taking for source code, consumer data, important document, and software keys. Anomaly network is used to trace traffic in which that security is difficult to in this method. Now anomaly network used for communication by specific of malware based on rule-based intrusion network tools. They are transmitted of stolen encrypt with remote server traffic to occur. It is detecting the TOR traffic and analysis to traffic package. In this analyze done by a single flow of packet. Source address, source port, destination address, and destination port are involved in each flow constitutes. TOR is a safeguard for the user. TOR transfer to information hacker get access to information. The IP address cannot directly predict the in this attack. A landscape sends alert for distill network. Let us apply to deep learning method easy to detect the attack and increase the security.

IV. The Importance of Encryption in Cybersecurity

1. What are encryption methods?

Encryption is the process of encoding information. This process converts the original representation of the information, known as plaintext, into an alternative form known as ciphertext. Only authorized parties can decipher a ciphertext back to plaintext and access the original information. Encryption does not itself prevent interference but denies the intelligible content to a would-be interceptor. For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm. It is possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, considerable computational resources and skills are required. An authorized recipient can easily decrypt the message with the key provided by the originator to recipients but not to unauthorized users. Historically, various forms of encryption have been used to aid in cryptography. Early encryption techniques were often utilized in military messaging. Since then, new techniques have emerged and become commonplace in all areas of modern computing. Modern encryption schemes utilize the concepts of public-key and symmetric-

key Modern encryption techniques to ensure security because modern computers are inefficient at cracking the encryption

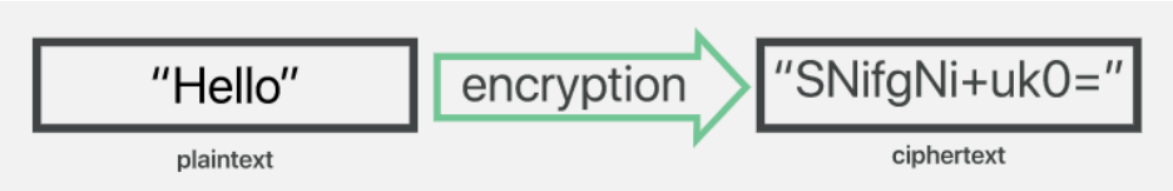


Figure 69: Encryption methods

2. What type of encryption

2.1. Data Encryption Standard (DES)

The Data Encryption Standard (DES) is a symmetric-key algorithm for the encryption of digital data. Although its short key length of 56 bits makes it too insecure for modern applications, it has been highly influential in the advancement of cryptography.
Developed in the early 1970s at IBM and based on an earlier design by Horst Feistel, the algorithm was submitted to the National Bureau of Standards (NBS) following the agency's invitation to propose a candidate for the protection of sensitive, unclassified electronic government data. In 1976, after consultation with the National Security Agency (NSA), the NBS selected a slightly modified version (strengthened against differential cryptanalysis, but weakened against brute-force attacks), which was published as an official Federal Information Processing Standard (FIPS) for the United States in 1977.
The publication of an NSA-approved encryption standard led to its quick international adoption and widespread academic scrutiny. Controversies arose from classified design elements, a relatively short key length of the symmetric-key block cipher design, and the involvement of the NSA, raising suspicions about a backdoor. The S-boxes that had prompted those suspicions were designed by the NSA to remove a backdoor they secretly knew (differential cryptanalysis). However, the NSA also ensured that the key size was drastically reduced so that they could break the cipher by brute force attack. The intense academic scrutiny the algorithm received overtime led to the modern understanding of block ciphers and their cryptanalysis.
DES is insecure due to the relatively short 56-bit key size. In January 1999, distributed.net and the Electronic Frontier Foundation collaborated to publicly break a DES key in 22 hours and 15 minutes (see chronology). There are also some analytical results that demonstrate theoretical weaknesses in the cipher, although they are infeasible in practice. The algorithm is believed to be practically secure in the form of Triple DES, although there are theoretical attacks. This cipher has been superseded by the Advanced Encryption Standard (AES). DES has been withdrawn as a standard by the National Institute of Standards and Technology.

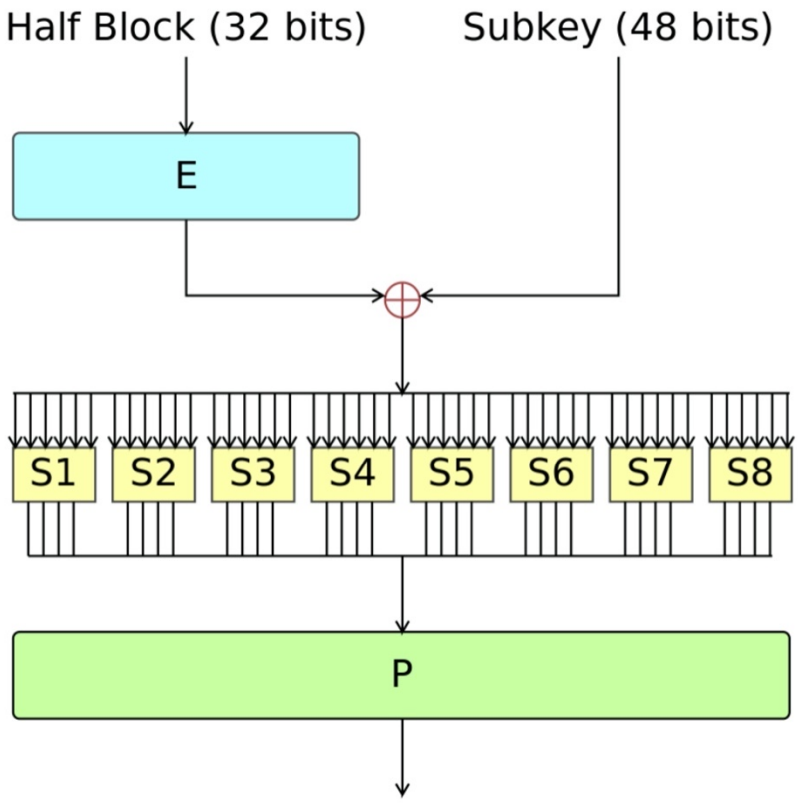


Figure 70: Data Encryption Standard (DES)

Description:

DES is the archetypal block cipher—an algorithm that takes a fixed-length string of plaintext bits and transforms it through a series of complicated operations into another ciphertext bitstring of the same length. In the case of DES, the block size is 64 bits. DES also uses a key to customize the transformation, so that decryption can supposedly only be performed by those who know the particular key used to encrypt. The key ostensibly consists of 64 bits; however, only 56 of these are actually used by the algorithm. Eight bits are used solely for checking parity, and are thereafter discarded. Hence the effective key length is 56 bits.
Like other block ciphers, DES by itself is not a secure means of encryption, but must instead be used in a mode of operation. FIPS-81 specifies several modes for use with DES. Further comments on the usage of DES are contained in FIPS-74. Decryption uses the same structure as encryption, but with the keys used in reverse order. (This has the advantage that the same hardware or software can be used in both directions.)

Overall structure
The algorithm's overall structure is shown in the picture below there are 16 identical stages of processing, termed rounds. There is also an initial and final permutation, termed IP and FP, which are inverses (IP "undoes" the action of FP, and vice versa). IP and FP have no cryptographic significance, but were included in order to facilitate loading blocks in and out of mid-1970s 8-bit based hardware.
Before the main rounds, the block is divided into two 32-bit halves and processed alternately; this criss-crossing is known as the Feistel scheme. The Feistel structure ensures that decryption and encryption are very similar processes—the only difference is that the

subkeys are applied in the reverse order when decrypting. The rest of the algorithm is identical. This greatly simplifies implementation, particularly in hardware, as there is no need for separate encryption and decryption algorithms.
The ⊕ symbol denotes the exclusive-OR (XOR) operation. The F-function scrambles half a block together with some of the key. The output from the F-function is then combined with the other half of the block, and the halves are swapped before the next round. After the final round, the halves are swapped; this is a feature of the Feistel structure which makes encryption and decryption similar processes.

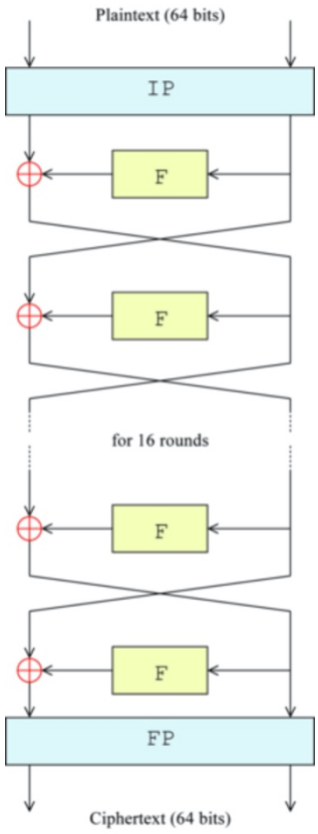


Figure 71: The overall structure of encryption methods

2.2. Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES), also known by its original name Rijndael, is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. AES is a subset of the Rijndael block cipher developed by two Belgian cryptographers, Vincent Rijmen and Joan Daemen, who submitted a proposal to NIST during the AES selection process. Rijndael is a family of ciphers with different key and block sizes. For AES, NIST selected three members of the Rijndael family, each with a block size of 128 bits, but three different key lengths: 128, 192 and 256 bits. AES has been adopted by the U.S. government and is now used worldwide. It supersedes the Data Encryption Standard (DES), which was published in 1977. The algorithm described by AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data.

In the United States, AES was announced by the NIST as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001. This announcement followed a five-year standardization process in which fifteen competing designs were presented and evaluated, before the Rijndael cipher was selected as the most suitable (see Advanced Encryption Standard process for more details). AES became effective as a federal government standard on May 26, 2002, after approval by the Secretary of Commerce. AES is included in the ISO/IEC 18033-3 standard. AES is available in many different encryption packages, and is the first (and only) publicly accessible cipher approved by the National Security Agency (NSA) for top secret information when used in an NSA approved cryptographic module

Description of the ciphers

AES is based on a design principle known as a substitution–permutation network, and is efficient in both software and hardware. Unlike its predecessor DES, AES does not use a Feistel network. AES is a variant of Rijndael, with a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. By contrast, Rijndael per se is specified with block and key sizes that may be any multiple of 32 bits, with a minimum of 128 and a maximum of 256 bits. AES operates on a 4 × 4 column-major order array of bytes, termed the state. Most AES calculations are done in a particular finite field. For instance, 16 bytes, (b0, b1…, b15) are represented as this two-dimensional array:

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

Figure 72: AES

The key size used for an AES cipher specifies the number of transformation rounds that convert the input, called the plaintext, into the final output, called the ciphertext. The number of rounds is as follows:

- 10 rounds for 128-bit keys.
- 12 rounds for 192-bit keys.
- 14 rounds for 256-bit keys.

Each round consists of several processing steps, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform ciphertext back into the original plaintext using the same encryption key.

High-level description of the algorithm

Key Expansion – round keys are derived from the cipher key using the AES key schedule. AES requires a separate 128-bit round key block for each round plus one more.

Initial round key addition: AddRoundKey – each byte of the state is combined with a byte of the round key using bitwise xor. 9, 11 or 13 rounds:

- SubBytes – a non-linear substitution step where each byte is replaced with another according to a lookup table.

- ShiftRows – a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.
- Mix Columns – a linear mixing operation which operates on the columns of the state, combining the four bytes in each column.
- AddRoundKey
- Final round (making 10, 12 or 14 rounds in total):
- SubBytes
- ShiftRows
- AddRoundKey

The SubBytes step

In the SubBytes step, each byte in the state is replaced a SubBytes using an 8-bit substitution box. This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over GF (28), known to have good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), and also any opposite fixed points. While performing the decryption, the InvSubBytes step (the inverse of SubBytes) is used, which requires first taking the inverse of the affine transformation and then finding the multiplicative inverse.

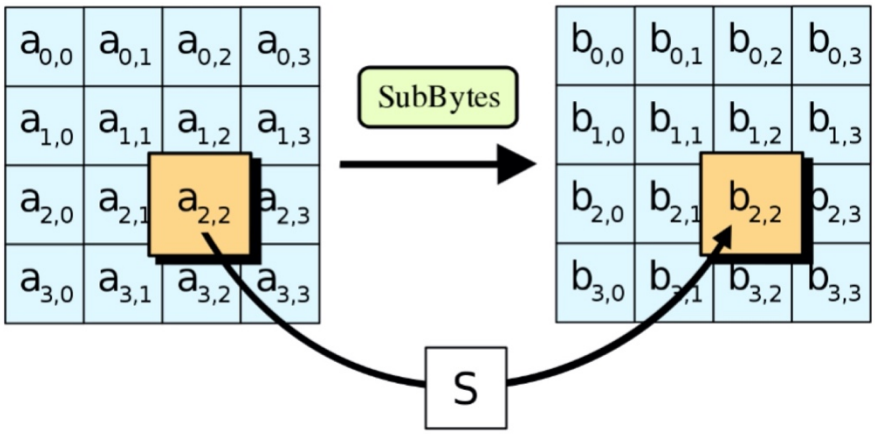


Figure 73: *High-level description of methods*

The ShiftRows step

The ShiftRows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left. Similarly, the third and fourth rows are shifted by offsets of two and three respectively. In this way, each column of the output state of the ShiftRows step is composed of bytes from each column of the input state. The importance of this step is to avoid the columns being encrypted independently, in which case AES degenerates into four independent block ciphers.

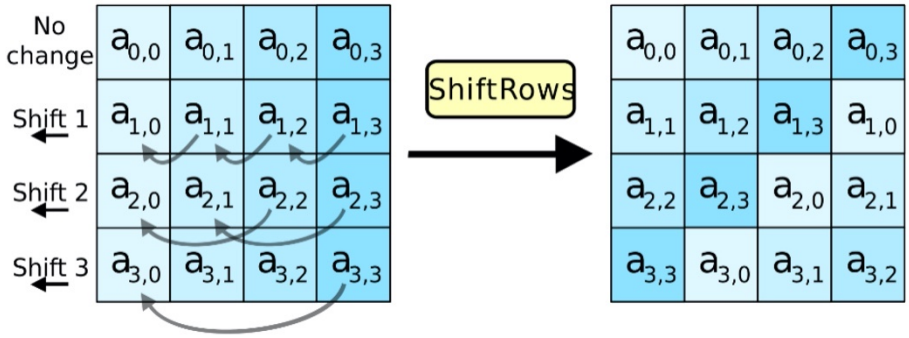


Figure 74: The ShiftRows step

The MixColumns step

In the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes. Together with ShiftRows, MixColumns provides diffusion in the cipher.

During this operation, each column is transformed using a fixed matrix (matrix left-multiplied by column gives new value of column in the state):

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} \qquad 0 \leq j \leq 3$$

Figure 75: MixColumns step

Matrix multiplication is composed of multiplication and addition of the entries. Entries are 8-bit bytes treated as coefficients of polynomial of order x7. Addition is simply XOR. Multiplication is modulo irreducible polynomial. X8 + x4 + x3 + x+1. If processed bit by bit, then, after shifting, a conditional XOR with 1B16 should be performed if the shifted value is larger than FF16(overflow must be corrected by subtraction of generating polynomial). These are special cases of the usual multiplication in GF (28).

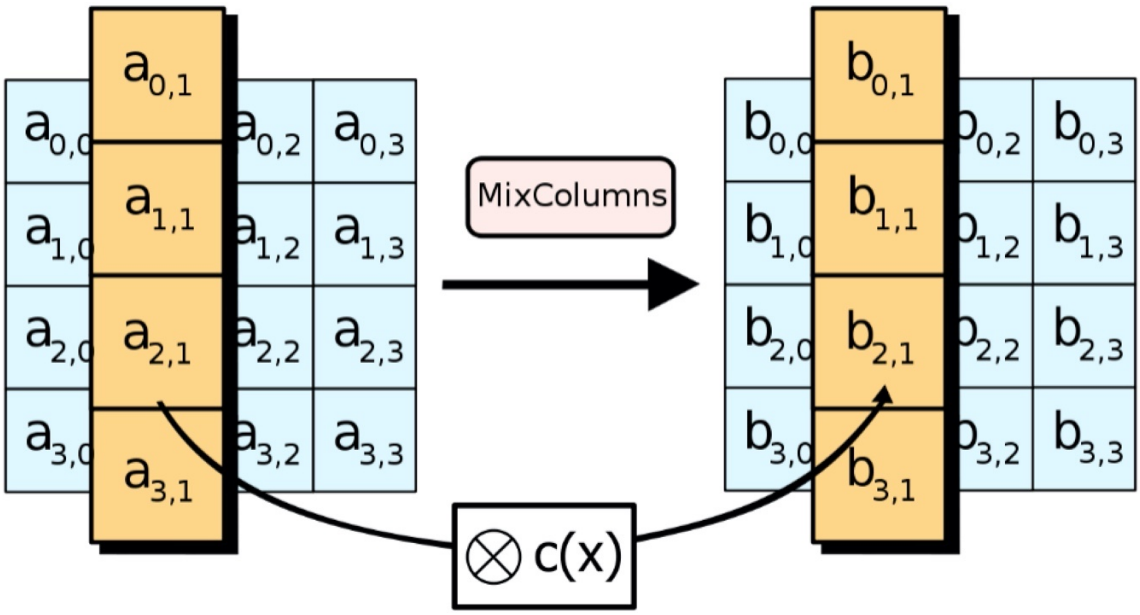


Figure 76: Matrix multiplication

The AddRoundKey step

In the AddRoundKey step, the subkey is combined with the state. For each round, a subkey is derived from the main key using Rijndael's key schedule; each subkey is the same size as the state. The subkey is added by combining each byte of the state with the corresponding byte of the subkey using bitwise XOR.

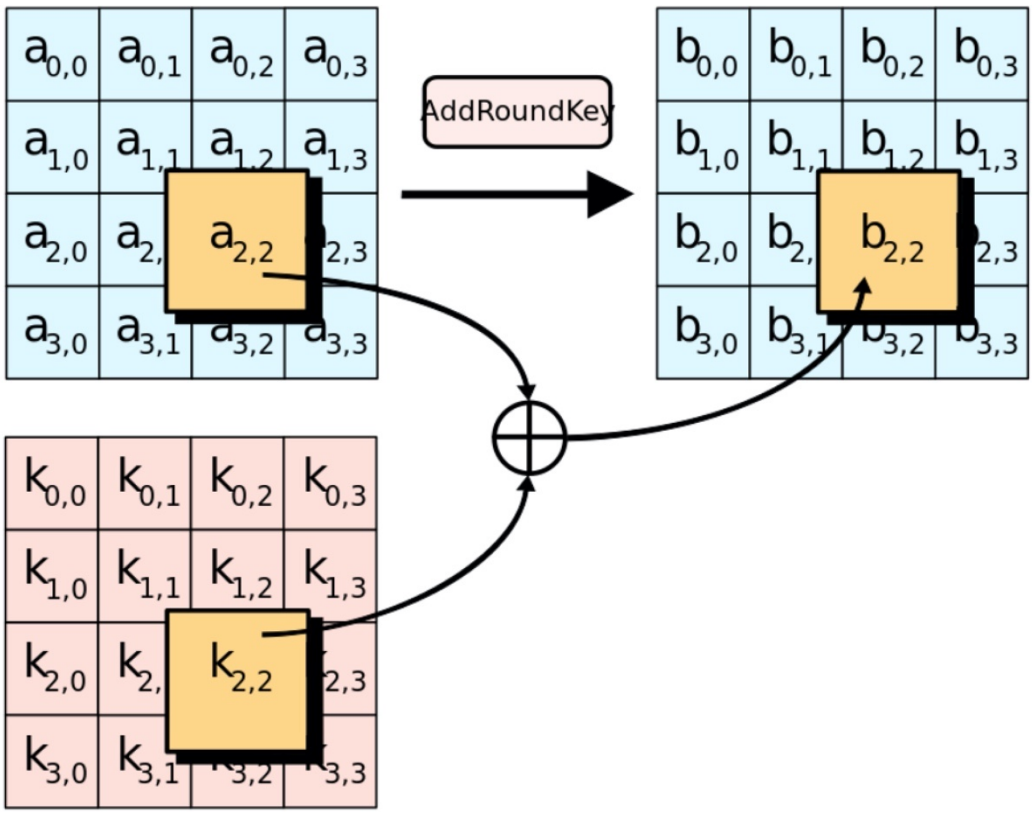


Figure 77: AddRoundKey

Optimization of the cipher
On systems with 32-bit or larger words, it is possible to speed up execution of this cipher by combining the SubBytes and ShiftRows steps with the MixColumns step by transforming them into a sequence of table lookups. This requires four 256-entry 32-bit tables (together occupying 4096 bytes). A round can then be performed with 16 table lookup operations and 12 32-bit exclusive-or operations, followed by four 32-bit exclusive-or operations in the AddRoundKey step. Alternatively, the table lookup operation can be performed with a single 256-entry 32-bit table (occupying 1024 bytes) followed by circular rotation operations. Using a byte-oriented approach, it is possible to combine the SubBytes, ShiftRows, and MixColumns steps into a single round operation.

2.3. Triple Data Encryption Standard (TripleDES)

Triple DES (3DES or TDES), officially the Triple Data Encryption Algorithm (TDEA or Triple DEA), is a symmetric-key block cipher, which applies the DES cipher algorithm three times to each data block. The Data Encryption Standard's (DES) 56-bit key is no longer considered adequate in the face of modern cryptanalytic techniques and supercomputing power. However, an adapted version of DES, Triple DES (3DES), uses the same algorithm to produce a more secure encryption.
While the government and industry standards abbreviate the algorithm's name as TDES (Triple DES) and TDEA (Triple Data Encryption Algorithm), RFC 1851 referred to it as 3DES from the time it first promulgated the idea, and this namesake has since come into wide use by most vendors, users, and cryptographers.

Algorithm

The original DES cipher's key size of 56 bits was generally sufficient when that algorithm was designed, but the availability of increasing computational power made brute-force attacks feasible. Triple DES provides a relatively simple method of increasing the key size of DES to protect against such attacks, without the need to design a completely new block cipher algorithm. A naive approach to increase strength of a block encryption algorithm with short key length (like DES) would be to use two keys (k1, k2) instead of one, and encrypt each block twice:
Ek2(Ek1((plaintext)). If the original key length is n bits, one would hope this scheme provides security equivalent to using key 2n bits long. Unfortunately, this approach is vulnerable to meet-in-the-middle attack: given a known plaintext pair (x, y) such that, y = Ek2(Ek1(x)) one can recover the key pair (K1, K2) in 2 n steps, instead of 22n steps one would expect from an ideally secure algorithm with 2n bits of key.
Therefore, Triple DES uses a "key bundle" that comprises three DES keys, K1, K2 and K3 each of 56 bits (excluding parity bits). The encryption algorithm is:

$$ciphertext = E_{K3}(D_{K2}(E_{K1}(plaintext)))$$

That is, DES encrypt with K1, DES decrypt with K2, then DES encrypt with K3.
Decryption is the reverse:

$$plaintext = D_{K1}(E_{K2}(E_{K3}(ciphertext)))$$

That is, decrypt with K3, encrypt with K2, then decrypt with K1.
Each triple encryption encrypts one block of 64 bits of data.

In each case the middle operation is the reverse of the first and last. This improves the strength of the algorithm when using keying option 2 and provides backward compatibility with DES with keying option 3.

Keying options

The standards define three keying options:

Keying option 1

All three keys are independent. Sometimes known as 3TDEA or triple-length keys.

This is the strongest, with 3 × 56 = 168 independent key bits. It is still vulnerable to meet-in-the-middle attack, but the attack requires 22 × 56 steps.

Keying option 2

K1 and K2 are independent, and K3 = K1. Sometimes known as 2TDEA or double-length keys. This provides a shorter key length of 112 bits and a reasonable compromise between DES and Keying option 1, with the same caveat as above.[15] This is an improvement over "double DES" which only requires 256 steps to attack. NIST has deprecated this option.[13]

Keying option 3

All three keys are identical, i.e. K1 = K2 = K3.

This is backward compatible with DES, since two operations cancel out. ISO/IEC 18033-3 never allowed this option, and NIST no longer allows K1 = K2 or K2 = K3.

Each DES key is 8 odd-parity bytes, with 56 bits of key and 8 bits of error-detection. A key bundle requires 24 bytes for option 1, 16 for option 2, or 8 for option 3.

NIST (and the current TCG specifications version 2.0 of approved algorithms for Trusted Platform Module) also disallows using any one of the 64 following 64-bit values in any keys (note that 32 of them are the binary complement of the 32 others; and that 32 of these keys are also the reverse permutation of bytes of the 32 others), listed here in hexadecimal (in each byte, the least significant bit is an odd-parity generated bit, it is discarded when forming the effective 56-bit keys):

```
01.01.01.01.01.01.01.01, FE.FE.FE.FE.FE.FE.FE.FE, E0.FE.FE.E0.F1.FE.FE.F1, 1F.01.01.1F.0E.01.01.0E,
01.01.FE.FE.01.01.FE.FE, FE.FE.01.01.FE.FE.01.01, E0.FE.01.1F.F1.FE.01.0E, 1F.01.FE.E0.0E.01.FE.F1,
01.01.E0.E0.01.01.F1.F1, FE.FE.1F.1F.FE.FE.0E.0E, E0.FE.1F.01.F1.FE.0E.01, 1F.01.E0.FE.0E.01.F1.FE,
01.01.1F.1F.01.01.0E.0E, FE.FE.E0.E0.FE.FE.F1.F1, E0.FE.E0.FE.F1.FE.F1.FE, 1F.01.1F.01.0E.01.0E.01,
01.FE.01.FE.01.FE.01.FE, FE.01.FE.01.FE.01.FE.01, E0.01.FE.1F.F1.01.FE.0E, 1F.FE.01.E0.0E.FE.01.F1,
01.FE.FE.01.01.FE.FE.01, FE.01.01.FE.FE.01.01.FE, E0.01.01.E0.F1.01.01.F1, 1F.FE.FE.1F.0E.FE.FE.0E,
01.FE.E0.1F.01.FE.F1.0E, FE.01.1F.E0.FE.01.0E.F1, E0.01.1F.FE.F1.01.0E.FE, 1F.FE.E0.01.0E.FE.F1.01,
01.FE.1F.E0.01.FE.0E.F1, FE.01.E0.1F.FE.01.F1.0E, E0.01.E0.01.F1.01.F1.01, 1F.FE.1F.FE.0E.FE.0E.FE,
01.E0.01.E0.01.F1.01.F1, FE.1F.FE.1F.FE.0E.FE.0E, E0.1F.FE.01.F1.0E.FE.01, 1F.E0.01.FE.0E.F1.01.FE,
01.E0.FE.1F.01.F1.FE.0E, FE.1F.01.E0.FE.0E.01.F1, E0.1F.01.FE.F1.0E.01.FE, 1F.E0.FE.01.0E.F1.FE.01,
01.E0.E0.01.01.F1.F1.01, FE.1F.1F.FE.FE.0E.0E.FE, E0.1F.1F.E0.F1.0E.0E.F1, 1F.E0.E0.1F.0E.F1.F1.0E,
01.E0.1F.FE.01.F1.0E.FE, FE.1F.E0.01.FE.0E.F1.01, E0.1F.E0.1F.F1.0E.F1.0E, 1F.E0.1F.E0.0E.F1.0E.F1,
01.1F.01.1F.01.0E.01.0E, FE.E0.FE.E0.FE.F1.FE.F1, E0.E0.FE.FE.F1.F1.FE.FE, 1F.1F.01.01.0E.0E.01.01,
01.1F.FE.E0.01.0E.FE.F1, FE.E0.01.1F.FE.F1.01.0E, E0.E0.01.01.F1.F1.01.01, 1F.1F.FE.FE.0E.0E.FE.FE,
01.1F.E0.FE.01.0E.F1.FE, FE.E0.1F.01.FE.F1.0E.01, E0.E0.1F.1F.F1.F1.0E.0E, 1F.1F.E0.E0.0E.0E.F1.F1,
01.1F.1F.01.01.0E.0E.01, FE.E0.E0.FE.FE.F1.F1.FE, E0.E0.E0.E0.F1.F1.F1.F1, 1F.1F.1F.1F.0E.0E.0E.0E,
```

With these restrictions on allowed keys, Triple DES has been reapproved with keying options 1 and 2 only. Generally, the three keys are generated by taking 24 bytes from a strong random generator and only keying option 1 should be used (option 2 needs only 16 random bytes, but strong random generators are hard to assert and it's considered best practice to use only option 1).

2.4. Rivest-Shamir-Adleman (RSA)

RSA (Rivest–Shamir–Adleman) is one of the first public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and distinct from the decryption key which is kept secret (private). In RSA, this asymmetry is

based on the practical difficulty of factoring the product of two large prime numbers, the "factoring problem". The acronym RSA is the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who publicly described the algorithm in 1977. Clifford Cocks, an English mathematician working for the British intelligence agency Government Communications Headquarters (GCHQ), had developed an equivalent system in 1973, which was not declassified until 1997.
A user of RSA creates and then publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers must be kept secret. Anyone can use the public key to encrypt a message, but only someone with knowledge of the prime numbers can decode the message. Breaking RSA encryption is known as the RSA problem. Whether it is as difficult as the factoring problem is an open question. There are no published methods to defeat the system if a large enough key is used.
RSA is a relatively slow algorithm, and because of this, it is less commonly used to directly encrypt user data. More often, RSA passes encrypted shared keys for symmetric key cryptography which in turn can perform bulk encryption-decryption operations at much higher speed.

Operation

The RSA algorithm involves four steps: key generation, key distribution, encryption and decryption. A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d and n such that with modular exponentiation for all integers m (with $0 \leq m < n$):

$$(m^e)^d = m \pmod n$$

and that knowing e and n, or even m, it can be extremely difficult to find d. The triple bar (≡) here denotes modular congruence.
In addition, for some operations it is convenient that the order of the two exponentiations can be changed and that this relation also implies:

$$(m^d)^e = m \pmod n$$

RSA involves a public key and a private key. The public key can be known by everyone, and it is used for encrypting messages. The intention is that messages encrypted with the public key can only be decrypted in a reasonable amount of time by using the private key. The public key is represented by the integers n and e; and, the private key, by the integer d (although n is also used during the decryption process. Thus, it might be considered to be a part of the private key, too). m represents the message (previously prepared with a certain technique explained below).

2.5. Blowfish

Blowfish is a symmetric-key block cipher, designed in 1993 by Bruce Schneier and included in many cipher suites and encryption products. Blowfish provides a good encryption rate in software and no effective cryptanalysis of it has been found to date. However, the Advanced Encryption Standard (AES) now receives more attention, and Schneier recommends *Twofish for modern applications.*
Schneier designed Blowfish as a general-purpose algorithm, intended as an alternative to the aging DES and free of the problems and constraints associated with other algorithms. At the time Blowfish was released, many other designs were proprietary, encumbered by

patents or were commercial or government secrets. Schneier has stated that, "Blowfish is unpatented, and will remain so in all countries. The algorithm is hereby placed in the public domain, and can be freely used by anyone."

Algorithm

Blowfish has a 64-bit block size and a variable key length from 32 bits up to 448 bits. It is a 16-round Feistel cipher and uses large key-dependent S-boxes. In structure it resembles CAST-128, which uses fixed S-boxes.

The adjacent diagram shows Blowfish's encryption routine. Each line represents 32 bits. There are five subkey-arrays: one 18-entry P-array (denoted as K in the diagram, to avoid confusion with the Plaintext) and four 256-entry S-boxes (S0, S1, S2 and S3).

Every round r consists of 4 actions:

Action 1	XOR the left half (L) of the data with the *r* th P-array entry
Action 2	Use the XORed data as input for Blowfish's F-function
Action 3	XOR the F-function's output with the right half (R) of the data
Action 4	Swap L and R

The F-function splits the 32-bit input into four eight-bit quarters, and uses the quarters as input to the S-boxes. The S-boxes accept 8-bit input and produce 32-bit output. The outputs are added modulo 232 and XORed to produce the final 32-bit output.

After the 16th round, undo the last swap, and XOR L with K18 and R with K17 (output whitening). Decryption is exactly the same as encryption, except that P1, P2, ..., P18 are used in the reverse order. This is not so obvious because xor is commutative and associative. A common misconception is to use inverse order of encryption as decryption algorithm (i.e. first XORing P17 and P18 to the ciphertext block, then using the P-entries in reverse order).

Blowfish's key schedule starts by initializing the P-array and S-boxes with values derived from the hexadecimal digits of pi, which contain no obvious pattern (see nothing up my sleeve number). The secret key is then, byte by byte, cycling the key if necessary, XORed with all the P-entries in order. A 64-bit all-zero block is then encrypted with the algorithm as it stands. The resultant ciphertext replaces P1 and P2. The same ciphertext is then encrypted again with the new subkeys, and the new ciphertext replaces P3 and P4. This continues, replacing the entire P-array and all the S-box entries. In all, the Blowfish encryption algorithm will run 521 times to generate all the subkeys - about 4KB of data is processed.

Because the P-array is 576 bits long, and the key bytes are XORed through all these 576 bits during the initialization, many implementations support key sizes up to 576 bits. The reason for that is a discrepancy between the original Blowfish description, which uses 448-bit keys, and its reference implementation, which uses 576-bit keys. The test vectors for verifying third party implementations were also produced with 576-bit keys. When asked which Blowfish version is the correct one, Bruce Schneier answered: "The test vectors should be used to determine the one true Blowfish".

Another opinion is that the 448 bits limit is present to ensure that every bit of every subkey depends on every bit of the key, as the last four values of the P-array don't affect every bit of the ciphertext. This point should be taken in consideration for implementations with a different number of rounds, as even though it increases security against an exhaustive attack, it weakens the security guaranteed by the algorithm. And given the slow initialization of the

cipher with each change of key, it is granted a natural protection against brute-force attacks, which doesn't really justify key sizes longer than 448 bits.

3. The role of encryption in Cybersecurity.

Encryption is a cybersecurity measure that protects private and personal data through the use of unique codes that scramble the data and make it impossible for intruders to read. Despite a data breach, encryption ensures that an institution's private data is safe, even when attackers get past the firewall.
Businesses are increasingly collecting a lot of private user data, so to prevent this data from getting into the hands of unauthorized agents, corporations need to ensure that they encrypt all the data in their possession.
The data encryption process is straightforward. An encryption key with a specific encryption algorithm is used to translate the plaintext data into unreadable data, also known as ciphertext. The scrambled data can only be decoded using the corresponding encryption key, so intruders will not be able to read the data when they get past the system security measures.
Secure Sockets Layer (SSL) is a data encryption mechanism that websites employ to protect vital user data. It prevents attackers from accessing sensitive user data that is moving to and from the website. The websites that have added SSL to secure their websites have a padlock symbol on their URLs and use "https" instead of "http" for their link address. SSL implementation assures the website's users that their online transactions are encrypted and sensitive user data is protected.

V. Cybersecurity Datasets for Deep Learning

NSL-KDD dataset
This dataset is proposed by Tavallaee and is recommended to solve some of the inherent problems of the KDD'99 dataset. Compared to the original KDD dataset, the NSL-KDD dataset has the following improvements: (1) it does not include redundant records, (2) it does not include duplicate records, (3) the number of selected records is organized as the percentage of records, and (4) the number of records is reasonable. Note that many papers on intrusion detection use both datasets together in performance evaluation (i.e., KDD Cup 1999 dataset and NSL-KDD dataset), and they typically find that the best results are found in the NSL-KDD dataset.
UNSW-NB15 dataset
This dataset is created by four tools, namely, IXIA PerfectStorm tool, Tcpdump tool, Argus tool, and Bro-IDS tool. These tools are used to create some types of attacks, including DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The UNSW-NB15 dataset contains approximately two million and 540,044 vectors with 49 features. In addition, Moustafa published a partition from this dataset which contains the training set (175,341 vectors) and the testing set (82,332 vectors).
CIC DoS dataset
This dataset contains application layer DoS attacks with 4 types of attacks using different tools. The CIC DoS dataset is proposed by Jazi which application layer DoS attacks are generally seen in high-volume (e.g., high-volume HTTP attacks generated using HULK (HTTP Unbearable Load King)) or low-volume variations (e.g., Low-volume DoS attacks). The High-volume HTTP attacks include DoS improved GET (Goldeneye), DDoS GET (ddossim), and DoS GET (hulk). The Low-volume HTTP attacks include slow-send body

(Slowhttptest), slow send body (RUDY), slow-send headers (Slowhttptest), slow send headers (Slowloris), and slow-read (Slowhttptest).
Bot-IoT dataset
This dataset contains more than 72.000.000 records, which includes DDoS, DoS, OS and Service Scan, Keylogging and Data exfiltration attacks. The Bot-IoT dataset is proposed by Koroniotis, which is new for the IoT environment compared to previous datasets. The authors employed the Node-red tool to simulate the network behavior of IoT devices. To link machine-to-machine (M2M) communications, the dataset uses the MQTT protocol, which is a lightweight communication protocol. However, there are five IoT scenarios used in the testbed, namely, weather station, smart fridge, motion activated lights, remotely activated garage door, and smart thermostat.

D. CASE STUDY

I. Application deep learning to Cybersecurity on the website in VietNam

Deep learning technology, a subset of machine learning algorithms (which is itself a subset of artificial intelligence algorithms), can predict and protect organizations from known and unknown cyberattacks in real-time while mitigating the problem of false positives. It is changing how organizations build and manage their cybersecurity stack. Many traditional solutions, as well, can only protect specific domains or operating systems (one vendor for Windows and Android, for example). A single platform that can handle any threat at any time is more viable. Before delving into deep learning and cybersecurity, it's important to identify why the cybersecurity model is broken.

From 2008 to 2018, the number of data breaches has doubled, from 636 to 1,244. Files and records exposed have also jumped from 35.7 million to 446.5 million over that same time period. The problem is getting worse, despite investment in cybersecurity increasing by 30%. Gartner has predicted that the market will be worth $248.6 billion by 2023, according to Statista. "There has been a huge growth in cybersecurity investment, but nothing has improved. In fact, it's got worse. The cybersecurity model is not working," stated Kaftzan.

Why?

1. Volume — more than 350,000 new malicious programs are created every day (mostly by machines). It is easy to modify existing malware and create a completely new cyber-attack. It is overwhelming.

2. A question of when not if — 67% of CIOs thought the possibility of their company experiencing a data breach or cyber-attack in 2018 was a given, according to a survey from the Ponemon Institute.

3. Cost — a big breach can cost an enterprise between $40-350 million.

4. The skills shortage in cyber — 69% of organizations say their cybersecurity teams are understaffed, while there will be as many as 3.5 million unfilled positions in the industry by 2021.

5. The complexity of cyber-attacks — the level of sophistication and complexity of cyber-attacks is increasing. AI-based malware and adversarial learning (using a neural network (DL or ML) to attack another neural network) are also beginning to threaten networks.

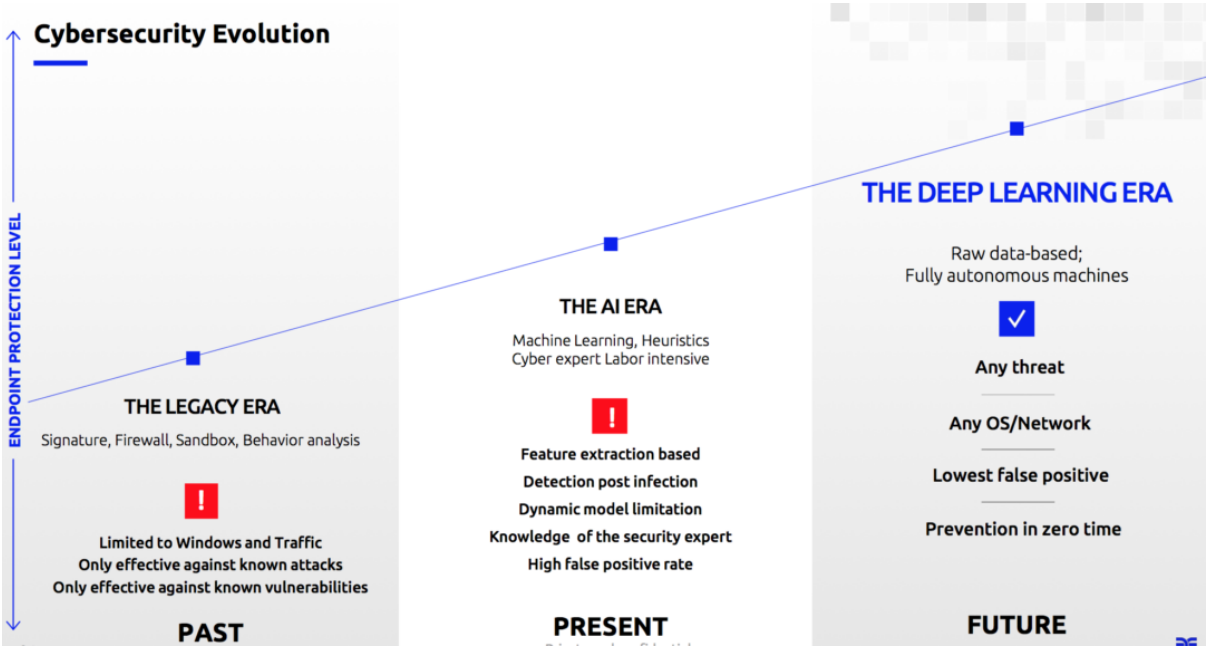


Figure DI-1: Cybersecurity evolution

The benefits of deep learning in cybersecurity

• Predictions of unknown threats.

• Zero-time prediction and detection.

• Zero-time classification.

• Works across any device, operating system, or file.

• Doesn't rely on the connection (edge deployment).

Growing prevalence of deep learning in real-world solutions

There is a growing prevalence of deep learning in real-world solutions, but the technology is lagging in cybersecurity:

• Computer vision: 98% deep learning, 2% traditional machine learning.

• Speech recognition: 80% deep learning, 20% traditional machine learning.

• Text understand 65% deep learning, 35% traditional machine learning.

• Cybersecurity: 2% deep learning, 98% traditional machine learning.

End-to-end deep learning for cybersecurity

An end-to-to-end deep learning framework can predict the mutations of existing malware and prevent them in real-time before the impact can be felt on a device or network. The algorithm is developed entirely on C/C++ and is optimized using NVIDIA's GPU training (before it is deployed on endpoints using regular CPUs).

Deep Instinct is pioneering deep learning in the cybersecurity space. But there are a number of challenges. The main one is that there are not enough experts that can build deep learning algorithms — they're all snapped up by the big companies, such as Baidu (speech recognition) and Google (NLP) after university.

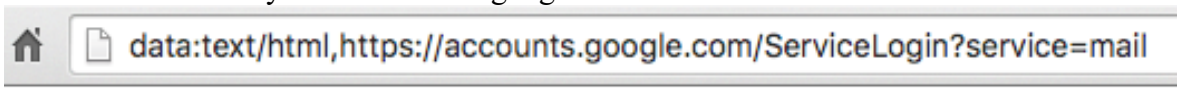
II. CASE STUDY: Website problems and Solution to solve it.

1. Website problem

1.1. The influence of phishing attack in website.

1.1.1. In Gmail

A new highly effective phishing technique targeting Gmail and other services has been gaining popularity during the past year among attackers. Over the past few weeks there have been reports of experienced technical users being hit by this. This attack is currently being used to target Gmail customers and is also targeting other services. The way the attack works is that an attacker will send an email to your Gmail account. That email may come from someone you know who has had their account hacked using this technique. It may also include something that looks like an image of an attachment you recognize from the sender. You click on the image, expecting Gmail to give you a preview of the attachment. Instead, a new tab opens up and you are prompted by Gmail to sign in again. You glance at the location bar and you see accounts.google.com in there. It looks like this….



You go ahead and sign in on a fully functional sign-in page that looks like this:

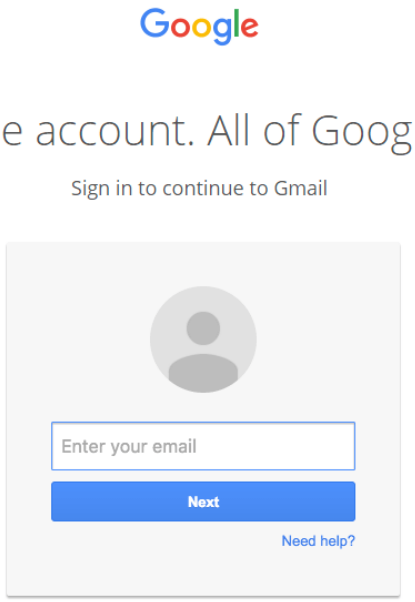
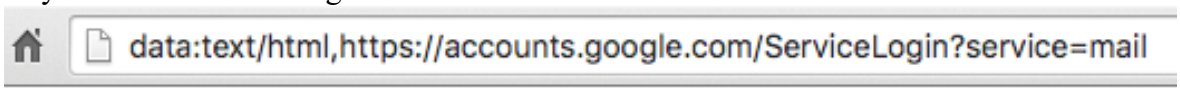


Figure 78: Gmail

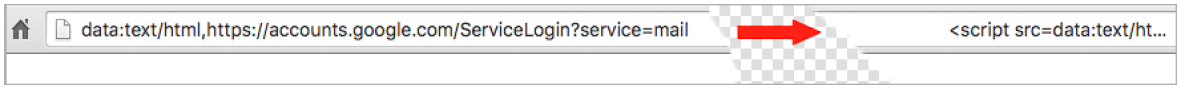
Once your complete sign-in, your account has been compromised. A commenter on Hacker News describes in clear terms what they experienced over the holiday break once they signed in to the fake page: "The attackers log in to your account immediately once they get the credentials, and they use one of your actual attachments, along with one of your actual subject lines, and send it to people in your contact list. For example, they went into one student's account, pulled an attachment with an athletic team practice schedule, generated the screenshot, and then paired that with a subject line that was tangentially related, and emailed it to the other members of the athletic team." The attackers signing into your account happens very quickly. It may be automated or they may have a team standing by to process accounts as they are compromised. Once they have access to your account, the attacker also has full access to all your emails including sent and received at this point and may download the whole lot. Now that they control your email address, they could also compromise a wide variety of other services that you use by using the password reset mechanism including other email accounts, any SaaS services you use and much more.
What I have described above is a phishing attack that is used to steal usernames and passwords on Gmail. It is being used right now with a high success rate. However, this technique can be used to steal credentials from many other platforms with many variations in the basic technique.

How to protect yourself against this phishing attack

You have always been told: "Check the location bar in your browser to make sure you are on the correct website before signing in. That will avoid phishing attacks that steal your username and password."
In the attack above, you did exactly that and saw 'accounts.google.com 'in the location bar, so you went ahead and signed in.



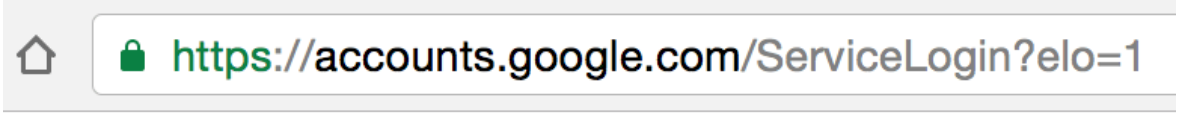
To protect yourself against this you need to change what you are checking in the location bar. This phishing technique uses something called a 'data URI' to include a complete file in the browser location bar. When you glance up at the browser location bar and see 'data: text/html….' that is actually a very long string of text. If you widen out the location bar it looks like this:



There is a lot of whitespace which I have removed. But on the far right you can see the beginning of what is a very large chunk of text. This is actually a file that opens in a new tab and creates a completely functional fake Gmail login page which sends your credentials to the attacker. As you can see on the far left of the browser location bar, instead of 'https' you have 'data: text/html,' followed by the usual 'https://accounts.google.com….'. If you aren't paying close attention you will ignore the 'data: text/html' preamble and assume the URL is safe. You are probably thinking you're too smart to fall for this. It turns out that this attack has caught, or almost caught several technical users who have either tweeted, blogged or commented about it. There is a specific reason why this is so effective that has to do with human perception. I describe that in the next section.

How to protect yourself

When you sign in to any service, check the browser location bar and verify the protocol, then verify the hostname. It should look like this in Chrome when signing into Gmail or Google:

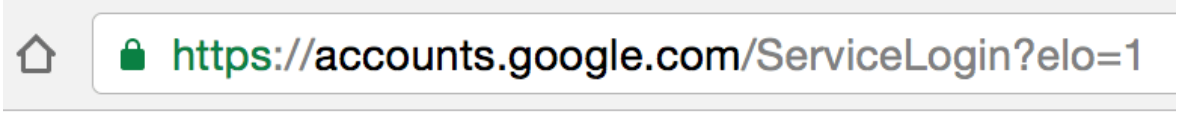


Make sure there is nothing before the hostname 'accounts.google.com' other than 'https://' and the lock symbol. You should also take special note of the green color and lock symbol that appears on the left. If you can't verify the protocol and verify the hostname, stop and consider what you just clicked on to get to that sign-in page.

Enable two factor authentications if it is available on every service that you use. GMail calls this "2- step verification" and you can find out how to enable it on this page. Enabling two factor authentication makes it much more difficult for an attacker to sign into a service that you use, even if they manage to steal your password using this technique. I would like to note that there is some discussion that indicates even two factor authentications may not protect against this attack. However, I have not seen a proof of concept, so I cannot confirm this. Why Google won't fix this and what they should do

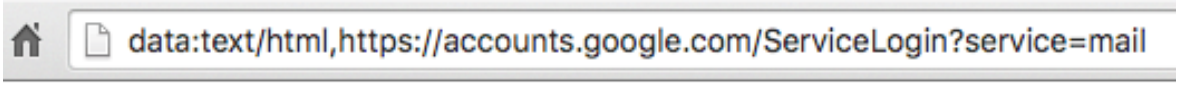
Google's response to a customer asking about this was as follows: "The address bar remains one of the few trusted UI components of the browsers and is the only one that can be relied upon as to what origin are the users currently visiting. If the users pay no attention to the address bar, phishing and spoofing attack are – obviously – trivial. Unfortunately, that's how the web works, and any fix that would to try to e.g. detect phishing pages based on their look would be easily by pass able in hundreds of ways. The data: URL part here is not that important as you could have a phishing on any http[s] page just as well."

This is likely a junior person within the organization based on the grammatical errors. I disagree with this response for a few reasons: Google have modified the behavior of the address bar in the past to show a green protocol color when a page is using HTTPS and a lock icon to indicate it is secure.



They also use a different way of displaying the protocol when a page is insecure, marking it red with a line through it:

During this attack, a user sees neither green nor red. They see ordinary black text:



That is why this attack is so effective. In user interface design and in human perception, elements that are connected by uniform visual properties are perceived as being more related than elements that are not connected. In this case the 'data: text/html' and the trusted hostname are the same color. That suggests to our perception that they're related and the 'data: text/html' part either doesn't matter or can be trusted. What Google needs to do in this case is change the way 'data: text/html' is displayed in the browser. There may be scenarios where this is safe, so they could use an amber color with a unique icon. That would alert our perception to a difference and we would examine it more closely. Update: How to

check if your account is already compromised. I've had two requests in the comments about this so I'm adding this section now. (at 9:39am Pacific time, 12:39am EST).
There is no sure way to check if your account has been compromised. If in doubt, change your password immediately. Changing your password every few months is good practice in general.
If you use Gmail, you can check your login activity to find out of someone else is signing into your account. Visit https://support.google.com/mail/answer/45938?hl=en for info. To use this feature, scroll to the bottom of your inbox and click "Details" (very small in the far lower right-hand corner of the screen). This will show you all currently active sessions as well as your recent login history. If you see active logins from unknown sources, you can force close them. If you see any logins in your history from places you don't know, you may have been hacked. There is a trustworthy site run by Troy Hunt who is a well-known security researcher where you can check if any of your email accounts have been part of a data leak. Troy's site is https://haveibeenpwned.com/ and it is well known in security circles. Simply enter your email address and hit the button. Troy aggregates data leaks into a database and gives you a way to look up your own email in that database to see if you have been part of a data breach. He also does a good job of actually verifying the data breaches he is sent.

Spread the word
I'll be sharing this on Facebook to create awareness among my own family and friends. This attack is incredibly effective at fooling even technical users for the reasons I have explained above. I have the sense that most ordinary users will be easy pickings. Please share this with the community to help create awareness and prevent this from having a wider impact.

Update: Official Statement from Google
This is an update at 11:30pm PST on Tuesday the 17th of January 2017. I was contacted by Aaron Stein from Google Communications. He has provided the following official statement from Google: "We're aware of this issue and continue to strengthen our defenses against it. We help protect users from phishing attacks in a variety of ways, including: machine learning based detection of phishing messages, Safe Browsing warnings that notify users of dangerous links in emails and browsers, preventing suspicious account sign-ins, and more. Users can also activate two-step verification for additional account protection."
I asked Aaron two follow-up questions: "Chrome 56 will include the text "Not secure" in the location bar on non-SSL websites where a page contains a password field or credit card input field. This is a fine example of a visual indication in the location bar that helps secure users. Are the Chrome dev team considering some visual indication in the browser location bar for data URI's? That would help defeat this attack because, currently, there is no visual indication of anything awry when viewing a phishing data URI. It's worth noting that the safe browsing system is currently unable to detect malicious data URI's because it is currently geared for traditional hostname-path URL's.
Second question: Emails that contain malicious data URI's are the attack vector in this case. Are the Gmail team considering any additional filtering or alerting related to data URI's as attachments in the Gmail web application? I think any guidance you can provide on the above two questions will go a long way to put Chrome and Gmail user's minds at ease."
He responded with: "I can't speak to things that aren't out yet, but *please* watch this space. Should have more to share soon". My thoughts on this response: I think this is a perfectly acceptable response from Google. To be clear, there are several teams within the Google organization that this affects: The Google Chrome browser team will be the ones who would implement any change in the location bar behavior when viewing a phishing data URI. The Gmail team would implement filtering and alerting within the Gmail application with a data

URI attachment is received with other associated phishing markers. The Google Safe Browsing team may add support for malicious data URI's in the GSB API and make that available to the Chrome browser team.
There may be other parts of the Google organization that touches including operations.
Asking Aaron to provide early guidance on how Google will mitigate this when it affects so many teams were a big ask, but I would be remiss if I didn't hit him with a couple of follow-up questions. The good news is that Google is aware of the issue and we have an official statement that indicates there will be something forthcoming in future releases of Chrome, Gmail and possibly other products that can help mitigate this.

1.1.2. In Facebook

When it comes to phishing, criminals put a lot of effort into making their attacks look legitimate, while putting pressure on their victims to take action. In today's post, we're going to examine a recent phishing attempt against me personally. This is an interesting attack, as it uses Google Translate, and targets multiple accounts in one go. Shortly after the New Year holiday, I received an email on my phone notifying me that my Google account had been accessed from a new Windows device. Since I didn't recall logging in via a new device, I decided to examine the email more thoroughly. I switched over to my laptop and logged into my personal Gmail account. One look at the sender address and my suspicions were confirmed. The email was a complete fake. The interesting thing is, the message looked much more convincing in its condensed state on my mobile device.

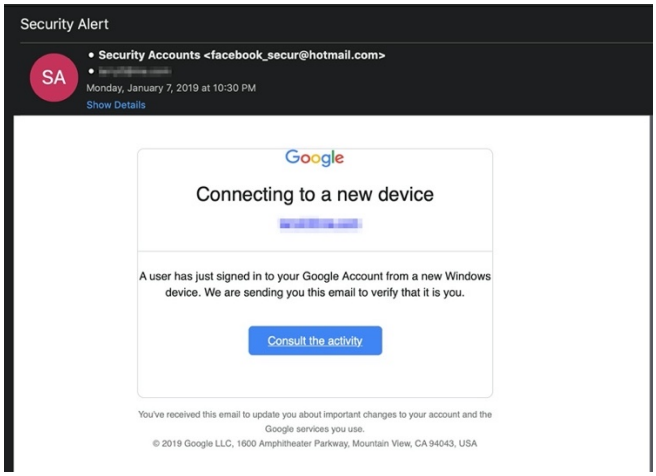


Figure 79: Account alert connects to new device

Random Emails
The message I received on January 7 was pretty basic. It alerted me to the fact that my Google account was accessed from a new device. While the condensed message on my mobile phone appeared legitimate, the full message when viewed on a computer has a number of problems.
First, the supposed security alert itself comes from a Hotmail account. Second, the entire address has nothing to do with Google. By using "facebook_secur", there is a chance a mobile user will assume the message came from Facebook's security team.
Taking advantage of known brand names is a common phishing trick, and it usually works if the victim isn't aware or paying attention. Criminals conducting phishing attacks want to throw people off their game, so they'll use fear, curiosity, or even false authority in order to make the victim take an action first, and question the situation later. When this happens, it

is entirely possible - expected, in some cases - that the victim isn't going to pay attention to little details that give the scam away. In my case, the attacker is using a mix of curiosity and fear. Fear that my account is compromised, and curiosity as to who did it.

The Attack

Clicking the embedded "Consult the activity" link brought me to the following page:

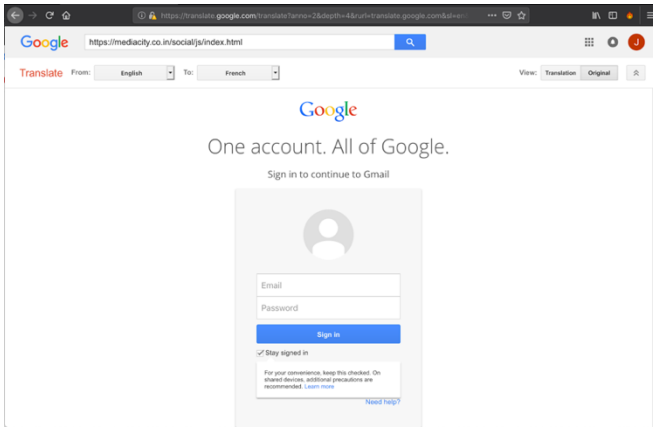


Figure 80 Consult the activity

Once the link is clicked, the page opens up to the form you see in the image above. This is the landing page. The criminal responsible for this phishing attack is doing something interesting with the landing page, because they're loading the malicious domain through Google Translate.

Using Google Translate does a number of things; it fills the URL (address) bar with lots of random text, but the most important thing visually is that the victim sees a legitimate Google domain. In some cases, this trick will help the criminal bypass endpoint defenses.

However, while this method of obfuscation might enjoy some success on mobile devices (the landing page is a near-perfect clone of Google's older login portal), it fails completely when viewed from a computer.

```
hxxps://translate.googleusercontent.com/translate_c?anno=2&depth=5&nv=1&rurl=translate.go
ogle.com&sl=en&sp=nmt4&tl=fr&u=hxxps://mediacity.co.in/social/js/index.html&xid=17259,1570
0022,15700124,15700186,15700191,15700201,15700237,15700242,15700248&usg=ALkJrhja
hTfiPwzWGQnU1YEBbrD706koDw
```

Side Note: Customers with Akamai's Enterprise Threat Protector (ETP) are shielded from this URL, as it already exists in our system. The domain, at the time of writing, has been compromised in order to be used for this phishing attack.

If you look at the image, you can clearly see the "mediacity" domain being translated. This should serve as an immediate red flag. For most people, this is where the attack would stop. They'll notice the fake login page and avoid entering their credentials. However, for those who do fall for the scam, the username and password entered will be collected and emailed to the attacker, triggering the second stage of this attack.

A Second Attack

Once the victim's credentials are sent to the attacker, something interesting happens. A second phishing attack starts. So, not only does the attacker target a victim's Google credentials but they attempt to hit victims twice, by forwarding them to a clone of Facebook's mobile login portal.

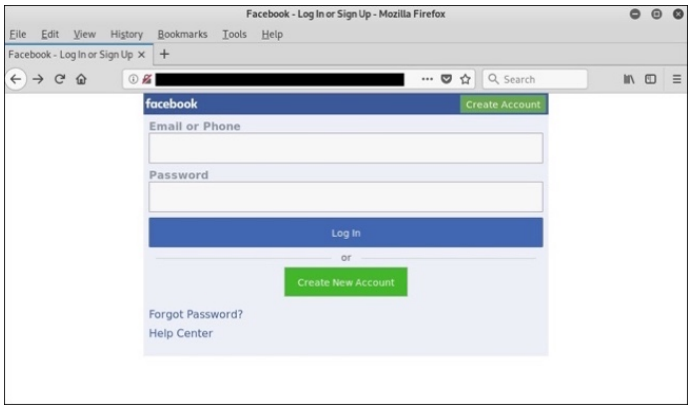


Figure 81: attack on Facebook

Since the Google email and landing page focused on mobile users, it only makes sense that the follow-on attack would do so as well. However, like the Google page, this Facebook landing page also uses an older version of the mobile login form. This suggests that the kit is old, and likely part of a widely circulated collection of kits commonly sold or traded on various underground forums.

The domain hosting the Facebook landing page is different from the domain hosting the Google one, but the two domains are linked via a script being used by the attacker. To give you an idea of what such a script looks like, here is the source code for kits similar to this one.

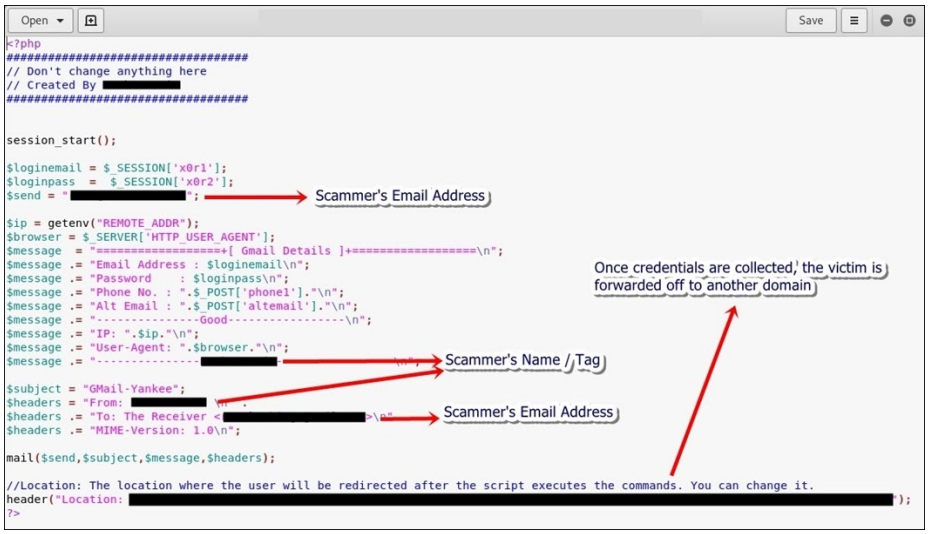


Figure 82: Script file in php

In this attack, once the Google credentials were recorded and emailed off to the attacker, the script above will format the message accordingly, and send the victim to the Facebook landing page. An example of what one of those emails look like is below on one of our lab systems.

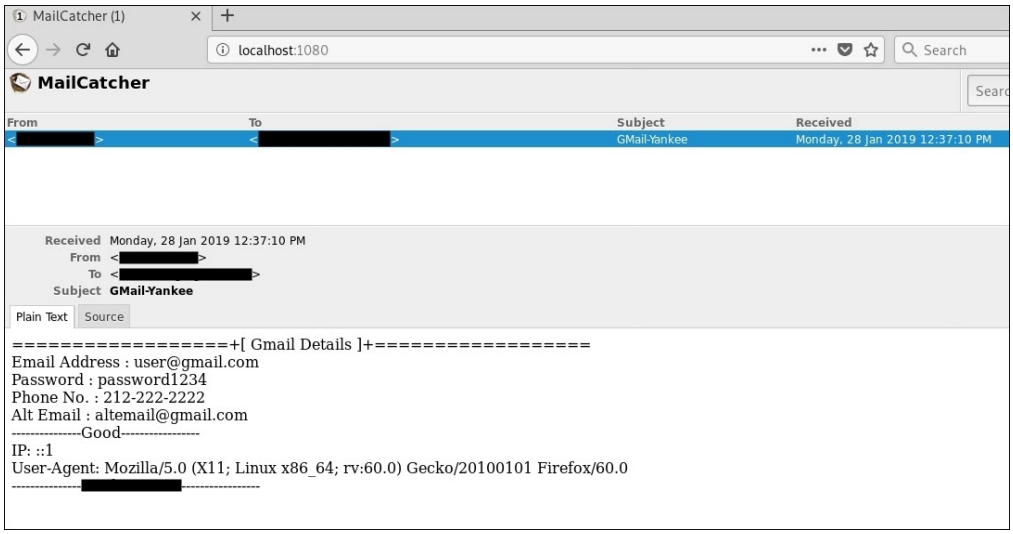


Figure 83: mail

The email records the victim's username and password, as well as other information including IP address and browser type. Some phishing kits will collect more information, such as location, and various levels of PII, which is usually shared or sold for use in credential stuffing attacks or additional phishing attacks.
It isn't every day that you see a phishing attack leverage Google Translate as a means of adding legitimacy and obfuscation on a mobile device. But it's highly uncommon to see such an attack target two brands in the same session. One interesting side note relates to the person driving these attacks, or at the least the author of the Facebook landing page - they linked it to their actual Facebook account, which is where the victim will land should they fall for the scam. Some phishing attacks are more sophisticated than others. In this case, the attack was easily spotted the moment I checked the message on my computer in addition to seeing it on my mobile device. However, other, more clever attacks fool thousands of people daily, even IT and Security professionals. The best defense is a good offense. That means taking your time and examining the message fully before taking any actions. Does the from address match what you're expecting? Does the message create a curious sense of urgency, fear, or authority, almost demanding you do something? If so, those are the messages to be suspicious of, and the ones most likely to result in compromised accounts.

1.1.3. In Bitcoin

Vulnerable wallets
There is a real vulnerability of Bitcoin wallets when it comes to hacking attacks and theft. A report by a team of researchers from Edinburgh University said they found weak spots in hardware wallets that can be exploited. According to the same research, even the heavily encrypted hardware wallets were still vulnerable due to that loophole.
Using malware, the scientists were able to intercept communication between the wallet and PCs. This security breach affects the privacy of Bitcoin users because their funds can easily be diverted to different accounts.

Hackers and cyber-attackers

The potential for a crippling attack directed at Bitcoin exchanges remains real. There have been significant attacks on exchanges before, but though Bitcoin's value slumped afterward, fears still abound of one that may completely cripple the popular cryptocurrency. We are not talking about an attack on the blockchain itself; that is almost a non-starter.

It is hacking major Bitcoin exchanges on the scale of Mt. Gox that I am thinking of. Reports circulated widely after the 2014 Mt. Gox heist indicates that hackers had been trying to get into the system for almost a year. When they did, they made off with 850,000 Bitcoins. At today's value, that would be $7.2 billion. Mt. Gox never recovered from the attack and filed for bankruptcy. Other major Exchanges like Bitfinex remain under threat, which is a security concern, too. Bitcoin is also threatened by Distributed Denial of Service (DDoS) attacks. A report by Imperva indicated that Bitcoin Exchanges had become favorites for DDoS attacks. The frequency is increasing, with Bitfinex, one of the largest exchanges, reporting that it had faced repeated DDoS attacks towards the end of 2017.

Selfishing mining

Bitcoin's continued use of proof-of-work consensus mechanism has another underlying threat. With some mining pools becoming powerful enough to command significant mining ratios, they may engage in selfish mining.

Also called block withholding, a pool may use their computational power to mine a block and then hide it from honest miners instead of broadcasting the new block to the network.

The selfish pool then attempts to find the second block while the rest grope in the dark. If the greedy miners manage to find a new block before the other miners, then broadcasting the two blocks makes the forked chain the longest. The selfish miners will be ahead of the other miners, getting all the rewards.

Such conspiracies, on a large scale, can be combined with the Sybil attack to cause considerable harm to mining because selfish miners can then use their power to invalidate transactions on the network.

Double spending

Although reinforcements have been instituted to mitigate this severe concern, fears still abound concerning this transaction risk to Bitcoin. Bitcoin is becoming increasingly sturdier against coordinated double-spends. However, some people might still be able to constitute attacks that would make them benefit from using the same coin twice in the same transaction. For instance, Bob purchases items from Alice and sends Alice x bitcoins.

At the same time, Bob executes a similar transaction to an address he controls using the same Bitcoins. Though Alice may believe that Bob has sent the money and may not bother to confirm, Bob's address may be credited with the transaction while Alice's won't get the contemplated transaction. Irreversibility then makes it pointless for Alice to get the transaction invalidated. And there is no recourse because Bitcoin is unregulated.

51 percent attacker

The so-called over 50 percent or 51 percent attack is a security concern for Bitcoin though not one that is easy to carry out. The increasing difficulty of mining Bitcoin has meant that miners get into pools to harmonize their computational power.

When a pool becomes too powerful that it can manage to command over 50 percent of the mining power, it then poses a threat to Bitcoin's network. If a group were to get this much power, then it could go on to manipulate transactions by either mining "invalid" blocks or double-spending. The use of ASICS mining rigs means a majority of miners can only do it through pools. Some of the pools have so much power that it can be misused. For instance, Antpool, the Chinese mining pool operated by Bitmain Tech., controls about 27 percent of the computational power. If it were to conspire with another pool, the combined force would

be dangerously close to 50 percent. Getting to that magic number would be a concern to Bitcoin users. However, genuine miners will always see the need to remain prudent. Fifty-one percent attacks are therefore unlikely to happen. The security concerns and risks facing Bitcoin are majorly related to the use of Bitcoin and not of the blockchain network. Most of them can, therefore, be remedied so as not to exacerbate problems associated with the cryptocurrency. All Bitcoin investors should be aware of these concerns and how they can affect investments.

1.2. What are the security risks?

Failure to cover cybersecurity basics

The common vulnerabilities and exploits used by attackers in the past year reveal that fundamental cybersecurity measures are lacking. Cyber criminals use less than a dozen vulnerabilities to hack into organizations and their systems, because they don't need more. The top 10 external vulnerabilities accounted for nearly 52% of all identified external vulnerabilities Thousands of vulnerabilities account for the other 48%. The top 10 internal vulnerabilities accounted for over 78% of all internal vulnerabilities during 2015. All 10 internal vulnerabilities are directly related to outdated patch levels on the target systems.

For example, something as simple as timely patching could have blocked 78% of internal vulnerabilities in the surveyed organizations. And the same goes for external security holes. Moreover, relying on antivirus as a single security layer and failing to encrypt data is an open invitation for attackers. It just screams: "open for hacking!". Not understanding what generates corporate cybersecurity risks. Companies often fail to understand "their vulnerability to attack, the value of their critical assets, and the profile or sophistication of potential attackers". This issue came up at the 2015 World Economic Forum and it will probably still be relevant for a few more years. Security risks are not always obvious. The categories below can provide some guidance for a deliberate effort to map and plan to mitigate them in the long term.

Figure 3. The likelihood of nine external threat vectors occurring
9 = most likely to 1 = least likely

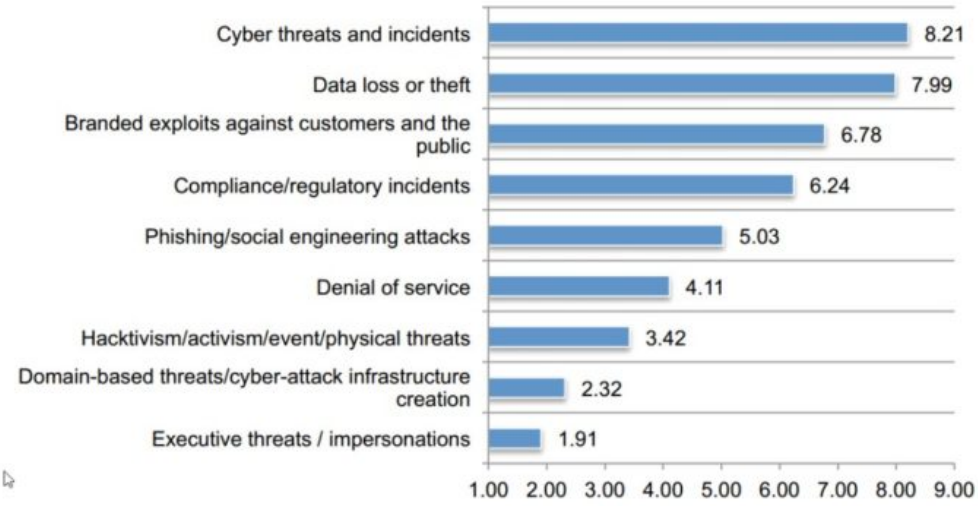


Figure 84: vulnerabilities threads

Technology isn't the only source for security risks. Psychological and sociological aspects are also involved. This is why company culture plays a major role in how it handles and perceives cybersecurity and its role.

Lack of a cybersecurity policy

Security standards are a must for any company that does business nowadays and wants to thrive at it. Cyber criminals aren't only targeting companies in the finance or tech sectors. They're threatening every single company out there. The increasing frequency of high-profile security breaches has made C-level management more aware of the matter. This is an important step, but one of many. External attacks are frequent and the financial costs of external attacks are significant. The 505 enterprises and financial institutions surveyed experienced an average of more than one cyber-attack each month and spent an average of almost $3.5 million annually to deal with attacks.

Not prioritizing the cybersecurity policy as an issue and not getting employees to engage with it is not something that companies nowadays can afford. This piece of advice shared in an article on Fortune.com is worth considering: Just as companies seek outside expertise for legal and financial matters, they should now be looking for experts in cybersecurity and data privacy. As part of their cybersecurity policy, companies should:

- Identify risks related to cybersecurity
- Establish cybersecurity governance
- Develop policies, procedures, and oversight processes
- Protect company networks and information
- Identify and address risks associated with remote access to client information and funds transfer requests
- Define and handle risks associated with vendors and other third parties
- Be able to detect unauthorized activity.
- Confusing compliance with cybersecurity

Another risk business has to deal with is the confusion between compliance and a cybersecurity policy. Ensuring compliance with company rules is not the equivalent of protecting the company against cyber-attacks. Unless the rules integrate a clear focus on security, of course. Enterprise risk management requires that every manager in the company has access to the parts of the security system that are relevant to them. Security is a company-wide responsibility, as our CEO always says. As a result, managers (and everyone else) should oversee how data flows through the system and know how to protect confidential information from leaking to cybercriminal infrastructure. Most companies are still not adequately prepared for – or even understand the risks faced: Only 37% of organizations have a cyber incident response plan. Clearly, there is plenty of work to be done here. The Carbon Lifeform – the weakest link: There are also other factors that can become corporate cybersecurity risks. They're the less technological kind. The human factor plays an important role in how strong (or weak) your company's information security defenses are. It turns out that people in higher positions, such as executive and management roles, are less prone to becoming malicious insiders. It's the lower-level employees who can weaken your security considerably. Be mindful of how you set and monitor their access levels. As you can see for this recent statistic, privilege abuse is the leading cause for data leakage determined by malicious insiders.

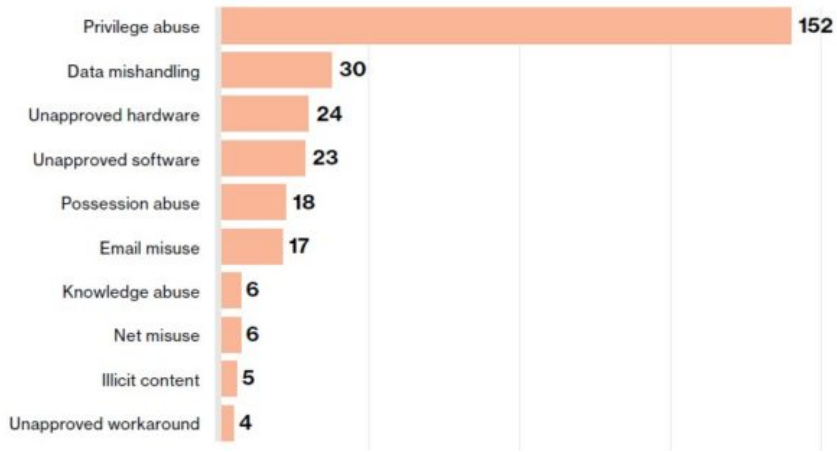


Figure 85: security risks

That is one more reason to add a cybersecurity policy to your company's approach, beyond a compliance checklist that you may already have in place. Protecting sensitive information is essential, and you need to look inside, as well as outside to map and mitigate potential threats. Bring your own device policy (BYOD) and the cloud

In the quest to providing your employees with better working conditions and a more flexible environment, you may have adopted the "Bring Your Own Device" policy. But have you considered the corporate cybersecurity risks you brought on by doing so?

One in five organizations suffered a mobile security breach, primarily driven by malware and malicious Wifi. Security threats to BYOD impose heavy burdens on organizations' IT resources (35%) and help desk workloads (27%). Despite increasing mobile security threats, data breaches and new regulations, only 30% of organizations are increasing security budgets for BYOD in the next 12 months. Meanwhile, 37% have no plans to change their security budgets. The bright side is that awareness on the matter of BYOD policies is increasing. When it comes to mobile devices, password protection is still the go-to solution. Overall, things seem to be going in the right direction with BYOD security. But, as with everything else, there is much more companies can do about it. Funding, talent and resources constraints We know that there are plenty of issues to consider when it comes to growing your business, keeping your advantages and planning for growth. So, budgets are tight and resources scarce. That's precisely one of the factors that incur corporate cybersecurity risks. Think of this security layer as your company's immune system. It needs funding and talent to prevent severe losses as a consequence of cyber-attacks.

A good approach would be to set reasonable expectations towards this objective and allocate the resources you can afford. It won't be easy, given the shortage of cybersecurity specialists, a phenomenon that's affecting the entire industry. No information security training. Employee training and awareness are critical to your company's safety. In fact, 50% of companies believe security training for both new and current employees is a priority, according to Dell's Protecting the organization against the unknown – A new generation of threats.

The specialists recommendation is to take a quick look at the most common file types that cyber attackers use to penetrate your system. This will tell you what types of actionable advice you could include in your employees' trainings on cybersecurity. The human filter can be a strength as well as a serious weakness. Educate your employees, and they might thank you for it. This training can be valuable for their private lives as well.

Lack of a recovery plan

Being prepared for a security attack means to have a thorough plan. This plan should include what can happen to prevent the cyber-attack, but also how to minimize the damage if is takes place. Unfortunately, the statistics reveal that companies are not ready to deal with such critical situations: Observing the trend of incidents supported since 2013, there has been little improvement in preparedness in 2015 there was a slight increase in organizations that were unprepared and had no formal plan to respond to incidents. Over the last three years, an average of 77% of organizations fall into this category, leaving only 23% having some capability to effectively respond.

If 77% of organizations lack a recovery plan, then maybe their resources would be better spent on preventive measures. This way, companies can detect the attack in its early stages, and the threats can be isolated and managed more effectively. But that doesn't eliminate the need for a recovery plan. There's no doubt that such a plan is critical for your response time and for resuming business activities.

Constantly evolving risks

There is one risk that you can't do much about: the polymorphism and stealthiness specific to current malware. Polymorphic malware is harmful, destructive or intrusive computer software such as a virus, worm, Trojan, or spyware. Its key asset is that it can change constantly, making it difficult for anti-malware programs to detect it. That is why you should take into account that your company might need an extra layer of protection, on top of the antivirus solution.

Your first line of defense should be a product that can act proactively to identify malware. It should be able to block access to malicious servers and stop data leakage. Part of this preventive layer's role is to also keep your system protected by patching vulnerabilities fast. As cyber risks increase and cyber-attacks become more aggressive, more extreme measures may become the norm. Such tactics include shutting down network segments or disconnecting specific computers from the Internet.

1.3. Why Websites Get Hacked

Every day, thousands of websites get hacked. WordPress sites make up a disproportionate percentage of those sites, since it powers over 30% of the web.

A lot of people think their sites are safe from attacks because they don't contain valuable and sensitive business information. However, there are plenty of other reasons why sites get hacked, such as: to spread malware, adding bandwidth to bot networks, which are often used for Denial of Service (DDoS) attacks, black-hat Search Engine Optimization (SEO), activism / hacktivism, just for practice and fun. The point is, no website is 100% exempt from the possibility of being targeted. Once it is online, it will be attacked.

1.4. Why is Website Security Important?

Web security is important to keeping hackers and cyber-thieves from accessing sensitive information. Without a proactive security strategy, businesses risk the spread and escalation of malware, attacks on other websites, networks, and other IT infrastructures. If a hacker is successful, attacks can spread from computer to computer, making it difficult to find the origin.

How Do I Make My Information on the Web More Secure?
The best line of defense on the web starts with user awareness. Avoid the risk of web security attacks and implement these 5 security tips:

1. Use Strong Passwords
It used to be that 3- or 4-character passwords would keep your information safe. However, as technology has advanced, so have the abilities and ways to crack passwords. Now, your passwords need at least 8 characters with a mixture of lower-case letters, capitals, numbers, and a special character like an exclamation mark is highly recommended.
Don't make your password a familiar phrase. It might be easy for you to remember the phrase "I love my children" but a password cracking software will break that in no time. A great idea is to take the first letter of a phrase you will remember and use those, like this:
"I love my children, John, Mary, and Phil" would be "ILm3c-JM&P". Never use a password twice. If someone hacks into any of your accounts then could access your bank accounts, your online purchase accounts, and any other important information.

2. Two-Factor Authorization.
A two-factor authorization comes in handy when a website recognizes a different IP address is used to login to a website like your Google account. You are immediately sent a text message with a phone number you registered with to confirm if it is you. If you didn't log in, you should immediately change the password to secure your account.

3. Always Use Secure Networks
When logging into financial and other crucial websites, look at the address bar before logging into your bank website and other sites on which you have personal information. If the address starts with HTTPS then you know it is secured (by the added "s"). If it doesn't, then you either have the wrong login page or it is possibly a spoof (fake) website.
Never click on a link in an email that seems suspicious. Better yet, never click on a link that comes from any crucial website such as your bank. Simply go to the website link you trust and have saved in your bookmarks to login, or call them. They will understand your caution.

4. Use More Than One Email Address
The email you use for your personal banking might be more secure if you use a different email for things like Facebook, Twitter, and even EBay. If someone were to hack into one then they would not automatically have access to the others.
5. Be Cautious About Posting Your Email Address Online
This is simply an invitation for spam if nothing else, but it also opens up a message of "Hey, hack me. Here's my email." Avoid posting your email address on forums, review sites, and message boards where spammers can easily pick up your address.

1.5. Secure website system in Viet Nam

Over 2,500 cyber-attacks on Vietnamese websites in Q3
More than 2,500 websites with Vietnamese domains were attacked in the third quarter, meaning Vietnam stands in 10th in terms of the largest number of cyber-attacks in the world. This number was tracked by the CyStack Attack Map system, developed by the Vietnam-based cyber security platform CyStack. It detected 127,367 attacks on websites around the world in the last three months, down slightly compared to the same period last year.

However, the number of attacks on Vietnamese websites increased by 113 per cent over a year earlier (from 1,183 to 2,523). The domain names on the receiving end of the most malicious attempts were .com, .vn and .net. “One website was being attacked every minute. When a website is successfully hacked, hackers can steal company data, change the website interface, plant malicious code or direct users to phishing sites,” the report said. CyStack research showed that 70 per cent of the attacks were targeted at websites using popular open source WordPress CMS (content management system), followed by Joomla and DNN. Vulnerabilities in websites using WordPress often stemmed from outdated plugins, themes or website versions, or users had downloaded plugins and themes from untrusted sources that had been installed with malicious code. In addition, the Linux operating system and Apache web server were also the most targeted by hackers with 64.8 per cent and 42.4 per cent, respectively. In the year to September, CyStack system recorded 450,000 website attacks around the world, of which 8,356 were made in Vietnam.

1.6. Website Security Framework

A cybersecurity framework is a comprehensive set of guidelines that help organizations define cybersecurity policies to assess their security posture and increase resilience in the face of cyberattacks. Cybersecurity frameworks formally define security controls, risk assessment methods, and appropriate safeguards to protect information systems and data from cyberthreats. While originally developed with large organizations and service providers in mind, cybersecurity frameworks can also be a valuable source of security best practices for medium and small businesses. Let’s have a look at the reasons for using a cybersecurity framework and see how you can find best-practice cybersecurity processes and actions to apply to web application security

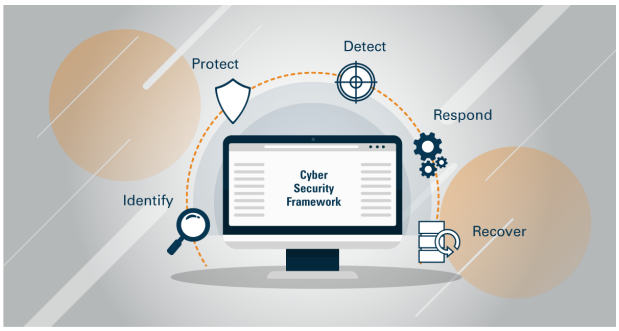


Figure 86: cybersecurity framework website

Why Cybersecurity Frameworks Were Developed

Cyberthreats have become a part of everyday life across the world, and a successful cyberattack, such as a denial of service or data breach, can have serious social, economic or even political consequences. Maintaining cybersecurity is now crucial for the operation of not only modern businesses and their supply chains, but also government institutions, markets, and entire economies. Data security and privacy are also high on the agenda, with the protection of personal data fast becoming a major concern for businesses, lawmakers, and the general public. As public and private organizations of all sizes were having to deal with the same cybersecurity events and challenges, it became clear that a common cybersecurity framework would benefit everyone by recommending best-practice policies, protective technologies, and specific activities related to information security and cybersecurity in general. Any organization’s internal policy will include at least some of those activities, and having a ready framework would be invaluable at the planning stage,

especially as organizations may lack the resources or technical competences to design their own policies from scratch.

Commonly Used Cybersecurity Frameworks

A cybersecurity framework can be any document that defines procedures and goals to guide more detailed cybersecurity policies. Existing documents that contain cybersecurity guidelines include: The NIST Cybersecurity Framework: Developed by the National Institute of Standards and Technology, this is probably the most widely used document for cybersecurity policy and planning.

- ISO 27001 Information Security Management: The International Organization for Standardization's guidelines for information security management systems (ISMS).
- CIS Critical Security Controls for Effective Cyber Defense (CIS Controls): A framework of prioritized actions to protect organizations from known cyberthreats.

Risk management frameworks: Documents such as NIST's Risk Management Framework (NIST SP 800-37 Rev. 2) or the ISO 27005:2018 standard (Information Security Risk Management) focus on risk management strategies, including risks related to cybersecurity. Industry-specific frameworks: Many industries have their own security standards that are required or recommended for these sectors, such as PCI DSS for electronic payment processing, HIPAA rules for healthcare, or COBIT for IT management and governance.

A Closer Look at the NIST Cybersecurity Framework

In 2013, a presidential executive order was issued in the United States, calling for a standardized cybersecurity framework that would describe and structure activities related to cybersecurity. In response to this, the NIST developed the Framework for Improving Critical Infrastructure Cybersecurity, commonly called the NIST Cybersecurity Framework. It is a comprehensive policy document intended to help organizations better manage and reduce cybersecurity risk and to facilitate communication related to risk and cybersecurity management. While the CSF was initially intended for companies managing critical infrastructure in the US private sector, it is widely used by public and private organizations of all sizes. The NIST CSF is divided into three main components to assist adoption by organizations: Framework core: This is the main informational part of the document, defining common activities and outcomes related to cybersecurity. Core information is divided into functions, categories, and subcategories. Framework profile: A subset of core categories and subcategories that an organization has chosen to apply based on its needs and risk assessments. Implementation tiers: A set of implementation levels intended to help organizations define and communicate their management approach and identified level of risk is their specific business environment. The framework core provides a clear structure of cybersecurity management processes, with five main functions: Identify, Protect, Detect, Respond, and Recover. For each function, multiple categories and subcategories are defined, and organizations can pick and mix to put together a set of items corresponding to their individual risks, requirements, and expected outcomes. Functions and categories have unique identifiers, so for example Asset Management within the Identify function is ID.AM, and Response Planning within the Response function is RS.RP. Each category includes a number of subcategories corresponding to appropriate activities, this time with numerical identifiers for subcategories. For example, subcategory Detection processes are tested under the Detection Processes category and Detect function is identified as DE. DP-3. Subcategories are accompanied by

informative references to the relevant sections of standards documents, allowing quick access to normative guidelines for each action.

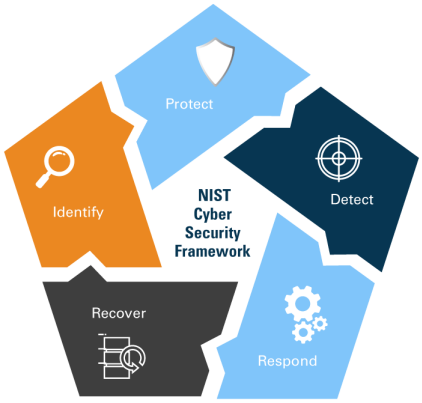


Figure 87: NIST cyber security Framework

How to Apply the NIST framework to Web Application Security

By its very nature, the NIST CSF has an extremely broad scope and covers far more activities than most organizations are going to need. To apply the framework to web application security, you can start by analyzing each of the five functions in the context of your existing and planned security activities and risk management processes. Then, you can select the categories and subcategories relevant to your specific needs and use them as the backbone of your own security policy to ensure you will cover all the required cybersecurity activities. For basic web application security, a skeleton cybersecurity policy would include at least the following subcategories for each function:

Identify:
ID.AM-2: Software platforms and applications within the organization are inventoried
ID.RA-1: Asset vulnerabilities are identified and documented

Protect:
PR.AC-4: Access permissions and authorizations are managed, incorporating the principles of least privilege and separation of duties
PR.DS-2: Data-in-transit is protected
PR. IP-10: Response and recovery plans are tested

Detect:
DE.AE-2: Detected events are analyzed to understand attack targets and methods
DE.CM-8: Vulnerability scans are performed

Respond:
RS.RP-1: Response plan is executed during or after an incident
RS.AN-1: Notifications from detection systems are investigated
Recover:
RC.RP-1: Recovery plan is executed during or after a cybersecurity incident
RC.CO-3: Recovery activities are communicated to internal and external stakeholders as well as executive and management teams

2. The solution to solve the problem in Website

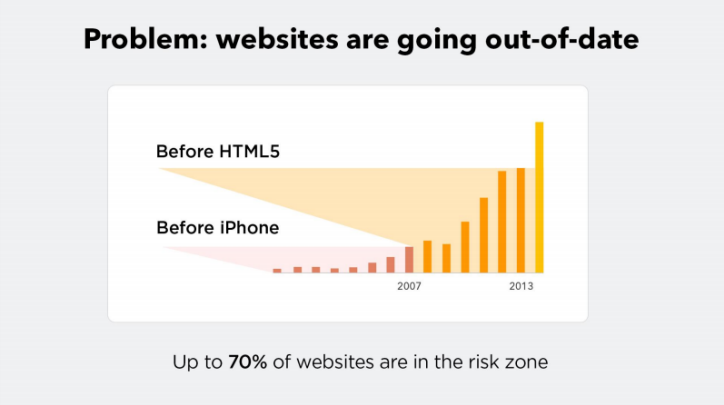


Figure 88: The data of websites in risk zone

The use of artificial intelligence by businesses is not a new concept. Anyone who has used Facebook, Google, Netflix, or Amazon is well aware of the tailored approach these companies use based on customer habits. And the use of customer driven data is not confined to high-tech companies alone. Businesses of all types apply machine learning to help drive a wide variety of decisions on marketing, product development, security, and even email spam filters.

One of the latest advances in artificial intelligence is deep learning. Deep learning uses neural networks to process enormous sets of data for diverse types of applications. For example, deep learning is used by autonomous cars for navigation, by security teams for video processing, and even to predict the result of court cases. All very impressive uses of deep learning, but can it solve more practical business problems? The answer is yes, of course, but like any tool deep learning must be used properly to be most effective.

Use Relevant and Clean Data

It seems obvious, but the first step to using machine learning of any type, including deep learning, is a good set of data. For example, if you want to predict the purchasing habit of 20 to 30-year-old men for your latest cologne, don't use data from five years ago that includes women customers.

Consider the Context

Using the cologne example, maybe the data you use shows a dramatic increase in the number of sales within the last year due to a promotion. Deep learning may over-predict the amount of cologne to market to existing customers based on raw data alone, if you ignore the context.

Continuously Revise Your Algorithm

Like any engineering tool, deep learning algorithms need to be revised and updated to reflect your business model. Plan for the long term and consistently upgrade hardware and software to get the most out artificial intelligence. Regular maintenance is key for top performance.

Use the Right Tools for the Job

The software used to develop deep learning solutions continues to evolve very rapidly. Finding the right combination of frameworks, tools, and libraries can be a real challenge. Make sure you have a viable plan for keeping all of the various modules up to date and interoperating properly and you will spare your data scientists a lot of headaches.

Bright Computing has the software you need to set your business up with flexible and robust machine learning solutions designed to meet your specific needs. Our software helps to turn your data into action.

E. CONCLUSION AND DEVELOPMENT

I. Conclusion

After studying and perform this project about "Deep learning in cybersecurity", we did the project and gain knowledge such as

- Understanding deep learning and cybersecurity concepts
- Know types of methods deep learning used in cybersecurity.
- Study the algorithm in each method
- Know how to solve case study
- Understand some methods and techniques in cybersecurity
- The advantages of using deep learning in cybersecurity, and know the future of cybersecurity is deep learning
- Know the application of deep learning in cybersecurity

Besides that, my group also gain a lot of useful knowledge and the power of its After studying this project, we gain a lot of useful things and achieve our favorite topic. It's still existed some obstacles, but I will base on this project and learn more in the future to have real project

II. Development

"The future of cybersecurity will have a heavy focus on using artificial intelligence (AI) to secure devices and systems in the increasingly connected world. With the internet of things (IoT) and connected devices proliferating at such an incredible rate, the ways in which we leave ourselves exposed to potential cyber-attacks are also increasing. Legacy systems simply do not have the capabilities to keep up with the evolving security threats, and relying solely on human oversight would prove woefully inadequate. Capable automated systems that can monitor, detect, manage, and prevent cyber-attacks in real time will be what drives cybersecurity going forward." Cybersecurity Ventures predicts there will be 3.5 million unfilled cybersecurity jobs by 2021, up from 1 million positions in 2014. As the labor crisis in our field worsens, AI and machine learning hold out great promise for eliminating some positions and increasing productivity by an order of magnitude for others. But the rub is that the world is facing a shortage of cybersecurity experts who know how to effectively evaluate, deploy and make the best use of these advanced technologies. AI and machine learning are not inherently cybersecurity technologies. So, there's going to be a big learning curve for organizations that adapt them for cyberdefense. There has been a myriad of groundbreaking cybersecurity solutions brought to market over the past decade and they haven't reduced the worker shortage -- which remains our biggest problem.

F. REFERENCES

Cyber Security Goals. (n.d.). Retrieved from javatpoint: https://www.javatpoint.com/cyber-security-goals

The Evolution of Deep Learning . (n.d.). Retrieved from medium: https://towardsdatascience.com/the-deep-history-of-deep-learning-3bebeb810fb2

Polyakov, A. (n.d.). *Machine Learning for Cybersecurity 101* . Retrieved from Dzone: https://dzone.com/articles/machine-learning-for-cybersecurity-101

Difference between classification and clustering in data mining. (n.d.). Retrieved from https://intellipaat.com/community/317/difference-between-classification-and-clustering-in-data-mining

Cook, K. (n.d.). *100+ Machine Learning Resources for Cybersecurity to Protect us from Cyberattacks*. Retrieved from houseofbots: https://www.houseofbots.com/news-detail/3683-4-100-Plus-machine-learning-resources-for-cybersecurity-to-protect-us-from-cyberattacks

Polyakov, A. (n.d.). *Machine Learning for Cybersecurity 101*. Retrieved from medium: https://towardsdatascience.com/machine-learning-for-cybersecurity-101-7822b802790b

Polyakov, A. (n.d.). *Machine Learning for Cybersecurity 101* . Retrieved from medium: https://towardsdatascience.com/machine-learning-for-cybersecurity-101-7822b802790b

Daly, C. (2018, 4 18). *Deep Learning Is The Future Of Cybersecurity* . Retrieved from AI business: https://aibusiness.com/document.asp?doc_id=760570&site=aibusiness

Corrales, J. C. (n.d.). *DEEP LEARNING IN CYBERSECURITY: THE DEFINITIVE TOOL*. Retrieved from buguroo: https://www.buguroo.com/en/blog/deep-learning-in-cybersecurity-the-definitive-tool

Mohamed , H., & Mohamed, H. (2012). *Artificial Neural Network*. Retrieved from sciencedirect: https://www.sciencedirect.com/topics/physics-and-astronomy/artificial-neural-network

Wikipedia. (n.d.). *Neural Turing machine* . Retrieved from wikipedia: https://en.wikipedia.org/wiki/Neural_Turing_machine

Convolutional Neural Networks for Visual Recognition. (n.d.). Retrieved from github: https://cs231n.github.io/neural-networks-1/

Jonnalagadda, V. K. (2018, December 6). *Sparse, Stacked and Variational Autoencoder*. Retrieved from medium: https://medium.com/@venkatakrishna.jonnalagadda/sparse-stacked-and-variational-autoencoder-efe5bfe73b64

Jonnalagadda, V. K. (2018, December 6). *Sparse, Stacked and Variational Autoencoder*. Retrieved from medium: https://medium.com/@venkatakrishna.jonnalagadda/sparse-stacked-and-variational-autoencoder-efe5bfe73b64

Mire, S. (2019 , June 26). *What's The Future of Cybersecurity? 39 Experts Share Their Insights*. Retrieved from disruptordaily: https://www.disruptordaily.com/future-of-cybersecurity/

Gibbons, L. (2018, July 17). *How to Use Deep Learning to Solve Your Business Problem* . Retrieved from brightcomputing: https://www.brightcomputing.com/blog/how-to-use-deep-learning-to-solve-your-business-problem

(2014, February 12). Retrieved from https://www.nist.gov/system/files/documents/cyberframework/cybersecurity-framework-021214.pdf

Banach, Z. (2020 , January 31). *Using a Cybersecurity Framework for Web Application Security*. Retrieved from Netsparker: https://www.netsparker.com/blog/web-security/cybersecurity-framework-web-application-security/

Wikipedia. (2020, June 23). *Neural Turing machine*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Neural_Turing_machine

Cyberinsiders. (n.d.). *A Brief History of Cybersecurity*. Retrieved from cybersecurity insiders: https://www.cybersecurity-insiders.com/a-brief-history-of-cybersecurity/

Wikipedia. (2020, June 21). *Computer_security*. Retrieved from wikipedia: https://en.wikipedia.org/wiki/Computer_security

Fortinet. (n.d.). *The Evolution of Cybersecurity*. Retrieved from fortinet: https://training.fortinet.com/local/staticpage/view.php?page=library_the-evolution-of-cybersecurity

Techmedics. (2019, October 17). *Techmedics*. Retrieved from 4 Real-Life Examples of Cyber Attacks: https://www.techmedics.com/4-real-life-examples-of-cyber-attacks/

Ismail, N. (2020, March 25). *Pioneering deep learning in the cyber security space: the new standard?* Retrieved from information-age: https://www.information-age.com/pioneering-deep-learning-cyber-security-new-standard-123488524/

Ismail, N. (2020, March 25). *Pioneering deep learning in the cyber security space: the new standard?* Retrieved from information-age: https://www.information-age.com/pioneering-deep-learning-cyber-security-new-standard-123488524/

Cicala, F. (2018, May 28). *Differentiable Neural Computers: An Overview*. Retrieved from Medium: https://towardsdatascience.com/rps-intro-to-differentiable-neural-computers-e6640b5aa73a