On Perfectly Secure Cryptography with Adversarial Neural Cryptography

By

Sylla Mohamed (mohamed.sylla@aims-senegal.org)
African Institute for Mathematical Sciences (AIMS), Senegal

Supervised by: Pr. Djiby SOW

Universite Cheikh Anta Diop, Senegal

Tutor : Michel Seck

February 5, 2020

Submitted in partial fulfilment of the requirements for the award of Master of Science in Mathematical Sciences at AIMS Senegal

AIMS | African Institute for Mathematical Sciences SENEGAL

DECLARATION

This work was carried out at AIMS Senegal in partial fulfilment of the requirements for a Master of Science Degree.

I hereby declare that except where due acknowledgement is made, that the work contained in this essay is my original work and that any work done by others or by myself previously has been acknowledged and referenced accordingly.

Student: Mohamed Sylla, January 23, 2020 [illegible]

Supervisor: Djiby Sow

ACKNOWLEDGEMENTS

First of all, I would like to express my thanks to Allah for all his power, grace and mercy without whom this research Project would never have been possible.

I would like to thank AIMS-SENEGAL (African Institute for Mathematical Sciences) for giving us the opportunity to be in the AIMS network and for believing in the potential of young people to develop Africa. Also thank you for all the material and financial resources that have enabled us to carry out this project under good conditions.

I would like to thank in particular Professor Djiby Sow, at Universite Cheikh Anta Diop de Dakar - Senegal (UCAD), who made this work possible. I thank him for his training and guidance. His availability, expert advice and support in providing me with all the tools necessary for this work have been invaluable. I will be eternally grateful to him for this extraordinary collaboration and for his honesty.

I would also like to thank my tutor, Mr. Michel Seck, who has never ceased to help, guide and advise me, for his availability and for his support for the smooth running of the research project.

I would like to thank Ms Nandi Lesli for her advice, her willingness to enlighten me in my work and her support.

I would also like to thank all the staff of AIMS Senegal for their invaluable help with advice, encouragement and unforgettable moments. I thank all my friends who gave me a good time. To all the SYLLA family for their encouragements, their unfailing support, their prayers, their advices, their encouragements, their love (Aminata Traoré, Seydou Thiam). To my brothers and friends:Nneka Okolo, Sawadogo Ibrahim, Karim Marche, Daimana Emanuel, Joana, Joy, Olyne, Khadim Sene, Aziz Drama, To all those who knew me and all those who loved me; To all those who have left me unforgettable memories... Thank you very much!!

DEDICATION

Thank you to the Almighty God Almighty "Allah" who gave me the courage, strength and patience to carry out this work. To the one who told me the right way by reminding me that will is always the key to success. Thank you Mummy To the one who without him I wouldn't be much without him. Thank you, Dad.

I would like to dedicate this work in particular to the spirit of my father because it who taught us to be honest and behave well. May God Almighty protect and bless him. And also to my mother, who worked for my success, through her love, support, all the sacrifices made and her precious advice, for all her help and presence in my life, receive through this work as modest as him, the expression of my feelings and my eternal gratitude. I would like to express my gratitude to all my friends, family and colleagues who have always supported and encouraged me in carrying out this work. Finally I thank my tutor Michel Seck.

Abstract

Nowadays the protection of communication is very important for any company. Several means and techniques have been developed to secure communication or data. With the exponential advance of artificial intelligence, which has many uses in almost every field of activity, it would be interesting to know how we can use artificial intelligence in cryptography to secure our communication or data, in other words, how artificial neural networks can use cryptographic algorithms to break or build a secure cryptosystem.

In this work, we review how artificial neural networks can learn to communicate an encryption scheme with certain instances of the One Time Pad (OTP) to secure communication,and we explain In which conditions artificial agents can learn to communicate securely.

Contents

1. Introduction

Machine learning is one of the fields of artificial intelligence that has greatly revolutionized the world of technology. However, the machine learning in Cryptography is not a new subject, Cryptography and the machine learning have many things in common (ALANI), such as the processing of large volumes of data and search spaces (C. Dwork, 2014). In 1991, Ronald Rivest (Rivest), one of the founders of RSA cryptography, gave a lecture on cryptography and the machine learning in ASIACYPT'1991. He discussed the differences and similarities between machine learning and cryptography. He also discussed the impact of cryptography on the machine learning and vice versa. Since this conference people are massively interested in this field and try to understand how to take advantage of this machine learning techniques in cryptography.

Machine learning and Cryptanalysis have much more in common than machine learning and cryptography. This can be explained by the fact that they have a common objective which is to search in a large research space. So the objective of cryptanalysis is to find the secret key to decipher the message, whereas the objective of the machine learning is to find an acceptable solution in a space solution.

1.1 Motivation

Nowadays data security is at the heart of all activities so it would be necessary to put in place a system to protect us from unauthorized third-party exchanges. So several techniques and methods are developed for data protection. Given the exponential evolution of artificial intelligence which has many applications than it would be interesting to know if artificial agent can communicate securely without a third party being able to understand the communication.

1.2 Problem Statement

Recently (Graves, 2013) and (Yu and Tsai) shown in their work the usefulness of machine learning in the field of cryptography such as facial recognition, speech recognition. The security of data or exchange of data is important in cryptography. It is important to know if we can build an artificial agent capable of communicating with another artificial agent a secure way without an adversary intervening and being able to understand. In other words, the problem is:

Can Neural networks learn to communicate securely over an unsecured channel using the One Time Pad algorithm?

1.3 Organization of work

The work is organized as follows:

1. In the first chapter we present some overview in cryptography. We recall the basics concept of encryption schemes and some notion on perfect secrecy.
2. In chapter 2, explain some concepts of Artificial Neural Network,and some model of neural like Multi-Layer perceptron.
3. In the last chapter, we review on this part how Neural Network can learn some instances of the One Time Pad, we review the original model ANC proposed by Abadi and Andersen and we explain the improvement of the ANC model call CPA-ANC propose by Murilo Coutinho.

2. Overview In Cryptography

2.1 Introduction

Cryptography is the art of hiding a message or clear information and finally making it incomprehensible to anyone who does not have the secret key. In other words, the fundamental objective of cryptography is to allow two people to communicate safely on an unreliable channel.

The message transmitted on the communication channel in such a way that an opponent, a third person, who has access to the information circulating on the communication channel, cannot understand what is being exchanged, in other words, guarantee the confidentiality, integrity , authentification of this information.

2.2 Basics Concept Of Cryptography

2.2.1 Cryptology

Cryptology resides in ancient Greece, it is a word composed of the element : ≪kryptos ≫ which means "hide" and ≪ logos ≫ which means "word". Cryptology has been used for many years to provide communications (military etc.). We can cite the example of the famous Roman emperor Julius Caesar who used an encryption algorithm to protect messages to his soldiers.

In (crypto) cryptology has two main components such that cryptography and cryptanalysis. Cryptography refers to all methods used to encrypt a message and finally make it incomprehensible to a third party or system. In other words, to ensure the safety and security of communications. Cryptanalysis is a method of decoding a cipher message. The objective is to find the message clearly without knowing the type of encryption algorithm used.

2.2.1.1 Cryptography

Cryptography is one of the disciplines of cryptology that focuses on protecting messages (ensuring confidentiality, authenticity, and integrity,Non-repudiation) often using secrets or keys(A. Menezes and Vanstone., 1992). It is different from steganography, which makes a message go unnoticed in another message, while cryptography makes a message unintelligible to anyone other than the rightful one.The word cryptography is a generic term referring to all the techniques used to encrypt messages, i.e. to make them unintelligible without specific action. The verb encrypts is sometimes used.

2.2.1.2 Cryptanalysis

Cryptanalysis corresponds to the methods used to analyze encrypted messages and "break" the cryptographic protection of these messages, i.e. break the cryptographic algorithm used, in order to find the clear information without knowing the decryption key.

2.3 Services of the Cryptography

The main goal of cryptography is to obtain secure communications over insecure channels. This goal is classically divided into four (4) services for security: confidentiality, data integrity, authentication (entities and data origin) and non-repudiation (digital signature).

2.3.1 Confidentiality

Confidentiality (secrecy) is a service used to ensuring that no one can read the message except the intended receiver. Data are kept secret from those without the proper credentials, even if that data travels through an insecure medium.

2.3.2 Integrity

This service allows us to guarantee that the content of the original message or the communication exchanged on the transmission channel has not been modified. Data integrity is a service which addresses the unauthorized alteration of data.

2.3.3 Authencity

Authentication is a service related to identification that ensures the identity of two entities that engage in communication or information exchange. And we can divide it into two parts, entity. Authentication and data. Authenticating the origin of the data implicitly ensures the integrity of. The data (because if a message is modified, the source changes).

2.3.4 Non-Repudiation

Non-Repudiation Allow to ensure that the signature that sends the message or data transmits on the communication channel cannot deny its signature. This ensures security. Non-repudiation is a service which prevents an entity from denying previous commitments or actions.

2.4 Encryption Schemes

An encryption scheme or cryptosystem (Buchmann, 2002) is a tuple ($\mathcal{A}$,$\mathcal{C}$,$\mathcal{P}$,$\mathcal{K}$,$\mathcal{E}$,$\mathcal{D}$) with the following properties:

- $\mathcal{A}$ denotes a finite set called the alphabet of definition. $\mathcal{A}^*$ denotes the set of all finite words over $\mathcal{A}$
- $\mathcal{P}$ denotes a set called the message space . An element of $\mathcal{P}$ is called a plaintext.
- $\mathcal{C}$ is a set. It is called the ciphertext space. Its elements are called ciphertext.
- $\mathcal{K}$ is a set. It is called the key space . Its elements are called keys.
- $\mathcal{E} = E_k : k \in \mathcal{K}$ is a family of functions $E_k : \mathcal{P} \to \mathcal{C}$. Its elements are called encryption functions.
- $\mathcal{D}$=D_k:$k \in K$ is a family of functions D_k: $\mathcal{P} \to C$. Its elements are called decryption functions.
- For each $e \in K$, there is $d \in K$ such that $D_d(E_e(p)) = p$ for all $p \in \mathcal{P}$.

Recall: The element (e, d) of $K \times K$ for which there exist two functions $E_e : \mathcal{P} \to \mathcal{C}$ and $D_d : \mathcal{C} \to \mathcal{P}$ such that $D_d o E_e(m) = m \in \mathcal{P}$, with a high probability are called keypair.

1. For a keypair (e, d)
   (a) the function E_e is injective (let invertible) and is called the encryption function.
   (b) the function D_d is called the decryption function.

In practice, for a secure communication over an insecure channel between Bob and Alice, the sender (Alice) encrypt a plain text (m) by computing the cipher text $y = E_e(m)$, and the receiver (Bob) decrypts the cipher text y by computing the plaintext $m = D_d(y) = D_d o E_e(m)$ (with a high probability). The decryption key for the encryption key e is $d = e$. This is, however, not true for every cryptosystem. To summarize encryption scheme is defined by three algorithms:

1. The key-generation algorithm is a probabilistic algorithm that output a key k chosen according to some distribution that is determined by the scheme.
2. The encryption algorithm E_n takes as input a key k generated and a plaintext m is a message and output a ciphertext C, denote by:

   $E_{nc_k}(m)$ which means that the encryption of the plaintext (m) using the key k.
3. The decryption algorithm D_{ec} takes as input a key k and a ciphertext C and output a plaintext m. Denoted the decryption of the ciphertext C using the key k it means $D_{ec_k}(m)$

2.5 Types of Encryption Schemes

To protect our communication and data exchange several cryptographic encryption methods have been implemented to ensure the security of the data exchanged,like Classical cryptography describing the period before computers were used. The processing was based on the letter and characters of a natural language(Caesar cipher) (V., 2011).

The methods used nowadays are more complex, but the principle remains the same, the fundamental difference is that modern methods (use of a computer) directly manipulate bits. Unlike the old methods that operated on alphabetic characters.

For two entities to communicate, it was necessary to establish certain conditions between its entities i.e. the secret key that is shared between these two and the algorithm to use they can exchange the message as long as the key remains secret.

In modern cryptography, all security is based on this secrete the key and not in the details of the algorithm used. This means that an algorithm can be published and analyzed, but the key must remain secret. Besides, we have modern cryptographic techniques such as:

1. Secret key cryptography or symmetric key encryption.
2. Public key cryptography or Asymmetric encryption.

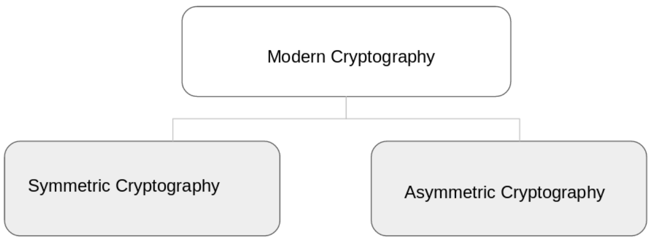


Figure 2.1: modern cryptographic techniques

Both of them have some advantages and disadvantages. The main difference between Symmetric and Asymmetric encryption is based on the key (VIDEAU, 2005).

2.5.1 Symmetric Key Encryption or private key encryption

Symmetric encryption(or private key encryption) uses the only the key for both encryption and decryption of the data. The key used for encryption and decryption is called the private key and only people who are authorized for the encryption/decryption would know it.

Symmetric cryptography or conventional cryptography is the oldest historically. It is extremely widespread because of its speed of encryption because it uses small keys(A. Menezes and Vanstone., 1992) and (Corporation, 2002).

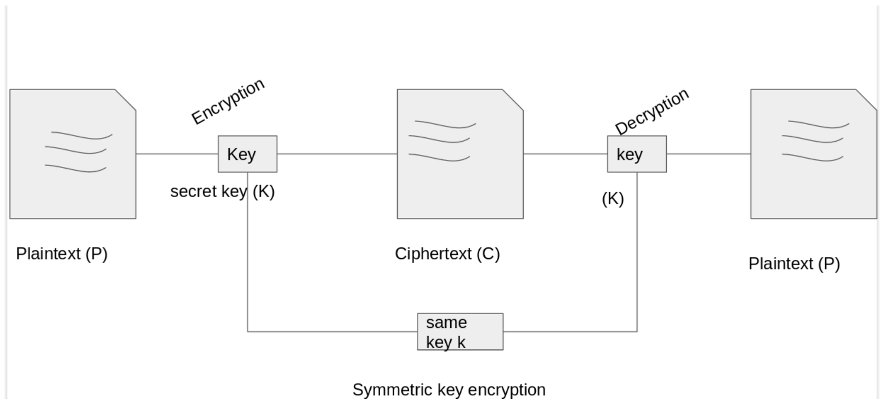


Figure 2.2: Symmetric encryption

This is the simplest type of encryption that involves only one secret key to encrypt and decrypt the information. It uses a secret key that can be a random number, word or string of letters. It means mixing with the plain text of a message to modify the content in a particular way. The sender and recipient must know the secret key used to encrypt and decrypt all messages. Therefore, the implementation (hardware and software) of symmetric cryptography can be very effective because of the speed of encryption and decryption of information. Symmetric encryption ensure that data encryption with one symmetric key cannot be decrypted with any other symmetric key. Therefore the symmetric key must be kept secret by both parties who use it to encrypt communication. Each party can be sure that communication with the other as long as the decrypted message continues to have to mean.

We know that the plaintext message is divided into successive blocks, each block being encrypted individually making it possible to have very powerful and simple encryption systems.

Symmetric encryption methods are naturally dived into two families, Block cipher(encryption) and Stream cipher: **Block cipher**, **Stream cipher**

2.5.1.1 Block Cipher(Encryption)

A block system, each clear text is divided into blocks of the same length and encrypted block by block(Dumont, 2010). The block ciphering is an encryption algorithm that encrypts a fixed size of n-bits of data known as a block in a single step... Block encryption algorithms encrypt a block of plaintext into a block of encrypted text by mixing it into an invertible block of encrypted text using a secret key.

The usual size of each block is 64 bits, 96 bits, 128 bits, 192 bits or 256 bits for equation cryptography. Block encryption is used (usually) when all data is available before the encryption process begins: Encryption of an email or long-term archived data.

- Confusion and Diffusion. A substitution in a round said to add confusion to the encryption process while a transposition/permutation add diffusion

1. Confusion is intended to make the relationship between the key and the encrypted text as complex as possible.
2. Diffusion refers to the rearranging or spreading bits in the message so that any redundancy in plaintext is distributed over the encrypted text. We can then say that around adds both confusion and diffusion to encryption.

- The general idea of block encryption(MOLLIN, 2007) is as follows:
  1. Replace characters with a binary code,
  2. Cut this string into blocks of given length,
  3. Encrypt a block by " adding" it bit by to key,
  4. Move some bits of the block,
  5. Replace operation three (3) a number of time if necessary,
  6. Go to the next block and return to step three(3) until the entire message is encrypted.

There are three (3) categories of encryption per block:

1. Encryption by substitution: Involve replacing symbols or groups of symbols with other symbols or groups of symbols in order to create confusion.
2. Encryption by transposition: consist in mixing symbols or groups of symbols of a clear message according to predefined rules, to create diffusion. These rules are determined by the encryption key. A sequence of transposition forms a permutation.
3. Encryption by Product: It is the combination of the two. Substitution or transposition encryption does not provide a high level or security, but by combining this two transformations, more robust encryption can be achieved. Most symmetric key algorithms use product-based encryption. A "round" is completed when both transformations have been done once (substitution and transformation).

The most well know algorithms of symmetric cryptography systems are: DES and AES.

DES(Data Encryption Standard):

The DES (A. Menezes and Vanstone., 1992) is a symmetric-key algorithm for the encryption electronic data. Although is short key length of 56 bits, criticized from the beginning, make it too insecure for most current applications, it was highly influential in the advancement of modern cryptography. DES, as stated above, is insecure. This is mainly due to the 56-bits, Key being too small.

AES(Advanced Encryption Standard):

AES (Buchmann, 2002), also known by its original name RIJNDOEL, is a specification for the encryption of electronic data established by the US National Institute of standard and Technology (NIST) in 2001. It uses the keys of length 128-bits, 192-bits and 256-bits.

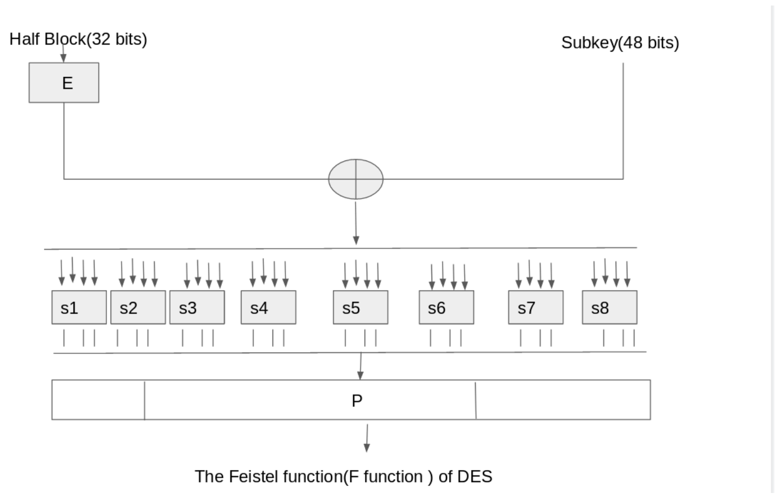


Figure 2.3: Feistel function

2.5.1.2 Stream Cipher (stream Encryption)

Remember that stream cipher uses a finite state machine initialized with a key and an initial vector to produce a long pseudo-random that is XORed with the plaintext to get the text in numbers (Buchmann, 2002). In general, flow encryption or stream cipher is used for as broad a real-time data encryption process as possible, hence the need for the very fast encryption process. Therefore the fact that the length of the block should be as short a possible the encryption algorithm should be as simple as possible.

For example, a vernam encryption: a small encryption block with a small block key k to produce a small encryption block text $C = m \oplus k$. If the encryption process for each small block is very

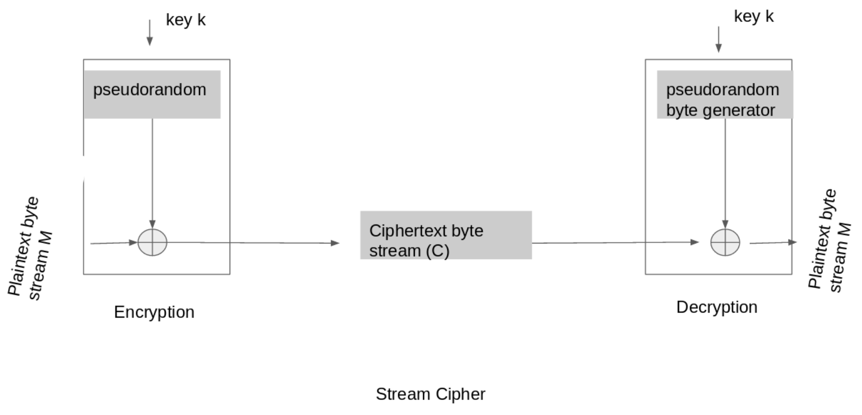


Figure 2.4: Stream Cipher

simple, for the security of stream cipher, we need.

1. to use a long sequence of key blocks(key stream) $k_0 k_1 ....k_r k_{r+1} .....$ to encrypt a long

sequence of message block $m_0m_1.......m_rm_{r+1}$,

2. to use the key-stream once (one-time pad),
3. to generate randomly the stream to ensure safety (security). It is important to focus on the key-stream generation process.

2.5.1.3 Limitations of Symmetric Encryption

Sharing the key. The biggest problem with symmetric key encryption is that you need a way to get the key from the person with whom you share data. If the shared key were to be captured by an unauthorized party. All security would be compromised. Therefore, you must have a secure way to obtain the key from the other party. Otherwise symmetric key encryption is the particular useful when you encrypt your own information rather than when you share encrypted information.

2.5.2 Asymmetric Cryptography(Encryption)

Key distribution problems are solved by key cryptography. This concept was introduced by Whitfield Diffie and Martin Hellman in 1976(DIFFIE and HELLMAN, 1976). Asymmetric encryption is also known as public-key cryptography, which is a relatively new method compared to symmetric encryption. Asymmetric encryption uses two keys to encrypt raw text. A public key that encrypts data and a corresponding private or secret key for decryption. It is important to note that anyone with a secret key can decrypt the message and that is why asymmetric encryption uses two related keys to enhance security. A public key is available free of charge to anyone who wishes to send you a message. The second private key is kept secret, so you can't know.

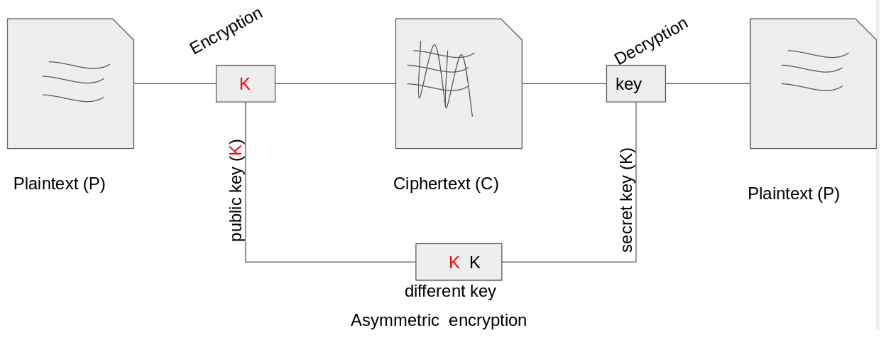


Figure 2.5: Asymmetric encryption

The security of the public key is not important because it is accessible to the public, Hence anyone can use it and can have the public key. The security of the public key is not important because it is accessible to the public, Hence anyone can use it and can have the public key. The asymmetric key has much higher power to ensure the security of the information transmitted during communication.

Asymmetric key encryption algorithm includes the following techniques in jonathan Katz and Lindell (2007):

- ELGmal
- RSA
- DSA
- ELLIPtic Curve , PKCS.

2.5.2.1 Limitations of ASymmetric Encryption

Asymmetric algorithms make it possible to overcome the problems associated with key exchange via an unsecured channel. However, the latter remains much less efficient in terms of computation time than symmetric algorithms.

2.6 The Shannon Theory of Secrecy

2.6.1 Elementary Probability Theory

The unconditional security of a cryptosystem abviously cannot be studied from the point of view of computational complexity because we allow computation time to be infinite(jonathan Katz and Lindell, 2007)

The appropriate framework in which to study unconditional security is probability. We need only elementary fact concerning probability. the main definition are : First, we define the idea of random variable.

Definition 2.6.1. A discrete random variable, say **X**, consist of a finite set X and a probability distribution defined on X. The probability that the random variable **X** takes on the value x is denoted $P_r[\mathbf{X} = x]$, somtime we will abbreviate this to $P_r[x]$ if the random variable **X** is fixed. It must be the case that $0 \leq P_r[x]$ for all $x \in X$, and $\sum_{x \in X} P_r[x] = 1$

Example 2.6.1. We could consider a coin toss to be a random variable defined on the set {heads, tails} The associated probability distribution would be

$$P_r[heads] = P_r[tails] = \frac{1}{2}$$

Suppose we have a random variable **X** defined on X. And $E \in X$. The probability that **X** takes on a value in the subset E is computed to be.

$$P_r[x \in E] = \sum_{x \in E} P_r[x] \tag{2.6.1}$$

the subset E is often called an event.

We next consider the concepts of joint and condition probabilities.

Definition 2.6.2. Suppose **X** and **Y** are random variables defined on finite set X and Y, respectively.The joint probability $P_r[x, y]$ is the probability that **X** takes on the value x and **Y** takes on the value y. The conditional probability$P_r[x|y]$ denotes the probability that **X** takes on the value x given that **Y** takes on the value y. The random variable **X** and **Y** are said to be independent random variables if

$$P_r[x, y] = P_r[x]P_r[y]$$

for all $x \in X$ and $y \in Y$

Join probability can be related to conditional probability by the formular

$$P_r[x, y] = P_r[x|y]P_r[y] \tag{2.6.2}$$

Interchanging x and y, we have that

$$P_r[x, y] = P_r[y|x]P_r[x] \tag{2.6.3}$$

From these two expressions we immediately obtain the following result, which is known as Bayes' Theorem.

Theorem 2.6.1. *(Bayes'theorem) If $P_r[y] > 0$, then:*

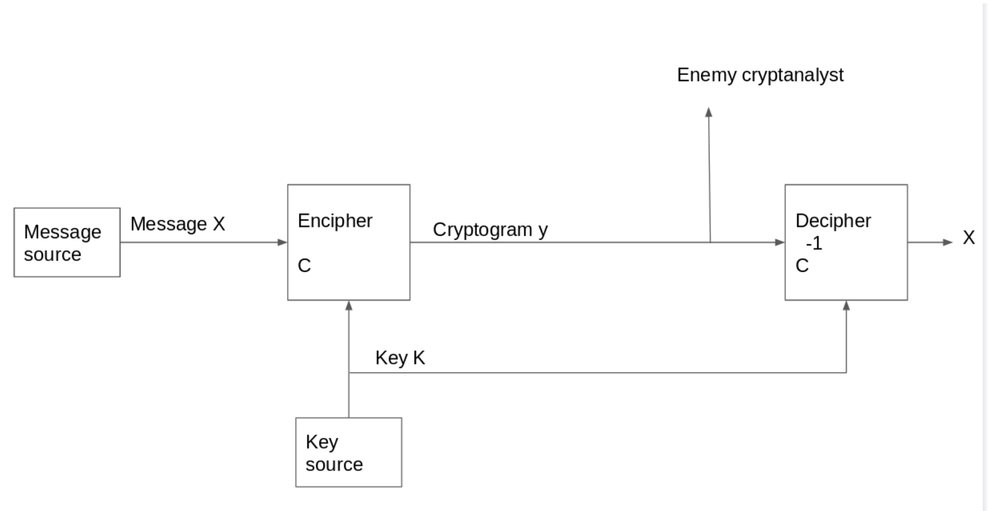
$$P_r[y|x] = \frac{P_r[x]P_r[x|y]}{P_r[y]}$$

Corollary. 2.6.1

X and **Y** are independent random variable if and only if $P_r[x|y] = P_r[x]$ for all $x \in$ **Y**

2.6.2 Secrecy of Communication

The purpose of encryption is to ensure communication secrecy. We assume that we want to communicate, which means to transmit information through a channel. The channel is not assumed to be secured.



The Shannon encryption model

Following the Shannon theory, we do not encrypt the fixed message, but message coming from a plaintext source. The plaintext source generated random text according to some given probability distribution.

Following the Shannon theory, cipher is given by:

1. A plaintext source (which is the corresponding distribution),
2. A secret key distribution,
3. A cipher text space C,
4. A rule which transforms any plaintext X and a key K by a cipher $Y = C_k(X)$,
5. A rule which enable recovering X from K and $Y = C_k(X)$ as $X = C_k^{-1}(Y)$.

A more intutive definition of cipher includes

1. A plaintext space, a ciphertext, a key space,
2. A key generation algorithm.
3. A encryption algorithm,
4. A decryption algorithm.

The definitions are related to conventional cryptography in (A. Menezes and Vanstone., 1992)

In this setting, the secret key K must still be transmitted in a secure way, so we need a secure channel. To summarize, in order to transmit a message securely, we first need to set up a key and transmit it securely. Is this a vicious circle? It is not for two reasons.

Firstly, the key can be a very short piece of information, and it can be easier to protect than protecting the message. We can afford using an expensive channel in order to send a short key securely. Later on, we use a long message securely thanks to encryption.

Secondly, even if the key is long (which is the case of the vernam cipher), it still makes sense to use an expensive channel to transmit it securely. The secure channel may not be available all the time or may have a longer transmission on delays. We can use it when available by anticipating that we will have to transmit a confidential document in the future. Later on, we can use any available channel with the fast transmission. This makes a secure channel virtually available. We when to use an expensive, slow, hard available, but confidential channel, we can transform any other channel into a confidential one by using the Shannon model of encryption.

2.6.3 Entropy

We see the concept of perfect secrecy. We use the key for only one encryption (symmetric). We now want to look at what happens as more and more plaintext are encrypted using the same key, and how likely a cryptanalyst will be able to carry out a successful ciphertext-only attack, given sufficient time.

To answer or study this question is the idea of entropy, a concept of information theory introduced by Shannon in 1948. Entropy can be considered a mathematical measure of information or uncertainty and is calculated based on a probability distribution.

Suppose we have a discrete random variable **X** which takes values from a finite set X according to a specified probability distribution. What is the information gained by the outcome of an experiment which takes place according to this probability distribution? Equivalently, if the experiment has not (yet) taken place, what is the uncertainty about the outcome? This quantity is called the entropy of **X** and is denoted by $H(\mathbf{X})$

Definition 2.6.3. Suppose **X** is a dicrete variable which takes on value from a finite set X. Then, the entropy of the random variable **X** is defined to be the quantity

$$H(\mathbf{X}) = -\sum_{x\in X} P_r[x]\log_2 P_r[x] \tag{2.6.4}$$

The joint entropy $H(\mathbf{X},\mathbf{Y})$ of two random variable is defined as the entropy of the joint variable $Z = (\mathbf{X},\mathbf{Y})$, i.e.

$$H(\mathbf{X},\mathbf{Y}) = -\textstyle\sum_{x,y} P_r[x,y]\log_2 P_r[x,y].$$

Note that **X** and **Y** are independent if and only if

$$P_r[x,y] = P_r[x]\times P_r[y]$$

for any x and y.

We further defined the conditional entropy $H(\mathbf{X}|\mathbf{Y})$ as

$$H(\mathbf{X}|\mathbf{Y}) = H(\mathbf{X},\mathbf{Y}) - H(\mathbf{Y}), \text{ i.e}$$
$$H(\mathbf{X}|\mathbf{Y}) = -\sum_{x,y} P_r[x,y]\log_2 P_r[x|y]$$

Theorem 2.6.2. *For any distribution,we have*

1. $H(\mathbf{X},\mathbf{Y}) \geq H(\mathbf{X})$ *with equality if and only if* $\mathbf{Y}$ *can be written as* $f(\mathbf{X})$.
2. $H(\mathbf{X}|\mathbf{Y}) \leq H(\mathbf{X}) + H(\mathbf{Y})$ *with equality if and only if* $\mathbf{X}$ *and* $\mathbf{Y}$ *are independent.*
3. *If* $P_r[x] \neq 0$ *for at least* n *values of* x *then* $H(\mathbf{X}) \leq \log_2(n)$ *with equality if and only if all nonzero* $P_r[x]$ *are equal to* $\frac{1}{n}$

2.6.4 Perfect Secrecy

Perfect secrecy refers to the posteriori distribution of the plaintext X after we know the ciphertext Y is equal to the a priori distribution of the plaintext:

The conditional distribution of $\mathbf{X}$ given $\mathbf{Y}$ is equal to the original distribution. Formaly, for all x and y such that $P_r[y] \neq 0$, we have $P_r[x|y] = P_r[x]$.

Theorem 2.6.3. *Perfect secrecy is equivalent to* $H(\mathbf{X}|\mathbf{Y}) = H(\mathbf{X})$ *and to the statistic independent between* $\mathbf{X}$ *and* $\mathbf{Y}$.

Proof

We have

$$H(\mathbf{X}|\mathbf{Y}) = H(\mathbf{X},\mathbf{Y}) - H(\mathbf{Y}) \leq H(\mathbf{X})$$

with equality if and only if $\mathbf{X}$ and $\mathbf{Y}$ are independent.

Thus $H(\mathbf{X}|\mathbf{Y}) = H(\mathbf{X})$ is equivalent to the independence of $\mathbf{X}$ and $\mathbf{Y}$.

If we have perfect secrecy, then

$$\frac{P_r[x,y]}{P_r[y]} = P_r[x|y] = P_r[x]$$

Theorem 2.6.4. *(Shannon 1948). Perfect secrecy implies* $H(K) \geq H(\mathbf{X})$

Proof

We first prove the intermediate property which holds in all case :

$H(\mathbf{Y}) \geq H(\mathbf{X})$.First, we have $H(\mathbf{Y}) \geq H(\mathbf{X}|K)$. We notice with the knowledge of K. given the same distribution for $\mathbf{X}$ and $\mathbf{Y}$, that $H(\mathbf{Y}|K) = H(\mathbf{X}|K)$. But since $\mathbf{X}$ and K are independent, we obtain $H(\mathbf{Y}|K) = H(\mathbf{X})$. We thus have $H(\mathbf{Y}) \geq H(\mathbf{X})$.

We now notice that when $\mathbf{X}$ is fixed, the K knowledge of K determines $\mathbf{Y}$. Furthermore, K and $\mathbf{X}$ are independent. Thus we have $H(\mathbf{Y}, K|\mathbf{X}) = H(K)$. Then we have $H(\mathbf{X}, \mathbf{Y}, K) \geq H(\mathbf{X}, \mathbf{Y})$. thus we have $H(K) \geq H(\mathbf{Y}|\mathbf{X})$. If we have perfect secrecy, we have

$$H(\mathbf{Y}|\mathbf{X}) = H(\mathbf{X}|\mathbf{Y}) + H(\mathbf{Y}) - H(\mathbf{X}) = H(\mathbf{Y}).$$

Thus we have

$$H(K) \geq H(\mathbf{Y})$$

Hence we obtain

$$H(K) \geq H(\mathbf{X})$$

Corollary. 2.6.4

$\mathbf{X}$ is an m-bit string and we want to achieve perfect secrecy for any distribution of $\mathbf{X}$, then the key must at least be represented with m-bits.

Proof. If we want to achieve perfect secrecy for any distribution of $\mathbf{X}$, we need to have $H(K) \geq H(\mathbf{X})$ for any distribution of m-bits string. For the uniform distribution we obtain $H(K) \geq m$. Now if k is the key length, we know that for any distribution of K, we have $H(K) \leq k$. thus we have $k \geq m$.

The corollary and the following result show that we cannot achieve perfect secrecy in a cheaper way than the vernam cipher.

□

2.6.5 Vernam Cipher (One-Time Pad)

With the industrial revolution, communication became automatic (ratio, etc.) Encryption and decryption had to be performed by a cryptographic device. The telewriter used the Baudot code in (jonathan Katz and Lindell, 2007) for which all characters are encoded into 5-bits.

The Vernam cipher (1926) is defined by

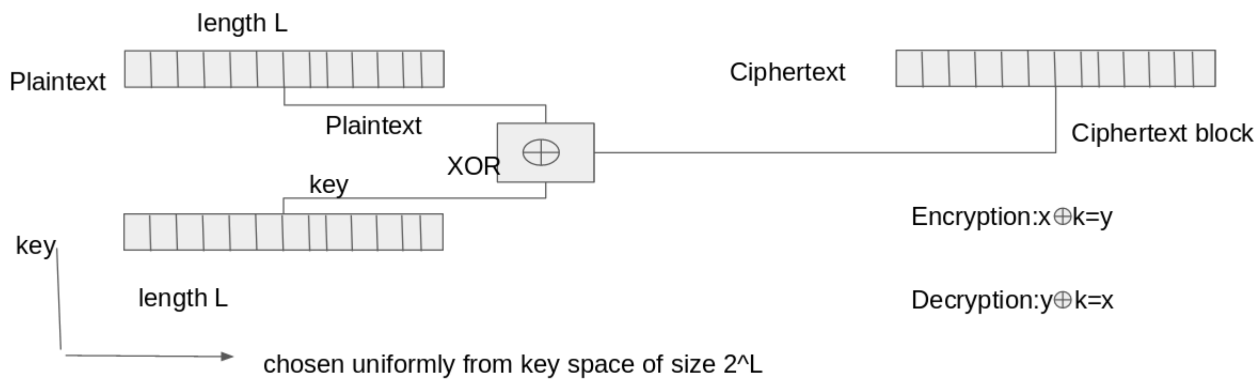
1. The palintext is a bit string : an element of $\{0, 1\}^n$
2. The secret key is uniformly distribured element of $\{0, 1\}^n$
3. The ciphertext is a $C_K(X) = X \oplus K$ where $\oplus$ is the bitwise **XOR**.

The Key is aimed at being used for only one plaintext. For this reason, this cipher is also known as **One-Time Pad** It was published in 1926 by Gilbert Vernam, from ATT. (It was actually invented at the end of the first world war, but since the war was over, people did not get any more interest in it until 1926.) The security was formally proven by Shannon, by using the notion of perfect secrecy. Later on, the Vernam cipher was used for the red telephone between Moscow and Washington. Note that of Venam cipher uses a key substitution. It is a kind of Vignere cipher with a binary alphabet and One-Time key.

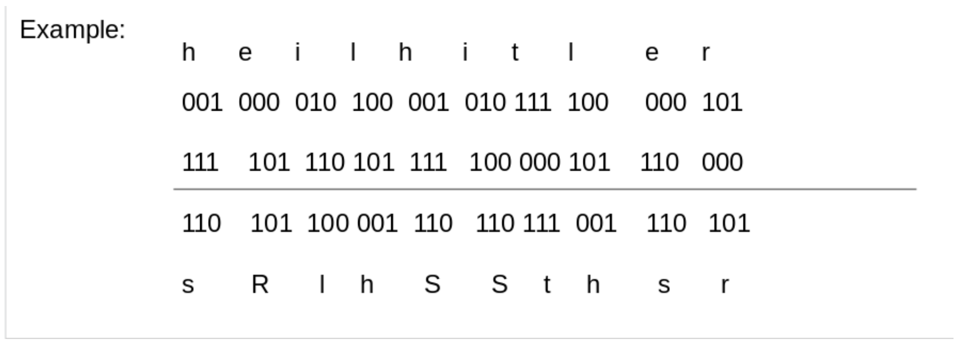
The drawBack of this are:

1. The key must be at least as long as the message,
2. It becomes insecure if a key is used twice,
3. The security result makes sense only when the source is truly random,
4. Some pseudorandom keys may have insecure implementation,
5. Randomness is expensive.

The security of venam cipher is indeed critically sensitive to the freshness requirement on the secret key. In the forties. The Soviet Intelligence agency KGB was using the Vernam cipher but was reusing some fragments of keys several times. This led the American counterpart NSA to decrypt message in the famous VERONA project.



Example 2.6.2. Encryption plaintext $\oplus$ key=ciphertext



Example:

h	e	i	l	h	i	t	l	e	r
001	000	010	100	001	010	111	100	000	101
111	101	110	101	111	100	000	101	110	000
110	101	100	001	110	110	111	001	110	101
s	R	l	h	S	S	t	h	s	r

Theorem 2.6.5. *The Vernam cipher provides perfect secrecy for any distribution of the plaintext.*

Proof: Let **Y** $= $ **X** $\oplus K$ be the ciphertext where **X** and K are independent bit string of length n, and K is uniformly distribution. for any x and y, we have

$$P_r[x,y] = P_r[x, K = x \oplus y]$$
$$= P_r[x] \times P_r[K = x \oplus y]$$
$$= P_r[x] \times 2^{-n}$$

By adding over all x we obtain that $P_r[y] = 2^{-n}$. We deduce that

$$P_r[x|y] = P_r[x].$$

For any x and y.

2.6.6 Limitations of Perfect Security

Key must be at least as long as the message .

Limits applicability if message is long **.** Key must be changed for every encryption - If the same key is used twice, then an adversary can compute the Xor of the message.

$$x_1 \oplus k = y_1$$
$$x_2 \oplus k = y_2$$
$$x_1 \oplus x_2 = y_1 \oplus y_2$$

The attacker can then do language analysis to determine y_1 and y_2

2.7 Cryptanalyts

2.7.1 principe and Basics

Shannons's Describe the conventional cryptosystem shannon (university of technology the Netherlands, 1996) . Figure 2.6, the general outline of a conventional cryptosystem is depicted.

Let $\mathcal{A}$ be a finite set, which we will call alphabet. we shall use $|A|$ to denote the cardinality of A. We shall often use $\mathbf{Z}_q = \{0, 1, \ldots. q-1\}$ as alphabet where we work with its elements modulo q.

A concatenation of n letters from A will be called an n-gram and denoted by $a = (a_0, a_1, a_2 \ldots., a_{n-1})$. Special cases are bi-grams $(n = 2)$ and tri-grams $(n = 3)$. The set of all n-grams from A will be denoted by A^n. A text is an element from $A^* = \bigcup_{n \geq 0} A^n$.

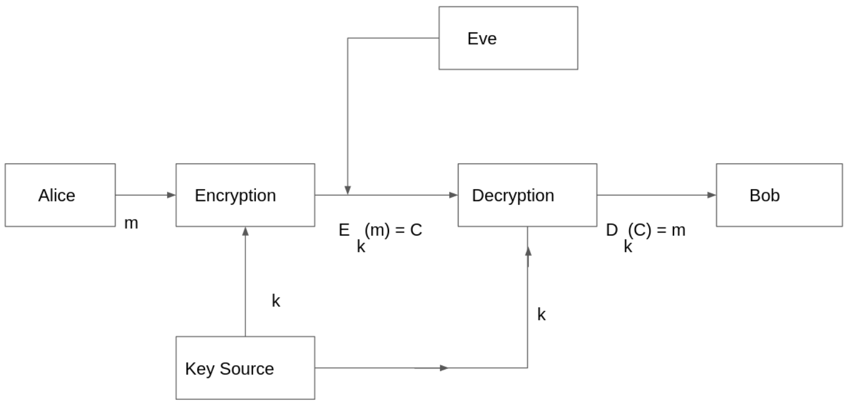


Figure 2.6: The conventional cryptosystem

Language is a subset A^*. Let A and B be two finite alphabets. Any one-to-one mapping to E of A^* to B^* is called a cryptographic transformation. In most practical situations, $|A|$ be equal $|B|$. Also often the E will map n-grams(to avoid data expansion during the encryption process). Let m, the message (a text from A^*) that Alice in figure 4.5. wants to sent in secrecy to Bob. Tnis is usually called the plaintext. Alice will first transform the plaintext into $C = E(m)$, the so called ciphertext.

Definition 2.7.1. A symmetric (or conventional cryptosystem E is a set of cryptography transformation $E = \{E_k|k \in K\}$ the index set K is called the key space, and its element k keys.

Since E_k is a one-to-one mapping, its inverse must exist. We shall denote it with D_k, E is encryption and D is decryption. One has $D_k(E_k(m)) = m$, for all $m \in A^*$ and key $k \in K$. If Alice wants to send the palintext m to Bob by means of the cryptographic transformation E_k, both Alice and Bob must know the choice of the key k. They will have agreed on the value of k by means a so-called secure channel. It could also be that Alice and Bob have, before hand, agreed on the choice of k.

Bob can decrypt (d) the ciphertext C by computing $D_k(c) = D_k(E_k(m)) = m$.

Normally, the same cryptosystem ∈ will be used a longtime and by many people, so it is reasonable to assume that his set of cryptographic transformation ∈ is also known to the cryptanalyst. It is the frequent changing of the key that has to provide the security of the data. This principle was already clearly stated by Dutchonor Auguste Kerckhoff (Kerckhoffs, 1883)in the 19th-century.

The cryptanalyst (Eve) who is connected to the transmission line can be :

. Passive (eavesdropping): the cryptanalyst tries to find m (or even better k) from C(and whatever further knowledge he has). By determinig k more ciphertext may be broken.

. Active (tampering): the cryptanalyst tries to actively manipulate the data that are being transmitted. For instance, he transmits his own ciphertext, retransmits old ciphertext and substitutes his own text for transmitted ciphertext etc .

2.7.2 The Levels of Attacks

In general, one discernes three levels of cryptanalysis:

1. Ciphertext only attack: only a piece of ciphertext is known to the cryptanalyst (and of then the context of the message).
2. Known plaintext attack : A piece of ciphertext with corresponding plaintext is known. If a system is ensure against this kind of attack the legitimate receiver does not have to destroys deciphered message.
3. Chosen plaintext attack : the cryptanalyst can choose any piece of plaintext and generated the corresponding ciphertext. The public-key cryptosystem has to be secure against this kind of attack. Differential cryptanalysis is an example of a chosen plain text attack.

3. Fundamentals of Artificial Neural Networks

3.1 Introduction

Artificial intelligence as imagined in the 1950s was more difficult to develop than expected. In recent years, the field has undergone a major revival with deep learning techniques inspired by neural networks in the brain. Current neural networks are improved versions of pioneering work based on connexionalism. However, the corresponding learning algorithms require active human participation.

The brain is a very complex, non-linear and parallel information processing system.It has the ability to organize its structural components, called neurons, by performing some calculations much faster than the fastest digital computer currently available. The brain regularly performs perceptual recognition tasks, such as recognizing a familiar face in an unknown scene.

A neural network is a machine designed to model how the brain performs a particular task. The network is implemented using electronic or simulated components. it resembles the brain in two ways(Vikas Gujra) :

1. The knowledge that the neural network learning comes from its environment through a learning process.
2. The connection forces of interneurons, known as synaptic weights, it allows to store the knowledge acquired by the neurone network.

3.2 Biological Neural Networks

Artificial Neural Networks(ANN) (Vikas Gujra) are mathematical models that are inspired by biology. The basis of these neurons is due to the fact that the artificial neuron, was originally derived from a desire to model the functioning of a biological neuron.

The cellular body or Soma or somatic cell that represents the control center makes the sum of the received information and after processing this information the results return is in the form of electrical signals. And bring back from the cellular body to the input of other neurons by the axon. The axons connecting the neurons. The neuron also consists of several branches called dendrites are the sensors of the neuron. They transmit the nerve impulses (information) generated by stimuli from their extremities to the perikaryon. The synapses of the neuron receive information from other neurons through the axon and thus allow the neurons to communicate with each other.

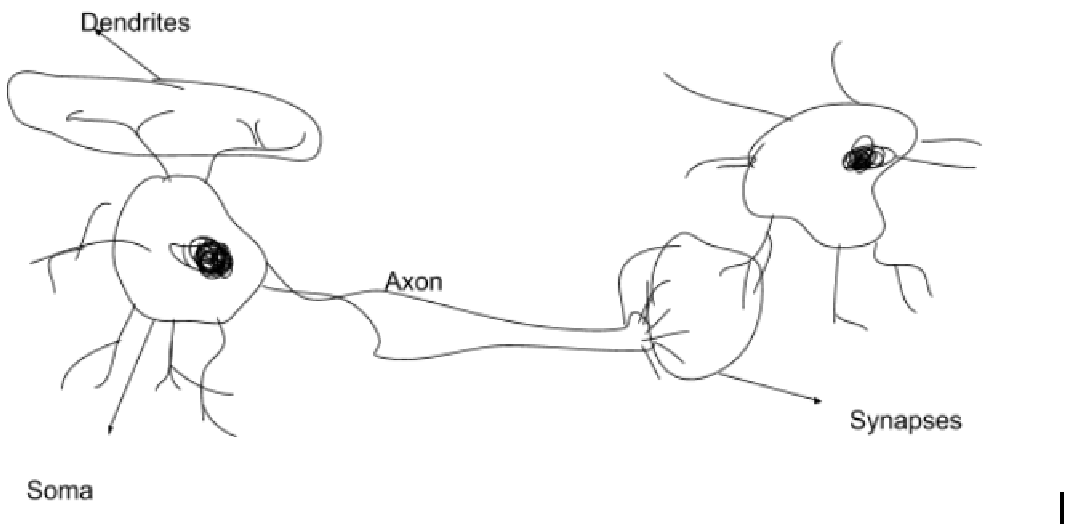


Figure 3.1: Neuron of brain

3.3 Model of Neuron

Several models of neural networks have been proposed by Aurelien Pottiez (2018). Each model is characterized by its learning rule, intra-neuronal functions (activation function) and inter-neuronal connection architecture. A neural model based on a single neuron is rarely used. We will present some models of artificial neural networks.

In this part we will consider three classical model an artificial neuron or processing unit refer in (YEGNANARAYANA).

3.3.1 McCulloch Pitts

In McCulloch Pitts (MP). (MP) model Figure 3.2 the activation is given by the weighted sum of its n input x_i and a bias term θ. The output signal (s) is non-linear function $f(x)$ of the activation value x. the equation of (MP) is define by:

activation : $x = \sum_{i=1}^{n} w_i \alpha_i - \theta$

Output signal : $S = f(x)$

We use non-linear function (ramp,binary, sigmoid) see the Figure 3.3:

only the binary function was used in the MP model.

Example 3.3.1.

For the NOR gate in table :

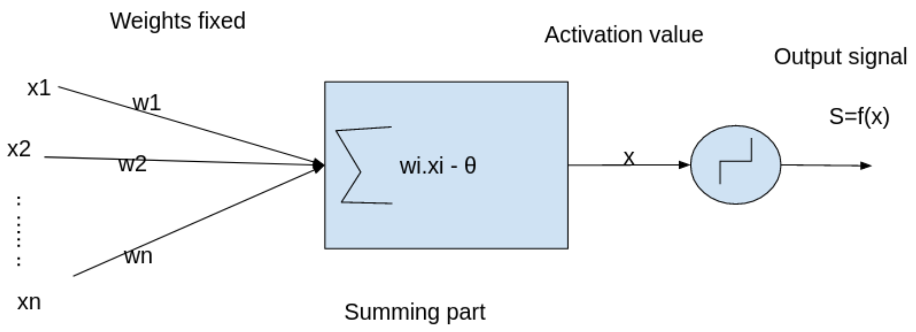


Figure 3.2: MCCulloch Pitts (MP)

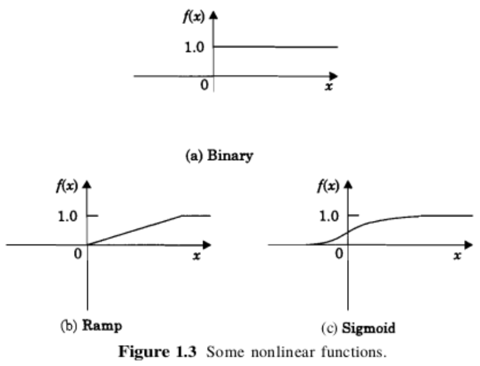


Figure 3.3: Non-linear Function

α_1	α_2	S
0	0	1
0	1	0
1	0	0
1	1	0

When any input to a NOR gate is energised by a logic "1" signal, then its output is "0". Otherwise the output is "1".

For the NANAD gate we can see the table.

α_1	α_2	S
0	0	1
0	1	1
1	0	1
1	1	0

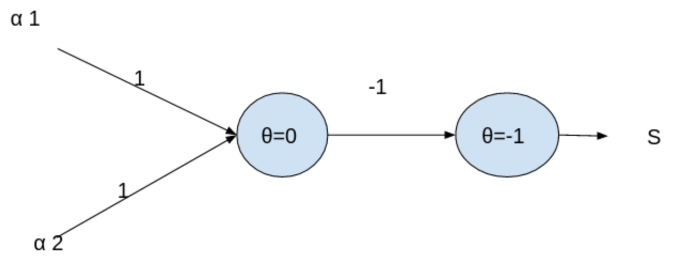


Figure 3.4: NOR gate

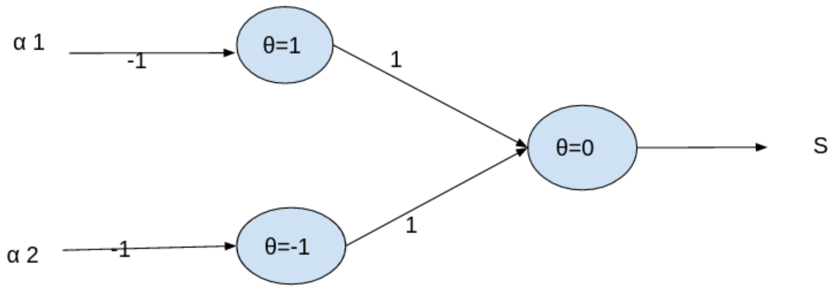


Figure 3.5: NAND gate

When any input to a NAND gate is energised by a logic "0" signal, then its output is "1". Otherwise the output is "0".

In this model a binary output function is used with the following

$$f(x) = \begin{cases} 0 \quad if \quad , x > 0 \\ \quad 1 \quad , if x \leq 0 \end{cases}$$

3.3.2 Adaline

ADAptive LiNear Element (ADALINE) is a model proposed by Widrow and is shown in the Figure 3.6

The main distinction between the Rosenblatt perceptron model and the widrow adaline model is that, in the ADaline the analog activation value x is compared with the target output (b). The output is a linear function of the activation value x.

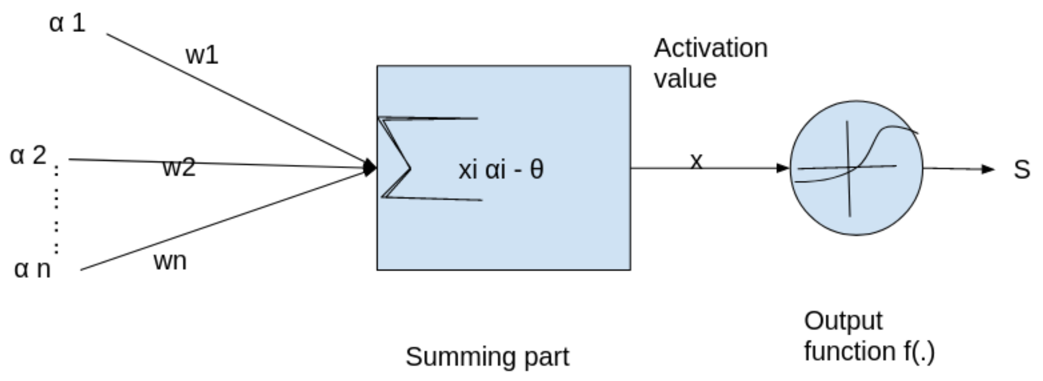


Figure 3.6: ADALINE Model

$$\begin{aligned} x &= \sum_{i=1}^{n} w_i \alpha_i - \theta \\ S &= f(x) = x \\ error\ \sigma &= b - S = b - x \\ weight\ change : \Delta w_i &= \eta \delta \alpha_i \end{aligned}$$

Where η is the learning rate parameter. This update rule minimises the mean squared error δ^2, averaged over all input. Hence it is called Least Mean squared (LMS) error learning law. This law is derived using the negative gradient of the error surface in the weight space. Hence it is also know as a gradient descent algorithm.

3.3.3 Perceptron

The perceptron, also called artificial neuron or formal neuron, the perceptron is a learning algorithm supervises for binary classifiers. Invented in 1957 by FranK Rosenblatt in (ROSENBLATT). It is inspired by a biological neuron.

The perceptron has some principal components we need to know.

1. The input layer that takes as input a vector $X(x_1, x_2, ..., x_n)$, the input represents the dendrites according the Figure 3.7 .
2. The $W(w_1, w_2, ...w_n)$ which represents the list of synaptic values or synaptic weights for each input, it is the same size as the input vector.

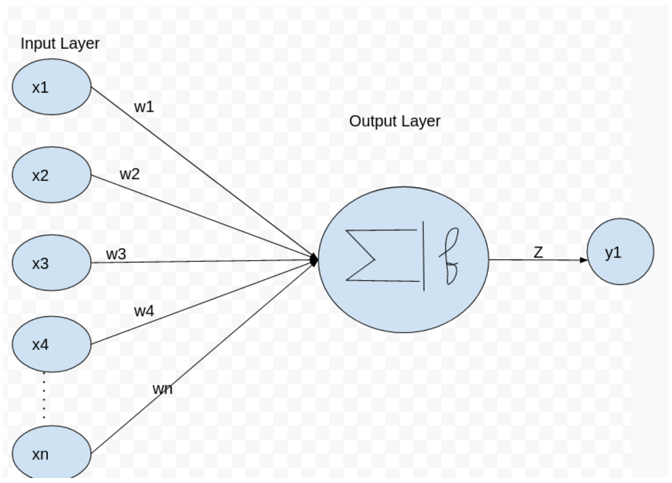


Figure 3.7: Simple Perceptron

3. The bias b this parameter influences the sensitivity of the neuron. On the Figure we don't add the bias.

$$y = \sum_{i=1}^{n-1} w_i x_i + b \tag{3.3.1}$$

   Generally the bias is a constant that is 1, only the weight changes.

4. A calculation module that has two main parts: pre-activation and the activation part.

For pre-activation it allows us to calculate the weighted sum of the inputs.

$$S = \left(\sum_{i=1}^{n-1} w_i x_i + b\right)$$

For activation this part takes in the weighted sum (activation function) and gives the output state of the neuron like $Z = f(S)$ (True/ False) relative to a given threshold.

$$Z = \begin{cases} 1, & \text{if } f(S) > 0 \\ 0, & \text{otherwise} \end{cases}$$

We have many activation function(Aurelien Pottiez, 2018) :

• Sigmoid function

The primary purpose of the function is to reduce the input value to a value between 0 and 1. In addition to expressing the value as a probability. The Sigmoid function has several faults:

1. It is not centered on zero, i.e. negative inputs can lead to positive outputs.

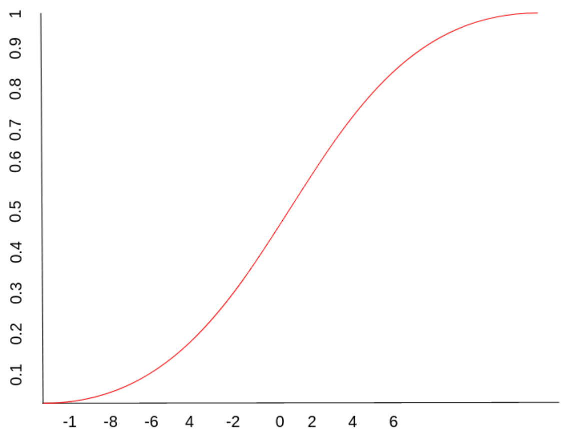


Figure 3.8: Sigmoid Function $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

2. Being quite flat, it has little influence on the neurons compared to other activation functions. The result is often very close to 0 or 1 causing saturation of some neurons.

3. It is expensive in terms of calculation because it includes the exponential function.

- **Tanh**:

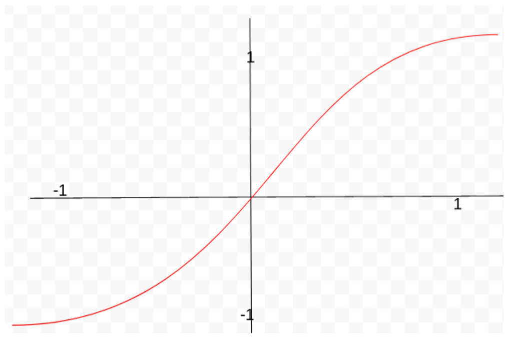


Figure 3.9: Sigmoid Function

The Tanh function is also known as "hyperbolic tangent."

This function is similar to the Sigmoid function. The difference with the Sigmoid function is that the Tanh function produces a result between -1 and 1. The Tanh function is in general terms preferable to the Sigmoid function because it is centered on zero. Large negative inputs tend towards -1 and large positive inputs tend towards 1.

Apart from this advantage, the Tanh function has the same other disadvantages as the Sigmoid function.

- **ReLu**:

To solve the problem of saturation of the two previous functions (Sigmoid and Tanh) there is the ReLU (Linear Grinding Unit) function. This is the most used function.

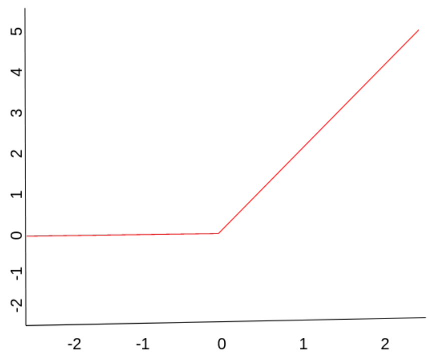


Figure 3.10: ReLu

The function ReLU is interpreted by the formula: $f(x) = \max(0, x)$. If the input is negative the output is 0 and if it is negative then the output is x. This activation function considerably increases the convergence of the network and does not saturate the network.

But the ReLU function is not perfect. If the input value is negative, the neuron remains inactive, so the weights are not updated and the network does not learn.

- **Leaky ReLU**:

The Leaky ReLU function is interpreted by the formula: $f(x) = \max(0.1x, x)$. The Leaky Relu function tries to correct the ReLU function when the input is negative. The concept of Leaky ReLU is when the input is negative, it will have a small positive slope of 0.1. This function somewhat eliminates the problem of inactivity of the reLU function for negative values, but the results obtained with it are not consistent. It still retains the characteristics of a ReLU activation function, i.e. it is computationally efficient, converges much faster, and does not saturate in positive regions.

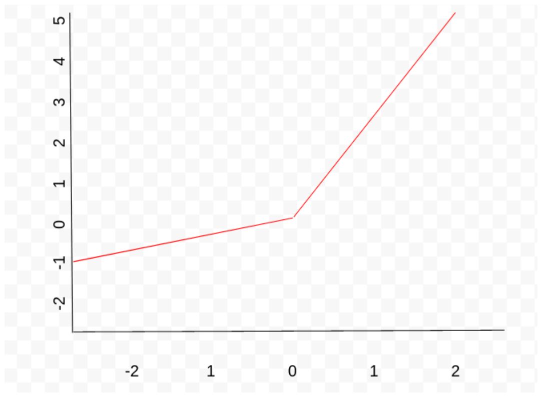


Figure 3.11: Leaky ReLu

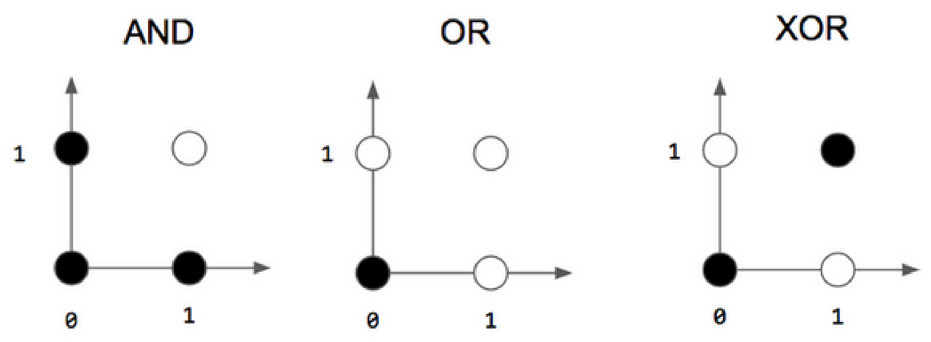
3.3.3.1 Limite of perceptron

Theorem 3.3.1. *A linear perceptron with n input threshold divides the space of the entries $\mathbf{R}^n$ in two subspaces delimited by a hyperplane.*

Theorem 3.3.2. *Any linearly separable assembly can be discriminated by a perceptron.*

Theorem 3.3.3. *XOR can not be calculated by a perceptron linear with threshold.*

The perceptron can only model a function if the data is linearly separable, which is an obvious liearite. The perceptron, because of its activation function, can only trace one line in space, which is precisely the problem with the XOR function.



Multilayer neural networks allow to solve these problems underlined by Minsky describe byMinsky on the perceptron.

3.3.4 Multi-Layers Perceptron(MLP)

3.3.4.1 Definition

A multilayer neural network, also called a multilayer perceptron define in (LeCun) and (S. Kouamo, b), corresponds to the assembly of several interconnected neurons (or perceptrons). It can have one or more hidden layers unlike a simple perceptron. The output of a layer is the input of the following layer. In multilayer or perceptron neural networks in our case, the neurons are arranged by layer. There is no connection between neurons in the same layer. the output of a layer is the input of the next layer, this means that each neuron in a layer is connected to all the other next neurons. Let us consider the full graph below

We notice from the diagram that for a multiple perceptron there are three main components, namely:

1. Input layer in which a set of data is given (a value vector).
2. Hidden layer we can have one or more hidden layers, this is the heart of our perceptron, where the relationships between the variables will be highlighted! The role of the hidden

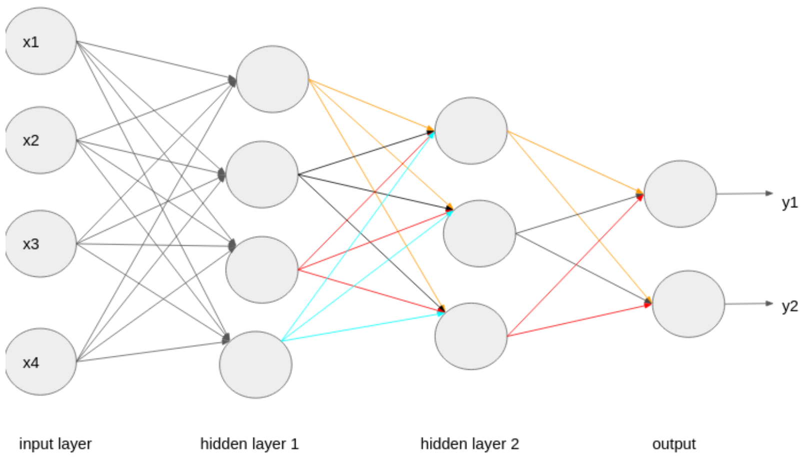


Figure 3.12: Multi Layers Perceptron MLP

layer is to prepare the data by mainly using non-linear activation functions in their neurons to present them to the output layer.

3. Output layer: this layer represents the final result of our network, its prediction

3.4 Gradient Descent

It's a quick reminder. Basically, we want to adjust a parameter in the neural network (call it w) so that the total error E decreases. We proceed as follows:

$$w = w - \alpha \frac{\partial E}{\partial w} \tag{3.4.1}$$

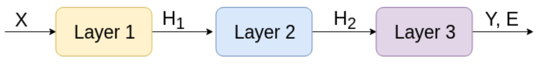
$\alpha \in [0, 1]$ that we have defined and which is called the learning rate. The most important here is $\frac{\partial E}{\partial W}$ (the derivative of E in relation to W). We must be able to find the value of this expression for any parameter of the neural network, whatever its architecture.

3.4.1 Forward Propagation

There will be at least two main elements, named the input layer X and an output layer Y.

$$X \rightarrow \boxed{\text{layer}} \rightarrow Y$$

Proposition 3.4.1. the output of a layer is the input of the following layer



This step is called the forward propagation algorithm: It consists of giving an input value X (image, text etc.) and it propagates through the artificial neural network to produce a result Y, and also an error named E. The objective will therefore be to reduce this error E with the gradient descent method that we will discuss later.

We will see how the forward pass algorithm is applied to the Fully Connected Layer (FC).

Definition 3.4.1. Fully Connected Layer (FC Layer), are the base layers that show the connection between input and output neurons. (see MPL).

We can calculate the value of each output neuron by this formula:

$$y_i = b_j + \sum_j x_i w_{ij}$$

The matrix form makes it possible to simplify this calculation

$$X = [x_1 \ \ ...x_i] \quad W = \begin{bmatrix} w_{11} & \cdots & w_{1j} \\ \vdots & \ddots & \vdots \\ w_{i1} & \cdots & w_{ij} \end{bmatrix} \qquad B = [b_1 \ \ ...b_j] \ \ Y = X * W + B$$

3.4.2 Backward Propagation

In this step we want to calculate the derivative of the error with respect to its input (Aflak). If we assume that we give a layer and the error derivative with respect to the output, then it must be able to calculate the error derivative with respect to the input. The error E is a scalar, X and Y are matrices.

$$\frac{\partial E}{\partial X} \leftarrow \boxed{\text{layer}} \leftarrow \frac{\partial E}{\partial Y}$$

$$\frac{\partial E}{\partial X} = [\frac{\partial E}{\partial x_1}\frac{\partial E}{\partial x_2}... \quad \frac{\partial E}{\partial x_i}]$$
$$\frac{\partial E}{\partial Y} = [\frac{\partial E}{\partial y_1}\frac{\partial E}{\partial y_2}... \quad \frac{\partial E}{\partial y_i}]$$

First we will try to explain the error E in relation to the output. If a layer has access to $\frac{\partial E}{\partial Y}$ where Y is its own output, then it is very easy to calculate the derivative of the error with respect to its $\frac{\partial E}{\partial W}$ parameters to adjust the parameters, regardless of the overall architecture of the neural network. So we will use derivation of compound functions.

$$\frac{\partial E}{\partial w} = \sum_J \frac{\partial E}{\partial y_j} \times \frac{\partial y_j}{\partial w}$$

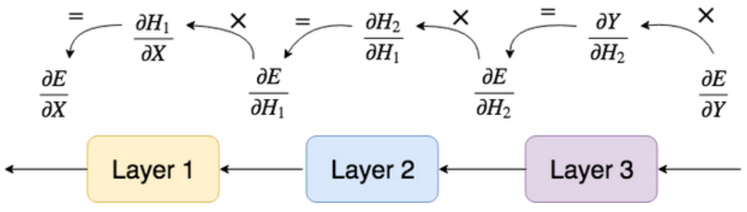
Given the fact that $\frac{\partial E}{\partial Y}$ we can calculate $\frac{\partial E}{\partial W}$!

Now we will calculate the derivative of the error E with respect to the input. From the property we know the output of one layer is the input of the next layer. It is said that $\frac{\partial E}{\partial X}$ for one layer will be $\frac{\partial E}{\partial Y}$ for the previous layer. Once equipped with its own $\frac{\partial E}{\partial Y}$, then the previous layer will be able to adjust its settings. The derivation of the compound functions is always used.

$$\frac{\partial E}{\partial x_i} = \sum_J \frac{\partial E}{\partial y_j} \times \frac{\partial y_j}{\partial x_i}$$

This derivation of the compound functions allows us understand the Backward propagation.

We can see the propagation of the error with respect to the output $\frac{\partial E}{\partial Y}$.


=
$\frac{\partial H_1}{\partial X}$
×
=
$\frac{\partial H_2}{\partial H_1}$
×
=
$\frac{\partial Y}{\partial H_2}$
×
$\frac{\partial E}{\partial X}$
$\frac{\partial E}{\partial H_1}$
$\frac{\partial E}{\partial H_2}$
$\frac{\partial E}{\partial Y}$
Layer 1
Layer 2
Layer 3

For the calculation of this step (FC Layer) we assume that we have the derivative of the error with respect to the output of this layer ($\frac{\partial E}{\partial Y}$). We need to consider the following:

1. The derivative of the error with respect to the parameters ($\frac{\partial E}{\partial W}, \frac{\partial E}{\partial B}$)
2. The derivative of the error with respect to the input ($\frac{\partial E}{\partial X}$)

For $\frac{\partial E}{\partial W}$, this matrix and W have the same size: $i \times j$ where i is the number of input neurons and j the number of output neurons. Then we need a derivative for each parameter:

$$\frac{\partial E}{\partial W} = \begin{bmatrix} \frac{\partial E}{\partial w_{11}} & \cdots & \frac{\partial E}{\partial w_{1j}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial w_{i1}} & \cdots & \frac{\partial E}{\partial w_{ij}} \end{bmatrix}$$

With the derivative of the compound functions we have:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_1} \times \frac{\partial E}{\partial w_{ij}} + ... + \frac{\partial E}{\partial y_j} \times \frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \times x_i$$

$$\frac{\partial E}{\partial W} = \begin{bmatrix} \frac{\partial E}{\partial y_1} \times x_1 & \cdots & \frac{\partial E}{\partial y_1} \times x_1 \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial y_1} \times x_i & \cdots & \frac{\partial E}{\partial y_j} \times x_i \end{bmatrix}$$

$$= \begin{bmatrix} x_1 \\ \vdots \\ x_i \end{bmatrix} \times [\frac{\partial E}{\partial y_1} \quad .... \quad \frac{\partial E}{\partial y_j}] = X^t \times \frac{\partial E}{\partial Y}$$

We have our first formula to adjust the weights and now we can calculate the derivative of the error with respect to the bias, $\frac{\partial E}{\partial B}$.

$$\frac{\partial E}{\partial B} = [\frac{\partial E}{\partial b_1} \; \frac{\partial E}{\partial b_2} \; \cdots \; \frac{\partial E}{\partial b_j}]$$

$\frac{\partial E}{\partial B}$ and B have the same dimension. The drift through the bias.

$$\frac{\partial E}{\partial b_{ij}} = \frac{\partial E}{\partial y_1} \times \frac{\partial y_i}{\partial b_j} + ... + \frac{\partial E}{\partial y_j} \times \frac{\partial E}{\partial b_j} = \frac{\partial E}{\partial y_j}$$

$$so$$

$$\frac{\partial E}{\partial B} = [\frac{\partial E}{\partial y_1} \; \frac{\partial E}{\partial y_2} \; \cdots \; \frac{\partial E}{\partial y_j}] = \frac{\partial E}{\partial Y}$$

Now that we have $\frac{\partial E}{\partial W}$ and $\frac{\partial E}{\partial B}$ we can adjust all the parameters of this layer in order to reduce the error. We still have to calculate $\frac{\partial E}{\partial X}$ so that the previous layer can do the same calculations.

$$\frac{\partial E}{\partial X} = [\frac{\partial E}{\partial x_1} \; \frac{\partial E}{\partial x_2} \; \cdots \; \frac{\partial E}{\partial x_i}]$$

Then we use the derivative of the compound functions

$$\frac{\partial E}{\partial x_i} = \frac{\partial E}{\partial y_1} \times \frac{\partial y_1}{\partial x_i} + ... + \frac{\partial E}{\partial y_j} \times \frac{\partial E}{\partial x_i}$$

$$= \frac{\partial E}{\partial y_1} \times w_{i1} + \; .... \; + \frac{\partial E}{\partial y_1} \times w_{ij}$$

The matrix form can be written as follows:

$$\frac{\partial E}{\partial X} = [(\frac{\partial E}{\partial y_1} \times w_{11} \; + ... + \; \frac{\partial E}{\partial y_1} \times w_{1j}) \; ... \; (\frac{\partial E}{\partial y_1} \times w_{i1} \; + ... + \; \frac{\partial E}{\partial y_1} \times w_{ij})]$$

$$= [\frac{\partial E}{\partial y_1} \; \cdots \; \frac{\partial E}{\partial y_j}] \times \begin{bmatrix} w_{11} & \cdots & w_{i1} \\ \vdots & \ddots & \vdots \\ w_{1j} & \cdots & w_{ij} \end{bmatrix} = \frac{\partial E}{\partial Y} \times W^t$$

• **Activation layer**:

We know that the machine will not learn anything from our model because the calculations we have made are linear. We will add non-linearity to our model by applying or using non-linear functions at the output of certain layers. To do this, we simply calculate the derivative of the error with respect to the input value $\frac{\partial E}{\partial X}$. Let f and f' be the activation function and its derivative, respectively.

For the propagation forward algorithm, we have a given input X, at the output we apply the activation function to each element of X. So X and Y have the same dimension.

$$Y = [f(x_1) \;\; ... \;\; f(x_i)] = f(X)$$

For backward propagation, we have $\dfrac{\partial E}{\partial Y}$, we will calculate $\dfrac{\partial E}{\partial X}$.

$$\frac{\partial E}{\partial X} = [\frac{\partial E}{\partial x_1} \;\; ... \;\; \frac{\partial E}{\partial x_i}]$$

$$= [\frac{\partial E}{\partial y_1} \times \frac{\partial y_1}{\partial x_1} \;\; ... \;\; \frac{\partial E}{\partial y_i} \times \frac{\partial y_i}{\partial x_i}]$$

$$= [\frac{\partial E}{\partial y_1} f\prime(x_1) \;\; ... \;\; \frac{\partial E}{\partial y_i} f\prime(x_i)]$$

$$[\frac{\partial E}{\partial y_1} \;\; ... \;\; \frac{\partial E}{\partial y_i}] \times [f\prime(x_1) \;\; ... \;\; f\prime(x_i)]$$

$$= \frac{\partial E}{\partial Y} \times f\prime(X)$$

• **Loss function**

In this part wewant to explain the loss function, but we know we have a lot of Loss function but we will just explain on activation function like Mean Squared Error (MSE).

We know that for a given layer, we assume that $\frac{\partial E}{\partial Y}$ was given (by the next layer). But what happens to the last layer? How does we get $\dfrac{\partial E}{\partial X}$? We simply give it manually, and that depends on how we define the error. The network error, which measures the degree of model performance for a given input, is defined by us. There are many ways to define error, and one of the best known is called Mean Squared Error (MSE).

$$E = \frac{1}{n}\sum_{i}^{n}(\hat{y}_i - y_i)^2 \tag{3.4.2}$$

Where $\hat{y}$ and y indicate respectively the desired output and the obtained output. we need now, with all other layers, to define $\frac{\partial E}{\partial Y}$.

$$\frac{\partial E}{\partial Y} = [\frac{\partial E}{\partial y_1} \;\; ... \;\; \frac{\partial E}{\partial y_i}]$$

$$= \frac{2}{n}[y_1 - \hat{y_1} \;\; ... \;\; y_i - \hat{y_i}]$$

$$= \frac{2}{n}(Y - \hat{Y})$$

Now it will be enough to give this value to the last layer during the back pass, which will allow her to adjust her parameters, then we calculates $\frac{\partial E}{\partial X}$ that we will pass to the front layer, which will do the same process in turn, etc.

In the next chapter we will explain the notion of artificial neural network in cryptography.

4. Neural Network In Cryptography

Artificial Intelligence (AI) is one of the domains that has seen a lot of progress, it applies today to almost all domains, and also many very useful applications such as facial recognition (Le, 2011), for the realization of certain tasks such as the detection of cancer in an image (M. M. Mehdy) we could even say that AI in some cases is better and faster than human beings.

So it is good to ask if it is possible for neural networks to create or break cryptographic algorithms. Several works have been done to solve this problem using machine learning techniques but do not talk about the concept of security.

Recently Abadi and Andersen proposed a GAN-based approach called GAN Cryptography or ANC (Adversarial Neural Cryptography). They show that artificial neural network or artificial agent can learn to communicate with symmetrical encryption to protect information from AI attackers. The idea is that ANC can be the basis of security itself. However the possible criticism of their work is due to the fact that they use convolutional neural networks (CNN) so they have not shown what cryptographic system their system has learned. And they not did really talk about the security system that the model learned.

The main objective in this work is to review how Artificial Neural Network (ANN) can learn some instance of the One Time Pad(OTP) algorithm to ensure the securitry of the cryptosyem and present the condition to improve the security of cryptosystem define by Murilo Coutinho. The One Time Pad (OTP) is the only one in cryptography the unbreakable algorithm, so we just want to use it to show that the AI can learn OTP and not to break it.

To reach this goal we will explain three possible cases with the agorithm for a simple neural network to learn some instance of the One Time Pad (Murilo Coutinho) to finally ensure the communication. The first case will consist of showing two simple neural networks such as Alice and Bob to learn to communicate without any adversary, they will communicate without a cryptographic scheme. But the cryptography scheme learned is not ensured. Then we test with the ANC model proposed by Abadi and Andersen , this will show that they can learn to communicate with a weak security encryption scheme. Finally we train Alice and Bob to learn how to communicate with the CPA-ANC model proposed by (Murilo Coutinho), this will show that they can communicate securely with high probability using OTP.

4.1 Notion of Artificial Neural network in Cryptography

Since the construction of neural networks in 1957 the example of the simple perceptron made by Frank Rosenblatt which allowed classification of data into two classes or solving other linear problems. But the limitations of the perceptron noted by Minsky imply that the perceptron cannot solve nonlinear problems. The simplest of neural networks cannot even compute XOR ($\oplus$). which is the basis of many cryptographic algorithms. With the advance of research, several models have been developed to solve non-linear problems.

We explain how neural networks can learn to protect the confidentiality of their data compared to other neural networks and become familiar with encryption-decryption using the OTP algorithm cryptography in (Murilo Coutinho). This is very promising because sometimes it is enough to know how to encrypt to protect information. What is even more interesting is that neural networks can also learn. Now we will present the system ANC (Abadi and Andersen) and explain how its works.

4.2 Adversarial Neural Cryptography

In this part we will describe the ANC model presented in (Abadi and Andersen).

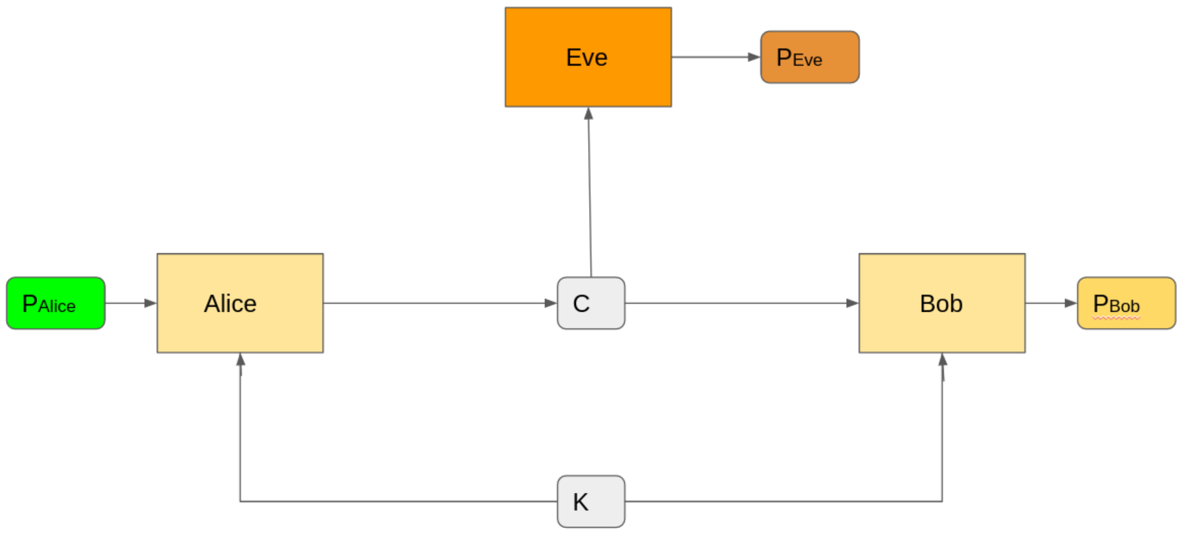


Figure 4.1: Adversarial Neural Cryptography

The ANC system is designed with classical cryptography in which we have three parties involved, namely Alice, Bob and Eve. Alice wishes to communicate an N-bit, P(plaintexts) message to Bob in a secure manner, so they share an N-bit key (which can be thought of as a password) to keep the message secret by encrypting and decrypting it. Here, the length of the message is equal to the length of the password. On the other hand, Eve does not have access to this key.

in the Figure 4.1,P is the plaintext message, C is the encrypted message and K is the share key between Alice and Bob. Alice takes the message and the key (P_{Alice}, K), and encrypts the message, producing an N-bit C cipher text. Bob receives the ciphertext and tries to decrypt it, getting an extra P_{Bob} for each ciphertext there is a key that is used K. Eve being a passive opponent can intercept the encrypted message C sent by Alice. She then decrypts the message herself without using the key, getting the recovery message P_{Eve}. These three parts are neural networks and very similar(same structure).

Mathematical Formular And Objective Function of ANC model

We describe here the ANC model , Alice , Bob and Eve are considered as neural networks with parameters, $\theta_A, \theta_B, \theta_E$, this is learning parameters of the neural network,

such as Alice output $E_A(\theta_A, P, K)$, Bob output $D_B(\theta_B, C, K)$and Eve output $D_E(\theta_E, C)$

To measure the distance between the plaintext message and the predict message defined by the distance.

$$d(P, P^{'}) = \frac{1}{N} \sum_{i=1}^{N} \mid P_i - P_i^{'} \mid \tag{4.2.1}$$

Eve's objective is to reconstruct the message with high precision. Then we can define Eve's loss function.

$$L_E(\theta_A, \theta_E, P, K) = d(P, D_E(\theta_E, C))$$
$$L_E(\theta_A, \theta_E, P, K) = d(P, D_E(\theta_E, E_A(\theta_A, P, K)))$$

Intuitively $L_E(\theta_A, \theta_E, P, K)$ explain how much Eve is wrong when the plaintext is P and and Key is K.

Denoted by $L_E(\theta_A, \theta_E)$ the global loss function over the distribution of plaintexts and the keys, $L_E(\theta_A, \theta_E)$ is naturaly the average over the plaintexts and the keys.

$$L_E(\theta_A, \theta_E) = \mathbb{E}_{P,K}[L_E(\theta_A, \theta_E, P, K)]$$

$$L_E(\theta_A, \theta_E) = \mathbb{E}_{P,K}[d(P, D_E(\theta_E, E_A(\theta_A, P, K)))]$$
$$L_E(\theta_A, \theta_E) = \mathbb{E}_{P,K}[\sum_{P_i, K_i} \mid P_i - D_E(\theta_E, E_A(\theta_A, P_i, K_i)) \mid]$$

$$L_E(\theta_A, \theta_E) = [\sum_{P_i, K_i} \mathbb{E}_{P,K} \mid P_i - D_E(\theta_E, E_A(\theta_A, P_i, K_i)) \mid]$$

Thus, the optimal Eve is found by minimizing the global loss function over parameters θ_E, θ_A

$$\Theta_E(\theta_A) = \arg\min_{\theta_E} L_E(\theta_A, \theta_E)$$

We note $\Theta_E(\theta_A)$ because the parameter of Eve depend on the parameter of Alice. We also describe Bob's loss function similar to that of Eve.

$$L_B(\theta_A, \theta_B, P, K) = d(P, D_B(\theta_B, E_A(\theta_A, P, K), K))$$
$$L_B(\theta_A, \theta_B) = \mathbb{E}_{P,K}[L_E(\theta_A, \theta_B, P, K)]$$

Alice and Bob want to communicate while hiding their communication from Eve. So we will define the loss function of Alice and Bob.

$$L_{AB}(\theta_A, \theta_B) = L_B(\theta_A, \theta_B) - L_E(\theta_A, \Theta_E(\theta_A)) \tag{4.2.2}$$

So Alice and Bob's Optimal function by minimizing $L_{AB}(\theta_A, \theta_B)$

$$(\Theta_A, \Theta_B) = \arg\min_{(\theta_A, \theta_B)}(L_{AB}(\theta_A, \theta_B)) \tag{4.2.3}$$

In the ANC model Abadi and Andersen showed that neural networks could communicate by encrypting and decrypting the message. But the model they learned said nothing about the security of the system. So for a good security system we put the strong opponent Eve into the ANC model (Murilo Coutinho) to force Alice and Bob to secure the cryptosystem. The name of this model is Chosen-Plaintext Attack-Adversarial Neural Cryptography (CPA-ANC).

4.3 Learning OTP with Chosen-Plaintext Attack Adversarial Neural Cryptography

In this part, we present an enhancement of ANC proposed by Murilo Coutinho, they adding the Chosen Plaintext Attack (CPA) type to ensure the robustness of the cryptosystem. Eve's work is much harder because she tries to guess a random message produced by Alice without knowledge of a key. While Alice and Bob make almost no effort to communicate and protect their communication from Eve. So it is good to know if it is possible to improve ANC consider a much more robust security model for Alice, Bob and Eve.

To explain this Eve have the choice to set up a CPA attack. So, Alice and Bob, to protect themselves against Eve they will have to find a much safer security system against the CPA attack. Fig. 4.2.

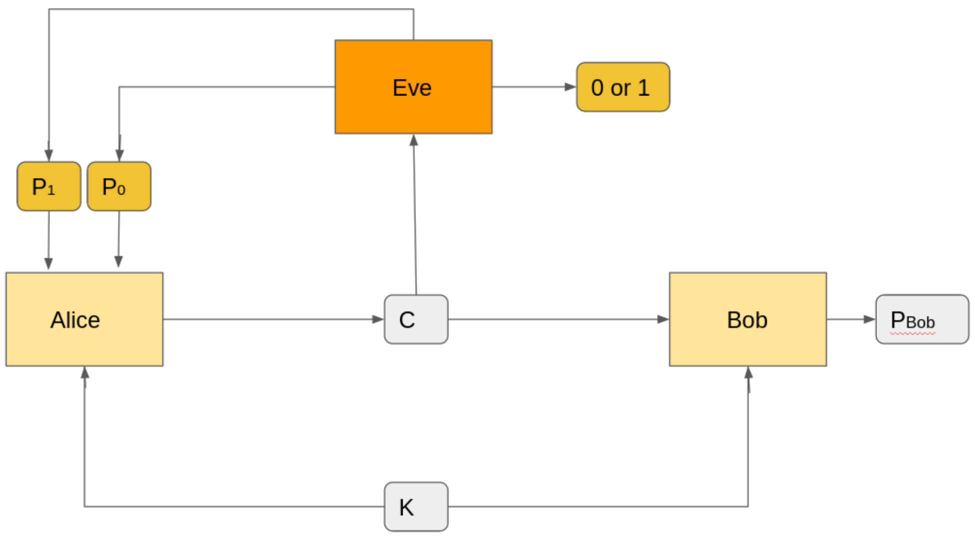


Figure 4.2: CPA-ANC

Note that Alice, Bob and Eve are neural networks(NN). Eve will choose two messages P_0 and P_1 and send both messages to Alice. Then Alice randomly chooses a message, then encrypts the message by producing the encrypted message C and send it to Bob and Eve, Bob must be able to decrypt the message using the key shared between him and Alice.

However, Eve will not try to decrypt the message produced by Alice, but rather she will try to guess the message is said to be 0 in output if it believes the message to be encrypted is P_0 or 1 if it believes the message to be encrypted is P_1. In this method, Alice and Bob must find a much better cryptosystem to protect their communication against Eve. Now we need for Neural Network (Alice, Bob and Eve) can learn the One Time Pad algorithm, we will explain a simple NN capable to learn the OTP (Murilo Coutinho).

4.4 Cryptography Network capable of learning the One Time Pad(CryptoNet)

We know that operation XOR is widely used in cryptography. For our NN to learn operation XOR we must generalize this operation. It is possible to generalize XOR by using the circle of the unit with the bits 0 at the angle 0 and 1 at the angle π. So the operation XOR is equivalent to the sum of the angles and the sum is a continuous operation. We can work with angle 0 and π.

The function used to generalize the XOR operation proposed by Murilo Coutinho:

$$f(b) = \arccos(1 - 2b) \tag{4.4.1}$$

b takes the values $\{0, 1\}$ and the inverse of f is as follows :

suppose that

$$\begin{aligned} f(b) = y \Leftrightarrow \arccos(1-2x) &= y \\ 1-2x &= \cos(y) \\ 2x &= 1 - cos(y) \\ x &= \frac{1-\cos(y)}{2} \end{aligned}$$

So the inverse function is:

$$f^{-1}(a) = \frac{1-\cos(a)}{2} \tag{4.4.2}$$

Example 4.4.1. Let define:

$f : \{0,1\} \longmapsto \{0,\pi\}$

$f^{-1} : \{0,\pi\} \longmapsto \{0,1\}$

$$f(b) \in \{0,\pi\} \begin{cases} if \quad b=0, f(0) = \arccos(1-2\times 0) = 0 \\ if \quad b=1, f(1) = \arccos(1-2\times 1) = \pi \end{cases}$$

$$f^{-1}(a) \in \{0,1\} \begin{cases} if \quad a=0, f^{-1}(0) = f^{-1}(a) = \dfrac{1-\cos(0)}{2} = 0 \\ if \quad a=\pi, f^{-1}(\pi) = f^{-1}(a) = \dfrac{1-\cos(\pi)}{2} = 1 \end{cases}$$

x	y	$x+y$	$f(a)^{-}1$
0	π	π	1
π	0	π	1
π	π	2π	0
0	0	0	0

x	y	$x \oplus y$
0	1	1
1	0	1
1	1	0
0	0	0

We can see the table the sum of the angles is equivalent to the operation XOR. Now we will explain the principle of how ANN capable of learning some instances of the One Time Pad.

Principe:

The algorithm CryptoNet it takes as input the bits of the plaintext vector $[P_0, P_1..., P_{n-1}]$ and the bits of the key vector $[k_0, k_1..., k_{n-1}]$ we apply the function on the bits and the transformation of all bit gives angles $[a_0, a_1..., a_{n-1}]$. The fully connected layer combines the angles to form the variables $[h_0, 1_1..., h_{n-1}]$ we can call this the sum of angles and we apply the inverse of the function on the combination of its angles, this transformation will give bits([0.,1.]) $[C_0, C_1..., C_{n-1}]$ which represents our encrypted message. The result of encryption will be a float number.

For the Fully connected layer :

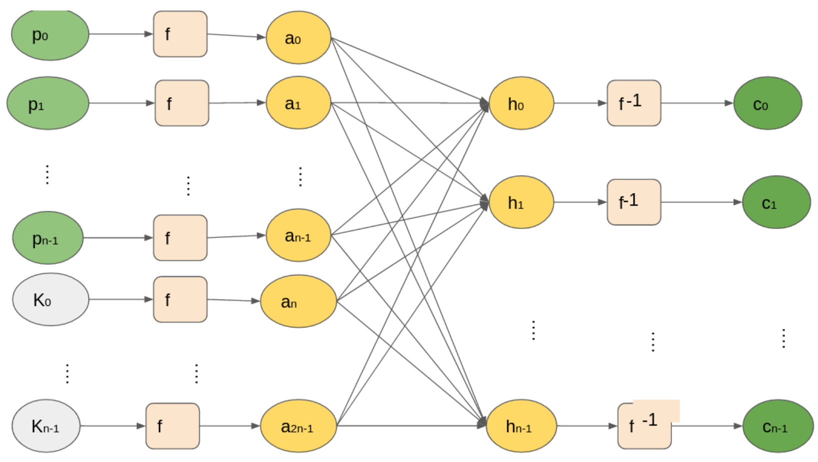


Figure 4.3: CryptoNet

$$\begin{pmatrix} h_0 \\ h_1 \\ \vdots \\ h_{n-1} \end{pmatrix}^T = \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \\ a_n \\ \vdots \\ a_{2n-1} \end{pmatrix}^T W_{2n,n},$$

Where $W_{2n,n}$ is the matrix of the weights with $2n$ rows and n columns, we define The Mathematical Formulation for CryptoNet such that :

$$C = \xi_n(W, P, K) \tag{4.4.3}$$

This CryptoNet can learn some instance of the OTP

Method to Learn OTP

In this part, we describe three different scenarios to reach the objective that is said to be communicated with a very high probability of security. First Alice and Bob learn to communicate without an adversary. And without an encryption scheme. Then Alice and Bob learn to use the ANC system but with a weak encryption scheme. Finally, Alice and Bob learn with CPA-ANC. They are able to communicate very securely and with a high probability using the One Time Pad.

Description Of the Method

We have two neural networks, Alice and Bob, using the same CryptoNet in equation (4.1.6). Alice and Bob here is to communicate in a single network. So Alice and Bob form a single network, define by:

$$E(B,K) = D(B,K) = \xi_n(W,B,K) \tag{4.4.4}$$

Where $E(B,K)$ is encryption of Alice and $D(B,K)$ is decryption of Bob. Since Alice and Bob are working together, then

$$\xi_n(W,\xi_n(W,P,K),K) = P \tag{4.4.5}$$

Where P is the plaintext and K is the key and W is the matrix of weigths.

For the test, W is the matrix of weight and it initialized randomly. To train the networks like in (Abadi and Andersen). I would like to point out that the process is time-consuming for the large keys and message, so we train the model with the small keys: 4-bit(n=4), 8-bit(n=8), 16-bit(n=16).

The concept of convergence is not defined here because when the Network of Eve change, the objective function of Alice and Bob also change. We have two criteria define by Murilo Coutinho such that :

1. Bob's decryption error is very close to zero and Eve's attacks are as bad as random guesses, then we stop (onvergence or success).
2. If the first stop criterion is not met in 100,000 revolutions, we stop(convergence or a failure.

We train this model using pytorch is a machine learning framework for python. We use Adam optimizer with learning rate =0.0002. To test the model we use the Algorithm 1.

```
Data: A training continuous CryptoNet $\xi_n(\mathbf{W},P,K)$
Result: Success or Failure
$\mathbf{W}_{round} \longleftarrow (\mathbf{W})$ round($\mathbf{W}$) transforms each entry of $\mathbf{W}$ to its closest integer
$\xi_n(\mathbf{W}_{round},P,K) \longleftarrow$ substitution of $\mathbf{W}$ by $\mathbf{W}_{round}$
for i=1 To T do
    if $\xi_n(\mathbf{W}_{round},\xi_n(\mathbf{W}_{round},P,K),K) \neq P$ then
        return Faillure
    else
        return Success
    end
end
```

Algorithm 1: Testing a discrete CryptoNet (Murilo Coutinho)

In this algorithm the input is the CryptoNet $\xi_n(\mathbf{W},P,K)$, after training we get the weigth that gives the good prediction. Recall the weigth is generated randomly and if we have finished the training and we have the weigth, we apply the function round to convert the weigths in integer number. Because when we train the model we initialized the weigth with float numbers. After converting the weiths to integer we substitute the float weigth in CryptoNet $\xi_n(\mathbf{W},P,K)$ by $\xi_n(\mathbf{W}_{round},P,K)$. Now we will use the same CryptoNet to decipher the message encrypted by the same CryptoNet like $\xi_n(\mathbf{W}_{round},\xi_n(\mathbf{W}_{round},P,K),K)$. Finally we compare the decipher to the plaintext, if the decipher is plaintext we says it's success else It's faillure.

4.5 Implementation

Training Without An Adversary

In this case we have two agent Alice and Bob, to communicate without opponent. We know Alice and Bob are the Neural Network. So we use the forward communication. And M is message generated randomly $[P^{(0)}, P^{(1)}, ..., P^{(M-1)}]$ and the keys $[K^{(0)}, K^{(1)}, ..., K^{(M-1)}]$. we use batch size is 4096 and minibatch is 512 like in (Abadi and Andersen) and we use Adam optimizer with the learning rate 0.0002 and we test for n=4-bit , n=8-bit and n=16-bit.

Alice and Bob want to communicate without an opponent (Eve) the code is explain in (Mohamed). And they work togather on a single network. So we only minimize Alice and Bob's loss function.

$$L_{AB} = \frac{1}{M}\sum_{i=0}^{M-1} d(P^{(i)}, \xi_n(W, \xi_n(W, P^{(i)}, K^{(i)}), K^{(i)})) \tag{4.5.1}$$

Training with ANC model

In this step, we introduce an adversarial. Because the first experience show in section result 4.6 Alice and Bob can communicate without error, but the model was learned could not find a secure network and without any notion of security And the next step Alice and Bob will communicate by a single neural network, and Eve will try to learn using the same NN, the input of Eve is ciphertext C and the output is plaintext predicted by Eve P.

$$P = D_E(\theta, C) \tag{4.5.2}$$

M example plaintexts $[P^{(0)}, P^{(1)}, ..., P^{(M-1)}]$ and the cipthertexts $[C^{(0)}, C^{(1)}, ..., C^{(M-1)}]$ loss function that Eve must minimize is :

$$L_E = \frac{1}{M}\sum_{i=0}^{M-1} d(P^{(i)}, D_E(\theta, C^{(i)})) \tag{4.5.3}$$

since Eve doesn't have plaintext so Eve applies a plaintext attack only. Alice and Bob want Eve not to understand the message, so the function of Alice and Bob that want to minimize to hide the message from Eve:

$$L = L_{AB} - \gamma \min(L_E, \frac{1}{2}) \tag{4.5.4}$$

Where L_{AB} is the loss function define in equation (4.5.1) and γ is hyperparameter(this parameter explains that it is very important to keep Eve's Error high than to reduce Bob's decryption error.). This loss function(minimum function) is to guarantee Alice and Bob would not try to maximize the loss function of Eve, because if Alice and Bob try to maximize the loss function of Eve. If Eve is wrong, Eve could just flip all bits in the next round achieving a correct guess (Murilo Coutinho).

In the test we train the networks with M random example plaintexts $[P^{(0)}, P^{(1)}, ..., P^{(M-1)}]$ and $[K^{(0)}, K^{(1)}, ..., K^{(M-1)}]$. We use the minibatch $M = 128$, the hyperparameter $\gamma = 5$,this hyperparameter means it's better to keep the error of Eve high than to reduce the decryption error of Bob. And we use L2 regularization explain in (Murilo Coutinho) with $\alpha = 0.1$(4-bits),$\alpha = 0.05$(8-bits) and $\alpha = 0.01$(16-bits). When we train Alice and Bob on five minbatches, we train Eve on 10 minibatches. Because she didn't take the key, so she need more computation than Alice and Bob. See code source of the implementation in (Mohamed)

Learning the One time with CPA-ANC

In this section we will explain the network of Eve and how it works like learn some instance of OTP algorithm in cryptography to force Alice and Bob network to communicate securely.

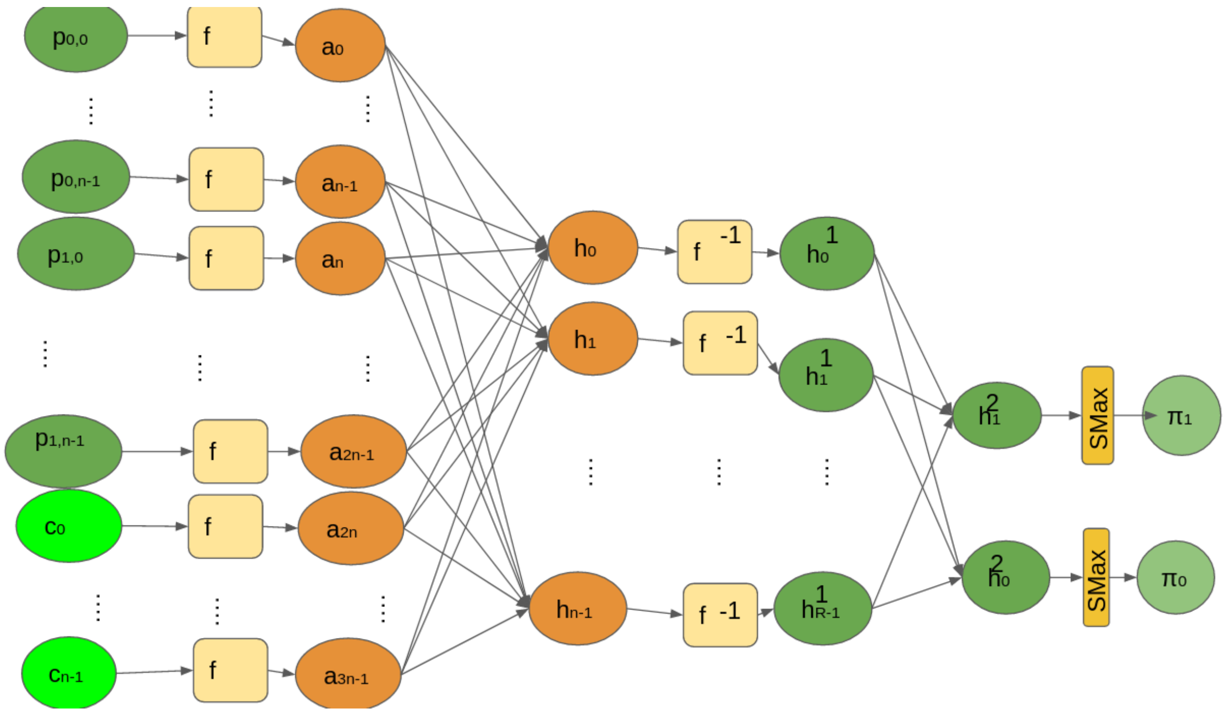


Figure 4.4: Network Eve

We know that Alice and Bob sharing on the single network(CryptoNet), to encrypt and decrypt. We talk before the network of Eve change a lot. In this experience Eve is a classifier, She wil take as input two plaintext like p_0, p_1, let say $p_0 = [p_{0,0}, p_{0,1}, .., p_{0,n-1}]$,$p_1 = [p_{1,0}, p_{1,1}, .., p_{1,n-1}]$ and she take the ciphertext $C_0 = [C_0, C_1, .., C_{n-1}]$.

Eve will try to classify the ciphertext C as 0 if it believes C come from p_0 and 1 if C come from p_1. Eve has the same structure of network (CryptoNet). The Network of Eve receives two plaintext p_0, p_1 and the ciphertext C and she apply the function f see the equation (4.4.1) to transform all the bit in angle between $\{0, \pi\}$, $f(a) \in \{0, \pi\}$ and $a \in \{0, 1\}$.

After transformation of all the input, A fully connected layer combine all the angle transformed by the function f in each neuron of the first hidden layer (Figure 4.5). The combination of each are called the total rule R, is composed by the generalization of the operation XOR. see equation (4.4.1). After combination of each angle into a total rule R,we apply the inverse function to transform each total rule R in the continuous bits.(4.4.2). After transformation in continuous bits , the second fully connected layer combine the results into logits applying a softmax layer. The result would be a probability like π_0 of ciphertext C being a ciphertext of P_0 and probability of π_1 of C being a ciphertext of P_1. Then the output of the network of Eve is define by :

$$\xi_n(\mathbf{W}, P_0, P_1, C) = \begin{cases} 0 & if \quad \pi_0 > \pi_1 \\ 1 & \quad otherwise \end{cases}$$

To learn some instance of OTP algorithm Eve will work using the network defined in Figure (4.5). To do this Eve's will minimize loss function cross entropy define by :

$$L_E = -\frac{1}{M}\sum_{i=0}^{M-1}\sum_{j=0}^{1}\lambda_j^i \log(\pi_j^i) \tag{4.5.5}$$

Where $\lambda_j^i = 1$ if C^i is the ciphertext of π_j^i, and $\lambda_j^i = 0$ otherwise. Eve will learn by minimizing L_E When Alice and Bob will try to minimize the function to hide the plaintext from Eve.

$$L = L_{AB} - \gamma \min(L_E, \frac{1}{2}) \tag{4.5.6}$$

L_{AB} is the loss function of Alice and Bob, L_E is the loss function of Eve. The code of implementation refer in (Mohamed)

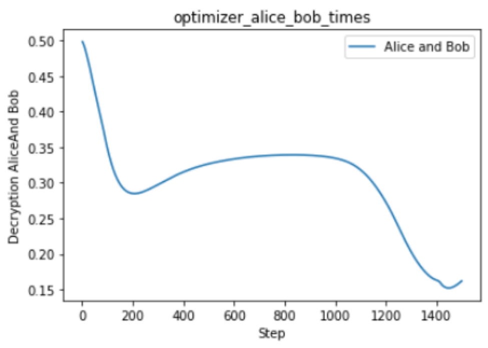
4.6 Results

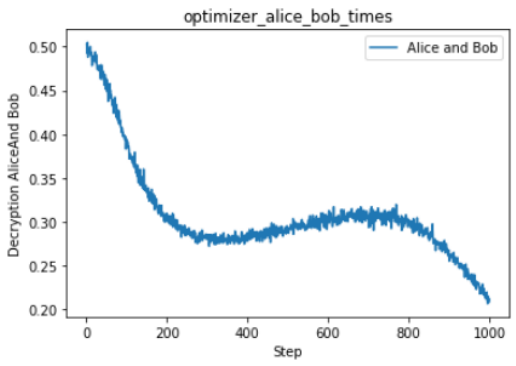
In this part, we will present differents results of these three cases we mentioned before see (section Implementation) finally to fix our objectives.

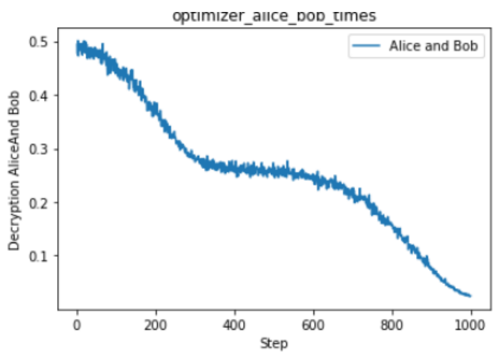
Training The Model Alice and Bob without an Adversary

For the first experience we can see the decryption errors when the size of the keys are (n=4,n=8 and n=16). The decryption decrease for all n in Figure 4.5 when Alice and Bob try to communicate.

After training the networks Alice and Bob can communicate without errors. All communication was successful. But about the security Alice and Bob arrive to communicate, but the communication is not secure and the networks can't learn some instance of One Time Pad, so the







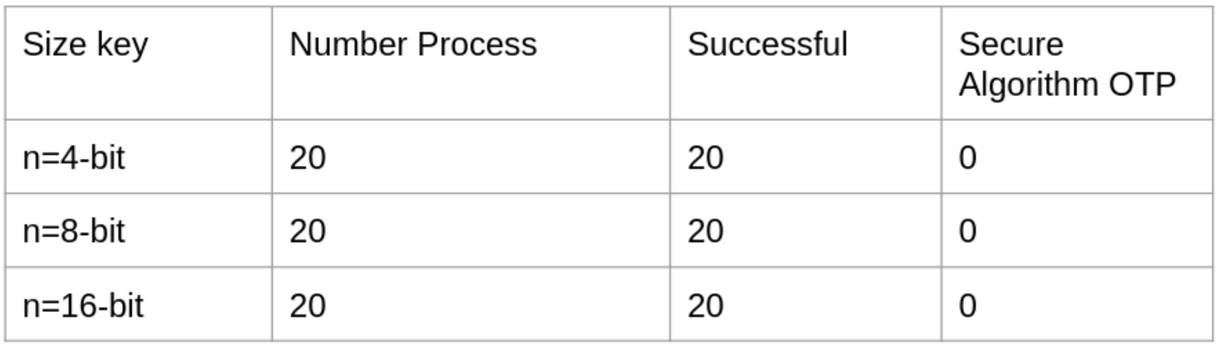
cryptosystem is not secure. The model learn some cryptography technique, but not some instance of OTP to encrypt the message. We can see the result of one process using the algorithm 1.

$$\xi_n(W_{round}, P, K) = \begin{bmatrix} p_2 \\ p_1 \\ p_3 \\ p_4 \end{bmatrix}^T$$

For more explanation we test the Algorithm for 10 trials, we can see all is successful but if we look clearly the model learned another scheme of cryptography permutation. we can say this system is not secure because a CryptoNet learned model is considered secure if it learned the OTP. For example, Alice and Bob learned the following model from CryptoNet . But this system don't learn the One Time Pad.

No	Plaintext Alice	Ciphertext	Plaintex Bob
0	[0. 0. 0. 1.]	[0., 0., 0., 1.]	[0. 0. 0. 1.]
1	[0. 1. 1. 1.]	[0., 1., 1., 1.]	[0. 1. 1. 1.]
2	[1. 0. 1. 1.]	[1., 1., 0., 1.]	[1. 0. 1. 1.]
3	[0. 0. 1. 1.]	[0., 1., 0., 1.]	[0. 0. 1. 1.]
4	[1. 1. 0. 1.]	[1., 0., 1., 1.]	[1. 1. 0. 1.]
5	[1. 0. 1. 0.]	[1., 1., 0., 0.]	[1. 0. 1. 0.]
6	[0. 1. 0. 0.]	[0., 0., 1., 0.]	[0. 1. 0. 0.]
7	[1. 1. 0. 1.]	[1., 0., 1., 1.]	[1. 1. 0. 1.]
8	[0. 1. 1. 0.]	[0., 1., 1., 0.]	[0. 1. 1. 0.]
9	[0. 0. 0. 0.]	[0., 0., 0., 0.]	[0. 0. 0. 0.]

The CryptoNet obtained is quite simple, so we can easily reason about its security without the help of an NN. see the table the CryptoNet can't leran the One Time Pad.



Size key	Number Process	Successful	Secure Algorithm OTP
n=4-bit	20	20	0
n=8-bit	20	20	0
n=16-bit	20	20	0

Figure 4.5: Test the netowrk Alice and Bob with Alogorithm 1

All process was successful without any error Alice and bob can communicate clearly without error. But the system is not secure.

Training with ANC Model

In this step we introduce an adversarial call Eve, Alice and Bob will try to communicate and Eve will try to undertand the message, she's apply only ciphertext attack because she's a passive attacker.

After training of Alice, Bob and Eve, we can see the decryption error of Alice, Bob and Eve. Alice

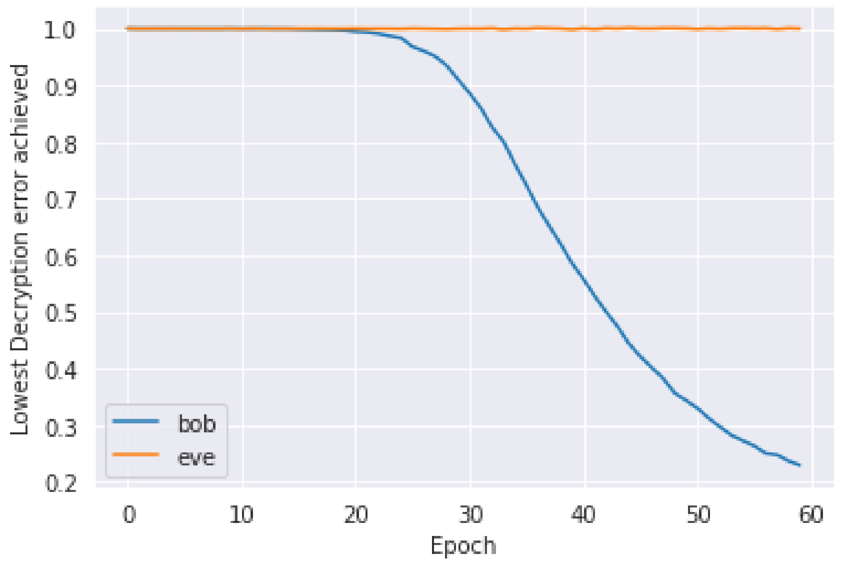


Figure 4.6: ANC Model

and Bob can commnucate. At the Begining Eve can understand the communication after some iteration she can't understand the communication on the figure 4.6

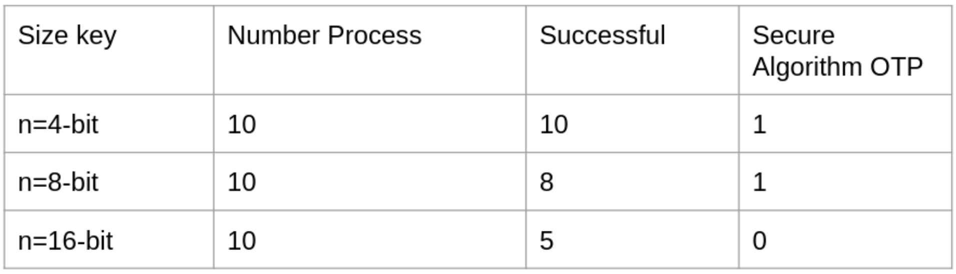
We can see the result of one process using the algorithm 1.

$$\xi_n(W_{round}, P, K) = \begin{bmatrix} p_1 \oplus k_0 \\ p_0 \oplus k_1 \oplus k_2 \oplus k_3 \\ p_2 \oplus k_3 \\ p_3 \oplus k_2 \end{bmatrix}^T$$

No	Plaintexts Alice	keys	Ciphertexts	Secure OPT
0	[1. 1. 1. 1.]	[0. 0. 1. 1.]	[1., 1., 0., 0.]	1
1	[0. 0. 1. 0.]	[0., 1., 1., 1.]	[0., 1., 1., 1.]	0
2	[1. 1. 0. 0.]	[1. 1. 1. 0.]	[1., 0., 1., 1.]	0
3	[0. 0. 0. 0.]	[0. 1. 1. 1.]	[0., 1., 0., 0.]	0
4	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1., 1., 1., 0.]	0
5	[0. 1. 1. 0.]	[0. 0. 1. 1.]	[0., 1., 0., 1.]	0
6	[0. 0. 0. 1.]	[1. 0. 1. 0.]	[0., 0., 0., 1.]	0
7	[0. 0. 1. 1.]	[1. 0. 0. 0.]	[0., 1., 1., 0.]	0
8	[1. 1. 1. 0.]	[1. 0. 0. 0.]	[1., 0., 1., 1.]	0
9	[1. 1. 0. 1.]	[0. 0. 1. 1.]	[1., 1., 1., 0.]	1

We just analyse if the model learn some instance of One Time Pad , used the Algorithm for 10 trial, all communication was successfull in some case but the cryptosystem is not secure. We ave some case the model learn some instance of One Time Pad with a weak probability. So this cryptosystem is not good.

After training we tested the model using the Algorithm 1 if the security is good.



Size key	Number Process	Successful	Secure Algorithm OTP
n=4-bit	10	10	1
n=8-bit	10	8	1
n=16-bit	10	5	0

Figure 4.7: Test the netowrk Alice and Bob with Alogorithm 1

We can see all process was successful in some case without any error only Alice and bob can communicate clearly. But the cryptosystem is not secure. The model learn in some case some instance of One Time Pad with a weak probability.

Learning the One time with CPA-ANC

In this part we want to show in this Alice And Bob can communicate securly. Eve is a strong adversarial that would force Alice and Bob to communicate securly. Because in the previous test like the model ANC where Alice and Bob would communicate and Eve will try to understand the communication (message) sending on the insecure channel.

In this experience Eve will not try to decrypte the message, but she will just guess if the message encrypted by Alice is P_0 or P_1 until we reconstruct the plaintext. We show that the model ANC (Abadi and Andersen) is not suficient to ensure the security of the cryptosystem and can't realy learn some instance of the algorithm OTP in some case (weak probability) see previous test. So we took a strong adversarial attacker called Eve. Eve will force Alice and Bob to communicate securely on the channel using some instance of OTP.

For the training part of the networks (Murilo Coutinho), we use a minibatch of $M = 128$, and the hyperparameter $\gamma = 7$ in equation 4.5.6.

We use the $L2$ regularization with (Goodfellow) :

$$\begin{cases} n = 4 - bit \quad , \quad \alpha = 0.1 \\ n = 8 - bit \quad , \quad \alpha = 0.015 \\ n = 4 - bit \quad , \quad \alpha = 0.015 \end{cases}$$

The network of Eve is the same structure like the network of Alice and Bob, but we add the second fully connected layer (see the Figure 4.5), we define the number of rules $R = 4 \times n$ in (Murilo Coutinho). The number of rules define the number of linear combinations that Eve can analyse together to attack.

The network becomes complex network when the key size increased. The strong NN of Eve depend of the number of the length of the key. We increase the number of linear combination until we get the value where Eve get the great powerfull to break the cryptosystem learning by Alice and Bob througth by CryptoNet to get the value $4 \times n$.

To train the Neural Network, Alice and Bob training for three (3) minibatches describe in (Murilo Coutinho) and for Eve Neural Network she will training for 60 minibatches. we give a computational advantage to Eve code source in (Mohamed).

We can see the decryption error of Bob's, Alice and Bob will try in this case to minimize Bob's decryption error and minimize Eve's classification rate. Bob decryption error decrease over time. Eve classification error increase in red color. when Alice and Bob learn a secure cryptosystem, in this case some instance of One Time Pad. Eve classification is no better than random.

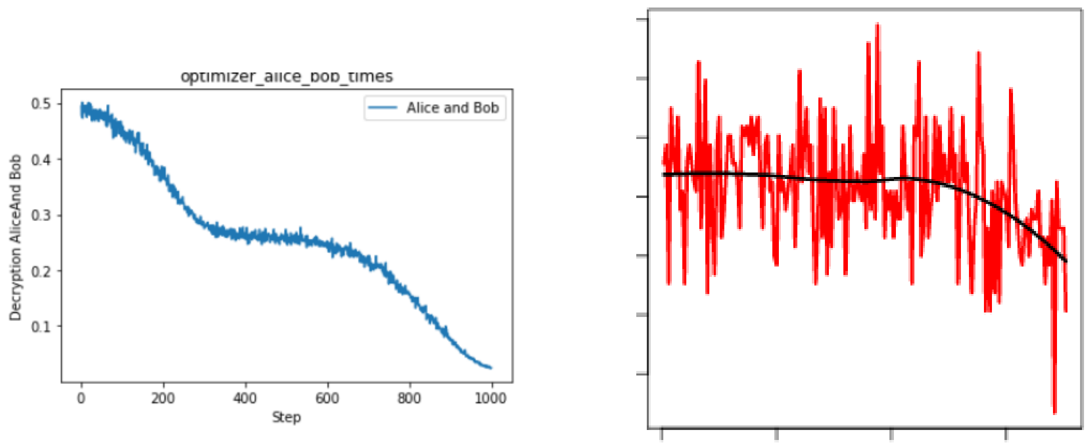


Figure 4.8: Bob's Decryption Error and Eve classification rate((Murilo Coutinho))

For 10 trials using the Algorithm 1 for each key size. Most of the trials was successul,Alice and Bob can communicate without error. Most of successful for our model in this case the networks learn the OTP, with a high probability.

Size key	Number Process	Successful	Secure Algorithm OTP
n=4-bit	10	10	8
n=8-bit	10	10	9
n=16-bit	10	10	10

Figure 4.9: CPA-ANC learn OTP

Alice and Bob can communicate securely if we have strong adversary. The adversary will force Alice and Bob to secure communication. So the CPA-ANC model proposed by Murilo Coutinho have high probability to find the secure cryptosystem than the original model proposed by Abadi and Andersen with a weak probability.

Limitation

The main limitation of this work is due to the size of the key. In our work we took keys length for $n = 4, n = 8, n = 16$. The size of the key and the message plaintext its like the same. When the size of key is big it consumse more time.

5. Conclusion

In this work, we have review how artificial neural networks can communicate securely on the insecure channels using some instances of One Time Pad. We have seen in which condition artificial neural networks can communicate securely.

We have first show that Alice and Bob can communicate without an error, but the cryptosystem learned is not secure, because some of this cryptosystem is indeed permutations.

Secondly, we have shown that if we introduce the notion of an adversarial, Alice and Bob can communicate clearly without error. But the cryptosystem generated is not secure, some of this cryptosystem is indeed Vignere cipher. We show in our experience all processes, the communication was successful in some cases and learn some instances of One time Pad in some cases with a weak probability. So the ANC model is not good. Finally, we have shown the security of the CPA-ANC model is good. all process to learn was an OTP with high probability, the cryptosystem is secure. So the general message here to get good security with high probability, The adversarial must be very powerful to force the system into a strong cryptosystem.

The limit of these models is due to the size of the key. For further work, one can continue to test the model to evaluate more parameters. And implement a parallel code to increase the performance and test the proposed in the work with the large keys. It remains an open problem whether a neural network can learn a secure cryptosystem in which a small key is used to encrypt a very long message like a block or stream cipher would do. And it would be interesting if we can prove the security of the cryptosystem to ensure the security with the model CCA-ANC(Chosen Ciphertext Attacks-Adversarial Neural Cryptography).

References

P. van Oorschot A. Menezes and S. Vanstone., editors. *Handbook of Applied Cryptography*. Number 746 in Contemporary Mathematics. 1997 by CRC Press, Inc., Providence, RI, 1992.

M. Abadi and D.G. Andersen. Learning to protect communications with adversarial neural cryptography. *arXiv preprint arXiv:1610.06918.*

Omar Aflak. Math of neural networks — from scratch in python. *medium*.

MOHAMMED M. ALANI. Applications of machine learning in cryptography: A survey. *arXiv*.

Marc Duquesnoy Aurelien Pottiez. *Intelligence artificielle et apprentissage de réseaux connexionnistes*. PhD thesis, Universite de Lille, 2018.

Patrick van der Smagt Ben Krose. *An introduction to Neural Network*. Eighth edition, 1996.

Benzerrouki Aicha Essedikia,Guemidi Zoulikha. Application des systèmes chaotiques à la cryptographie. Memoire.

Johannes A. Buchmann. *Introduction to Cryptography*. Springer-Verlag New York Berlin Heidelberg, 2002.

et al C. Dwork, A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9:211–407, 2014.

PGP Corporation. *An Introduction to Cryptography*. PGP Corporation, 2002.

crypto. La cryptographie de l'antiquité à l'internet. François Bergeron et Alain Goupil,Université du Québec à Montréal.

WHITFIELD DIFFIE and MARTIN E. HELLMAN. New directions in cryptography. *IEEE*, 22, 1976.

Renaud Dumont. *Cryptographie et Securite informatique*. Universite de Liege, 2010.

Y.; Courville A Goodfellow, I.; Bengio. Deep learning. *MIT Press: Cambridge, MA, USA, 2016. Available online: http://www.deeplearningbook.org*.

A.R.; Hinton G Graves, A.; Mohamed. Hinton, g. speech recognition with deep recurrent neural networks. *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2:6645–6649, 2013.

Donald Olding HEBB. The organization of behavior. *New York, Wiley Sons*.

J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the UnitedStates of America*.

jonathan Katz and Yehuda Lindell. *introduction to Modern Cryptography*. Series Editor Douglas R. Stinson, 2007.

Auguste Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires http://www.petitcolas.net/fabien/kerckhoffs/*, pages 5–38, 161–191, 1883.

Thai Hoang Le. Applying artificial neural networks for face recognition. *Hindawi Publishing Corporation*, 2011:16, 2011.

Yann LeCun. a learning scheme for asymmetric threshold networks. *Proceedings of Cogni-tiva*.

E. F. Shair N. I. Md Saleh and C. Gomes M. M. Mehdy, P. Y. Ng. Artificial neural networks in image processing for early detection of breast cancer. *Hindawi*.

S. Papert M. Minsky. Perceptrons. *MIT Press Cambridge*.

Sylla Mohamed. On perfectly secure cryptography with adversarial neural cryptography. *https://github.com/msylla01/On-perfect-secure.git*.

RICHARD A. MOLLIN. *An INTRODUCTION to CRYPTOGRAPHY second edition*. Taylor and Francis Group, LLC, 2007.

Fabio Borges Murilo Coutinho, Robson Albuquerque. Learning perfectly secure cryptography to protect communications with adversarial neural cryptography. *Sensors*.

R. L. Rivest. Cryptography and machine learning. *Springer Berlin Heidelberg*.

Frank ROSENBLATT. The perceptron : A perceiving and recognizing automa-ton (project para). *Bulletin of Mathematical Biophysics*.

C. Tangha. S. Kouamo. Image compression with artificial neural network. *Advance in Intelligent system and computing*, a.

C. Tangha. S. Kouamo. Fingerprint recognition with artificial neural networks: Application to e-learning. *Journal of Intelligent Learning system And Application*, b.

Y.-B. Sheng and L. Zhou. Distributed secure quantum machine learning,. *Science Bulletin*.

Djiby SOW. Number theory based cryptography part i: Basic notions in cryptography. Technical Report 97, African Institute for Mathematical Sciences, 2019.

Douglas R. Stinson. *CRYPTOGRAPHY Theory and Practice Third Edition*. Taylor and Francis Group, LLC, 2006.

Henk C.A.van Tilborg Eihdhoven university of technology the Netherlands. *Fundamentals of Cryptography*. KLUWERAC ADEMIC PUBLISHERS Boston/Dordrecht/London, 1996.

Alice Land Benoit V. *Panorama des algorithme de cryptographie*. Universite de Nante, 2011.

M. VIDEAU. *Critere de sécurite des algorithmes de chiffrement à cle secrete*. Phd, Université de Paris 6(France), 2005.

Satish K. P. Vikas Gujra. Cryptography using artificial neural networks. *ResearchGate*.

B. Widrow and M.E. Hoff. Adaptive switching circuits. *IRE WESCON Convention Record*, 2: 96–104, 1988.

McCulloch.A W.Pitts W.S. logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*.

X. Xu. Adaptive intrusion detection based on machine learning: feature extraction, classifier construction and sequential pattern prediction. *International Journal of Web Services Practices*, 2:49–58, 2006.

B. YEGNANARAYANA, editor. *ARTIFICIAL NEURAL NETWORKS*.

Z. Yu and J. J. Tsai. A framework of machine learning based intrusion detection for wireless sensor networks. *IEEE International Conference*.