AUTOMATIC DETECTION OF CLASSIC ENCRYPTION METHODS BASED ON NEURAL NETWORK MODEL

Nasriddinov Javokhir
Master's student at the National University of Uzbekistan
E-mail: abdurrahmonumar6@gmail.com
Khushvaqtov Jaloliddin
Master's student at the National University of Uzbekistan
E-mail: jaloliddin0611@gmail.com
Abdusattorov Akbar
3rd year student at Tashkent University of Information Technologies
E-mail: a.sh.abdusattorov@gmail.com

Abstract: This study addresses the issue of automatic detection of text encryption algorithms using artificial neural programs. An artificial neural network model was trained on a dataset based on the classical encryption algorithms Caesar, Vigenere, Gronsfeld, and RailFence (or ZigZag) methods. The model was able to distinguish each encryption method on the test set.

Keywords: Vigenere, Caesar, Gronsfeld. ZigZag, artificial neural network, classical encryption, cryptanalysis

Аннотация: В этом исследовании рассматривается проблема автоматического обнаружения алгоритмов шифрования текста с использованием искусственных нейронных программ. Модель искусственной нейронной сети была обучена на наборе данных, основанном на классических алгоритмах шифрования методами Caesar, Vigenere, Gronsfeld и RailFence (или ZigZag). Модель смогла различить каждый метод шифрования на тестовом наборе.

Ключевые слова: Виженер, Цезарь, Гронсфельд., ZigZag, искусственная нейронная сеть, классическое шифрование, криптоанализ

Annotatsiya: Mazkur tadqiqotda sun’iy neyron tarmoqlar yordamida matnlarni shifrlash algoritmlarining avtomatik aniqlanish masalasi ko‘rib chiqildi. Klassik shifrlash algoritmlari hisoblangan Sezar, Vijener, Gronsfeld va RailFence(yoki boshqacha aytadigan bo‘lsak, ZigZag) usullari asosida yaratilgan ma’lumotlar to‘plamida sun’iy neyron tarmoq model asosida o‘qitildi. Model test to‘plamida har bir shifrlash usulini farqlay oldi.

Kalit so‘zlar: Vijener, Sezar, Gronsfeld. ZigZag, sun’iy neyron tarmoq, klassik shifrlash, kriptotahlil

Introduction

ISSN 3030-3907

Clustering problems are one of the most common problems in artificial intelligence. The choice of loss function is very important when analyzing the results of Deep Learning models.

Classical encryption algorithms are alphabetic encryption algorithms, which are often used for scientific experiments in modern cryptography. Caesar, Vigenere, Gronsfeld algorithms are clear examples of classical encryption algorithms.

LITERATURE ANALYSIS

Caesar. The substitution method is calculated by shifting the letters by the number k in the Caesar method. Julius Caesar himself used this method when k=3. The main problem with this method is that the same letters are replaced by the same letters in turn.

For encryption: $C_i = (m_i + k) mod\ 26$

Decryption: $m_i = (26 + C_i - k) mod\ 26$

Table 1.

The place of letters in the alphabet

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Example. Given the word Hello and k=5. Its encryption is as follows: Since H-8, e-5, l-12, o-15, the number k is added to each element: Ciphertext = {13,10,17,17,20}. These numbers are now converted to letters: MJQQT. The word Hello is encrypted as mjqqt.

Gronsfeld. The Gronsfeld algorithm is a polyalphabetic substitution algorithm. It was created in the 17th century in Germany by Count Gronsfeld. In this method, the chosen key must be equal to the length of the plaintext. If the key length is not equal to the length of the text, it is written repeatedly from the beginning until it is equal. The key must necessarily consist of numbers (0-9). The Gronsfeld encryption works as follows.

For each letter of the plaintext, the corresponding element of the key is obtained. The letter is then mapped to the key number in the alphabet, just like in the Caesar method. If it exceeds the length of the alphabet, then the remainder when divided by the length of the alphabet is obtained. Since the English alphabet is often chosen as the alphabet, the remainder when divided by 26 is often obtained. Mathematical formula: $C_i = (P_i + K_i) mod\ 26$

Here, C is the index of the ciphertext letter, P is the index of the plaintext letter, K is the number in the key

ISSN 3030-3907

Table 2.

Gronsfeld encryption algorithm

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
9	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I

Example: Plaintext=”HELLO”, key=32452.

H+3=>K, E+2=>G, L+4=>P, L+5=>Q, O+2=>Q. Ciphertext: KGPQQ

The decryption process is considered the inverse of the encryption process, and in Gronsfeld it is found by the following formula:

$$P_i = (C_i - K_i) mod\ 26$$

Example: Ciphertext – “KGPQQ”, key =32452:

It is known that K=10, G=6, P=15, Q=16.

$$P_0 = (10 - 3) mod\ 26 = 7$$
$$P_1 = (6 - 2) mod\ 26 = 4$$
$$P_2 = (15 - 4) mod\ 26 = 11$$
$$P_3 = (16 - 5) mod\ 26 = 11$$
$$P_4 = (16 - 2) mod\ 26 = 14$$

After finding that P={7,4,11,11,14}, the exact same characters in the alphabet are taken and the plaintext is found. That is, P=”HELLO”.

Vigenere. The Vigenere cipher is a polyalphabetic cipher, as opposed to a simple substitution cipher, which is monoalphabetic. In a polyalphabetic cipher, the mapping between plaintext and ciphertext is reversed, meaning that multiple ciphertext symbols are used to replace a single plaintext symbol. This means that there is a one-to-many relationship between each letter and its replacement letters.

Table 3.

Vigenere algorithm table

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B

ISSN 3030-3907

C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	P	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Q	Q	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
R	R	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
S	S	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
T	T	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
U	U	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
V	V	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
W	W	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
X	X	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Y	Y	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
Z	Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

RailFence. The ZigZag or Railfence method Rail Fence Cipher is a type of transposition cipher. This means that it encrypts text by changing the order of the letters in the text. In Rail Fence, the key is the number of lines. The following steps are used in the Rail Fence encryption technique.

1. The plaintext message is written as a sequence of diagonals in n lines
2. The plaintext written in step 1 is read as a sequence of lines.

Example:

Plain Text-: MORNING , Key n=2.

M	→	R	→	I	→	G
→	O	→	N	→	N	

Cipher Text- MRIGONN

ISSN 3030-3907

Due to the ease of creating models using the TensorFlow and Keras libraries of the Python programming language, the model was built on them. To create it, the text and keywords provided were converted to numbers using Tokenizer.

20% of the data provided was used for testing, and 80% for training. This model used the softmax activation function and the Adam optimizer.

RESULTS

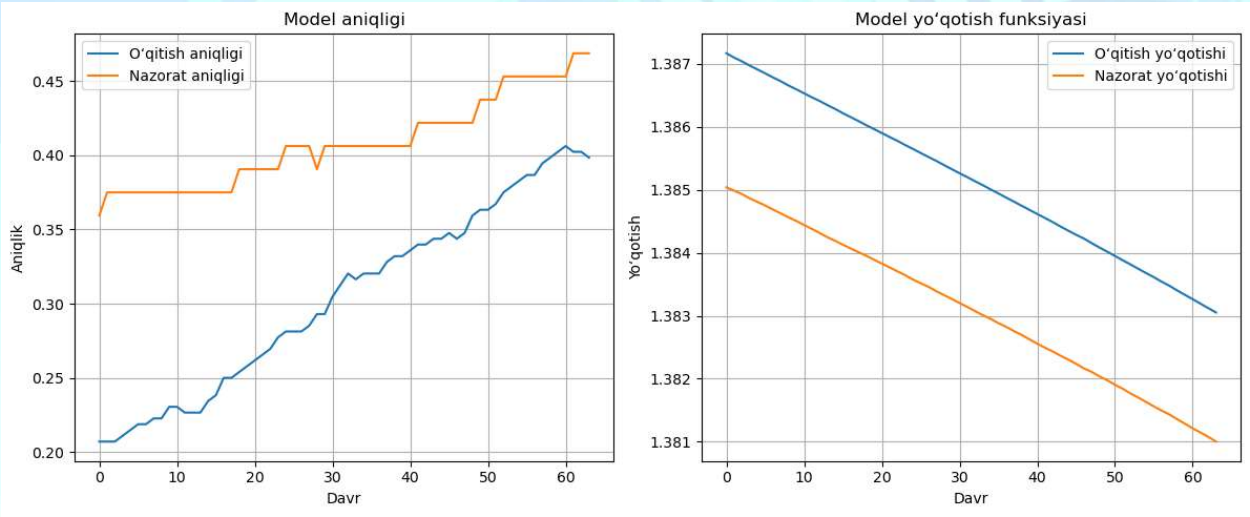


Figure 1. Results graph with 100 elements from each class, epochs=64

In Experiment 1, the dataset created using 4 encryption methods has 100 elements from each method. The epoch here is taken as 64.

As for the results, the ciphertext "KGPQQ" and the key "32452" are given. As a result of the prediction of the model we built, the encryption algorithm was determined to be Gronsfeld. The training accuracy is -40%..

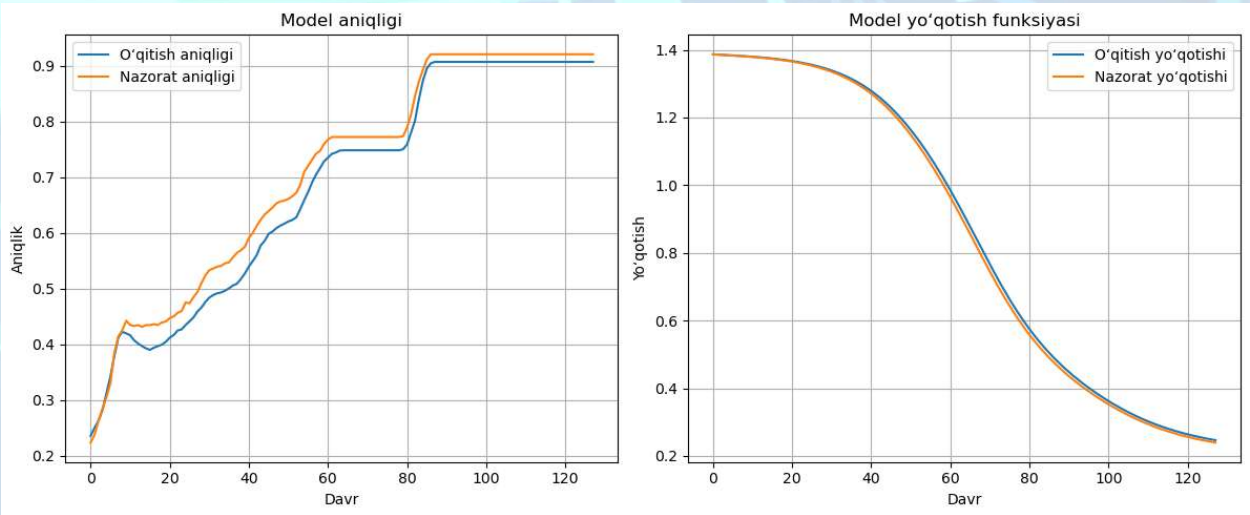


Figure 2. Results graph with 1000 elements from each class, epochs=128

In the next experiment, 1000 elements were generated using each method. This time, the epoch was chosen to be 128.

When the ciphertext was “KHOOR” and the key was “3”, the model predicted that the method was RailFence. The training accuracy was -89.69%..

ISSN 3030-3907

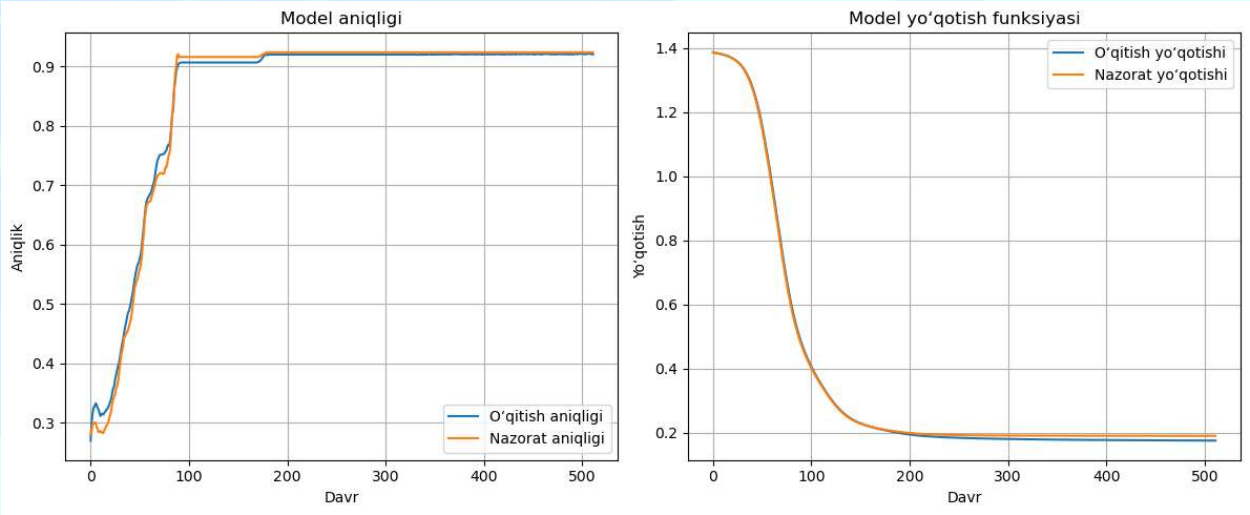


Figure 3. Results graph with 1000 elements from each class, epochs=512

In the final experiment, 1,000 examples from each method were trained over 512 epochs.

The word "HOMLF" is encrypted, and when "HELLO" is used as the keyword, the trained model detects that it is a Vigenere encryption algorithm. The training accuracy is 89.84%.

Table 4.

Results

Elements	Epochs	Learning rate	Training Accuracy
400	64	0. 000005	40%
4000	128	0. 000005	89.69%
4000	512	0.000005	89.84%

CONCLUSION

In this study, the possibility of automatically identifying types of classical encryption algorithms was studied by building an artificial neural network model. A neural network model was trained on a dataset created based on the given encryption algorithms. As can be seen from the experimental results, the model is confused in distinguishing between the Caesar and RailFence algorithms, which operate on almost the same key. It identifies other algorithms with high accuracy.

REFERENCES

1. Nasriddinov, J. I. o'g'li ., & Abdusattorov, A. S. o'g'li . (2025). “GRONSFELD SHIFRLASHI VA UNI SUN'IY NEYRON TARMOQLARI YORDAMIDA UNI O'QITISH”. Educational research in universal sciences, 4(6), 60–64. https://doi.org/10.5281/zenodo.15337777

2. M. Aripov, A.S. Matyakubov, “Axborotlarni himoyalash usullari”, Toshkent 2014 y. 97 bet

ISSN 3030-3907

3. N. McClure “TensorFlow Machine Learning Cookbook”, Birmingham,2017,370 p
4. S.K. Ganiyev, M.M. Karimov ,K.A. Tashev ,”AXBOROT XAVFSIZLIGI”, Toshkent 2016-yil, 366 bet
5. Nivedhitha Ramarathnam Krishna, “Classifying Classic Ciphers using Machine Learning”, SAN JOSE STATE UNIVERSITY, May 2019

ISSN 3030-3907