Wykorzystanie sieci neuronowych w kryptologii.

Piotr Kotlarz
Uniwersytet Kazimierza Wielkiego, Bydgoszcz
piotrk@ukw.edu.pl

Promotor rozprawy doktorskiej dr hab. Zbigniew Kotulski

Plan prezentacji

-Obszary zastosowań sieci neuronowych w kryptologii

-Inspiracja do zajęcia się tematem wykorzystania sieci neuronowych do szyfrowania

- Czy można nauczyć sieć neuronową szyfrowania ?

- Rozważania protokołu kryptograficznego

- Podsumowanie

Obszary zastosowań sieci neuronowych w kryptologii

Najczęstsze zastosowania narzędzi sztucznej inteligencji to systemy wykrywania anomalii w systemach komputerowych (Intrusion Detection System)

Wykorzystania sieci neuronowych w kryptoanalizie

Realizacja systemów wymiany klucza

[1] "Cooperating attackers in neural cryptography", Lanir N. Shacham, Einat Klein, Rachel Mislovaty, Ido Kanter, and Wolfgang Kinzel, PHYSICAL REVIEW E 69, 066137 (2004)
[2] "Analysis of Neural Cryptography", Alexander Klimov, Anton Mityaguine, and Adi Shamir, Computer Science department, The Weizmann Institute, Rehovot 76100, Israel.
[3] "Neural cryptography with feedback", Andreas Ruttor and Wolfgang Kinzel, Institut f¨ur Theoretische Physik, Universit¨at W¨urzburg, Am Hubland, 97074 W¨urzburg, Germany Lanir Shacham and Ido Kanter, Department of Physics, Bar Ilan University

Inspiracja do zajęcia się tematem wyk. sieci neuronowych do szyfrowania

Bezpieczeństwo protokołów kryptograficznych, zależy między innymi od jakości zaimplementowanych w nich algorytmów szyfrujących.

Co się stanie jeśli jeden z tych zaimplementowanych algorytmów składających się na protokół przestanie być bezpieczny ?

Inspiracja do zajęcia się tematem wyk. sieci neuronowych do szyfrowania

Przykład trochę przewrotny.

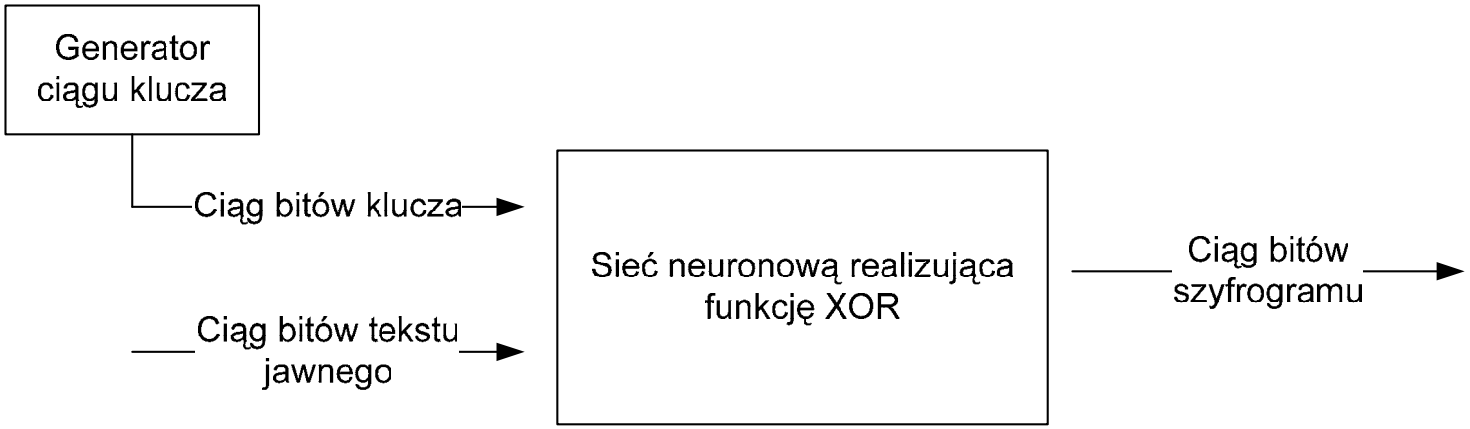
Szyfr Vernama z 1917

$$C = M \oplus K, c_i = m_i \oplus k_i$$

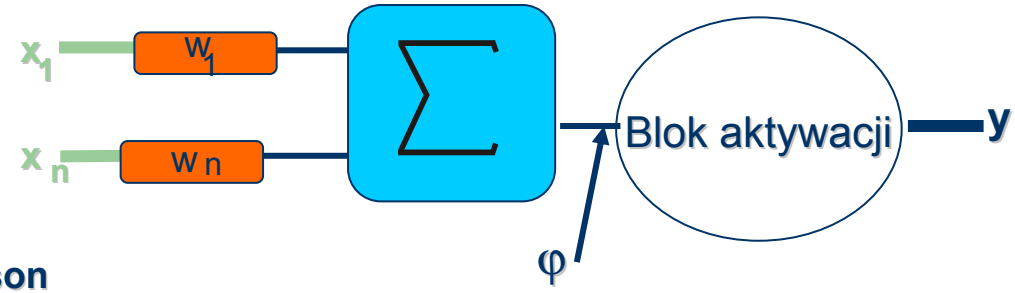
$$M = C \oplus K, m_i = c_i \oplus k_i$$

M=m1,m2,…,mi, ciąg bitów tekstu jawnego

K=k1,k2,…,ki *, losowy ciąg bitów klucza*



Podstawowy element sieci neuronowej



$$\varphi = \sum_{i=1}^{N} X_i * W_i = X * W^T$$

$$y = f(\varphi)$$

$$f = \begin{cases} 1, & \varphi > p \\ 0, & \varphi \le p \end{cases} \qquad y = \frac{1}{1 + e^{(-\beta\varphi)}}$$

Wprowadzenie do szyfrowania

Szyfr symetryczny

$$C = F_{K_1}(M) \qquad M = D_{K_1}(M)$$

$$M = D_{K_n}(E_{K_n}(M))$$

Szyfr asymetryczny

$$C = F_{K_2}(M) \qquad M = D_{K_1}(C)$$

$$M = D_{K_1}(E_{K_2}(M))$$

Wprowadzenie do szyfrowania

Szyfr symetryczny – elementarne przekształcenia kryptograficzne

permutacja

S={1,2,3,4,5}, permutacja p:S⟶S

$$p = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 4 & 2 & 1 \end{pmatrix}$$

wiersz górny jest dziedziną a dolny obrazem odwzorowania p

$$p^{-1} = \begin{pmatrix} 3 & 5 & 4 & 2 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

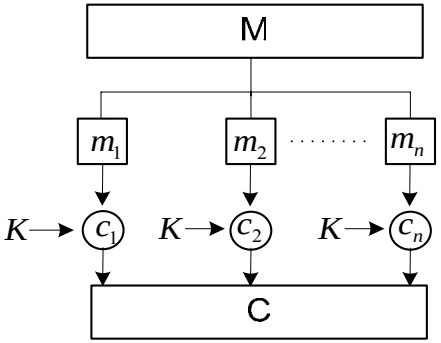
podstawienie

podstawienie v:J→S

w kryptografii J - alfabet jawny, S-alfabet szyfrowy

Wprowadzenie do szyfrowania

Szyfry blokowe



$$C = (c_1, c_2, ..., c_n) = (E_K(m_1), E_K(m_2), ..., E_K(m_n))$$

$$M = (m_1, m_2, ..., m_n) = (D_K(c_1), D_K(c_2), ..., D_K(c_n))$$

** Tryb elektronicznej książki kodowej (ECB)*

Czy można nauczyć sieć neuronową szyfrowania ?

Sieci neuronowe nie służą do tego !!!

Kryptografia wymaga dużej dokładności

Sieci neuronowe w ogólnym przypadku dają odpowiedzi bardziej ogólne lub przybliżenia wartości dokładnych.

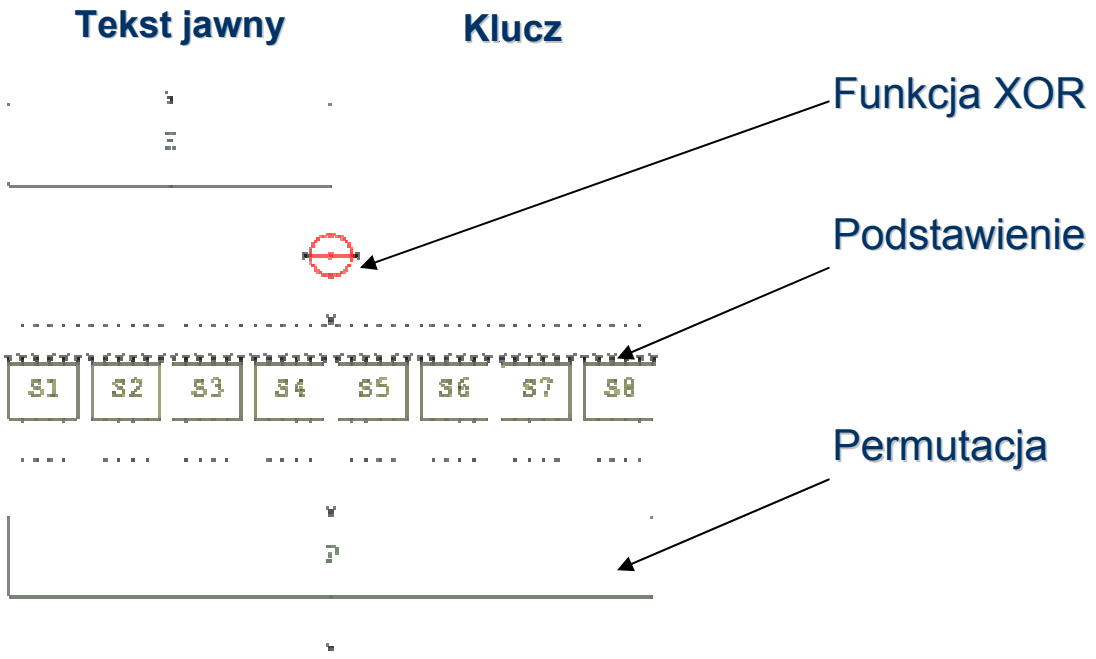
Wobec tego czy sieć neuronowa w ogóle może realizować szyfrowanie ?

Do czego zmierzamy ?

Rozwiązać problem realizacji elementarnych przekształceń szyfrujących za pomocą sieci neuronowej.

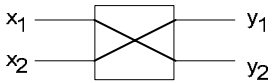
Zaproponować alternatywę dla obecnego podejścia do realizacji algorytmów kryptograficznych (szyfrowanie, funkcje skrótu)

Co nam się udało,

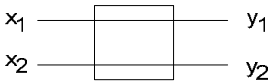


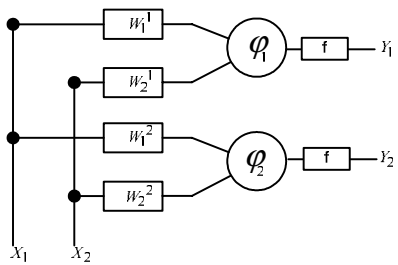
Realizacji permutacji za pomocą sieci neuronowej ,

$$\sigma_A = \begin{pmatrix} x_1 & x_2 \\ y_1 & y_2 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$$



$$\sigma_B = \begin{pmatrix} x_1 & x_2 \\ y_1 & y_2 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix}$$





$$\varphi_n = \sum w_i^n x_k$$

$$f = \begin{cases} 1, & \varphi > p \\ 0, & \varphi < p \end{cases}$$

$$y_l = f(\varphi_n)$$

φ_n - potencjał membranowy
n – nr neuronu
i – nr wejścia neuronu
k – nr wejścia sieci
l – nr wyjścia sieci

Realizacji permutacji za pomocą sieci neuronowej ,

$$\sigma_A = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

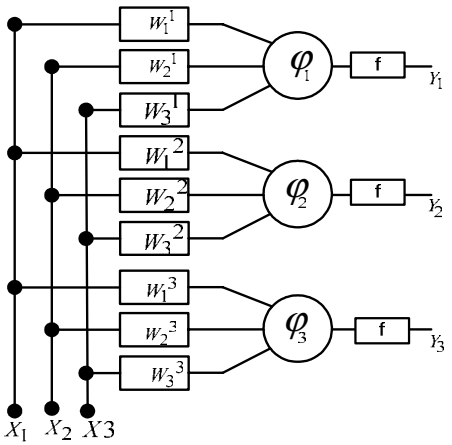
$$\sigma_B = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}$$

$$\sigma_C = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}$$

$$\sigma_D = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}$$

$$\sigma_E = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$$

$$\sigma_F = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$



Realizacji permutacji za pomocą sieci neuronowej ,

Permutacje

Sieć z wagami zmiennoprzecinkowymi

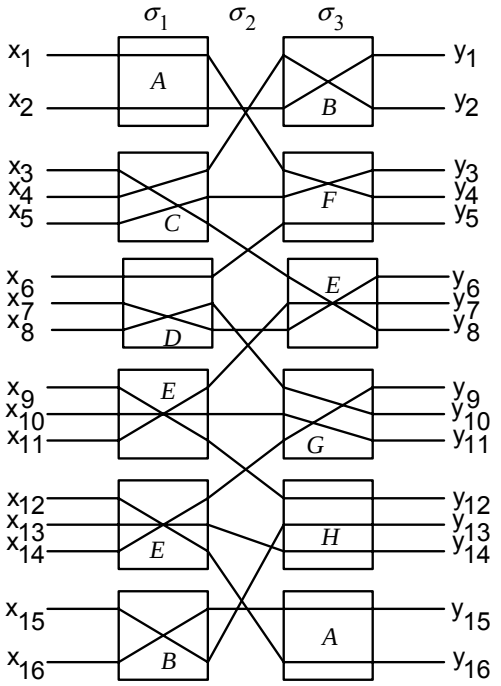
$$\sigma = (\sigma_1 \circ \sigma_2 \circ \sigma_3) =$$

$$= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\ 1 & 2 & 5 & 3 & 4 & 6 & 8 & 7 & 11 & 10 & 9 & 14 & 13 & 12 & 16 & 15 \end{pmatrix} \circ$$

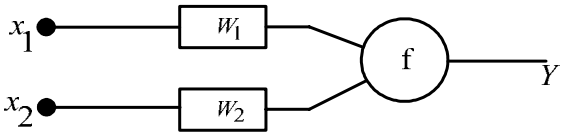
$$\circ \begin{pmatrix} 1 & 2 & 5 & 3 & 4 & 6 & 8 & 7 & 11 & 10 & 9 & 14 & 13 & 12 & 16 & 15 \\ 3 & 2 & 6 & 1 & 4 & 5 & 8 & 9 & 12 & 10 & 7 & 16 & 14 & 11 & 13 & 15 \end{pmatrix} \circ$$

$$\circ \begin{pmatrix} 3 & 2 & 6 & 1 & 4 & 5 & 8 & 9 & 12 & 10 & 7 & 16 & 14 & 11 & 13 & 15 \\ 4 & 1 & 8 & 2 & 3 & 5 & 6 & 10 & 12 & 11 & 7 & 16 & 14 & 9 & 13 & 15 \end{pmatrix} =$$

$$= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\ 4 & 1 & 8 & 2 & 3 & 5 & 6 & 10 & 12 & 11 & 7 & 16 & 14 & 9 & 13 & 15 \end{pmatrix}$$



Wykorzystanie sieci składającej się z neuronów boolowskich,



$$y = \begin{cases} 1, & dla \quad (w_1 + 1)x_1 + (w_2 + 1)x_2 \geq 2 \\ 0, & dla \quad (w_1 + 1)x_1 + (w_2 + 1)x_2 < 2 \end{cases}$$

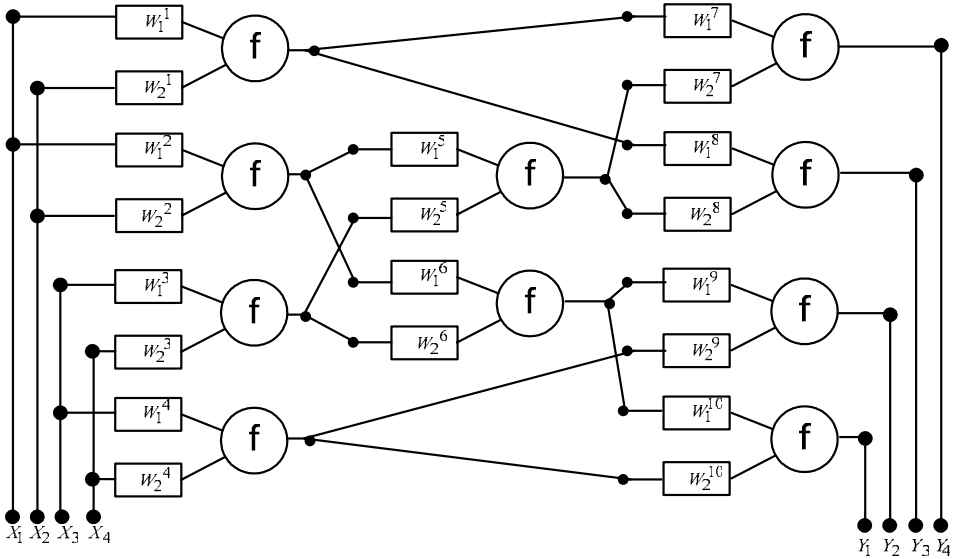
wartości wag i wejść neuronu mogą przyjmować jedynie wartości 0 lub 1

neuron będzie realizował przy danej kombinacji stanów wag (W) jedną z czterech funkcji boolowskich: AND, LEFT, RIGHT lub OR.

Wykorzystanie sieci składającej się z neuronów boolowskich,

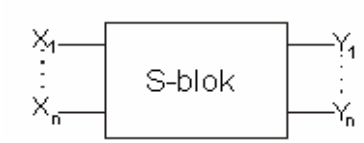
Permutacje

$$\sigma_A = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 2 & 4 \end{pmatrix}$$



Realizacji podstawienia za pomocą sieci neuronowej ,

Podstawienie

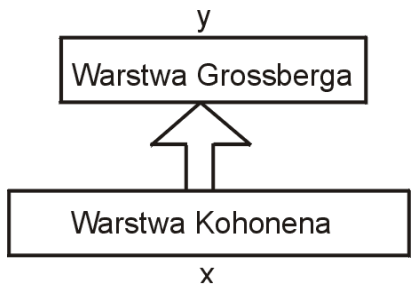


Kolumna 13

111010

Wiersz 3

Realizacja wybierania z tablicy
„Sieć przesyłająca żeton"



S_1

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

$10_{10} = 1010_2$

* Do deszyfrowania używany jest ten sam s-blok, dzięki zastosowaniu tego samego klucza

Oprogramowanie wykorzystywane do badań

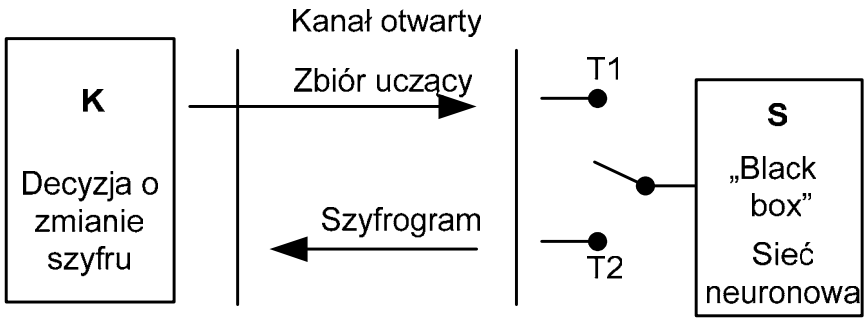
Po wielu próbach i doświadczeniach z wykorzystaniem gotowych narzędzi, do realizacji sieci neuronowych (Matlab, Sphinx, Neuronix, Neurosolution …)

Podjęto decyzję o stworzeniem oryginalnego oprogramowania o możliwościach:

- Budowanie bloków elementarnych z pojedynczych neuronów
- Budowanie z bloków elementarnych dowolnej sieci szyfrującej
- Możliwość szczegółowego obserwowania procesu uczenia
- Realizacja trzech trybów pracy neuronowego układu szyfrującego
    - uczenie
    - szyfrowanie
    - aktualizacja parametrów sieci

Sieć neuronowa jako alternatywa dla układu programowalnego

Ogólna koncepcja protokołu

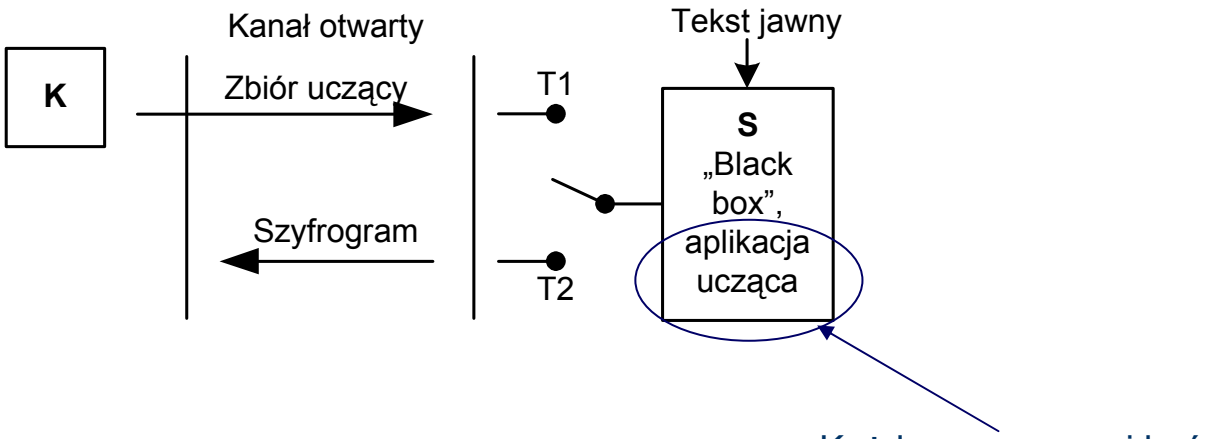


K – klient
S – serwer
T1 – tryb pracy serwera, zmiana algorytmu
T2 – tryb pracy serwera, szyfrowanie

Jeden klient może „posiadać" wiele serwerów

Jak zmieniać realizowany algorytm szyfrujący

Uczenie po stronie serwera
(Przesyłanie zbioru uczącego)



-przesyłany jest kompletny zbiór uczący

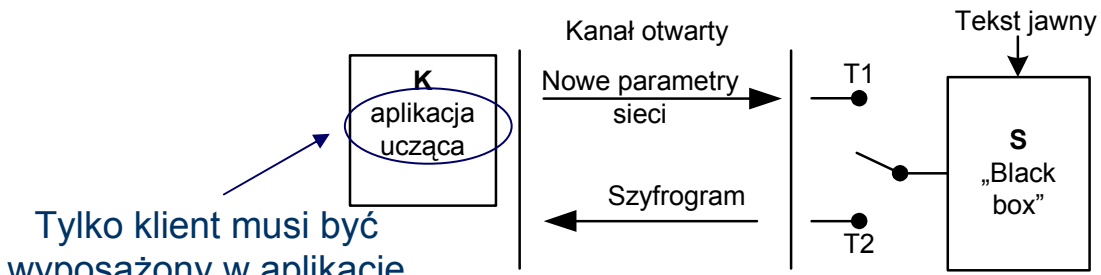
-zbiór uczący to „wiedza” o tym jak szyfrować

-konieczność zachowania w tajemnicy struktury sieci

Jak zmieniać realizowany algorytm szyfrujący

Uczenie po stronie klienta

(Przesyłanie nowych wartości wag)



$$W_A = [w_{A1} \dots w_{A4}], W_B = [w_{B1} \dots w_{B4}], W_C = [w_{C1} \dots w_{C6}], W_D = [w_{D1} \dots w_{B6}],$$
$$W_E = [w_{E1} \dots w_{E6}], W_F = [w_{F1} \dots w_{F6}], W_G = [w_{G1} \dots w_{G6}], W_H = [w_{H1} \dots w_{H6}]$$

-przesyłany jest komplet wag

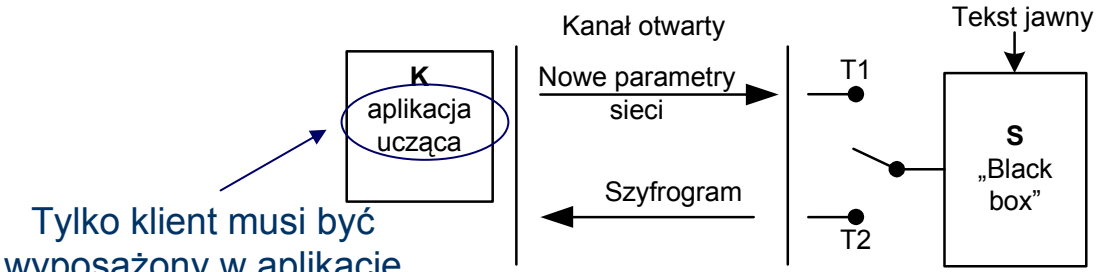
-komplet wag to „wiedza" o tym jak szyfrować

-konieczność zachowania w tajemnicy struktury sieci

Jak zmieniać realizowany algorytm szyfrujący

Uczenie po stronie klienta

(przesyłanie różnicy wag)



$$R = W \pm W'$$

$$W'_A = [W_{A1} \pm W_{R1}, \ldots, W_{A4} \pm W_{R4}], W'_B = [W_{B1} \pm W_{R1}, \ldots, W_{B4} \pm W_{R4}], \ldots$$

przesyłanie wartości wag - funkcja XOR
sieć boolowska

$$w_i^j(n+1) \oplus w_i^j(n) = w_i^j(k)$$
$$w_i^j(n+1) = w_i^j(k) \oplus w_i^j(n)$$

wij(k) – wartości wysyłane do serwera
(n+1) – nowa wartość wagi, (n) wartość poprzednia

-przesyłana tylko różnica

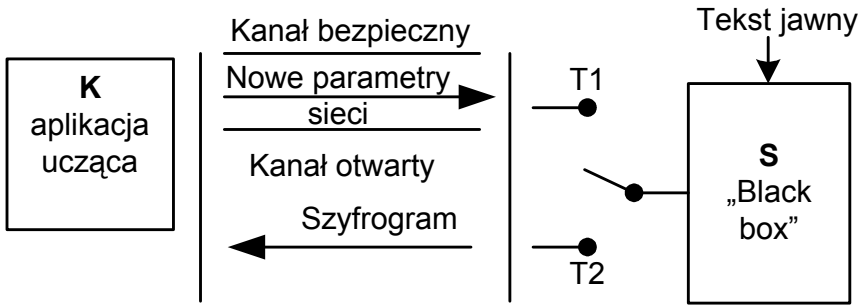
-różnica to zbyt mało aby odgadnąć algorytm szyfrowania

-konieczność prowadzenia przez intruza stałego nasłuchu

-konieczność posiadania przez intruza wiedzy na temat stanu początkowego sieci

Jak zmieniać realizowany algorytm szyfrujący

Wykorzystanie kanału bezpiecznego



Przykładowy atak siłowy

Neurons	No. of neurons within a network	Plaintext length (in bits)	Average training time (sec.)	Time of the brute-force plaintext attack (sec.)
Perceptrons	4	4	0.03	1099.20
Perceptrons	16	8	0.32	6306.60
Perceptrons	36	16	1.21	More than 168000
Boolean neurons	10	4	0.02	4.09
Boolean neurons	24	8	0.1	136.12
Boolean neurons	36	16	0.8	739.23

Table 1. Results of simulations for different neural networks

Podsumowanie

Do czego dążymy ?

Zrealizować za pomocą sieci neuronowej algorytm DES

Opracowanie metod i reguł pozwalających na bezpieczną modyfikację algorytmu szyfrującego.

Sieć neuronowa samoucząca się szyfrować
Raczej nie realne

Cz. 2

-analiza i wyniki porównania pracy szyfrującego układy neuronowego różnych trybach pracy (iteracyjny, potokowy, hybrydowy)

-analiza odporności proponowanego rozwiązania na kryptoanalizę różnicową

-implementacja algorytmu DES (może AES)

-implementacja funkcji skrótu

Dziękuję za uwagę.

Piotr Kotlarz
piotrk@ukw.edu.pl