Neural Network Based Cryptanalysis of PRESENT Lightweight Block Cipher

Michael Gamota

Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, United States of America

Abstract

Substitution-Permutation Networks (SPNs) are the backbone of block cipher encryption schemes. These networks are responsible for introducing confusion and diffusion, ideas introduced by Claude Shannon, which are required for a secure block cipher. (Shannon 1949) This work implements an existing ”lightweight block cipher” called PRESENT (Bogdanov et al. 2007) with non original, custom substitution boxes, permutation boxes, and number of encryption rounds. Quantitative analysis is used to calculate metrics which represent confusion and diffusion for these modified versions of the cipher and the performance of a neural network trained on plaintext-ciphertext pairs is measured and reported. Plaintexts of random bits and English words are both tested. Neural network size (hidden layer size) is also varied.

This work is motivated by the increased popularity of and ease of access to neural network based machine learning models. This work seeks to contribute to understanding how this shift may impact security of “lightweight block ciphers”, specifically PRESENT.

Introduction

Moore’s Law (Schaller 1997) and Dennard Scaling (Dennard et al. 1974) have allowed classical computers and accelerators to become more powerful and software tools like Pytorch have enabled individuals with access to consumer grade hardware to architect, train, test, and deploy neural networks. Neural networks are capable of ”learning” non-linear functions, so exploration of their use in cryptanalysis of block ciphers, where non-linearity must be introduced to ensure security, is natural. (Sze et al. 2017).

Substitution boxes (S-boxes) map a certain bitstring to a different bitstring. Permutation boxes (P-boxes) rearrange the bits in a bitstring. “Lightweight block ciphers” like PRESENT [1] which are based on SPNs are optimized for efficient hardware implementation and have real world utility, making them a target for attacks.

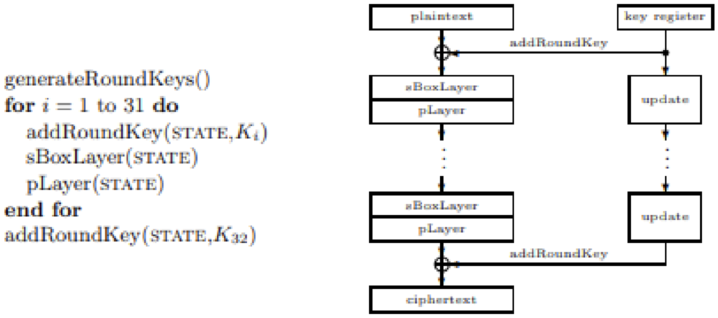
Statistical cryptanalysis methods like differential cryptanalysis can be successful when there is limited non-linearity introduced in the cipher. (Trappe and Washington 2002) This can be the result of poorly crafted (very linear) S-boxes. Non-linearity is strongly tied to the ”confusion” element of a strong block cipher. P-boxes are responsible for rearranging bits, this is tied to the

”diffusion” element of a strong block cipher. Intuitively, if a single bit is changed in the input plaintext, and if there is no rearranging of bits, this single bit will propagate through the cipher and only affect one block of the ciphertext.

Given a fixed key, this work assesses the ability of various sized neural networks to recreate plaintexts given a ciphertext after being trained on a dataset of plaintext-ciphertext pairs. Both random bit plaintexts and English word plaintexts are assessed.

Related Work

The base cipher architecture for this work is a ”lightweight block cipher” called PRESENT (Bogdanov et al. 2007). A Python implementation (Oosterlynck and Teuwen 2008) of this was adapted to allow for user defined S-boxes, P-boxes, and number of cipher rounds. A diagram is shown below.



```
generateRoundKeys()
for $i = 1$ to 31 do
    addRoundKey(STATE,$K_i$)
    sBoxLayer(STATE)
    pLayer(STATE)
end for
addRoundKey(STATE,$K_{32}$)
```

(Kim et al. 2023) presents neural network based attacks on different lightweight block ciphers: S-AES, S-DES, and S-SPECK. The attacks are all key recovery attacks, and the researchers find that they can predict certain bits with greater than 50% accuracy, but larger key lengths become harder to attack.

(Jain, Kohli, and Mishra 2021) implements a neural network distinguisher for a reduced round (3-6 inclusive) version of the PRESENT cipher (and others). They find that after 6 rounds the output of PRESENT cannot be distinguished from a random output using their neural network. For another lightweight block cipher, SIMECK, they find the cutoff to be 7 rounds.

(Jeong, Ahmadzadeh, and Moon 2024) This paper performs key recovery, plaintext recovery, and encryption emulation attacks on 5 different block ciphers. They also look at sentence-based text encryption and word-based text encryption. This is particularly interesting because it seems like a practically relevant attack because widely used secure messaging platforms encrypt text.

(Gérault et al. 2025) is a survey paper which explores current works and outlines best practices for future work in Neural Cryptanalysis. It provides a comprehensive overview of neural cryptanalysis attacks as well as a framework and best practices for future research.

Methods

Modifications to PRESENT Cipher

In order to perform this study, it was necessary to modify an existing implementation of the PRESENT cipher to allow for custom S-boxes, P-boxes, and reduced round versions.

Using the framework for quantifying S-box nonlinearity described in (Cruz Jiménez 2018), I performed a random search of the 4-bit S-box space and found an S-box which has a nonlinearity of 2, compared to the original PRESENT S-box which has a nonlinearity of 4.

Input (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output (Hex)	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Table 1: Original PRESENT S-Box (Non-linearity 4)

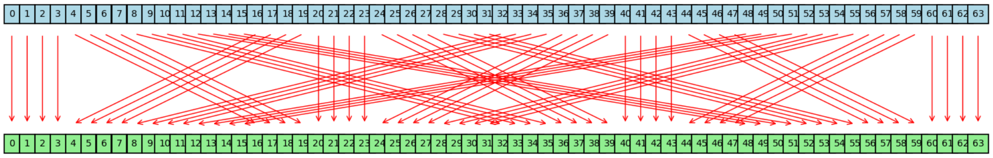
Input (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output (Hex)	1	6	D	0	4	2	A	5	7	8	3	C	B	F	9	E

Table 2: Medium linearity S-Box (Non-linearity 2)

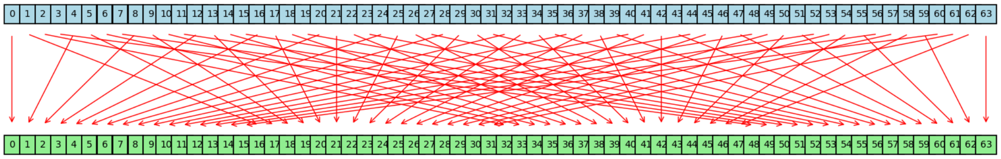
Input (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output (Hex)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Table 3: Perfectly Linear S-Box (Non-linearity 0)

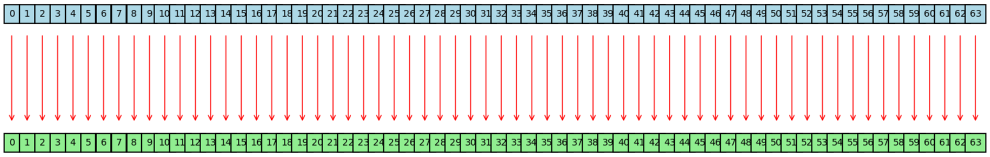
One metric useful for measuring the effect of a P-box on the overal cipher is to measure the avalanche effect (Stallings 2010) . The avalanche effect is used to describe how changing one bit in the input has an effect on the output. Ideally, a one bit change should lead to a change in 50% of the output bits. For purposes of this project, I chose to use 3 different P-boxes. The original P-box, a ”weak” P-box which has more localized mappings, and a trivial P-box which does not rearrange bits at all. The spatial mappings of all 3 P-boxes are shown below.



”Medium” P-Box



Original PRESENT P-Box



Trivial P-Box

In addition to modifying the substitution and permutation networks, the number of cipher rounds has also become a tunable parameter. The default PRESENT cipher uses 32 rounds. At the start of every round, the input is XORed with a 64-bit key for that round, which has been developed from an initial 80-bit key. In this work I investigate neural network effectiveness against 1-10, 16, and 32 round versions of PRESENT.

Intuition for Nonlinearity Metric

The most complicated math of this project is the calculation of the nonlinearity of an S-box(Cruz Jiménez 2018). This method, shown below, is based on the Walsh Transform, a domain transformation used in signal processing.

$$W_{a,b} = \sum_{x \in F_2^n} (-1)^{\langle a,x \rangle \oplus \langle b,\Phi(x) \rangle}.$$

Where x is the input to the S-Box, $\Phi(x)$ is the output of the S-box, and a and b are bitstrings of length n.

The complete forumla for nonlinearity is shown below.

$$\mathrm{NL}(\Phi) = 2^{n-1} - \frac{1}{2} \max_{a \neq 0,\, b} |W_{a,b}|\,.$$

The idea is that by apply a bitmasks to the S-box inputs and outputs (a and b, respectively) to isolate how specific bit positions or several input bit positions relate to those bit position(s) in the output. The goal is to check across all substitution mappings and across all possible subsets of those mappings (achieved with bitmasks) to find the maximum possible “bias”.

This “bias” is determined by how often a bit or subset of bits is flipped when passing through the S-box. The Walsh Transform sum increases or decreases depending on whether a bit gets flipped.

If a 4-bit S-box always flips the highest bit, the magnitude of the Walsh transform will continually grow as x iterates. When testing with the bitmasks $a = b = 1000$, the exponent in the Walsh Transform across x summation evaluates to:

$$\mathbf{1000 \cdot 0000 \ \oplus \ 1000 \cdot 1000 = 0 \ \oplus \ 1 = 1}$$
$$\mathbf{1000 \cdot 0001 \ \oplus \ 1000 \cdot 1001 = 0 \ \oplus \ 1 = 1}$$

$$\mathbf{1000 \cdot 0010 \ \oplus \ 1000 \cdot 1010 = 0 \ \oplus \ 1 = 1}$$

$$\mathbf{1000 \cdot 0011 \ \oplus \ 1000 \cdot 1011 = 0 \ \oplus \ 1 = 1}$$

$$...$$

$$\mathbf{1000 \cdot 1111 \ \oplus \ 1000 \cdot 0111 = 0 \ \oplus \ 1 = 1}$$

This means the Walsh Transform will evaluate to -16.

$$\mathrm{NL}(\Phi) = 2^{4-1} - \frac{1}{2}\left|-16\right|$$

$$\mathrm{NL}(\Phi) = 8 - 8 = 0$$

This means that this S-box will get a nonlinearity value of 0, meaning it is very linear.

Plaintext Experiments

Since one of the motivating themes behind this project is applicability, I thought it would be interesting to compare effectiveness of a neural network based plaintext reconstruction attack when trained and tested on English text data compared to random bitstrings. One prevalent example of real-world encryption use is encrypted messaging apps, reconstructing plaintext from ciphertext (using a fixed key across training and testing) would be of interest to malicious parties.

All plaintexts used are 64-bits. The random plaintext data is a 64-bit random string. The English language plaintext words are padded to 64-bits. Each ASCII character is 8 bits.

Neural Network Specifications

The architecture used in this work is a basic MLP with 64 input neurons (one for each bit of ciphertext), varying hidden layers and hidden layer sizes, and a 64 neuron output layer (one for each bit of predicted plaintext).

To broaden experiment variables, I decided to use varying sizes of neural network. In all of my experiments, the number of hidden layers in the neural network is equal to the number of rounds of the cipher. In addition to this scaling, different hidden layer sizes were used. Hidden layer sizes of 128, 256, 512, and 1024 were all tested. The goal was to investigate possible relationship between number of neurons and bit prediction accuracy.

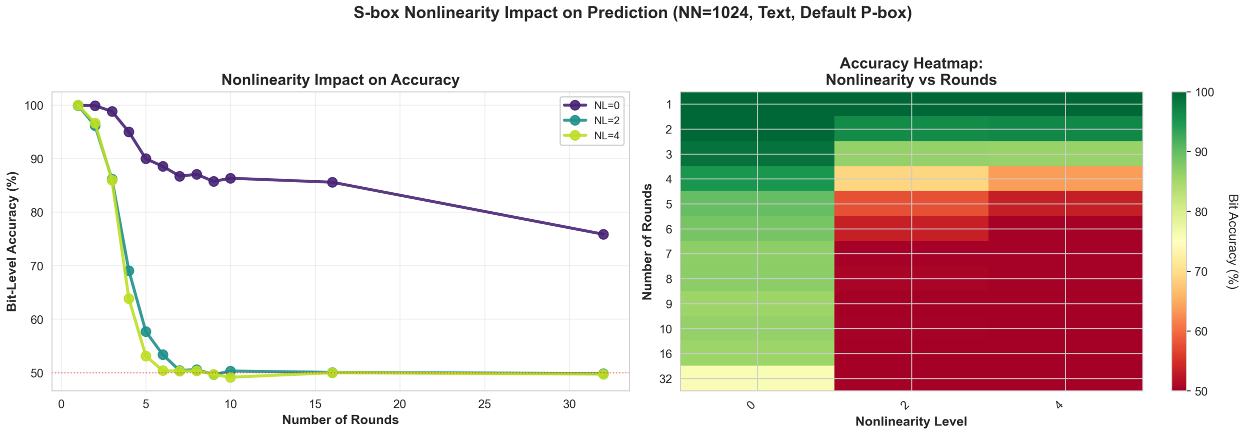
ReLU activation was applied on all hidden layers and sigmoid was used on the output layer. The output values were rounded up or down to convert to bits. Binary Cross Entropy loss and Adam optimization were used in training. 900 samples were used for training and 140 were used for testing for both random and English word plaintexts.

Overall Flow

1. A random 80-bit round key is generated using a fixed seed value.
2. A script is ran to generate plaintext-ciphertext pairs for the each S-box, P-box, and number of rounds configuration for all text data and all random bit data. The avalanche effect and non-linearity of each configuration is calculated and included in the output file name.
3. The MLP is trained on these ciphertext plaintext pairs according to the specifications in the above section. Every plaintext, ciphertext, and predicted plaintext in the test set, are recorded with their respective bit prediction accuracy in a .csv file.
4. The plaintext, ciphertext, and predicted plaintexts are converted from binary to ASCII for English text.
5. A visualization script is ran to plot relationships between neural network performance, plaintext type, and cipher configuration.

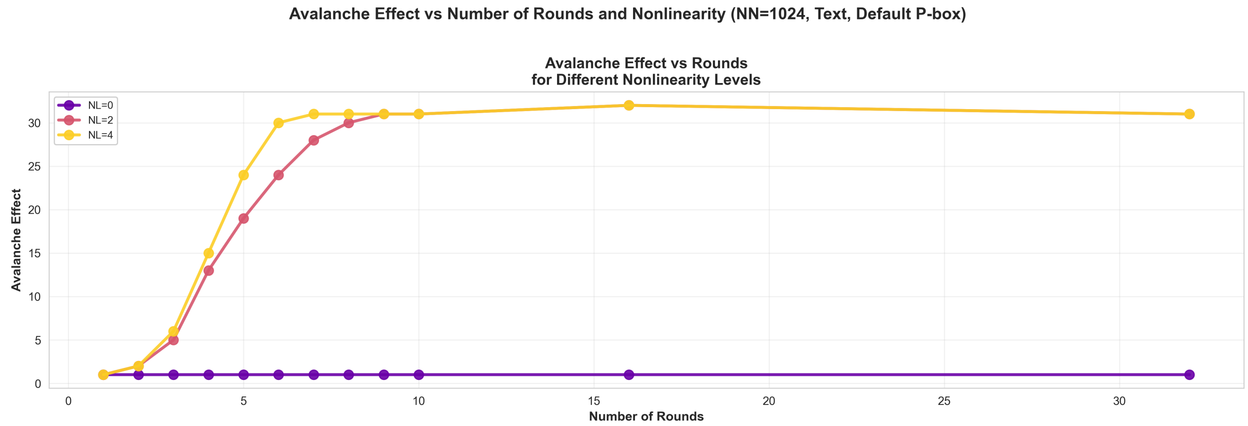
Results and Discussion

Accuracy vs Nonlinearity



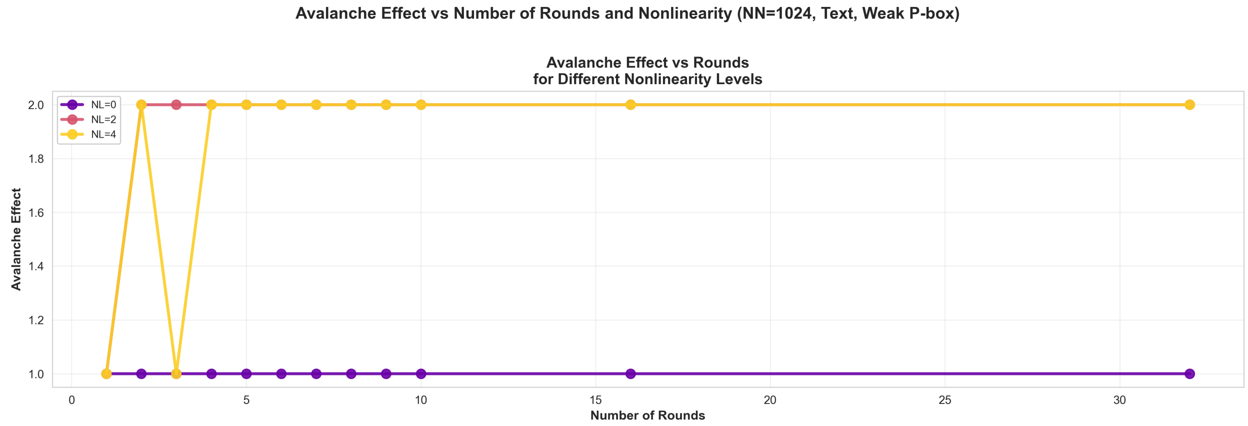
It is seen here that even a version of the PRESENT cipher with a more linear S-box is still effective against a neural network based plaintext reconstruction attack when a full round version is used. For reduced round versions of the cipher, there is a slight advantage to the original S-box.

Avalanche Effect vs Rounds


Avalanche Effect vs Number of Rounds and Nonlinearity (NN=1024, Text, Default P-box)

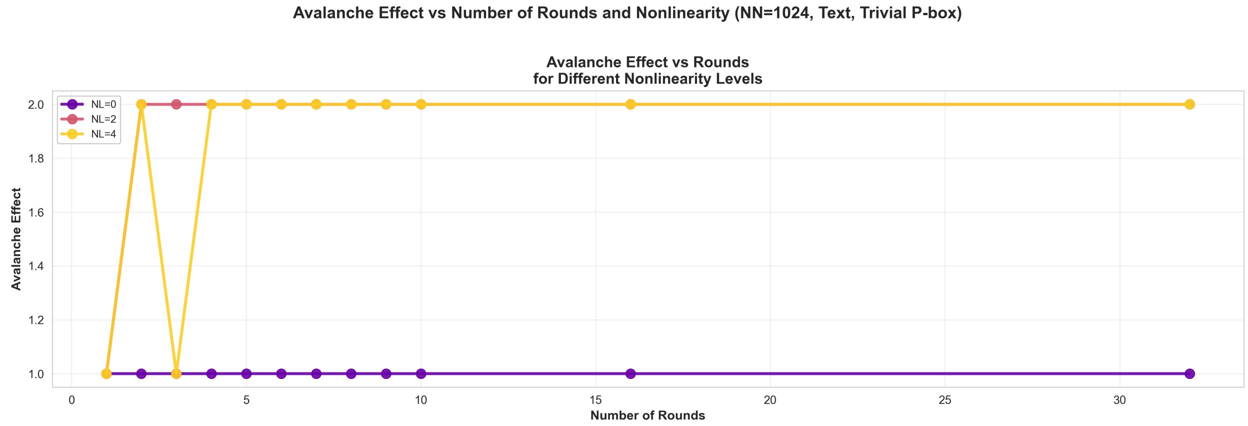
Avalanche Effect vs Rounds
for Different Nonlinearity Levels

Avalanche Effect vs Rounds


Avalanche Effect vs Number of Rounds and Nonlinearity (NN=1024, Text, Weak P-box)

Avalanche Effect vs Rounds
for Different Nonlinearity Levels

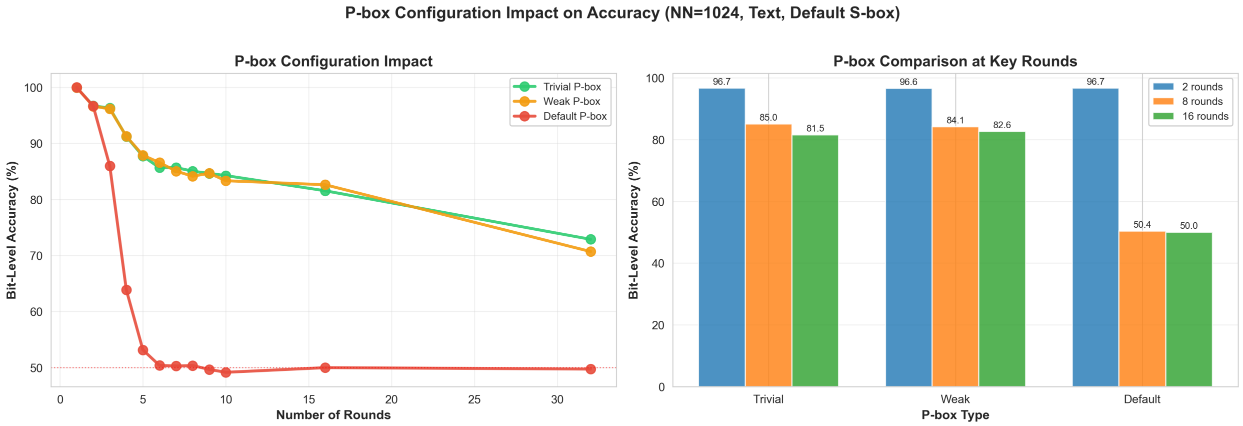
Avalanche Effect vs Rounds


Avalanche Effect vs Number of Rounds and Nonlinearity (NN=1024, Text, Trivial P-box)

Avalanche Effect vs Rounds
for Different Nonlinearity Levels

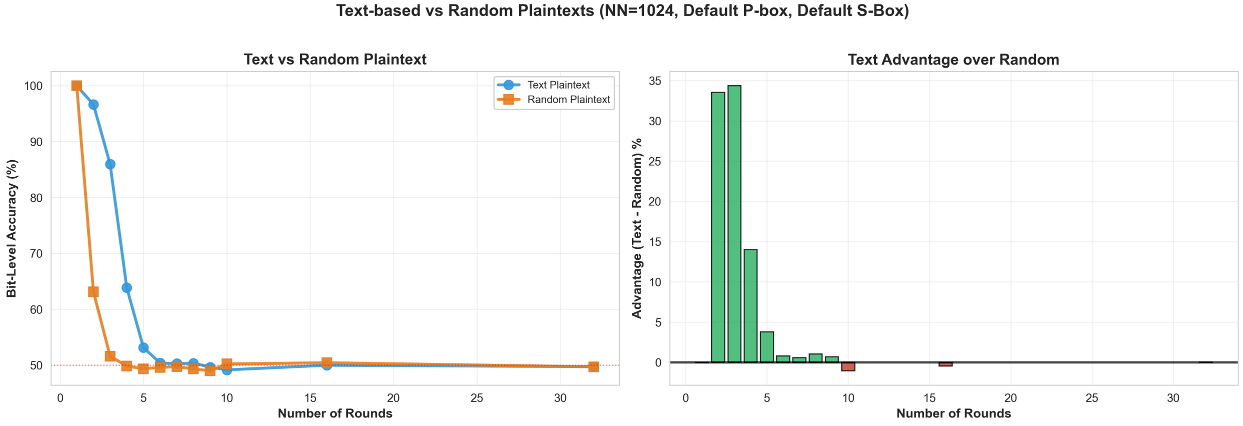
The above three plots show that the original PRESENT P-box achieves close to the ideal avalanche effect of 32. The role of S-box design is also demonstrated. The trivial and weak P-boxes show that having a poorly designed P-box cannot be overcome by a well designed S-box.

Accuracy vs P-box (Default S-Box)



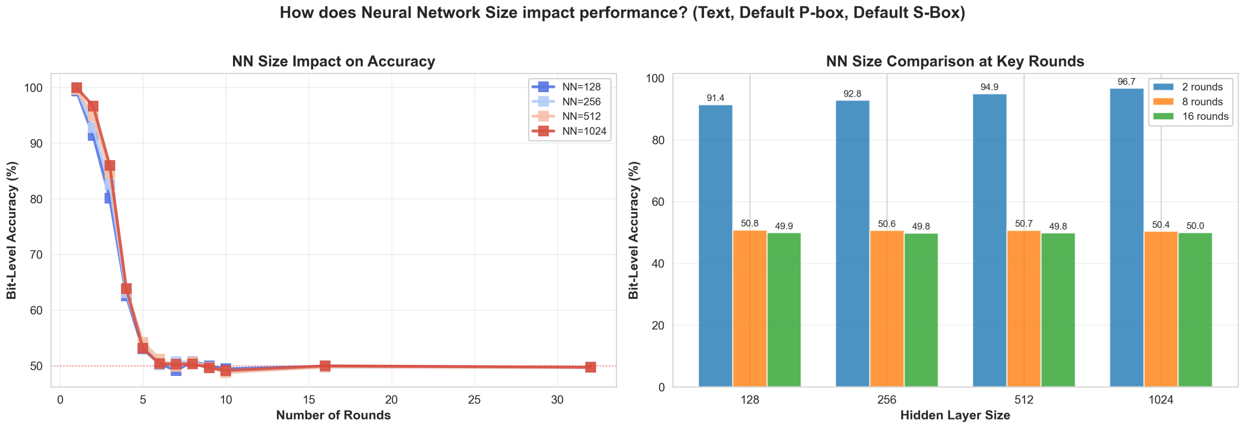
Even at full rounds, a well designed P-box is needed for bit accuracy to converge to 50

Random vs English Plaintext



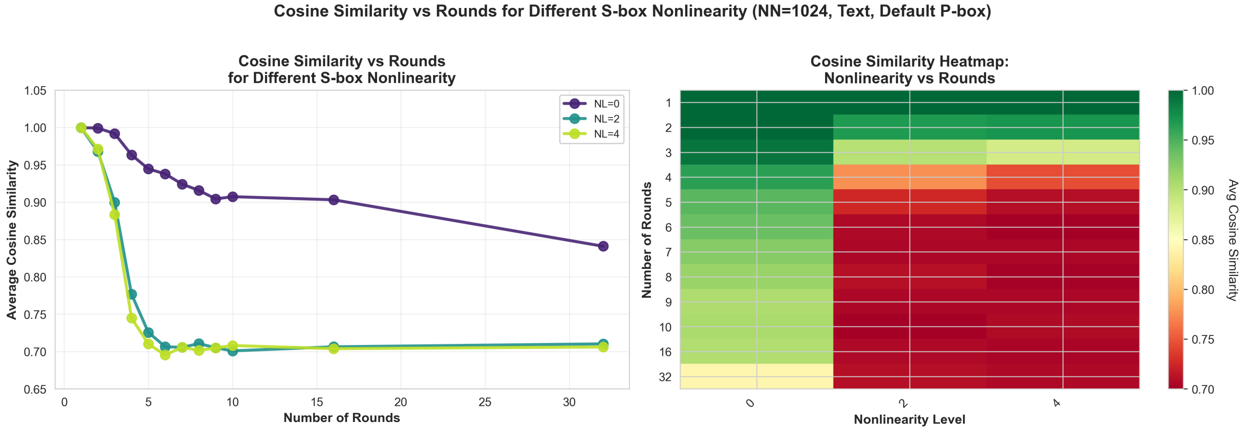
Interestingly, there is a difference in the ability of a neural network to predict plaintext bits given a ciphertext when it is known that the input is English plaintext(the model is purely trained on English plaintext-ciphertext pairs). Certain features like padding may contribute to predictability.

Accuracy vs Neural Network Size

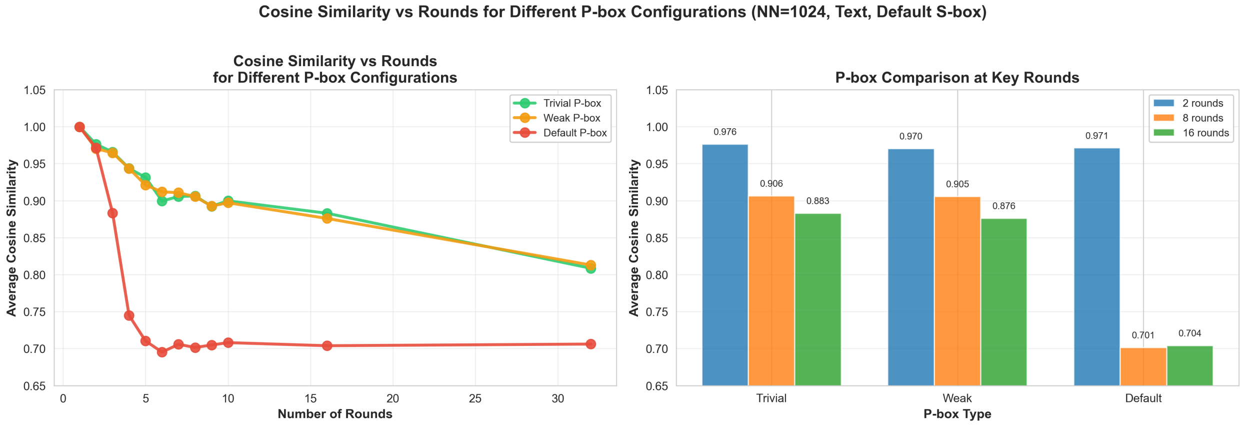


Based on the neural network sizes that I used, it can be seen that there is not an advantage to using a larger neural network when it comes to bit level accuracy.

Cosine Similarity vs Nonlinearity



Cosine Similarity vs P-Box



The intuititon for looking at cosine similarity is to get some sort of confidence metric for the neural network's prediction. Since the output of the neural network is a float between 0-1, there is rounding done to convert to a binary value. Looking at cosine similarity, we see that the trends are similar to the ”Accuracy vs P-Box” and the ”Accuracy vs Nonlinearity”

Example Predictions

? is used to substitute padding bits and bytes which are not ASCII characters.

Predicted Plaintext	True Plaintext
dog?????	dog?????
not???	hot???
some????	some????
darc????	dark????

Table 4: 96.65% accuracy, 2 rounds, Default S-box, Default P-Box

Predicted Plaintext	True Plaintext
aog?????	dog?????
contkfn?	mention?
lftter??	faster??
soice???	voice???

Table 5: 91.25% accuracy, 4 rounds, Default S-box, Weak P-Box

Predicted Plaintext	True Plaintext
cav?????	dog?????
cadder??	center??
cat?????	cat?????
cat?????	man?????

Table 6: 86.76% accuracy, 8 rounds, Trivial S-box, Trivial P-Box

Predicted Plaintext	True Plaintext
cieet???	door????
cieet???	dog?????
cieet???	blank???
cieet???	hot?????
cieet???	amazing?

Table 7: 49.72% accuracy, 32 rounds, Original S-box, Original P-Box

Future Work

Future work extending this could look at using larger neural networks, different neural network architectures, full sentences, or numbers with specific format or embedded codes like credit card numbers where some of the digits indicate the card provider. More works that investigate cipher resistance to attacks that a malicious party would attempt in a real-world scenario increases the possible impact of a cryptanalysis project.

Conclusion

This work emphasizes the importance of nonlinear S-box design and designing P-boxes which ensure an avalanche effect which affects nearly half of output bits when one input bit is changed. Traditionally, these principles of confusion and diffusion have been emphasized for resistance to differential cryptanalysis, where cipher outputs are evaluated for distinguishability between a random number and a meaningful encryption of some plaintext. This work provided an analysis of lightweight block cipher strength against a neural network plaintext reconstruction attack, which differs from differential cryptanalysis but has a practical motivation of preventing individual data from being leaked.

References

Bogdanov, Andrey, Lars Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew Robshaw, Yannick Seurin, and C. Vikkelsoe. 2007. "PRESENT: an ultra-lightweight block cipher." *Lect Note. Comput. Sci.* 4727 (September): 450–466. https://doi.org/10.1007/978-3-540-74735-2_31.

Cruz Jiménez, Reynier Antonio de la. 2018. "On some methods for constructing almost optimal S-Boxes and their resilience against side-channel attacks." *IACR Cryptol. ePrint Arch.* 2018:618. https://api.semanticscholar.org/CorpusID:51801605.

Dennard, R.H., F.H. Gaensslen, Hwa-Nien Yu, V.L. Rideout, E. Bassous, and A.R. LeBlanc. 1974. "Design of ion-implanted MOSFET's with very small physical dimensions." *IEEE Journal of Solid-State Circuits* 9 (5): 256–268. https://doi.org/10.1109/JSSC.1974.1050511.

Jain, Aayush, Varun Kohli, and Girish Mishra. 2021. *Deep Learning based Differential Distinguisher for Lightweight Block Ciphers.* arXiv: 2112.05061 `[cs.CR]`. https://arxiv.org/abs/2112.05061.

Jeong, Ongee, Ezat Ahmadzadeh, and Inkyu Moon. 2024. "Comprehensive Neural Cryptanalysis on Block Ciphers Using Different Encryption Methods." *Preprints* (May). https://doi.org/10.20944/preprints202405.2022.v1. https://doi.org/10.20944/preprints202405.2022.v1.

Kim, Hyunji, Sejin Lim, Yeajun Kang, Wonwoong Kim, Dukyoung Kim, Seyoung Yoon, and Hwajeong Seo. 2023. "Deep-Learning-Based Cryptanalysis of Lightweight Block Ciphers Revisited." *Entropy* 25 (June). https://doi.org/10.3390/e25070986.

Schaller, R.R. 1997. "Moore's law: past, present and future." *IEEE Spectrum* 34 (6): 52–59. https://doi.org/10.1109/6.591665.

Shannon, C. E. 1949. "Communication theory of secrecy systems." *The Bell System Technical Journal* 28 (4): 656–715. https://doi.org/10.1002/j.1538-7305.1949.tb00928.x.

Stallings, William. 2010. *Cryptography and Network Security: Principles and Practice.* 5th. USA: Prentice Hall Press. ISBN: 0136097049.

Sze, Vivienne, Yu-Hsin Chen, Tien-Ju Yang, and Joel Emer. 2017. *Efficient Processing of Deep Neural Networks: A Tutorial and Survey.* arXiv: 1703.09039 `[cs.CV]`. https://arxiv.org/abs/1703.09039.

Trappe, Wade, and Lawrence C. Washington. 2002. *Introduction to cryptography: With coding theory.* Prentice Hall.