INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv

Neural Network Based Data Encryption: A Comparison Study among DES, AES, and HE Techniques

Sin-Qian Yeow [a], Kok-Why Ng [a,*]

[a] *Faculty of Computing and Informatics, Multimedia University, 63100 Cyberjaya, Selangor, Malaysia*
*Corresponding author: *kwng@mmu.edu.my*

***Abstract*—With the improvement of technology and the continuous expansion and deepening of neural network technology, its application in computer network security plays an important role. However, the development of neural networks is accompanied by new threats and challenges. This paper proposes to encrypt the weight data using encryption algorithms and embed image encryption algorithms to improve protected data security further. The purpose is to address the feasibility and effectiveness of using modern encryption algorithms for data encryption in machine learning in response to data privacy breaches. The approach consists of training a neural network to simulate a model of machine learning and then encrypting it using Data Encryption Standard (DES), Advanced Encryption Standard (AES), and Homomorphic Encryption (HE) techniques, respectively. Its performance is evaluated based on the encryption/decryption accuracy and computational efficiency. The results indicate that combining DES with Blowfish offers moderate encryption and decryption speeds but is less secure than AES and HE. AES provides a practical solution, balancing security and performance, offering a relatively swift encryption and decryption process while maintaining high security. However, Fernet and HE present a viable alternative if data privacy is a top priority. Encryption and decryption times increase with file size and require sufficient computational resources. Future research should explore image encryption techniques to balance security and accurate image retrieval during decryption. Advanced privacy-preserving approaches, such as differential privacy and secure multi-party computation, may enhance security and confidentiality in digital encryption and decryption processes.**

***Keywords*—Cryptography; data encryption standard; advanced encryption standard; homomorphic encryption; image encryption; machine learning.**

Manuscript received 15 Jan. 2023; revised 22 Jun. 2023; accepted 18 Sep. 2023. Date of publication 30 Nov. 2023.

I. Introduction

In recent years, neural networks have evolved considerably and found various applications driven by technological advances. Artificial Intelligence (AI) is being applied to increasing use cases in a growing number of sectors by providing new opportunities for intelligent and automated problem-solving in decision-making. Applications in a growing number of sectors. The benefits of artificial intelligence are significant and undeniable. However, the development of AI also comes with new threats and challenges that relevant professionals must face. ENISA published a report on the AI threat landscape in 2020, regarding the agency's positive mapping of the AI cybersecurity ecosystem and its threat landscape [1]. This threat sets the stage for upcoming cybersecurity policy initiatives and technical guidance and highlights the associated challenges. This research is based on three encryption algorithms, namely DES with Blowfish, AES, and Fernet with HE (Homomorphic Encryption), for securing YOLOv5 [2], [3] model files and images. Each algorithm is discussed, analyzed, and implemented as a separate program with improved methodologies.

A. Homomorphic Encryption in Neural Network

Homomorphic encryption can be used to perform operations on data without decryption. For backdoor attacks, neural networks provide misclassification rules or patterns specific to the neural network as triggers. When triggered, the neural network predicts its output according to the misclassification rules, allowing the attacker to supervise the neural network's output. Moradiya et al. [4] proposed using homomorphic encryption to prevent such vulnerabilities. Prior work of Nandakumar et al. [5] introduced the use of FHE (Fully Homomorphic Encryption) to facilitate the training of neural network models on encrypted data. This allows user authorization by using their secret key and sharing

encrypted data with the developer. The service developer can train the model without seeing the underlying data. Since the final model will also be encrypted, the service developer has no idea of the data or the learning model. As a result, the final model is only available for users with access to the private key. This prevents the hiding of the pattern to any extent. Onoufriou et al. [6], [7] adopted high-level Homomorphic Encryption (FHE) for encrypted neural network inference and sampling theory of FHE corresponding neural networks with their open source and recurrent examples. Hong et al. [8] presented an estimate that led to SoftMax activation in the model using HE and a process for good encoding information to reduce the calculation cost. Nugent [9] proposed a method for detecting privacy fraud on encrypted transactions using homomorphic encryption. The conventional ResNet-20 model is implemented with the RNS-CKKS FHE in the academic research publication of Lee et al. [10], and the resulting model is verified with the CIFAR-10 dataset and specified model parameters. This investigation confirms that the new model outperforms the original ResNet-20 model on unsecured information from the CIFAR-10 dataset by 98.43%. Besides, Lou et al. [11] demonstrated an accumulation of shifts based on LHE-enabled deep neural networks (SHE) for fast and accurate inference of encrypted data. The results show that SHE achieves state-of-the-art inference accuracy on MNIST and CIFAR-10 with a reduction of 76.21% over the previous LHE CNN.

B. AES in Neural Network

Previous researchers have introduced AES in neural networks to address security issues. In order to get outputs that match the conventional key extensions used in the conventional AES method, Yasin et al. [12] changed the standard Advanced Encryption Standard (AES) algorithm. They did this using a quick, inexpensive training process based on the Levenberg-Marquardt algorithm. Ameen et al. [13] use a nonlinear Nto design and implement AES with some modifications to resist the attack. In the NN design, the encryption and decryption processes are carried out using a symmetric key cipher. The key used in the encryption and decryption process is the initial weight of the ANN. In 2019, an AES neural system with diagonally weighted coefficients was proposed by Lytvyn et al. [14] to provide a new asymmetrically formed key for each input image. The AES cryptosystem consists of four layers working in parallel. Each neural network layer has an input vector of 4 bytes. Kwon et al. [15] proposed an attack model that can recover secret keys using a strength analysis attack based on a deep learning CNN.

Additionally, academic research by Sangeetha et al. [16] uses deep neural networks for AES encryption and image decryption. The algorithm has been implemented in MATLAB software, and the results have been studied. According to the research results, the cryptographic system of the chaotic and deep neural network is combined, and the image encryption security is improved. In recent works, Liu et al. [17] proposed an Advanced Encryption Standard (AES) encryption technique based on memetic neural networks. This paper focuses on the proposed image authentication model. Peleshchak et al. [18] a new two-stage encryption method to increase the cryptographic strength of the AES algorithm.

C. DES in Neural Network

Mundra et al. [19] explored the cryptanalysis of the DES 64-bit symmetric encryption algorithm using a deep neural network approach for DES optimization. A backpropagation technique [20] with multiple hidden layers and advanced activation functions was used, and the gradient loss problem was also addressed. Furthermore, the implementation results showed an accuracy rate of 90%, and their proposed technique was compared with existing techniques on three parameters: time, loss, and accuracy, all of which were significantly improved. To simulate the DES algorithm, Dhia et al. [21] built an ANN system that is used to attack this algorithm. The main dependency is on the processes that represent plaintext/ciphertext, so these processes (extended permutations, S-box permutations [22], and P-box permutations [23]) are expressed in the proposed neural network model. In 2013, Záluský et al. [24] designed a modified algorithm for the Data Encryption Standard (DES), which uses a neural network.

The proposed DES algorithm is compared with a standard algorithm, and its encryption and decryption capabilities are evaluated. In the first test, the data is encrypted with the standard DES algorithm and then decrypted with the proposed algorithm. The other tests are performed in the opposite order. The test is performed on 100 randomly generated 64-bit blocks. Both tests demonstrate the proper functioning of the proposed DES algorithm with a feedforward neural network. Fan et al. [25] simulate the DES decryption process using the backpropagation algorithm. A neural network simulator for decrypting the target ciphertext is built by inputting many plaintexts and ciphertext pairs and decrypting the given ciphertext. The cryptanalysis [26] problem can be said to be something unknown, and researchers have proposed the use of artificial neural networks to improve the performance of DES algorithms, Yousif [27] studies the implementation of DES to reduce its execution time and make DES more suitable to apply and improve its utility in the field of neural networks.

D. SDES in Neural Network

The SDES algorithm is similar to the original DES algorithm, but in a simplified version. SDES is created for educational purposes. It involves a smaller algorithm with fewer parameters and is easier to understand for DES. It requires only an 8-bit block, whereas the original DES requires 64 bits, which amounts to an eight-fold simplification. Alallayah et al. [28] propose a new complementary model discussed in the block cryptosystem (SDES) approach. The neural recognition model is constructed to achieve two goals. The first goal is to build a neural model simulator for the target cryptosystem, and the second goal is (cryptanalysis) to determine the key for a given plaintext-ciphertext pair. The concept of an equivalent cryptosystem that is 100% identical to the unknown system is summarized, meaning that an unknown hardware or software cryptosystem can be reconstructed without knowledge of its internal circuitry or algorithms. Mundra et al. studied the cryptanalysis of 64-bit DES symmetric encryption algorithms using a deep neural network approach to optimize DES. A backpropagation technique with multiple hidden layers and advanced activation functions is used, and the problem of gradient loss is also solved.

Furthermore, the implementation results show 90% accuracy, and the proposed technique is compared with existing techniques in three parameters: time, loss, and accuracy, all of which improve significantly. The network is an ideal tool for identifying black box systems. Alallayah et al. [29] discuss the simplified DES (SDES) block encryption system. The neural recognition model was constructed to achieve two objectives. The first goal is to build a neural model of the target encryption system simulation, while the second goal is (cryptanalysis) to determine the key of a given plaintext-ciphertext pair.symmetric encryption algorithm using a deep neural network approach for DES optimization. A backpropagation technique with multiple hidden layers and advanced activation functions was used, and the gradient loss problem was also addressed.

II. Materials and Method

This section describes the materials and methods comprising an integrated development environment, presumptions, data set, overall process, and proposed algorithms.

A. Integrated Development Environment

The operating system is Windows 11 with Nvidia CUDA 11.7 virtual CPU. Python is employed as the web programming language due to its widespread usage and compatibility with various platforms. Pycharm [30] is a Py-based framework for development that delivers several necessary tools for Python developers. These tools are tightly integrated to create a comfortable environment for accessing the command line, connecting to a database, creating a virtual environment, and managing projects.

B. Presumptions

- The generated model must be complete.
- The weights used for encryption must be complete.
- The weight of data before encryption is achievable.
- The weight file must be a pt. file.
- The complete model must be usable.
- Weight data before encryption and after decryption must be unaltered.
- Image photos must be frontal, clear, and unobstructed.
- Image photos must be recognizable.
- The displayed image must be a whole, not just a part of it.
- The photo must have no misleading ornamentation in the background.

C. Data Set

To collect data required in this study, search for data and information as appropriate references to support the truth of the material description, theory, and discussion. In this paper, the image datasets are collected from open sources, such as Google Image and the sites of car rental shops. The total number of image samples used for this research is 100. Labeling is a Python-based graphical image annotation tool that uses PyQt for its graphical user interface to tag pictures. Annotations are saved as text files in the YOLO format. The CreateML and PASCAL VOC formats are also supported. The item's region and class will be determined during the labeling process. The information format that was selected follows the YOLO principle. This data contains information on the objects' class, x-y coordinates, length, and width [31].

D. Overall Process

The research methods employed in this paper involve the implementation of three encryption algorithms, namely DES with Blowfish [32], AES, and Fernet [33] with HE (Homomorphic Encryption), for securing YOLOv5 model files and images. Each algorithm is implemented as a separate program with specific methodologies and techniques. The overall flowchart of the prototype is shown in Fig. 1.

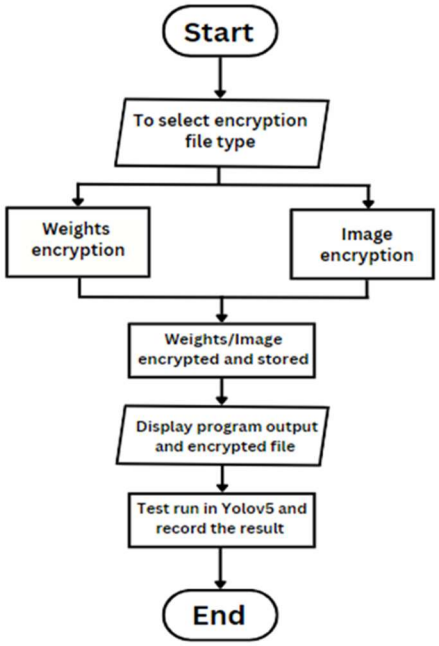


Fig. 1 Overall flowchart

E. Proposed Algorithms and Implementation

The modern methodologies meticulously devised for the encryption and decryption of sensitive data utilizing three sophisticated encryption paradigms: Data Encryption Standard (DES), Advanced Encryption Standard (AES), and Homomorphic Encryption (HE).

1) Data Encryption Standard (DES) Algorithm: The DES algorithm, renowned for its historical significance and cryptographic robustness, constitutes the cornerstone of our encryption framework. The DES encryption procedure initiates by prompting the user to select both a YOLOv5 model file and an image file, intended for encryption. Subsequently, the selected model file and image file are directed to the encryption process. In this process, both the model file and image file undergo two successive encryption rounds as illustrated in Fig. 2 and 3. In the first encryption round, the DES algorithm is applied to the model file. Following this, the Blowfish algorithm is executed for further encryption, adhering to the same process as applied to the image file. Notably, the DES algorithm necessitates a unique key, while the Blowfish algorithm requires an initialization vector (IV). To this end, the DES key and Blowfish IV are generated. The temporal efficacy of the encryption process is evaluated utilizing the time.perf_counter() function. The encrypted model is subsequently displayed in a text widget, serving the dual purpose of user visualization and verification.

For decryption, the DES decryption key and Blowfish decryption IV are retrieved. The time taken for decryption is assessed using the time.perf_counter() function. Both the decrypted model and image files are presented to the user for validation and visualization.

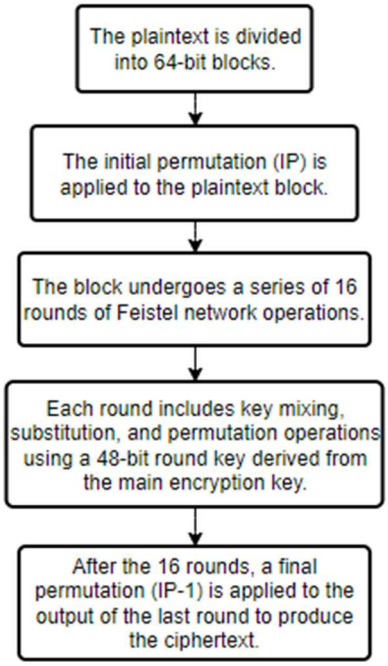


Fig. 2 DES encryption process

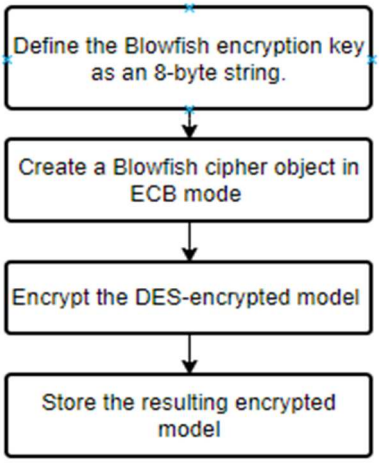


Fig. 3 Blowfish encryption process

2) Advanced Encryption Standard (AES) Algorithm: The AES encryption edifice embodies modern cryptographic intricacies. The AES encryption edifice, an embodiment of modern cryptographic intricacies. The AES encryption commences by designing an intuitive graphical user interface (GUI) using the Tkinter library. This GUI empowers the user to opt for a YOLOv5 model file and an image file intended for encryption, mirroring the process employed in the DES algorithm. Fig. 4 shows that AES encryption necessitates a 256-bit encryption key and an Initialization Vector (IV) in alignment with the DES encryption procedure. These values are programmatically generated prior to the encryption process. The commencement of the encryption process is marked by initiating a timer via the time.perf_counter() function to gauge the encryption duration. Distinctly, the selected model and image files are subjected to separate AES encryption utilizing the Cipher Block Chaining (CBC) mode and padding. The encryption process transforms the original data into ciphertext, rendering it indecipherable without the corresponding decryption key. Upon the completion of the encryption process, the timer is halted. The encrypted model and image files are stored for future utilization, and a user-centric success message is conveyed. In conjunction, the encrypted model data is showcased to the user via the Tkinter Text widget, offering visual feedback. Subsequently, during the decryption phase, the AES decryption algorithm is concurrently applied to both the model and image files. The 256-bit AES algorithm in CBC mode with padding is instrumental in the decryption process. The decryption endeavors utilize the same encryption keys and IVs as generated during the encryption phase. The temporal assessment of decryption is facilitated by time.perf_counter() function. The decrypted model data is presented within the GUI, accompanied by the creation of the decrypted image.

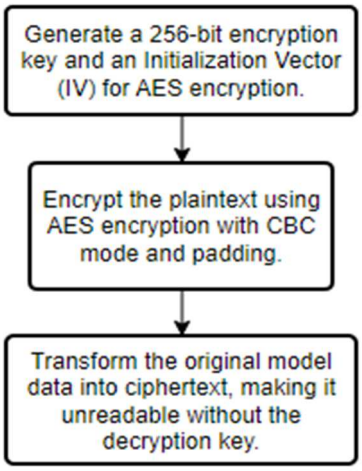


Fig. 4 AES encryption process

3) Homomorphic Encryption (HE) Algorithm: Within the vanguard of progressive cryptographic endeavors, the Homomorphic Encryption (HE) paradigm presents itself as an embodiment of cryptographic pioneering, designed to navigate the intricacies of secure data manipulation while transcending traditional cryptographic limitations. Figure 5 underscores the orchestration of the Homomorphic Encryption (HE) algorithm. The selected files are consecutively encrypted, first utilizing the Fernet encryption scheme, followed by the Paillier homomorphic encryption [34]–[36] technique to encrypt the Fernet Key. This involves the generation of a public-private key pair, with the encryption procedure relying on the public key. Subsequent to encryption, both the model and image files are stored, with the private key reserved for future decryption endeavors. Throughout the encryption process, informative message boxes furnish users with feedback. The GUI also facilitates the visualization of the encrypted model and image data in textual format. The decryption phase necessitates the loading of the private key for its execution. The decrypted model and image files are subsequently stored, concluding the process. The GUI further plays an integral role by displaying the decrypted model and image data to the user.

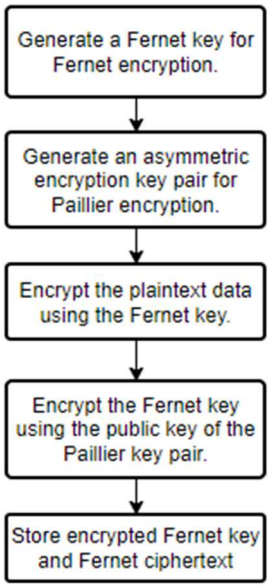


Fig. 5 Fernet key encryption of HE algorithm

III. Results and Discussion

This section depicts the results and discussion of the experiments carried out in this research by comparing various model and image encryption approaches.

A. Attack Methods on Ciphertext

The tests in this paper are the results of tests performed on encryption. The encrypted model files will be used to change extensions, autopsy tool [37] inspection, cipher identifiers [38], modification headers, and attacks on keys. These attacks will be performed with different applications. Table I referred. The encryption tests on the cryptographic files were performed to determine the resistance of the encryption methods implemented in this study to such attacks.

TABLE I
THE ATTACKS METHODS ON CIPHERTEXT AND THE RESULTS

Attack methods	**DES & Blowfish**	**AES**	**HE & Fernet**
Change Extension	Content remains unchanged	Content remains unchanged	Content remains unchanged
Autopsy Tool Examination	Not able to track the file content	Not able to track the file content	Not able to track the file content
Cipher Identifier	Not accurate	Not accurate	Not accurate
Modify Header	Not success	Not success	Not success
Attack On Key	Easy to detect	Easy to detect	Difficult to detect

From the result of the attack test on the ciphertext, it can be seen that:

- The test of changing the extension of the cipher text file did not work, and even if the file was changed in extension, it would not imply that it could be restored to the original file in another format.
- Test on Autospy tool on the cipher text verified that the encryption algorithm achieved data protection, which encrypts the weights, biases, etc. Before encryption, sensitive data in plain text was readable by text in Latin script, and after encryption, data files could not be forced to be read by some specific software.
- Test of the ciphertext on the cipher identifier resulted in the encrypted data format not being easily recognized. Since the plain text itself was only partially readable after being converted to Latin, it was even more impossible for the encrypted text to be read by humans. Even with some cryptographic identifiers, verifying what the encrypted cipher text represents was impossible, and the result was not accurate.
- Attacking a modified header on the cipher text was unwise because the plaintext header in the encrypted file result would be altered with the application of different encryption algorithms. Therefore, the cipher text cannot be decrypted even if it has been changed back to the header of the normal pt file.

Things turn out differently when the side channel used to attack the key. DES keys with shorter length, using only 56-bit keys, are vulnerable to brute-force attacks. There were three attacks known that can break the full 16 rounds of DES with less complexity than a brute-force search: differential cryptanalysis (DC) [39], [40]. In order to increase its security level, Blowfish encryption has been implemented on it. However, Blowfish and DES have relatively smaller key sizes. Thus, it might not significantly enhance overall security. While AES, as a symmetric encryption algorithm supporting key sizes of 128, 192, and 256 bits, means that the encryption algorithm is secure, but does not mean that the hardware and software are secure. There are possible to crack the information. The Fernet key, as a symmetric encryption key, incorporated the HMAC-SHA256-based Message Authentication Code (MAC) using the AES algorithm in CBC mode, providing a high-security level. On the other hand, Paillier encryption has been extensively studied and is considered secure against certain attacks, such as brute force attacks and selective plaintext attacks. Therefore, the combined encryption scheme is theoretically more secure than DES algorithm and AES algorithm.

B. Performance of Encryption Algorithms on Model

1) Accuracy of Model Ciphertext Decryption: The above experiment analyses the accuracy between the original and encrypted weights using *WinMerge* [41]. In addition, by running the *Yolov5* program, we can also determine if the decrypted cipher text has been altered or if the decryption is incomplete. The encryption algorithms involved in this investigation are AES, DES, Blowfish, Fernet, and Paillier, with homomorphic encryption. Figure 6 and Table II show the accuracy of ciphertext decryption. In the test, the performance of Paillier alone was tested in addition to the mixed encryption of Fernet and Paillier. We can clearly see that all three encryption schemes decrypt the cipher text perfectly, except for Paillier alone.

TABLE II
ACCURACY OF MODEL DECRYPTION

Encryption Algorithm	**DES & Blowfish**	**AES**	**HE**	**Fernet & HE**
Accuracy	100%	100%	0%	100%

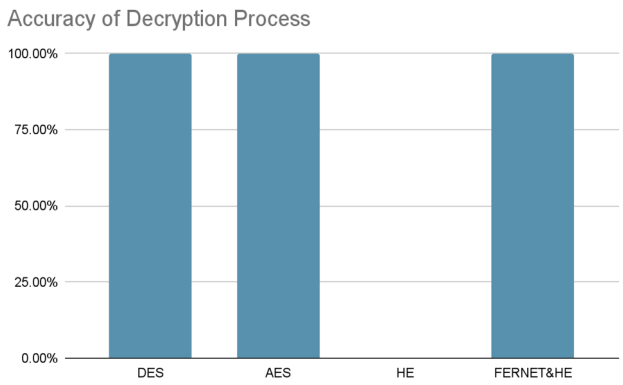


Fig. 6 Accuracy of model decryption

2) Speed Performance: To acquire more trustworthy findings, each category in this experiment is repeated twice. The speed performance results of DES, AES, and HE are recorded. Based on the test result data in Table III. We can see the encryption and decryption times for different algorithms and weight files. The encryption and decryption times vary according to the size of the algorithm and the weight file. In general, heavier files require greater encryption and decryption time compared to lighter files. DES with Blowfish exhibits moderate encryption and decryption speeds among the various techniques. However, it is vital to remember that DES and Blowfish are regarded as relatively weak and less safe compared to AES and HE. Moreover, AES shows a fair mix between security and performance. It delivers fairly rapid encryption and decryption speeds while maintaining high security. Furthermore, in conjunction with the Homomorphic Encryption (HE) algorithm, the Fernet encryption system demonstrates reasonably fast encryption and decryption speeds in files of varying weights. This demonstrates that Fernet, in conjunction with HE, can perform efficient encryption and decryption operations. As a result, the encryption and decryption times for the YOLOv5n, YOLOv5m, Exp l, Exp X, and Exp X6 weight files grow as the file size rises. This is understandable because bigger data needs more processing resources for encryption and decryption procedures.

TABLE III
SPEED OF ENCRYPTION ALGORITHMS ON DATA OF VARIOUS SIZES

Algorithm/Weight File	DES & Blowfish	AES	HE	Fernet & HE
Yolov5n - 3,967KB	Encryption Time: 0.11193 seconds Decryption Time: 0.09539 seconds	Encryption Time: 0.020959 seconds Decryption Time: 0.021053 seconds	Encryption time: 0.207021 seconds Decryption time: 0.205406 seconds	Encryption Time: 0.08403 seconds Decryption Time: 0.08161 seconds
Yolov5n - 3,967KB	Encryption Time: 0.11193 seconds Decryption Time: 0.09539 seconds	Encryption Time: 0.020959 seconds Decryption Time: 0.021053 seconds	Encryption time: 0.207021 seconds Decryption time: 0.205406 seconds	Encryption Time: 0.08403 seconds Decryption Time: 0.08161 seconds
Best.pt - 14,112KB	Encryption Time: 0.55761 seconds Decryption Time: 0.30662 seconds	Encryption Time: 0.069078 seconds Decryption Time: 0.064736 seconds	Encryption time: 0.492445 seconds Decryption time: 0.318864 seconds	Encryption Time: 0.68819 seconds Decryption Time: 0.42561 seconds
Yolov5m - 41,804KB	Encryption Time: 1.70342 seconds Decryption Time: 1.40162 seconds	Encryption Time: 0.726855 seconds Decryption Time: 0.479657 seconds	Encryption time: 0.647585 seconds Decryption time: 0.497766 seconds	Encryption Time: 1.57127 seconds Decryption Time: 0.77389 seconds
Exp l - 113,095KB	Encryption Time: 4.74583 seconds Decryption Time: 3.36297 seconds	Encryption Time: 2.459054 seconds Decryption Time: 1.349515 seconds	Encryption time: 2.613739 seconds Decryption time: 1.743041 seconds	Encryption Time: 3.96074 seconds Decryption Time: 2.51225 seconds
Exp X - 178,058KB	Encryption Time: 6.82246 seconds Decryption Time: 6.10376 seconds	Encryption Time: 3.817086 seconds Decryption Time: 2.034902 seconds	Encryption time: 8.330284 seconds Decryption time: 5.875806 seconds	Encryption Time: 6.65213 seconds Decryption Time: 4.34867 seconds

3) Performance of Encryption Algorithms on Large File: The psutil Python package was used in this test to gather data on RAM and CPU usage. To get the proportion of CPU usage, the psutil.cpu_percent() method is used, which measures the CPU usage over a 15-second period. The returned value represents the average CPU usage for that period. In Table IV, it shows the testing results on the performance of encryption algorithms on large file:

- CPU Usage: The CPU usage varies depending on the encryption technique. The AES method uses the least CPU (6.9), while the HE algorithm uses the most (19.). DES&Blowfish and Fernet&HE come somewhat in the middle, with CPU usages of 7.5 and 7.6%, respectively.
- RAM Memory Usage: The amount of RAM used varies depending on the encryption mechanism. The DES hybrid method consumes 72.5% of the RAM memory (7.151 GB), whereas the AES, HE, and Fernet&HE algorithms consume 70.6% (6.787 GB), 94.5% (9.257 GB), and 78% (7.135 GB), respectively. It is important to note that if the available memory is restricted, high RAM memory utilization may have an influence on overall system performance.
- Average Required Time: The average time indispensable for encryption and decryption processes varies. The average time for DES&Blowfish and Fernet&HE is 12 seconds and 13 seconds, whereas AES is the quickest with an average duration of 6

seconds. With an average time of 18 seconds, HE takes the longest.

TABLE IV
COMPARISON OF PERFORMANCE OF ENCRYPTION ALGORITHMS ON LARGE FILE (304,296KB)

Algorithm	DES & Blowfish	AES	HE	Fernet & HE
CPU Usage (Exp X6)	RAM used: 72.5%	RAM used: 70 .6%	RAM used: 94.5%	RAM used: 78%
	RAM Used (GB): 7.151395	RAM Used (GB): 6.786684	RAM Used (GB): 9.256743	RAM Used (GB): 7.134695
	CPU usage: 7.5	CPU usage: 6.9	CPU usage: 19.8	CPU usage: 7.6
Average Require Time	12 seconds	6 seconds	18 seconds	13 seconds

In conclusion, regarding the performance of encryption algorithms on huge files, AES has the fastest average time and the lowest CPU utilization. DES&Blowfish and Fernet&HE have comparable average times, with DES methods using the least CPU. HE uses the most CPU and has the longest average time. RAM memory utilization is relatively high for all algorithms, with HE uses the most RAM memory. It is critical to evaluate these aspects and establish a balance between them.

C. Performance of Encryption Algorithms on Image

1) Comparison between original image, encrypted image, decrypted image: This implementation has embedded image encryption to provide higher privacy protection and sensitive data, respectively applying DES [42], AES[43], and openCV image encryption algorithms. The results are shown in Table V. While openCV image encryption is implemented using OpenCV's GaussianBlur function [44]. The blurring effect is characterized by a kernel size of 305, which can lead to severely blurred images. This excessive blurring has the potential to make it challenging to decrypt the image and recover the original details accurately. The encrypted image of openCV was not successfully decrypted because the kernel size chosen was 305, which could lead to excessive blurring, making it difficult to recover the original image accurately. The GaussianBlur function used for encryption causes information loss in the image, especially with a large kernel size. This loss of information during encryption can make it challenging to recover the original image during decryption fully. In a sense, this is also a very secure method for one-way images. All three methods are well protected on the bright side of image privacy, and the vital information in the image is not visible to the human eye.

2) Attacks on encrypted image: In this research, ELA image analysis [45], lightwave adjustment, and Zsteg analysis have been applied to encrypted images to test the resistance of encrypted images (Figure 6). In *Photo-Forensics*'s ELA analysis (Error Level Analysis), the JPEG quality is set to 90%, the error scale is set to 20, and the opacity value is specified as 0.52, indicating the degree of transparency of an element in the image. The original image is not shown here in any place. Lightwave's adjustment attacks by *paint.net* do not show any trace of the original image provided. However, Zsteg [46] points out that the images have traces of being encrypted, such as PGP secret subkeys, and VISX image files. Although one may know that these are encrypted images, nonetheless, they are not easily cracked. While using a Steganography decoder application, the attempts to decode the images were unsuccessful, meaning that the decoding process could not identify the encryption algorithm, indicating that the original images could not be fully recovered or restored.

3) Accuracy of Image Ciphertext Decryption: Fig. 7, Table VI, and Table VII show a comparison of the accuracy of image decryption with various encryption algorithms. In this test, a separate HE encryption test is added. This test was used IMGonline.com. [47] to analyze the accuracy between the original image and the decrypted image. The 100% accuracy of DES and AES shows that these algorithms effectively decrypt images encrypted with their corresponding keys. This reliability can be attributed to their strong cryptographic properties and extensive testing and analysis. On the other hand, the accuracy provided for HE (Homomorphic Encryption) is 0%. Homomorphic encryption is an encryption scheme that allows to compute of encrypted data without decrypting it.

TABLE V
COMPARISON AMONG DES, AES AND OPENCV IMAGE ENCRYPTION SCHEMES ON ORIGINAL IMAGE, ENCRYPTED IMAGE AND DECRYPTED IMAGE

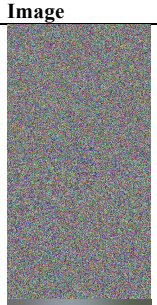
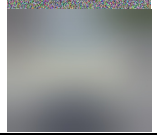
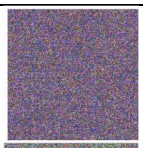
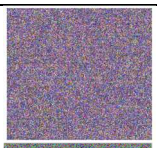
	Original Image	Encrypted Image	Decrypted Image
DES			
AES			
openCV			

TABLE VI
ATTACK METHODS ON ENCRYPTED IMAGE

Attack Methods	DES	AES	OpenCV & Fernet
ELA (JPEG quality:90% Error scale:20 opacity:0.52)			
Lightwave			
ZSteg Steganography	PGP Secret Sub-key - Not success	PGP Secret Sub-key - Not success	VISX image file Not reversible

However, it is known to be computationally intensive and inefficient compared to symmetric encryption algorithms such as DES and AES. 0% accuracy means that homomorphic

encryption's decryption process is unsuccessful or inapplicable in this case. The accuracy of 66% for OpenCV indicates that OpenCV is an open-source computer vision library that is partially successful in image decryption. openCV is designed primarily for image processing tasks and may not directly provide encryption or decryption functions. Therefore, the accuracy provided may be related to some specific image decryption techniques or methods implemented using OpenCV. In summary, DES and AES show high accuracy rates for image decryption, while HE appears to be ineffective in this regard. The accuracy provided for OpenCV shows moderate success, probably due to the specific image decryption techniques implemented using the library.

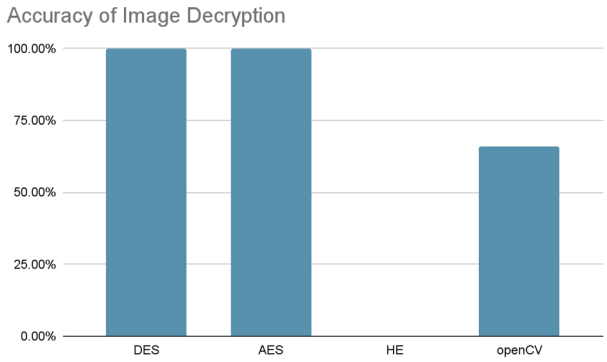


Fig. 7 Accuracy of image decryption

TABLE VII
ACCURACY OF MODEL DECRYPTION

Encryption Algorithm	DES	AES	HE	openCV
Accuracy	100%	100%	0%	100%

IV. CONCLUSION

This paper examines the application of neural network-based data encryption techniques to solve data security and privacy challenges in digital contexts. The effectiveness and performance of several encryption techniques, such as DES, Blowfish, AES, and homomorphic encryption, are compared and examined to protect CNN model weights and input images. AES strikes a balance between security and performance, providing relatively fast encryption and decryption speeds while maintaining high security. DES with Blowfish combination methods exhibits moderate encryption and decryption speeds, but it is considered relatively weak and less secure than AES and Homomorphic Encryption (HE). The combination of Fernet and HE demonstrates efficient encryption and decryption operations, making it a viable option.

However, it must be noted that encryption and decryption times increase with file size and require appropriate computational resources. Additional performance aspects were investigated, such as CPU utilization, RAM utilization, and the average time necessary for encryption and decryption. Several assaults were carried out on the encrypted pictures to test their resilience.

For future work, more study may be required in image encryption techniques to find a satisfactory compromise between security and the ability to retrieve the initial image within decryption [48] accurately, [49]. Advanced privacy-preserving approaches, such as differential privacy [50] and secure multi-party computation [51], may add security and secrecy during digital encryption and decryption.

ACKNOWLEDGMENT

We thank Multimedia University, Cyberjaya for supporting this article.

REFERENCES

[1] T. Munusamy and T. Khodadi, "Building Cyber Resilience: Key Factors for Enhancing Organizational Cyber Security," *Journal of Informatics and Web Engineering*, vol. 2, no. 2, pp. 59–71, Sep. 2023, doi: 10.33093/jiwe.2023.2.2.5.

[2] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 1066–1073. doi: 10.1016/j.procs.2022.01.135.

[3] J. Terven and D. M. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," *arXiv preprint*, Apr. 2023, [Online]. Available: http://arxiv.org/abs/2304.00501

[4] A. Dalvi, A. Jain, S. Moradiya, R. Nirmal, J. Sanghavi, and I. Siddavatam, "Securing Neural Networks Using Homomorphic Encryption," in *International Conference on Intelligent Technologies, CONIT 2021*, Jun. 2021, pp. 1–7. doi: 10.1109/CONIT51480.2021.9498376.

[5] K. Nandakumar, N. Ratha, S. Pankanti, and S. Halevi, "Towards deep neural network training on encrypted data," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, IEEE Computer Society, Jun. 2019, pp. 40–48. doi: 10.1109/CVPRW.2019.00011.

[6] J. W. Lee *et al.*, "Privacy-Preserving Machine Learning With Fully Homomorphic Encryption for Deep Neural Network," *IEEE Access*, vol. 10, pp. 30039–30054, 2022, doi: 10.1109/ACCESS.2022.3159694.

[7] G. Onoufriou, M. Hanheide, and G. Leontidis, "EDLaaS: Fully Homomorphic Encryption Over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting," *arXiv preprint*, Oct. 2022, [Online]. Available: http://arxiv.org/abs/2110.13638

[8] S. Hong, J. H. Park, W. Cho, H. Choe, and J. H. Cheon, "Secure tumor classification by shallow neural network using homomorphic encryption," *BMC Genomics*, vol. 23, no. 1, Dec. 2022, doi: 10.1186/s12864-022-08469-w.

[9] D. Nugent, "Privacy-Preserving Credit Card Fraud Detection using Homomorphic Encryption," *arXiv preprint*, Nov. 2022, [Online]. Available: http://arxiv.org/abs/2211.06675

[10] J.-W. Lee *et al.*, "Privacy-Preserving Machine Learning with Fully Homomorphic Encryption for Deep Neural Network," *IEEE Access*, vol. 10, pp. 30039–30054, Jun. 2022, doi: 10.1109/ACCESS.2022.3159694.

[11] Q. Lou and L. Jiang, "SHE: A Fast and Accurate Deep Neural Network for Encrypted Data," *Adv Neural Inf Process Syst*, May 2019, [Online]. Available: http://arxiv.org/abs/1906.00148

[12] Y. K. Yasin, P. Siddeeq, Y. Ameen, D. Hassan, and A. Chiad, "Advanced Encryption Standard (AES) Enhancement Using Artificial Neural Networks," *Int J Sci Eng Res*, vol. 5, no. 10, 2014, [Online]. Available: http://www.ijser.org

[13] Siddeeq. Y. Ameen and A. H. Mahdi, "AES Cryptosystem Development Using Neural Networks," *International Journal of Computer and Electrical Engineering*, vol. 3(2), pp. 315–318, 2011, doi: 10.7763/IJCEE.2011.V3.333.

[14] V. Lytvyn, I. Peleshchak, R. Peleshchak, and V. Vysotska, "Information Encryption Based on the Synthesis of a Neural Network and AES Algorithm," *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT 2019 - Proceedings*, pp. 447–450, Jul. 2019, doi: 10.1109/AIACT.2019.8847896.

[15] H. Kwon, H. Yoon, and K. W. Park, "Multi-Targeted Backdoor: Indentifying Backdoor Attack for Multiple Deep Neural Networks," *IEICE Trans Inf Syst*, vol. E103.D, no. 4, pp. 883–887, Apr. 2020, doi: 10.1587/TRANSINF.2019EDL8170.

[16] Sangeetha S and Haseena P, "Image Encryption using Deep Neural Networks based Chaotic Algorithm," *International Research Journal*

of Engineering and Technology, 2020, Accessed: Oct. 21, 2023. [Online]. Available: www.irjet.net

[17] Y. A. Liu *et al.*, "A dynamic AES cryptosystem based on memristive neural network," *Scientific* Reports *2022 12:1*, vol. 12, no. 1, pp. 1–11, Jul. 2022, doi: 10.1038/s41598-022-13286-y.

[18] R. Peleshchak, V. Lytvyn, N. Kholodna, I. Peleshchak, and V. Vysotska, "Two-Stage AES Encryption Method Based on Stochastic Error of a Neural Network," *Proceedings - 16th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2022*, pp. 381–385, 2022, doi: 10.1109/TCSET55632.2022.9766991.

[19] A. Mundra, S. Mundra, J. S. Srivastava, and P. Gupta, "Optimized deep neural network for cryptanalysis of DES," *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 5, pp. 5921–5931, May 2020, doi: 10.3233/JIFS-179679.

[20] Y. Chauvin, "A Back-Propagation Algorithm with Optimal Use of Hidden Units," *Neural Information Processing Systems*, 1988.

[21] N. Dhia and K. Al-Shakarchy, "Simulating DES Algorithm Using Artificial Neural Network," *Journal of Kerbala University*, vol. 10, 2012.

[22] M. A. B. Farah, R. Guesmi, A. Kachouri, and M. Samet, "A new design of cryptosystem based on S-box and chaotic permutation," *Multimed Tools Appl*, vol. 79, no. 27–28, pp. 19129–19150, Jul. 2020, doi: 10.1007/S11042-020-08718-8/METRICS.

[23] Z. Tolba, M. Derdour, M. A. Ferrag, S. M. Muyeen, and M. Benbouzid, "Automated Deep Learning BLACK-BOX Attack for Multimedia P-BOX Security Assessment," *IEEE Access*, vol. 10, pp. 94019–94039, 2022, doi: 10.1109/ACCESS.2022.3204175.

[24] R. Záluský, D. Ďuračková, V. Sedlák, and T. Kováčik, "The Use of Neural Network For Data Encryption Standard (DES)," May 2013.

[25] S. Fan and Y. Zhao, "Analysis of des Plaintext Recovery Based on BP Neural Network," *Security and Communication Networks*, vol. 2019, 2019, doi: 10.1155/2019/9580862.

[26] A. Benamira, D. Gerault, T. Peyrin, and Q. Q. Tan, "A Deeper Look at Machine Learning-Based Cryptanalysis," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12696 LNCS, pp. 805–835, 2021, doi: 10.1007/978-3-030-77870-5_28/COVER.

[27] Y. E. Yousif, "Improving The Efficiency of Des Algorithm Using Neural Networks," *International Journal of Engineering Applied Sciences and Technology*, vol. 5, no. 1, pp. 26–29, May 2020, doi: 10.33564/IJEAST.2020.V05I01.004.

[28] K. M. Alallayah, W. F. Abd El-Wahed, M. Amin, and A. H. Alhamami, "Attack of Against Simplified Data Encryption Standard Cipher System Using Neural Networks," *Journal of Computer Science*, vol. 6, no. 1, pp. 29–35, Jan. 2010, doi: 10.3844/JCSSP.2010.29.35.

[29] K. M. Alallayah, A. H. Alhamami, W. A. El-Wahed, and M. Amin, "Applying neural networks for simplified data encryption standard (SDES) cipher system cryptanalysis," *˜The œinternational Arab journal of information technology*, 2012.

[30] Q. Hu, L. Ma, and J. Zhao, "DeepGraph: A PyCharm Tool for Visualizing and Understanding Deep Learning Models," *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, vol. 2018-December, pp. 628–632, Jul. 2018, doi: 10.1109/APSEC.2018.00079.

[31] P. van Lunteren, "EcoAssist: A no-code platform to train and deploy custom YOLOv5 object detection models," *J Open Source Softw*, vol. 8, no. 88, p. 5581, Aug. 2023, doi: 10.21105/JOSS.05581.

[32] H. Dibas and K. E. Sabri, "A comprehensive performance empirical study of the symmetric algorithms:AES, 3DES, Blowfish and Twofish," *2021 International Conference on Information Technology, ICIT 2021 - Proceedings*, pp. 344–349, Jul. 2021, doi: 10.1109/ICIT52682.2021.9491644.

[33] E. Karuna Wijaya, R. Kumala, and B. Soewito, "Improving Security and Imperceptibility Using Modified Least Significant BIT and Fernet Symmetric Encryption," *J Theor Appl Inf Technol*, vol. 15, p. 17, 2022, Accessed: Oct. 21, 2023.

[34] M. Ghadamyari and S. Samet, "Privacy-Preserving Statistical Analysis of Health Data Using Paillier Homomorphic Encryption and Permissioned Blockchain," *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, pp. 5474–5479, Dec. 2019, doi: 10.1109/BIGDATA47090.2019.9006231.

[35] A. A. Alqarni, "A Secure Approach for Data Integration in Cloud using Paillier Homomorphic Encryption," *Albaha University Journal of Basic and Applied Sciences*, vol. 5, no. 2, pp. 15–21, 2021, Accessed: Oct. 21, 2023. [Online]. Available: https://portal.bu.edu.sa/web/jbas/

[36] T. B. Ogunseyi and T. Bo, "Fast Decryption Algorithm for Paillier Homomorphic Cryptosystem," *Proceedings of 2020 IEEE International Conference on Power, Intelligent Computing and Systems, ICPICS 2020*, pp. 803–806, Jul. 2020, doi: 10.1109/ICPICS50287.2020.9202325.

[37] G. Grispos and K. Bastola, "Cyber autopsies: The integration of digital forensics into medical contexts," *Proc IEEE Symp Comput Based Med Syst*, vol. 2020-July, pp. 510–513, Jul. 2020, doi: 10.1109/CBMS49503.2020.00102.

[38] E. Leierzopf, V. Mikhalev, N. Kopal, B. Esslinger, H. Lampesberger, and E. Hermann, "Detection of Classical Cipher Types with Feature-Learning Approaches," *Communications in Computer and Information Science*, vol. 1504 CCIS, pp. 152–164, 2021, doi: 10.1007/978-981-16-8531-6_11/COVER.

[39] G. Wang and G. Wang, "Improved Differential-ML Distinguisher: Machine Learning Based Generic Extension for Differential Analysis," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12919 LNCS, pp. 21–38, 2021, doi: 10.1007/978-3-030-88052-1_2/TABLES/6.

[40] M. Cao and W. Zhang, "Related-Key Differential Cryptanalysis of the Reduced-Round Block Cipher GIFT," *IEEE Access*, vol. 7, pp. 175769–175778, 2019, doi: 10.1109/ACCESS.2019.2957581.

[41] Adam Bertram, "How to Use WinMerge to Compare Files," Ipswitch.

[42] X. Zhang, L. Wang, G. Cui, and Y. Niu, "Entropy-Based Block Scrambling Image Encryption Using des Structure and Chaotic Systems," *Int J Opt*, vol. 2019, 2019, doi: 10.1155/2019/3594534.

[43] D. M. Alsaffar *et al.*, "Image Encryption Based on AES and RSA Algorithms," *ICCAIS 2020 - 3rd International Conference on Computer Applications and Information Security*, Mar. 2020, doi: 10.1109/ICCAIS48893.2020.9096809.

[44] H. Singh, "Advanced Image Processing Using OpenCV," *Practical Machine Learning and Image Processing*, pp. 63–88, 2019, doi: 10.1007/978-1-4842-4149-3_4.

[45] W. P. Sari and H. Fahmi, "The Effect of Error Level Analysis on The Image Forgery Detection Using Deep Learning," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, vol. 6, no. 3, pp. 187–194, Aug. 2021, doi: 10.22219/KINETIK.V6I3.1272.

[46] K. Sharma, A. Aggarwal, T. Singhania, D. Gupta, and A. Khanna, "Hiding Data in Images Using Cryptography and Deep Neural Network," *Journal of Artificial Intelligence and Systems*, vol. 1, no. 1, pp. 143–162, Dec. 2019, doi: 10.33969/AIS.2019.11009.

[47] J. Deepika, C. Rajan, and T. Senthil, "Security and Privacy of Cloud-And IoT-Based Medical Image Diagnosis Using Fuzzy Convolutional Neural Network," *Comput Intell Neurosci*, vol. 2021, 2021, doi: 10.1155/2021/6615411.

[48] X. Chai, Y. Wang, Z. Gan, X. Chen, and Y. Zhang, "Preserving privacy while revealing thumbnail for content-based encrypted image retrieval in the cloud," *Inf Sci (N Y)*, vol. 604, pp. 115–141, Aug. 2022, doi: 10.1016/J.INS.2022.05.008.

[49] J. Jain and A. Jain, "Securing E-Healthcare Images Using an Efficient Image Encryption Model," *Sci Program*, vol. 2022, 2022, doi: 10.1155/2022/6438331.

[50] M. U. Hassan, M. H. Rehmani, and J. Chen, "Differential Privacy Techniques for Cyber Physical Systems: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 1, pp. 746–789, Jan. 2020, doi: 10.1109/COMST.2019.2944748.

[51] B. Knott, S. Venkataraman, A. Hannun, S. Sengupta, M. Ibrahim, and L. van der Maaten, "CrypTen: Secure Multi-Party Computation Meets Machine Learning," *Adv Neural Inf Process Syst*, vol. 7, pp. 4961–4973, Sep. 2021, Accessed: Oct. 21, 2023. [Online]. Available: https://arxiv.org/abs/2109.00984v2.