Homomorphic Encryption in Image Classification

Shoukath Siddiqui
Department of Computer Science
BITS Pilani Dubai Campus
Dubai, UAE
f20220037@dubai.bits-pilani.ac.in

Abstract: This paper introduces the concept of homomorphic encryption with image classification algorithms. Image Classification is a technique that extracts relevant features from the image may it be positional data or qualitative data. Homomorphic encryption is a technique by which encrypted data used in calculations with each other to produce the encrypted result. This paper explores popular image classification algorithms like Convolutional Neural Networks, LeNet-5 and AlexNet. Additionally, how Homomorphism has evolved over the years is also mentioned.

Keywords: Homomorphic Encryption, Image Classification, CNNs, Machine learning;

1. Introduction

Since the evolution of digital imaging and the advent of the internet, large datasets of images have given modern image classification algorithms the power to use comprehensive and diverse data to create robust models that accurately tokenize as well as classify the data. Today we have apps that recognize and match a products using a single image, facial recognition algorithms and advanced vision-based systems.

A. History of Image Analysis

One of the first systems for visual classification developed was Bidirectional Associative Memory (BAM) [1]. It was used to recognize printed characters and was very susceptible to position and rotational orientation. The limitation of primitive algorithms is that they can only solve linear problems whereas image processing deals with a lot of variability.

In 1963, Roberts submitted his thesis paper detailing a visual machine model that identified features from an image and produced its corresponding three-dimensional structure [2]. As seen in Figure 1. Robert's method was revolutionary at its time. Although it only operated on predetermined 3D objects by exploiting perspective rules and making a computer determine said objects position via mapping edges on a 2D image with a mathematical equation, it was still a revolutionary technique for its time.

In 1983, Kunihiko Fukushima introduced the world to the Neocognitron, a precursor to modern Convolutional Neural Networks (CNNs) as we know. The Neocognitron is based on a similar paper from human retinal biology where it was observed that neurons close together interact with each other. Similarly, the Neocognitron performs operations on pixels together and this output with S-cells that choose features from previous iterations and C-cells. This method has now become ubiquitous with CNNs as most image analysis models are performed with a CNN.

Yann LeCun published an article on image classification using the MNIST handwritten numbers [4]. The neural network developed in this paper, named as LeNet-5 was a Convolutional Neural Network that generalized the findings of K Fukushima and achieved a mean error rate of 0.8% with one of lowest memory requirements needed by an algorithm.

With the popularization of Deep Learning and new large image data sets such as ImageNet with 3.2 million labeled images general image classification was one field that many researchers were working towards [5]. AlexNet was one such model. It won the 2012 ImageNet Large Scale Visual Recognition Challenge ILSVRC by 10 percentage points, a feat never done before. AlexNet implemented its algorithm on GPUs making deep Neural Network training practical. They even pioneered a new way of thinking about activation functions with the development of ReLU, which was computationally efficient compared to tanh or sigmoidal.

The network consisted of convolutions and fully connected layers. The convolutions applied multiple kernels on the input layer and extracted certain features. It was observed that after a few layers the output of one particular kernel had a high distinction for facial features. The kernels although very primitive were still capable of performing image classification. This made training the model faster as the size of the model was small.

As seen in Fig 1. The initial kernels only boost features based on orientations and color.

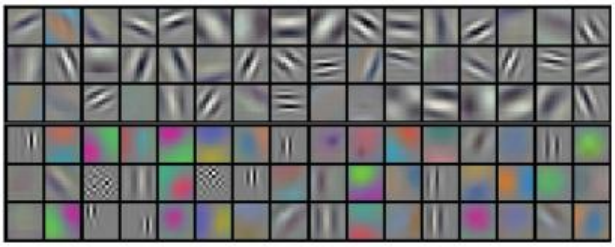

Figure 1. Kernel mappings of AlexNet.

B. Modern Uses of Image Analysis

Modern deep learning networks, especially convolutional neural networks (CNNs) and transformer-based models, have transformed many industries by making advanced computer imaging analysis possible.

Facial recognition technology, powered by deep learning models, is widely used for purposes ranging from unlocking smartphones and categorizing photos to security applications in airports and to check in conformation hardware at work or home.

CNNs and other deep learning models are used in medical imaging to diagnose conditions from X-rays, MRIs, and CT scans, improving diagnostic accuracy and speeding up the detection of diseases like cancer, Alzheimer's, and more. One of the first techniques utilizing imaging via neural networks was a medical diagnostic system to classify tumors in mammograms into cancerous or benign.

CNNs are used in self-driving cars to identify objects, lanes, pedestrians, and traffic signals, while other computer vision systems monitor drivers' alertness and detect distractions to enhance road safety.

In retail, computer vision models analyze customer behavior, detect trends, and improve store layouts and stock management.

Platforms like Facebook, Instagram, and YouTube use deep learning models to classify and filter content, and personalize content for users. Additionally, models detect copyright infringement of video using proprietary software.

In agriculture, CNNs help monitor crop health, soil conditions, and pest infestations using drone and satellite imagery, enabling more efficient farming practices. And, in environmental monitoring, deep learning models analyze satellite images to track deforestation, pollution, wildlife habitats, weather patterns and land changes.

C. Implementation of Neuro-Systems

As more industries adopt the use of Neural Networks within their products, different methods of performing operations on data emerges.

Cloud compute hosts the database on a server with dedicated graphics without the need for local storage. Cloud models scale faster and updating the model version is often immediate. Although cloud models offer security of the model the question of privacy with respect to the data being sent is raised.

Edge computing is a solution where the model lives much closer to the users, i.e. the model is often distributed to the user leading to faster latency and complete data sovereignty. However edge systems are limited by the power of the hardware of the end user, they also do not update immediately. Edge systems are also susceptible to leaking of the training dataset by reverse engineering the model.

Another powerful method to distribute model architecture is Federated Learning. The model is distributed to end users and this local model is trained on the data that it encounters. The local models will be aggregated ever so often into the main model that is stored on the cloud. Again, it is susceptible to similar privacy issues as compared to Edge Computing.

D. Privacy Solutions

Homomorphic encryption is a technique by which incoming data to a neuro model is encrypted in a way that preserves the necessary features required by the model. This form of encryption is very necessary when it comes to verifying sensitive data that is sent over to the cloud. Many companies are offering AI solutions to their users, and as large-scale deep learning models require a lot of computational ability it is not something you can actively distribute and preserving data sanctity.

In the case of Edge AI, where the software is distributed to end users, the security of the model becomes more pertinent. We also have to think about federated learning architecture, where each local model has to update its own structure. However, such functionality can be accomplished by normal program encryption techniques.

2. Homomorphic Encryption

Again, Homomorphic Encryption is scheme from which the encrypted data i.e. the ciphertext can still be operated upon by a function to get results from it without exposing the original data to anyone. This technique is widely used in privacy preserving Machine Learning algorithms and as of late researchers, mathematicians and computer scientist form the Homomorphic Encryption Standardization Consortium that builds guidelines and rules for open Homomorphic Encryption standards around the world.

A. Introduction

Let's say that Alice wants to perform a certain computation on her data, let's say x. Alice will then encrypt this data with and sends it to Bob $c \leftarrow ENC(x, pk)$.

Bob will now perform the operations on this data $r \leftarrow FUNC(f, c)$ and sends it to Alice.

Alice will now decode these encrypted results and extract the value of the function computation $y \leftarrow DEC(r, sk)$.

The above paradigm works if $y = f(x)$ for all valid (pk, sk) pairs.

Homomorphic Schemes are evaluated on 3 distinct attributes, correctness of the input function, privacy between Alice and Bob and finally compactness.

B. Types of Homomorphic Encryption

For a system to be homomorphic it should solve the following condition:

$$E(x) * E(y) = E(x \circ y)$$

If the $\circ$ operator is addition, then it refers to Additive Homomorphism and if the operator is multiplication, it is Multiplicative Homomorphism where the $*$ operator refers to some mathematical operation between two encrypted ciphertexts.

Based on what our system is capable of we categorize our system as Partially Homomorphic Encryption (PHE), Somewhat Homomorphic Encryption (SHE) and Fully Homomorphic Encryption (FHE).

Partially Homomorphic Encryption (PHE)

When the algorithm can only satisfy one of Additive or Multiplicative Homomorphism, it is known as PHE. Some examples are

- RSA: (satisfies multiplicative homomorphism) where if $RSA(x)$ and $RSA(y)$ are two operations then $RSA(x) \times RSA(y) = RSA(x \times y)$ where $RSA(x) = x^e \bmod n$.
- Pallier: (satisfies additive homomorphism) where if $PAL(x)$ and $PAL(y)$ are two operations then $PAL(x) \times PAL(y) = PAL(x + y)$ where $PAL(x) = g^x \cdot r^n \bmod n^2$ and g, n, r are predetermined numbers.

Somewhat Homomorphic Encryption (SHE)

Although SHE algorithms support both Additive and multiplicative Homomorphism unlike the previous PHE algorithms that can operate on numerous encrypted cyphertexts, SHE algorithms are limited in operations. This due to the exponential nature of noise growth leading to loss of data.

Fully Homomorphic Encryption (FHE)

Both PHE and SHE algorithms have existed for a long time, but FHE algorithms that offer both Additive and Multiplicative Homomorphism are considered modern systems of cryptography.

Gentry in 2006 created one of the first FHE algorithms to exist. They tackled the problem of noise in SHE by creating a bootstrapping algorithm. Bootstrapping essentially partially decrypts the current ciphertext and re-encrypts them. This makes sure that the noise in the encryption is limited to a certain level.

Gentry's scheme introduced the concepts of lattice-based cryptography, a lattice is a set of points in n-dimensional space. This lattice is picked in such a way that it remains unchanged over certain operations making it versatile for cryptographic-based systems. However, this solution was not without problems, finding the shortest vector, i.e. the shortest vector problem (SVP) in the lattice proved to be computationally intensive and learning with errors (LWE problem) becomes impossible without the secret key [6].

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan developed what we now call as the BGV scheme making an efficient FHE that improves on the previous scheme by making noise management faster. They did this by implementing modular switching that means as the number of computations increased the ciphertext modulus was reduced. By making the modulus smaller the noise gain per computation decreased too. A side benefit of modulur switching allows BGV to operate with ciphertexts of multiple different moduli. This achieves lower noise without using bootstrapping and without having to reveal the secret key. While on Gentry's previous scheme performing ciphertext multiplications resulted in an increase of the degree, in BGV we use key switching and re-linearization to create an equivalent ciphertext, key pair with reduced dimensions to keep calculations fast. By this time the LWE problem which is based on ideal lattices had expanded to the ring-learning with errors problem (Ring-LWE). The Ring-LWE problem operates on a ring rather than a vector space. This adjustment allows BGV to work with polynomial arithmetic rather than matrix operations, which is significantly more efficient. Apart from setting new benchmarks for FHE algorithms BGV also utilized a ton of optimizations such as batch querying and making use of bootstrapping only when absolutely necessary [7].

Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN) is the primitive implementation of the approach of FHE by Cheon Jung Hee, Kim Andrey, Kim Miran, and Song Yongsoo (CKKS) [8]. CKKS is able to perform calculations on numbers with approximate arithmetic. CKKS takes certain inspirations from BGV while providing with its own new paradigms. Just like BGV it rescales the dimension after performing an operation. It introduces some level of noise and is able to get an approximate answer for the computation without needing to deploy intense schemes to determine the precise value of the operands. Unlike integer-based schemes such as BGV, CKKS is able to operate on real valued data. CKKS encodes both real and complex numbers as polynomials through a process called as phase encoding. CKKS is also powerful in the way it allows you to change the precision of the operations based on a parameter. Similar to BGV, CKKS relies on the Ring Learning with Errors (Ring-LWE) problem for security. Ring-LWE involves distinguishing between noisy polynomial parameters over rings and is computationally impossible without the private key. This provides CKKS with quantum-resilient security and efficient encryption.

Feature	*Gentry's 2006 FHE*	*BGV (Brakerski-Gentry-Vaikuntanathan)*	*CKKS (Cheon-Kim-Kim-Song)*
Development Year	2006	2014	2018
Main Goal	Demonstrate FHE feasibility	Improve efficiency, reduce noise, enable complex operations	Support approximate arithmetic for real-valued data
Supported Data Types	Integers	Integers	Real and complex numbers

Noise Management	Bootstrapping required frequently to reset noise	Modular switching, key switching, optional bootstrapping	Rescaling after multiplication to manage precision and noise
Noise Reduction Technique	Bootstrapping	Bootstrapping (optional due to modular switching)	Rescaling after each operation instead of bootstrapping
Mathematical Foundation	Ideal lattices	Ring Learning with Errors (Ring-LWE)	Ring-LWE with complex number support
Multiplication Handling	Increases ciphertext dimension, relies on bootstrapping	Re-linearization reduces dimension after multiplication	Rescaling reduces dimension after multiplication
Encoding of Data	Binary integers	Integer polynomials	Complex numbers
Modulus Switching	Not present	Present	Present
Key Techniques	Bootstrapping, squashing, lattice-based cryptography	Modular switching, key switching, re-linearization, batching	Rescaling, batching, complex encoding, parallelization
Efficiency	Computationally intensive due to frequent bootstrapping	More efficient; bootstrapping optional, suitable for deeper computations	Highly efficient for floating-point operations; rescaling minimizes need for bootstrapping
Batching	Not supported	Supported; allows parallel operations on multiple data slots	Supported; essential for vector data
Ciphertext Structure	Single ciphertext for each value	Ciphertexts can hold multiple values (batched)	Supports encoding multiple complex numbers per ciphertext
Applications	Concept proof of FHE, theoretical exploration	General-purpose FHE for integers, suitable for deep computation without frequent bootstrapping	Machine learning, image processing, and other tasks requiring real-number arithmetic
Performance and Practicality	Highly impractical for real-world use due to high computation	Improved practicality, especially for integer-based computations	Practical for real-number applications, well-suited to encrypted machine learning
Encryption Overhead	High, complex bootstrapping for each operation	Reduced overhead due to modular switching	Minimal overhead in approximate calculations, rescaling is efficient
Security Basis	Ideal lattice problem	Ring-LWE	Ring-LWE with added support for approximate arithmetic
Supported Operations	Additions and multiplications on encrypted data; slow due to frequent noise management	Additions, multiplications; noise management improved	Approximate additions, multiplications, and divisions with real numbers
Approximate Arithmetic	Not supported	Not supported	Supported
Use of Floating-Point Numbers	Not supported	Not supported	Supported, designed for approximate floating-point calculations
Example of Usage	Proof of concept	Encrypted databases, private queries	Privacy-preserving machine learning, encrypted statistical analysis
Primary Limitations	Frequent bootstrapping requirement, high computational cost	Higher efficiency but still limited to integers; some bootstrapping required for very deep computations	Precision loss due to approximate arithmetic, not suitable for applications needing exact results

Table 1. Differences between FHE schemes

3. Literature Review

[10] saw through with the implementation of a PHE, pallier system with a linear regression model as the Machine Learning framework. For a simple algorithm like Linear Regression an additive PHE is more than enough to satisfy the requirements of homomorphic encrypted operations with the model.

[11] implements a neural network with federated learning capabilities as well as homomorphic encryption techniques. They use a distributed deep learning algorithm to make it parallelable and reducing training overhead. To implement the concept of Secure Multi Party Computation (SMPC) [12], they used a PHE with Pallier encryption. They argued that in a Multi-Layer Perceptron (MLP) the homomorphic algorithm only has to solve addition problems and hence can be considered a FHE in this particular use case. The paper also implemented an optimized Pallier algorithm which was used to reduce network training performance. Their final model known as Pallier Federated Multi-Layer Perceptron (PFMLP) constructed along with a Key Management Center (KMC). Their results showed that the PFMLP had an accuracy of 92% compared to a normal MLP with 90%.

[13] constructed a new scheme for classification called as FHE scheme for Greater Than Relation or FHE4GT. This FHE can compute whether one of the two encrypted inputs is greater than the other without knowing private key. The FHE4GT scheme was used in a new algorithm that was inspired from BGV to perform homomorphic classification. They used a Support Vector Machine model in their classifier to perform linear operations on the operands. The paper successfully performs GT classification without losing privacy.

[14] proposes a novel system called GuardML for privacy-preserving machine learning (PPML). GuardML employs Hybrid Homomorphic Encryption (HHE). HHEs combine symmetric-key encryption with homomorphic encryption to improve efficiency. In HHEs, data is first encrypted locally using a symmetric key. Then, the symmetric key is encrypted using a HE schemes. The paper describes two protocols, 2GML and 3GML, that learn the classification outcomes over encrypted data. To demonstrate its practicality, the researchers implemented an HHE-based Machine Learning application for classifying heart disease based on electrocardiogram (ECG) data. The experiments indicate that GuardML achieves slightly less but still competitive accuracy to plaintext inference while significantly reducing computational and communication costs.

[15] tested the implementation of a FHE with a Convolution Neural Network and determined that it was too computationally intensive to be practically feasible. The activation function crucial in CNNs is ReLU, and with most homomorphic models it doesn't have a clean representation in crypto space. Many different HE models approximate it to a polynomial. Models like PEGASUS [16] use a lookup table to determine activation function output. The authors of this paper concluded that CNNs with FHEs were limited by the activation functions. They proposed a new activation function, Self-Learnable Activation Function (SLAF) and a way for this function to operate with convolutions. The SLAF is a n-degree polynomial function with learnable parameters attached to each power. While training this function can adjust itself and simulate non-linear behavior. SLAF used along a neural network can provide seamless Homomorphic functionality the network. The SLAFNN migrates the issue of noise by migrating the data to lower order polynomials.

[17] proposes a novel and secure approach for medical image analysis using a distributed data architecture, federated learning, and Partially Homomorphic Encryption. Data fabric as a platform that maps the data to be easily discoverable, intersected, integrated, easily managed, and easily accessed across multiple environments. The authors address the challenges of managing and analyzing sensitive medical data while adhering to regulations like HIPAA and GDPR. Their architecture allows multiple healthcare centers to collaborate in training machine learning models without sharing raw patient data. They train this data on pituitary tumor classification, achieving 83% accuracy using a customized convolutional neural network model. They highlighted that using a PHE significantly limited their possibilities in models and data preprocessing methods. They mentioned that prioritizing data preprocessing along with FHE schemes is crucial for future endeavors.

[18] displayed the importance and feasibility of implementing fingerprint recognition using HE. The paper used Siamese Neural Network, a powerful neural network that calculates similarity between two objects, in this case fingerprints. The Siamese NN consists of two CNNs with same weights that computes similarity. They used the CKKS scheme to perform comparisons on different encrypted inputs. Their implementation, Blind-Touch achieved an F1 score of 92-98% on popular fingerprint databases such as PolyU and Sokoto.

4. Proposed Solution

We have seen many solutions to the problem of creating a Homomorphic satisfactory Machine Learning model. Although the type of homomorphic encryption scheme depends on our models' capabilities, FHE seems to be the most powerful and intuitive scheme available. As certain papers mentioned, having properly formatted data and focusing on data preprocessing will improve one's implementation. Another issue with HE is space, as encrypted ciphertexts can require large numbers and can often scale out of proportion.

As for image analysis, Convolutional Neural Networks are the most powerful image classification paradigm developed. However, they do not play well with HE and become computationally intensive to solve. To solve this, we can create our activation function that converts a convolution to a matrix multiplication operation. Conversely, we can use standard Multi-Layer Perceptron that can function out of the box with HE.

5. Model Development

For the MNIST handwriting dataset a CNN was developed with 1x4 convolutional and 3 fully connected layers. The model had an accuracy of 98%. The weights inferred from this model were transferred to a new model that supported Homomorphic encryption. To optimize the inference the converted the kernel mappings to a matrix mapping. Again, before encryption the we use im2col transformation to divide the current image by kernel sections and align them into rows. Since our Homomorphic model is accepting transformed inputs, this approach makes it a little more secure as the CKKS parameters are increased.

The model had an accuracy of 98% on both standard and Homomorphic Encrypted training data. However, it must be noted that each encrypted inference took 8000x times more than the standard tests. This is because the encryption transferred approximately 1KB of data into 50KB. With more data the computation load increases. Each inference took 0.826s on a *Ryzen 7 4800H @ 4.0GHz*.

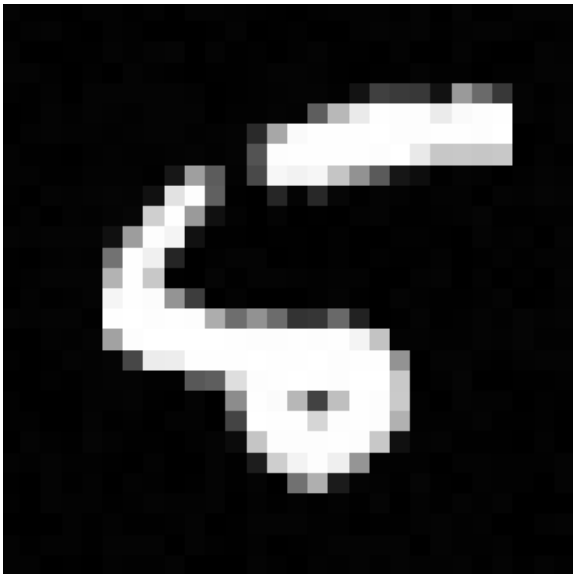

Figure 2. Plaintext sample from MNIST.

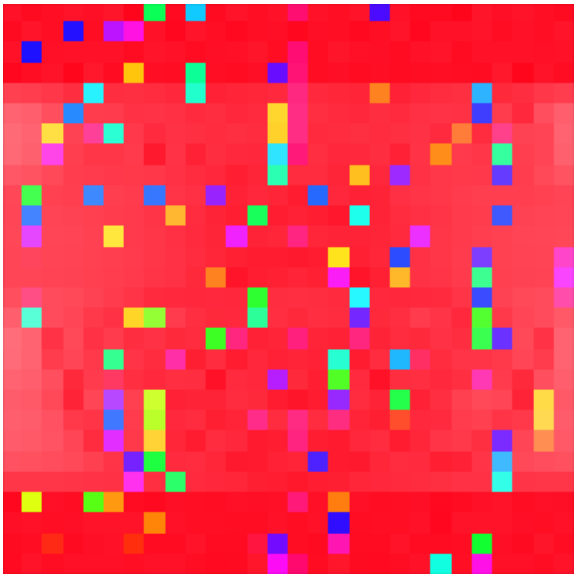

Figure 3. Encrypted Image of Fig 2. hue corresponds to argument angle and intensity to magnitude.

6. Proposal for Homomorphic Encrypted Federated Learning Model

With the popularization of Edge Computing and Edge AI, Federated model have become popular. Federated Learning distributes the model to the participants where they can use this model to run their own analysis and additionally train their local model to personalize their own experience. Over time the new weights of the model are aggregated and everyone's local model is updated.

Our proposal involves securing the updating of our new weights by encrypting the aggregates, such that the global model can be updated without revealing any individual's weights. However, since the global model needs to decrypt the computed/aggerated weights to update the plain weights, it will need the secret key. However, if the global model possesses the secret key, it may directly decrypt the incoming updates. The first solution to this problem we propose is having a trusted key bearer that only decrypts the aggregated result and sends it back to the global model. The second more thorough solution is to use multiparty computation techniques where each end user has their own partial decryption key and individually cannot decrypt the ciphertext. RSA an additive homomorphic encryption scheme is perfect for such a task and V. Vaikuntanathan in [19] describes the way it can be used in multiparty decryption.

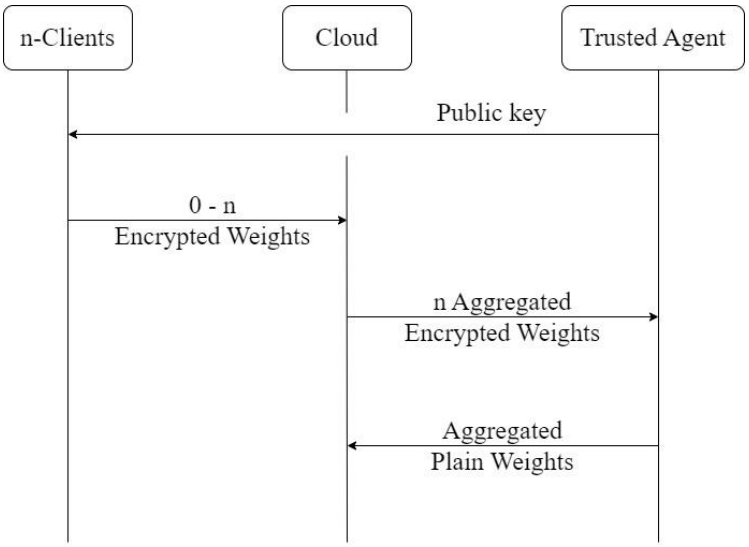

Figure 4. Proposed solution with trusted authority.

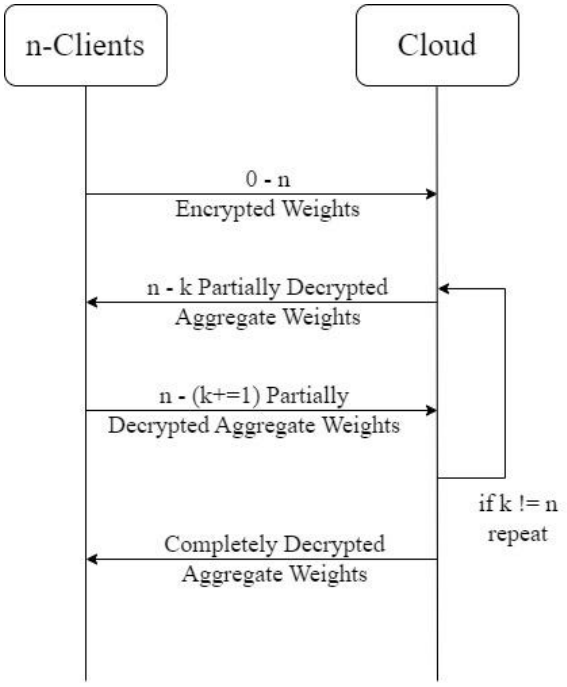

Figure 5. Proposed solution with multiparty decryption.

7. Conclusion

This paper consists of a literature survey that saw the evolution of Image classification from fitting points to predetermined 3-dimensional shapes to Convolutional Neural Networks. We also mentioned how these systems are used in modern day application. As many of these applications deal with sensitive user data a need for Encrypting said data but also being able to perform computations on it emerges. Homomorphic encryption solves this dilemma for us, by encrypting data in such a way that applications can operate on it and still produce a consistent result. We discussed the types of Homomorphic Encryption and the evolution of Fully Homomorphic Encryption and their security. We then analyzed papers that implemented Homomorphic systems along with neural networks or had a new method of tacking issues that arise with neuro-HE machines. We then proposed an optimal solution for those who want to build a homomorphic image classification neuro system. Finally, we looked at a modern way of integrating homomorphic encryption with federated learning and provided two solutions for it. Investigating efficient multi-party homomorphic decryption is something we leave for the future.

References

[1] B. Kosko, "Bidirectional associative memories," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 49-60, Jan.-Feb. 1988, doi: 10.1109/21.87054

[2] L. G. Roberts, “MACHINE PERCEPTION OF THREE-DIMENSIONAL SOLIDS,” May 1963.

[3] K. Fukushima, S. Miyake and T. Ito, "Neocognitron: A neural network model for a mechanism of visual pattern recognition," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 826-834, Sept.-Oct. 1983, doi: 10.1109/TSMC.1983.6313076.

[4] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[5] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[6] C. Gentry, “Fully homomorphic encryption using ideal lattices,” *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing - STOC ’09*, 2009, doi: https://doi.org/10.1145/1536414.1536440.

[7] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(Leveled) Fully Homomorphic Encryption without Bootstrapping,” *ACM Transactions on Computation Theory*, vol. 6, no. 3, pp. 1–36, Jul. 2014, doi: https://doi.org/10.1145/2633600.

[8] J. H. Cheon, A. Kim, M. Kim, and Y. Song, “Homomorphic Encryption for Arithmetic of Approximate Numbers,” *Advances in Cryptology – ASIACRYPT 2017*, pp. 409–437, 2017, doi: https://doi.org/10.1007/978-3-319-70694-8_15.

[9] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, “Bootstrapping for Approximate Homomorphic Encryption,” *Cryptology ePrint Archive*, 2018. https://eprint.iacr.org/2018/153 (accessed Nov. 09, 2024).

[10] S. Behera and J. R. Prathuri, "Application of Homomorphic Encryption in Machine Learning," *2020 2nd PhD Colloquium on Ethically Driven Innovation and Technology for Society (PhD EDITS)*, Bangalore, India, 2020, pp. 1-2, doi: 10.1109/PhDEDITS51180.2020.9315305.

[11] H. Fang and Q. Qian, “Privacy Preserving Machine Learning with Homomorphic Encryption and Federated Learning,” *Future Internet*, vol. 13, no. 4, p. 94, Apr. 2021, doi: https://doi.org/10.3390/fi13040094.

[12] R. Canetti, U. Feige, O. Goldreich, and M. Naor, “Adaptively secure multi-party computation,” *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC ’96*, 1996, doi: https://doi.org/10.1145/237814.238015.

[13] S. Arita and S. Nakasato, "Fully Homomorphic Encryption for Classification in Machine Learning," *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*, Hong Kong, China, 2017, pp. 1-4, doi: 10.1109/SMARTCOMP.2017.7947011.

[14] E. Frimpong, K. Nguyen, Mindaugas Budzys, T. Khan, and Antonis Michalas, “GuardML: Efficient Privacy-Preserving Machine Learning Services Through Hybrid Homomorphic Encryption,” *arXiv (Cornell University)*, Apr. 2024, doi: https://doi.org/10.1145/3605098.3635983.

[15] J. Xiong, J. Chen, J. Lin, D. Jiao, and H. Liu, “Enhancing privacy-preserving machine learning with self-learnable activation functions in fully homomorphic encryption,” *Journal of Information Security and Applications*, vol. 86, pp. 103887–103887, Sep. 2024, doi: https://doi.org/10.1016/j.jisa.2024.103887

[16] W. -j. Lu, Z. Huang, C. Hong, Y. Ma and H. Qu, "PEGASUS: Bridging Polynomial and Non-polynomial Evaluations in Homomorphic Encryption," *2021 IEEE Symposium on Security and Privacy (SP)*, San Francisco, CA, USA, 2021, pp. 1057-1073, doi: 10.1109/SP40001.2021.00043.

[17] S. A. Rieyan *et al.*, “An advanced data fabric architecture leveraging homomorphic encryption and federated learning,” *Information Fusion*, p. 102004, Sep. 2023, doi: https://doi.org/10.1016/j.inffus.2023.102004.

[18] H. Choi, S. S. Woo, and H. Kim, “Blind-Touch: Homomorphic Encryption-Based Distributed Neural Network Inference for Privacy-Preserving Fingerprint Authentication,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 20, pp. 21976–21985, Mar. 2024, doi: https://doi.org/10.1609/aaai.v38i20.30200.

[19] A. López-Alt, E. Tromer, and V. Vaikuntanathan, “On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption,” Proceedings of the 44th symposium on Theory of Computing - STOC ’12, 2012, doi: https://doi.org/10.1145/2213977.2214086.