Blockchain-based Multi-Diagnosis Deep Learning Application for Various Diseases Classification

Hakima Rym Rahal (hakimarym.rahal@univ-biskra.dz)
University of Biskra
Sihem Slatnia
University of Biskra
Okba Kazar
University of Biskra
Ezedin Barka
University of the United Arab Emirates Al Ain
Saad Harous
University of Sharjah

Research Article

Keywords: Blockchain, Smart Contracts, Deep Learning, Model Ensembling, Medical Data Security

Posted Date: May 2nd, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2860508/v1

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at International Journal of Information Security on August 7th, 2023. See the published version at https://doi.org/10.1007/s10207-023-00733-8.

Blockchain-based Multi-Diagnosis Deep Learning Application for Various Diseases Classification

Hakima Rym Rahal[a,1], **Sihem Slatnia**[b,1], **Okba Kazar**[c,1,2], **Ezedin Barka**[d,2], **Saad Harous**[e,3]

[1]LINFI Laboratory, Computer Science Department, University of Mohammed Khider Biskra
[2]College of Information Technology, University of the United Arab Emirates Al Ain, United Arab Emirates
[3]Computer Science Department, College of Computing and Informatics, University of Sharjah, Sharjah 27272

Received: date / Accepted: date

Abstract Misdiagnosis is a critical issue in healthcare, which can lead to severe consequences for patients, including delayed or inappropriate treatment, unnecessary procedures, psychological distress, financial burden, and legal implications. To mitigate this issue, we propose using deep learning algorithms to improve diagnostic accuracy. However, building accurate deep learning models for medical diagnosis requires substantial amounts of high-quality data, which can be challenging for individual healthcare sectors or organizations to acquire. Therefore, combining data from multiple sources to create a diverse dataset for efficient training is needed. However, sharing medical data between different healthcare sectors can be problematic from a security standpoint due to sensitive information and privacy laws. To address these challenges, we propose using Blockchain technology to provide a secure, decentralized, and privacy-respecting way to share locally trained deep learning models instead of the data itself. Our proposed method of model ensembling, which combines the weights of several local deep learning models to build a single global model, that enables accurate diagnosis of complex medical conditions across multiple locations while preserving patient privacy and data security. Our research demonstrates the effectiveness of this approach in accurately diagnosing three diseases (Breast cancer, Lung cancer, and Diabetes) with high accuracy rates, surpassing the accuracy of local models and building a multi-diagnosis application.

[a]e-mail: hakimarym.rahal@univ-biskra.dz
[b]e-mail: sihem.slatnia@univ-biskra.dz
[c]e-mail: o.kazar@univ-biskra.dz
[d]e-mail: ebarka@uaeu.ac.ae
[e]e-mail: harous@sharjah.ac.ae

Keywords Blockchain · Smart Contracts · Deep Learning · Model Ensembling · Medical Data Security

1 Introduction

Misdiagnosis can result in severe consequences for patients, such as delayed or inappropriate treatment, unnecessary treatments, psychological distress, financial strain, and legal implications. The patient's condition can worsen or become life-threatening due to delayed or inappropriate treatment. Furthermore, unnecessary treatments, such as surgeries or medications, can not only cause harm to the patient but also waste valuable resources and increase medical costs. Misdiagnosis can also have psychological effects on the patient, such as anxiety and post-traumatic stress disorder. In addition, it can cause a financial burden for the patient and legal consequences, including medical malpractice claims and lawsuits. To avoid misdiagnosis, it is crucial to use reliable diagnostic methods, especially in complicated cases. Deep learning can have a vital impact in decreasing the occurrence of misdiagnosis by examining vast quantities of medical data and recognizing complex patterns and correlations that may be difficult for human specialists to detect. Using deep learning algorithms, medical professionals can detect uncommon patterns in patient data, which can indicate a specific illness or condition, leading to more accurate diagnoses and a reduced risk of misdiagnosis. The implementation of deep learning technology in healthcare can enhance diagnostic precision, lower the possibility of misdiagnosis, and ultimately lead to improved patient outcomes. Accurately building and implementing deep learning models for medical diagnosis requires substantial amounts of high-quality data for training.

However, it is possible that a single healthcare sector or organization may not have sufficient data to develop a dependable deep learning model. To mitigate this, it is necessary to combine data from various healthcare sectors to create a diverse dataset for the efficient training of deep learning models. By combining data from multiple sources, deep learning models can detect patterns and connections across various medical conditions, resulting in more precise and dependable diagnoses. Moreover, deep learning models can recognize uncommon or intricate medical conditions that may be challenging for individual healthcare sectors or organizations to comprehend. Sharing medical data between different healthcare sectors can be problematic from a security standpoint for various reasons. Firstly, medical records contain sensitive information related to a patient's health, which is protected under privacy laws. As a result, healthcare providers must ensure that this information is not shared with any unauthorized individuals or organizations. Secondly, in the event of medical data breaches, the consequences can be severe, ranging from financial losses and damage to reputation to potential harm to patients. Therefore, healthcare providers must take the necessary precautions to safeguard their data and prevent unauthorized access. Thirdly, healthcare providers must comply with complex regulations such as HIPAA, which govern the collection, storage, and sharing of medical data, adding to the challenge of sharing data securely. To address these challenges, blockchain technology can provide several solutions. First, it enables a decentralized and secure storage and sharing of medical data, where each participant retains control over their data and decides who can access it. The data is encrypted using cryptographic protocols, and only authorized parties can access it, ensuring data privacy. Second, the immutable nature of blockchain ensures that medical data cannot be tampered with, which makes it a reliable and trustworthy source of information. This, in turn, increases the integrity of medical data and mitigates the risk of data breaches. Third, blockchain technology enables healthcare providers to comply with regulations such as HIPAA by creating a transparent and traceable system that records all access to medical data. This helps healthcare providers demonstrate their compliance with regulatory requirements and avoid possible penalties. In summary, blockchain technology offers a secure, efficient, and privacy-respecting way to share medical data while ensuring data integrity and compliance with regulations.

In our examination of relevant literature, we assessed various studies that employed blockchain technology to safeguard crucial medical information for the purpose of developing diagnostic systems. H. Hasanova et al. suggested a new algorithm, called Sine Cosine Weighted K-Nearest Neighbor (SCA-WKNN), which uses machine learning to predict heart disease. The algorithm learns from data stored in blockchain, which is a secure and tamper-resistant source of information for patient data. To measure its effectiveness, the SCA-WKNN algorithm was compared to other algorithms in terms of accuracy, precision, recall, F-score, and root mean square error. The results showed that the SCA-WKNN algorithm outperformed both W K-NN and KNN algorithms with a maximum accuracy rates difference of 4.59% and 15.61%, respectively [1]. M. Abraham et al. presented in the study referenced in [2] a predictive technique for ovarian cancer using microscopic images of protein expression data from the "Human Protein Atlas" dataset. The study developed a CNN Siamese network-based detection system that can identify the cancer-causing mutation. This technique allows a secure exchange of healthcare data, patient information, and model-predicted ovarian cancer results using blockchain technology. The effectiveness of the procedure was validated using the CRYPTO++ standards, and the model achieved an accuracy of 86%. However, it's worth noting that the model provides predictions based on unseen data and has only been tested on a small sample of classes. R. Kumar et al. suggested [3] a combination of smart contracts, private blockchain, and deep learning techniques called PBDL. The PBDL approach starts with blockchain technology to register and verify the cooperating parties using zero-knowledge proof, followed by a smart contract-based agreement method. The validated data is then used to create a novel DL approach called SA-BiLSTM using Stacked Sparse Variational AutoEncoder (SSVAE) and Self Attention-based Bidirectional Long Short Term Memory. The SSVAE is used to modify the structure of healthcare data, while SA-BiLSTM identifies and improves the threat detection technique. The security evaluation and testing results showed that the PBDL technique outperformed previous methods with an accuracy rate of 99.89% for ToN-IoT and 99.98% for IoT-Botnet. The study suggested testing the software-defined network version of the PBDL approach to verify its effectiveness and flexibility. E. A. Mantey et al. proposed a new approach to access control, which enables users to create their own access policies without having to get permission from data owners. To achieve this, the design uses an Access Control Repository (ACR) as part of an identity-based access control system. The ACR is then integrated into smart contracts, which are distributed across a Blockchain network by data owners to ensure the security of personal data and prevent sensi-

tive data leaks. Additionally, the article explains a technique for identifying COVID-19 in x-ray images using Deep Learning, Keras, and Tensor flow, which yielded impressive results with an accuracy of 90-95% [4]. T. E. Tan et al. used deep learning algorithms to construct a platform capable of identifying severe myopia and myopic macular degeneration. The development of three deep learning models and the utilization of different datasets from various countries for training and testing enabled the achievement of this goal. Blockchain technology was employed to address the issue of securely transmitting data and models between healthcare institutions. The accuracy of the model was 91.3% for high myopia and 96.9% for myopic macular degeneration. Moreover, the deep learning algorithms surpassed the performance of six human specialists, accurately detecting myopic macular degeneration with 97.8% precision and extreme myopia with 97.3% precision [5]. H. Subramanian et al. developed and tested a digital pathology system that prioritizes confidentiality, security, and personalization. They utilized smart contracts based on Ethereum, the nonfungible token (NFT) standard, and the Interplanetary File System to ensure the system's distributed nature and data storage. The proposed method was implemented into existing data systems and is expected to enhance the speed and accuracy of diagnoses, as well as increase the availability of professional pathological assessments [6].

These studies and works share a common limitation that should not be overlooked. If any of the parties involved in a cooperative system decide to leak medical sensitive data intentionally or unintentionally, it has significant consequences and poses a significant threat to the system's integrity and the privacy of patients. The potential impact of a such data leak cannot be overstated, as medical records contain personal and sensitive information that is protected by privacy laws and ethical considerations. Medical data leaks can lead to severe consequences, including identity theft, blackmail, discrimination, and harm to the patient's reputation, among other risks. Moreover, such a leak could damage the trust and credibility of the entire healthcare system, leading to a loss of public confidence.

The focus of our research is to tackle this significant limitation of using blockchain technology that has implications for the privacy of patient data. To overcome this limitation, we propose a stronger method that is built based on model ensembling, which leverages the strengths of multiple local deep learning models to train a more powerful and robust global deep learning model. The key idea behind this method is to combine the weights of several local models, each of which has been trained on data from a specific location or source, into a single global model that can make accurate predictions on a wide range of data. To ensure the secure transfer of the models between different locations, we have utilized blockchain technology, which offers an immutable and decentralized platform for data sharing and collaboration. With this approach, we can guarantee the privacy and security of patient data and prevent unauthorized access or tampering.

Based on this technique we succeeded in building a multi-diagnosis application that can accurately diagnose 3 diseases (Breast cancer, Lung cancer, and Diabetes). Supposing that 3 different hospitals were collaborating to build this application, 2 hospitals build 3 local ANN models for each disease using their private data, then sharing their models with the third hospital. The third hospital builds 3 global deep learning models for each disease by combining the weights of the previous local models and training the global ones on his private data. Then using these global models to build the application that was shared again with the other hospitals through blockchain.

The accuracy of the Breast cancer global model reached 97.44% (overcoming the local models with a difference of 2.7% for the first model and 2.71% for the second one). As for the Lung cancer global model, it showed accuracy up to 98.51% (overcoming the local models with a difference of 2.99% for the first model and 5.97% for the second one). And the last is the Diabetes global model which reached 97.14% (overcoming the local models with a difference of 11.43% for the first model and 2.85% for the second one).

The rest of the paper is organized as follows: Our methodology is explained in Section 2, Section 3 represents the implementation of the methodology. Section 4 includes the complexity calculations for each model, The obtained results are shown in Section 5, Section 6 represents and overview of the application, and finaly the paper is concluded in Section 7.

2 Proposed Methodology

In the field of healthcare, data security and privacy are crucial aspects that must be taken into consideration when developing deep learning models for medical diagnosis. The potential risks of data leaks or breaches can have serious consequences for patients and healthcare providers alike. To address this issue, we propose a novel method based on Blockchain technology that allows for the secure sharing of locally trained deep learning models on private data.

Our method involves using Blockchain to create a decentralized and secure network for sharing the locally

trained models on private data. This means that sensitive patient data will not be shared with anyone.

In addition to the security benefits, our method optimizes the overall performance of the global deep learning model using model ensembling. By combining the weights of the locally trained models, we can create a more accurate and reliable diagnosis model that is less susceptible to misdiagnosis. This is particularly important in the field of medical diagnosis, where even small errors can have significant consequences for patients.

Overall, our proposed method offers a unique solution to the challenge of developing deep learning models for medical diagnosis while ensuring the security and privacy of sensitive patient data. By leveraging Blockchain technology and model ensembling, we are able to achieve both improved performance and enhanced data security, leading to better outcomes for both patients and healthcare providers.

An overview of the proposed methodology is represented in Figure 1.

As we can see in Figure 1, each collaborator hospital trains a local deep learning model using its private data then shares it in blockchain except for the last hospital that is going to build the global model, the last hospital obtains the local models of the other hospitals from the blockchain, extracts their final weights, and uses them to train the global model on its private data.

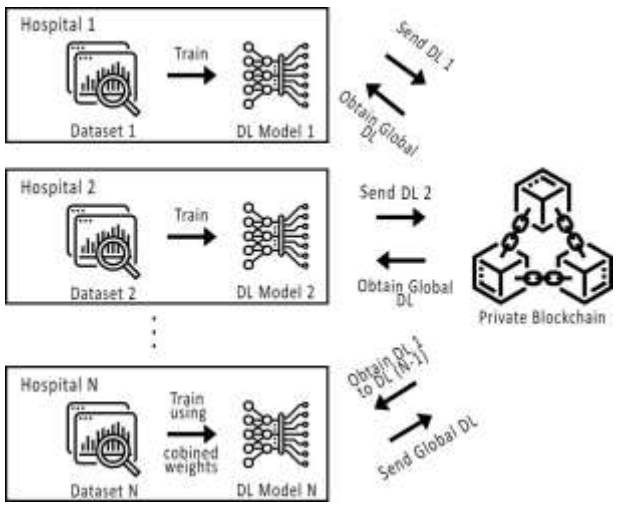

Fig. 1 Overview of our method

3 Implementation

Suppose that there are 3 hospitals, each with an amount of patient data that could potentially be shared and used to develop a strong deep learning model. However, due to the sensitive nature of patient data, it is crucial to ensure the privacy and security of the data throughout the development process. To address this issue, a solution is to build local models in the first 2 hospitals based on their own private data, and then share these models through blockchain technology with the third hospital. In this case, three different diseases (Breast cancer, Lung cancer, and Diabetes) are the focus, and each disease has two local models developed in hospital 1 and hospital 2. Blockchain technology offers an immutable and secure platform that can facilitate data sharing while maintaining privacy, security, and integrity. By leveraging blockchain, hospitals can securely share their locally developed models without compromising the privacy of patient data, and be certain that the deep learning models have not changed.

Hospital 3, extracts the final weights of model 1 and model 2 for each disease, combines them, and uses them to train a global model on its own private data. By using the locally developed models from multiple hospitals, the global model can learn from a more extensive range of data, making it more accurate and robust, this is what's called model ensembling.

In the end, hospital 3 creates a multi-diagnosis application using these global models.

3.1 Datasets

To validate our methodology we used 3 different datasets.

3.1.1 Breast Cancer Wisconsin (Diagnostic) DataSet

The Breast Cancer Wisconsin (Diagnostic) dataset [7] [8] is a valuable resource in the field of machine learning, particularly for developing models related to breast cancer diagnosis. This dataset, which we examined in our evaluation, can be found in the Kaggle repository and was acquired from UCI Machine Learning. It consists of 569 instances, each comprising of an ID number, 30 different characteristics, a binary class indicating whether the lump is malignant or benign, and an empty attribute, making a total of 33 attributes. The data for this dataset was collected by analyzing fluid samples from patients with solid breast lumps, using Xcyt, a graphic computer software program that is user-friendly and effective. Xcyt computed 10 features for each cell sample using a curve-fitting technique, which were then used to calculate the mean value, extreme value, and standard error for each feature, resulting in a 30 real-valued vector for each image in the dataset. This dataset is vital for developing models that can improve

the accuracy of breast cancer diagnosis. It provides a large and diverse collection of data that can be used for training and testing these models. The detailed characteristics and binary class labels in the Breast Cancer Wisconsin (Diagnostic) dataset make it an invaluable tool for machine learning researchers and practitioners working to advance the field of cancer diagnosis.

3.1.2 Lung Cancer Prediction Dataset

The Lung Cancer Data is available on data.world [9] and Kaggle [10] is from the Cancer Data Health Program (CDHP), which is an organization that provides access to publicly available cancer data. The CDHP collects data from various sources, such as the National Cancer Institute's Surveillance, Epidemiology, and End Results (SEER) program, and makes it available for research purposes. This database is a collection of 1000 instances with 23 attributes that provide information about the symptoms, risk factors, and potential diagnosis of lung cancer. This database is a crucial asset for medical researchers and practitioners who are seeking to understand and treat this deadly disease. The attributes within the database capture a broad range of factors that may contribute to an individual's risk of developing lung cancer. These include demographic information such as age, gender, and smoking history. One of the key features of this database is the inclusion of risk level categories, which are classified as Low, Medium, and High. This categorization provides valuable insights into the severity of an individual's lung cancer risk, and can help medical professionals to develop targeted interventions and treatment plans. In addition to the risk level categories, each instance within the database is also assigned a unique index number and patient ID. Overall, this database is a vital resource for medical researchers and practitioners alike, providing a wealth of information about the factors that contribute to lung cancer risk, and helping to guide the development of effective prevention and treatment strategies for this devastating disease.

3.1.3 Diabetes UCI Dataset

The early stage diabetes risk prediction dataset is an essential resource for researchers and healthcare practitioners working to improve the early detection and prevention of diabetes. This dataset contains clinical and demographic data from 520 patients who were screened for diabetes at Sylhet Diabetes Hospital in Bangladesh. The 16 variables included in the dataset provide valuable information on the patient's age, gender, and diabetes symptoms, which are critical indicators of diabetes risk. The dataset also includes information on whether the patient has diabetes (or is at risk of developing diabetes) or not. This information can be used to develop machine learning models that predict the risk of early-stage diabetes accurately. By identifying individuals who are at risk of developing diabetes at an early stage, healthcare practitioners can implement preventive measures to reduce the risk of complications and improve patient outcomes. Moreover, researchers can use this dataset to investigate new diagnostic techniques and treatments for diabetes. The dataset is publicly available on the UCI Machine Learning Repository [11] and Kaggle [12], and has been widely used in research studies related to diabetes risk prediction, highlighting its significance as a valuable resource for the scientific community.

3.2 Preparation of the datasets

Assuming that there are 3 hospitals, we aim to utilize the technique illustrated in Figure 1. To achieve this, it is necessary to divide the dataset into three distinct datasets that share the same structure. The following steps are applied to the three datasets. We loaded the CSV file 'data.csv' into the dataset variable using the csv.reader() function. The header row is extracted and removed from the dataset variable. The dataset is then shuffled using the random.shuffle() function to randomize the order of the rows. The size of each part is determined by dividing the total number of rows by 3 and storing the result in the part_size variable. Next, we created three new CSV files for each part using the open() function and wrote the header row to each file using the csv.writer() function. To split the dataset into the three parts, we iterated through the shuffled dataset using a for loop. Each row is written to the appropriate part CSV file based on its index using the csv.writer() function. The first part will receive the first part_size number of rows, the second part will receive the next part_size number of rows, and the third part will receive the remaining rows. Finally, the code closes the three new CSV files using the close() method to free up system resources. Currently, we possess three distinct csv datasets, which have the same structure for every disease category. Table 1 displays the count of instances and characteristics for each section of the datasets, following their division.

It is apparent that Parts 1 and 2 comprise 189 occurrences and 30 characteristics, while Part 3 encompasses 191 occurrences (giving a total of 569) and 30 features for the dataset of Breast cancer. Furthermore, we have eliminated the ID and Unnamed features as they are not required in any of the datasets. Moreover, we have

Table 1 Information on Different Parts of the Datasets for Three Diseases

Disease	Dataset Parts	Instances	Features
Breast Cancer	Part 1	189	30
	Part 2	189	30
	Part 3	191	30
Lung Cancer	Part 1	333	23
	Part 2	333	23
	Part 3	334	23
Diabetes	Part 1	173	16
	Part 2	173	16
	Part 3	174	16

transformed the ”diagnosis” column values, representing the class, from ”B” and ”M” to binary values 0 and 1, respectively. As for Lung cancer dataset Parts 1 and 2 comprise 333 occurrences and 23 characteristics, while Part 3 encompasses 334 occurrences (giving a total of 1000) and 23 features. In addition, we have eliminated the Patient ID and Index features as they are not required in any of the datasets. Also, we have transformed the ”Level” column values, representing the class, from ”Low”, ”Medium” and ”High” to Integer values 0, 1, and 2, respectively. And Parts 1 and 2 of Diabetes dataset comprise 173 occurrences and 16 characteristics, while Part 3 encompasses 174 occurrences (giving a total of 520) and 16 features. Also, we transformed all the features values from ”Yes” and ”NO, to binary values 1 and 0 respectively. After that, the different segments of each datasets parts are divided into two groups: one comprising the independent variables (x), and the other, the dependent variable (y). Next, the dataset is divided into two sets for training and testing, respectively, utilizing the train_test_split function. Lastly, the data is standardized by employing the StandardScaler object from scikit-learn, which scales the characteristics of the dataset to have an average of 0 and a variance of 1.

3.3 Deep Learning Models

An Artificial Neural Network (ANN) is a type of machine learning model inspired by the human brain, composed of interconnected nodes that receive, process and produce output data. The ANN is trained through a process called backpropagation that adjusts the weights of the connections between the nodes in the network. ANN has been successfully used in a variety of applications such as image recognition, natural language processing, speech recognition, and financial forecasting. It has been developed by many notable scientists and researchers, and its development continues to be an active area of research [13].

In our study we built two types of ANN, one for binary classification (In the cases of Breast Cancer and Diabetes Classification), and the other one is for categorical classification of Lung Cancer with 3 classes.

3.3.1 Binary Classification ANN (Breast Cancer and Diabetes)

The ANN model was created using the Sequential() function from the Keras library. The model is initialized as an empty sequence of layers. Next, two dense layers were added to the model using the add() function. The first dense layer has 30 neurons with the Rectified Linear Unit (ReLU) activation function, and the input layer is automatically added with the input dimension specified by the variable ’number_features’. The BatchNormalization layer (Batch normalization is a technique that improves the training of deep neural networks by normalizing the inputs in each batch to have zero mean and unit variance. This can help to reduce overfitting and improve generalization.) is added after the first dense layer to normalize the inputs. Dropout regularization is also applied after BatchNormalization with a rate of 0.5 to randomly set 50% of the neuron outputs to zero during training to reduce overfitting. The second dense layer is added in the same way with 30 neurons, and another BatchNormalization and Dropout layers are added after this layer. Finally, an output layer with a single neuron and sigmoid activation function is added for binary classification. The model is then compiled with the Adam optimizer and binary cross-entropy loss function. The accuracy metric is also specified to monitor the performance of the model during training. Callbacks are set to monitor the validation loss during training and stop the training if the validation loss does not decrease for 25 consecutive epochs to avoid overfitting. The model is then trained using the fit() function with the training and validation data. The batch size is set to 32, and the number of epochs is set to 100. The callbacks are also passed to the fit() function to monitor the performance of the model during training. Figure 2 is an example of the model.

The mathematical explanation is as follows:

- **Notation:** Let us define the following notation used in the equations:

- x: the input vector of size *number features*
- $W^{(i)}$: the weight matrix of layer i
- $b^{(i)}$: the bias vector of layer i
- $z^{(i)}$: the linear combination of the inputs and weights of layer i
- $a^{(i)}$: the output (activation) vector of layer i
- $p^{(i)}$: the dropout probability of layer i

- $BN^{(i)}$: the batch normalization transformation of layer i
- y_{pred}: the predicted output of the network

- Equations: The equations for each layer are as follows:

Input layer:

There is no equation for the input layer, as it just passes the input vector x to the first hidden layer.

First hidden layer:

$$z^{(1)} = W^{(1)}x + b^{(1)}$$
$$a^{(1)} = \text{ReLU}(z^{(1)})$$
$$a^{(1)'} = BN^{(1)}(a^{(1)})$$
$$a^{(1)''} = \text{Dropout}(a^{(1)'}, p^{(1)})$$

where ReLU(z) = max(0, z) is the rectified linear unit activation function, BN (a) is the batch normalization transformation, and Dropout(a, p) randomly drops out a fraction p of the activations a during training to prevent overfitting.

Second hidden layer:

$$z^{(2)} = W^{(2)}a^{(1)''} + b^{(2)}$$
$$a^{(2)} = \text{ReLU}(z^{(2)})$$
$$a^{(2)'} = BN^{(2)}(a^{(2)})$$
$$a^{(2)''} = \text{Dropout}(a^{(2)'}, p^{(2)})$$

Output layer:

$$y_{pred} = \sigma(W^{(3)}a^{(2)''} + b^{(3)})$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid activation function.

During training, the network is trained to minimize the binary cross-entropy loss function $L(y, y_{pred})$ using the Adam optimizer. The loss function is defined as:

$$L(y, y_{pred}) = -[y \log(y_{pred}) + (1 - y) \log(1 - y_{pred})]$$

where y is the true label (0 or 1), and y_{pred} is the predicted output of the network.

The accuracy metric is calculated as the percentage of correctly classified examples in the validation set.

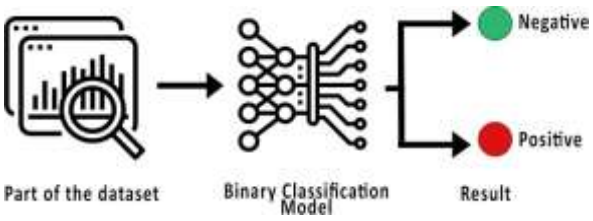

Fig. 2 Binary Classification Model

3.3.2 Categorical Classification ANN (Lung Cancer case)

The Artificial Neural Network (ANN) model is created using Sequential() function by creating an empty sequence of layers that the model will contain. The model architecture is built by adding two dense layers to the model using the add() function. The first dense layer has 30 neurons with the Rectified Linear Unit (ReLU) activation function, and the input dimension is automatically added with the number of features specified by the variable 'number_features'. A BatchNormalization layer is added after the first dense layer to normalize the inputs, which can reduce overfitting and improve generalization. Dropout regularization is applied after the BatchNormalization layer with a rate of 0.5, which randomly sets 50% of the neuron outputs to zero during training to reduce overfitting. The second dense layer is added in the same way with 30 neurons, and another BatchNormalization and Dropout layers are added after this layer. An output layer with 3 neurons and softmax activation function is added for multi-class classification. The model is compiled using the Adam optimizer and categorical cross-entropy loss function. The accuracy metric is specified to monitor the performance of the model during training. Callbacks are set to monitor the validation loss during training and stop the training if the validation loss does not decrease for 10 consecutive epochs to avoid overfitting. The model is then trained using the fit() function with the training and validation data. The batch size is set to 32, and the number of epochs is set to 37. The callbacks are also passed to the fit() function to monitor the performance of the model during training. The training history is stored in the 'history' object. Figure 3 is an example of the model.

The mathematical explanation is as follows:

3.4 Notation

Let us define the following notation used in the equations:

– *num classes*: the number of output classes

The equations for layer 1 and 2 are the same as the binary classification.

The output layer is as follows:

3.4.1 Output layer

$$y_{pred} = softmax(W^{(3)}a^{(2)''} + b^{(3)})$$

where $softmax(z) = \frac{e^z}{\sum_{i=1}^{num\_classes} z_i}$ is the softmax activation function for multi-class classification.

During training, the network is trained to minimize the categorical cross-entropy loss function $L(y, y_{pred})$ using the Adam optimizer. The loss function is defined as:

$$L(y, y_{pred}) = -\sum_{i=1}^{num\ classes} y_i \log(y_{pred_i})$$

where y is the true label (one-hot encoded vector) and y_{pred} is the predicted output of the network.

The accuracy metric is calculated as the percentage of correctly classified examples in the validation set.

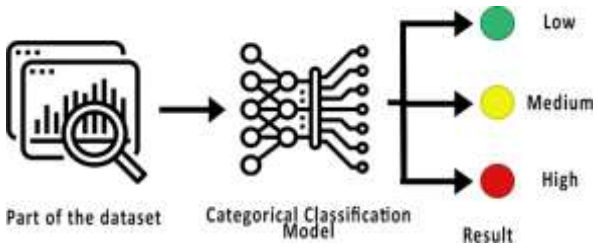

Fig. 3 Categorical Classification Model

3.5 Model Ensembling method

Model ensembling is an effective technique in machine learning that can enhance the predictive performance of individual models. The fundamental concept of ensembling is to amalgamate multiple models that are trained on the same dataset to create a more precise global model. These models are trained on diverse subsets of the data or using distinct algorithms, which enables them to learn varying aspects of the problem and create their own forecasts. These forecasts are then combined using different methods to produce a final prediction that surpasses any individual prediction. Ensembling has a major advantage of reducing overfitting, a situation where a model becomes too complex and closely fits the training data, resulting in a weak generalization ability when confronted with new data. The use of multiple models in ensembling permits each model to learn different patterns in the data, decreasing the likelihood of overfitting and advancing the generalization performance of the final model [14].

In our context, the approach to model ensembling involves training the first two artificial neural networks (ANNs) of each hospital on the first two parts of the datasets for each disease. The third ANN then learns from the combined knowledge of the first two models, leading to a significant improvement in the overall accuracy of the ensemble model. After the first two models are trained, their saved files are transmitted via blockchain to the third hospital, which is responsible for combining the knowledge of the two models. To achieve this, the weights of each model are extracted using the get_weights() method and stored in separate variables. The weights are then combined by iterating through each set of weights and performing element-wise addition and division. A list comprehension is used to apply the operation (w1 + w2) / 2.0 to each pair of weights returned by the zip() function. The resulting combined_w variable contains a list of combined weights that can be used to initialize the third neural network model.

The use of blockchain technology in the transfer of the saved model files provides a secure and transparent way of exchanging information between different hospitals. By combining ensemble learning with blockchain technology, healthcare industries can effectively improve the accuracy and robustness of prediction models while maintaining sensitive data's security and privacy.

3.6 Ethereum Private Blockchain

3.6.1 Blockchain

Blockchain technology has been a revolutionary innovation that has transformed the way data is stored, processed, and shared across various industries. The fundamental concept of blockchain is based on its decentralized and distributed digital ledger, which is managed by a network of computers or nodes, rather than a single central authority. This technology is designed to provide a high level of security, transparency, and tamper-proof data management that is suitable for applications requiring high trust and security. The core principle of blockchain technology is that each block in the chain contains a unique cryptographic signature that links it to the previous block in the chain, creating an unbroken and tamper-proof chain of records [15]. This unique

feature makes it impossible for any party to modify or manipulate the data stored in the blockchain, thus ensuring the integrity and security of the data. The decentralized nature of blockchain technology provides several benefits compared to traditional databases that are managed by a central authority. With blockchain technology, the network of computers or nodes that manages the blockchain is spread across the globe, making it almost impossible for any single party to control or manipulate the data stored in the blockchain. Furthermore, the decentralized nature of blockchain technology provides a higher level of security, as there is no single point of failure that can compromise the entire system. Another advantage of blockchain technology is its transparency. All transactions or data added to the blockchain are visible to all participants in the network, creating a high level of transparency and accountability. This feature ensures that any attempts to tamper with or manipulate the data can be easily detected and traced back to the source. The potential applications of blockchain technology are virtually limitless [16], and it has already found its use in various industries. In the healthcare industry, blockchain technology is being used to create more secure and transparent systems for managing patient data. By utilizing blockchain technology, healthcare providers can securely share and access patient data across different healthcare providers without compromising the patient's privacy. In the supply chain industry, blockchain technology is being used to create more transparent and efficient supply chains that reduce fraud and increase efficiency [17].

3.6.2 Ethereum Blockchain

The Ethereum blockchain technology is a game-changing innovation that has elevated the concept of blockchain to a whole new level. It is a decentralized and distributed digital ledger technology that has attracted widespread attention due to its adaptability and versatility. Ethereum is founded on the same technology as Bitcoin but with some crucial differences that set it apart from its predecessor. One of the most significant differences between Ethereum and Bitcoin is its use of a Turing-complete programming language. This means that developers can use Ethereum to create complex and sophisticated decentralized applications (dApps) that can perform a diverse range of functions. This contrasts with Bitcoin, which was primarily designed as a digital currency and has limited functionality.

Solidity, the Turing-complete programming language used by Ethereum, is a powerful tool that enables developers to write smart contracts. Smart contracts are self-executing contracts that are programmed to execute automatically when certain conditions are met. They are a critical aspect of the Ethereum ecosystem, and they have a limitless range of potential applications in various sectors, such as finance, real estate, supply chain management, and many more [18].

3.6.3 IPFS

IPFS (InterPlanetary File System) is a cutting-edge technology that aims to revolutionize how files are stored and shared online. IPFS is designed as a decentralized protocol and network that enables files to be stored and retrieved in a distributed manner, removing the need for a centralized server. This ensures that files are not stored in a single location, making it more robust against failures, censorship, and other security risks. The IPFS architecture is based on a peer-to-peer network, where files are split into smaller pieces and distributed across a network of nodes. This ensures that users can retrieve files from multiple nodes, resulting in faster and more efficient file transfers. Additionally, IPFS utilizes a content-addressed system where each file is identified by a unique hash, enabling the file to be located and retrieved from any node on the network. This ensures that files are easily accessed and shared across different platforms and devices. IPFS has already garnered a significant following in the tech community and has been utilized in various applications, such as decentralized social networks, distributed file storage systems, and blockchain-based applications. As the internet continues to evolve, IPFS has the potential to become a crucial infrastructure component that enables a genuinely decentralized and open web [19].

3.6.4 Our Implementation

First, hospital 1 and hospital 2 upload their models into IPFS, by defining an array called ”fileUploads” that contains an object representing the file that will be uploaded. This object includes the file path and its contents, which are read and encoded in base64 format using the ”fs” library. We defined an asynchronous function called ”uploadToIpfs” that uploads the file to IPFS. The function takes the fileUploads array as a parameter and uses it to specify the ABI (Application Binary Interface) for the uploaded file. After that, we called the uploadToIpfs function. The output is a hash of the location of the model file on IPFS.

Now that we have the link where the model is stored on IPFS we can send it through Blockchain to the third hospital. The owner of the model creates a smart contract using Solidity that manages a link storage system

with three private variables: link, owner, and authorizedNode, of type string, address, and address, respectively. The link variable stores the link, owner is the creator of the contract, and authorizedNode is the address of the node authorized to retrieve the link. The constructor initializes the owner variable to be the address of the user who deployed the contract. A setLink function allows the owner to set the link value. An onlyOwner modifier is used to restrict access to the function to only the contract owner. A getLink function allows the authorizedNode to retrieve the link value. A require statement checks if the sender of the transaction is equal to the authorizedNode. If the condition is not met, the function will not execute, and an error message will be displayed. A setAuthorizedNode function allows the owner to set the authorizedNode variable to a new address. The onlyOwner modifier is used to restrict access to the function to only the contract owner. The modifier onlyOwner is used to check if the caller is the owner of the contract. It uses the require statement to check if the sender of the transaction is equal to the owner. If the condition is true, the function will continue its execution and the rest of the code will be executed. If the condition is not met, an error message will be displayed, and the function will not execute. Figure 4 and Figure 5 are overviews of the system.

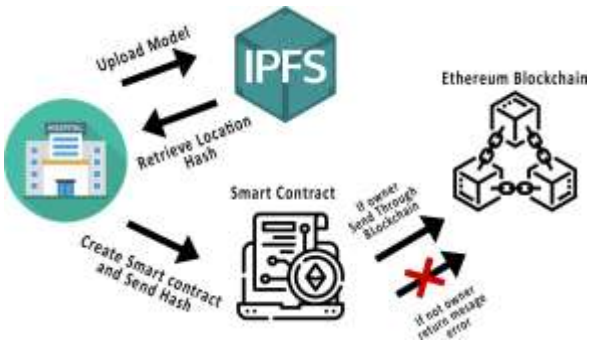

Fig. 4 Case of sending the link of the model location

In the end, the third hospital creates a multi-diagnosis application using the global models it created and shares it back in the same manner with hospital 1 and hospital 2 through blockchain.

4 Complexity Calculation

This section provides a comparison between the complexity of the ANN models (between ANN1/ANN2 and ANN3).

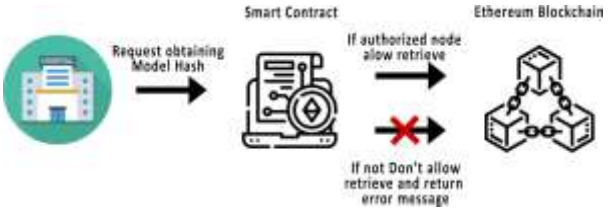

Fig. 5 Case of retrieving the link of the model location

4.1 Binary classification

4.1.1 ANN1/ANN2

The time complexity of model creation is O(1) because it is a simple assignment statement that takes constant time to execute, regardless of the input size.

The time complexity of the first layer is O(num_features * 30) because it involves multiplying the number of input features by the number of neurons in the layer. The space complexity is O(num_features * 30 + 30) because it creates a weight matrix with dimensions number_features x 30 and a bias vector with dimensions 1 x 30.

The time complexity of the Batch Normalization and Dropout is O(1) because they involve simple computations that take constant time. The space complexity of each line is O(30) because they add 30 values to the model's state.

The time complexity of the second layer is $O(30^2)$ because it involves multiplying the number of neurons in the previous layer by the number of neurons in the current layer. The space complexity is $O(30^2 + 30)$ because it creates a weight matrix with dimensions 30 x 30 and a bias vector with dimensions 1 x 30.

The time complexity and space complexity of Batch Normalization and Dropout are the same as described above.

The time complexity of the output layer is O(30) because it involves multiplying the number of neurons in the previous layer by 1. The space complexity is O(30 + 1) because it creates a weight matrix with dimensions 30 x 1 and a bias vector with dimensions 1 x 1.

The time complexity of the compilation of the model is O(1) because it involves setting values of certain parameters and takes constant time to execute.

The time complexity and space complexity of the Early Stopping callback are both O(1) because it involves creating a new object and setting its parameters.

The time complexity of the model training is O(100 * N) because it involves training the model for 100 epochs, where N is the total number of samples. The

space complexity is O(1) because it only stores the training history object.

- The total time complexity of the code is:

O(number_features * 30 + 30^2 + 50 * N)

- The total space complexity of the code is:

O(number_features * 30 + 30^2 + 30 + 1)

4.1.2 ANN3

The creation of the model, the first 2 layers, the Batch Normalization, the Dropout, and the output layer have the same complexity as ANN1/ANN2.

Extracting the weights has a time complexity of O(1) because it simply returns a reference to an already allocated memory location where the weights are stored. The space complexity is O(total number of weights) because it creates a new list to store the weights (for weights1 and weights2).

The time complexity of the creation of the combined weights that will be used in the model ensembling is O(total number of weights) because it has to iterate through both weight lists and perform a calculation on each pair of corresponding weights. The space complexity is also O(total number of weights) because it creates a new list to store the combined weights.

The model compilation complexity is the same as ANN1/ANN2.

The time complexity of setting the combined weights to the model is O(total number of weights) because it has to iterate through the weights and biases of the model and set each one to its corresponding value in the combined weights list. The space complexity is also O(total number of weights) because it does not allocate any additional memory.

The Early Stopping callback and the training complexities are the same as ANN1/ANN2.

- The total time complexity of the code is:

O(number_features * 30 + 30 + 30^2 + total number of weights + 100 * N)

- The total space complexity of the code is:

O(number_features * 30 + 30^2 + 30 + 31 + total number of weights + 1)

4.2 categorical classification

4.2.1 ANN1/ANN2

the creation of the model, the first 2 layers, the Batch Normalization, and the Dropout have the same complexity as the binary classification model.

The time complexity for the output layer is O(30 * 3) because each unit in the previous layer connects to every unit in this layer, and the space complexity is O(30 * 3 + 3) because there are 30 weights and 3 biases.

The time complexity of the compilation and Early Stopping Callback are the same as the binary classification model.

The time complexity of the training is proportional to the number of epochs multiplied by the number of training samples (N). In this case, the time complexity is O(37 * N). The space complexity is O(1) because it only stores the training history object.

- The total time complexity of the code is:

O(number_features * 30 + 30^2 + 30 * 3 + 37 * N)

- The total space complexity is:

O(number_features * 30 + 30^2 + 30 * 2 + 30 * 3 + 3 + 1).

4.2.2 ANN3

The creation of the model, the first 2 layers, the Batch Normalization, the Dropout, and the output layer have the same complexity as ANN1/ANN2.

Extracting the weights has a time complexity of O(1) because it simply returns a reference to an already allocated memory location where the weights are stored. The space complexity is O(total number of weights) because it creates a new list to store the weights (for weights1 and weights2).

The time complexity of the Creation of the combined weights that will be used in the model ensembling is O(total number of weights) because it has to iterate through both weight lists and perform a calculation on each pair of corresponding weights. The space complexity is also O(total number of weights) because it creates a new list to store the combined weights.

The model compilation complexity is the same as ANN1/ANN2.

The time complexity of setting the combined weights to the model is O(total number of weights) because it has to iterate through the weights and biases of the model and set each one to its corresponding value in the combined weights list. The space complexity is also O(total number of weights) because it doesn't allocate any additional memory.

The Early Stopping callback and the training complexities are the same as ANN1/ANN2.

Time Complexity: O(number_features * 30 + 30^2 + 30 * 3 + total number of weights + 37 * N)

Space Complexity: O(number_features * 30 + 30^2 + 30 * 3 + total number of weights + 30 * 3 + 3 + 1)

5 Results

The results shown by each model are represented in Tables 2, 3, and 4, and Figures 6 to 17.

5.1 Breast Cancer

Table 2 represents the results obtained by the three ANN models trained using the breast cancer dataset in terms of Accuracy, Precision, Recall, and F1-score. Figures 6, 7, and 8 represent graphic curves of the Training vs Validation Accuracy and loss for each ANN. Figure 9 represents a comparison between the results obtained from each model. Figure 10 represents a confusion matrix of the performance of the model that made predictions on a set of 39 instances.

The matrix has two rows and two columns. The first row corresponds to the instances that belong to the Malignant class and the second row corresponds to the instances that belong to the Benign class. The first column represents the instances that were predicted as positive by the model and the second column represents the instances that were predicted as negative.

In this case, the confusion matrix shows that: There are 22 instances that truly belong to the Malignant class and the model correctly predicted all of them (i.e., true Malignant). There are 16 instances that truly belong to the Benign class and the model correctly predicted all of them (i.e., true Benign). There is 1 instance that truly belongs to the Malignant class but the model predicted it as Benign (i.e., false Benign). There are no instances that truly belong to the Benign class but the model predicted them as Malignant (i.e., false Malignant).

Table 2 Breast Cancer Results

	Accuracy	Precision	Recall	F1-score
ANN 1	94.74%	100%	90%	95%
ANN 2	94.73%	100%	86%	92%
ANN 3	97.44%	100%	94%	97%

5.2 Lung Cancer

Table 3 represents the results obtained by the three ANN models trained using the Lung cancer dataset in terms of Accuracy, Precision, Recall, and F1-score. Figures 11, 12, and 13 represent graphic curves of the Training vs Validation Accuracy and loss for each ANN. Figure 14 represents a comparison between the results

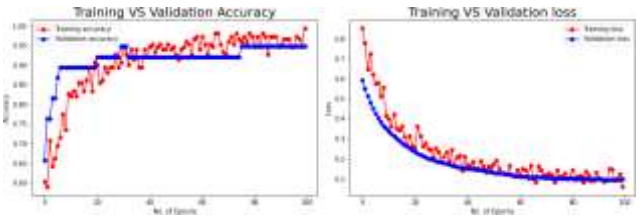

Fig. 6 Training vs Validation Accuracy and Loss of ANN 1

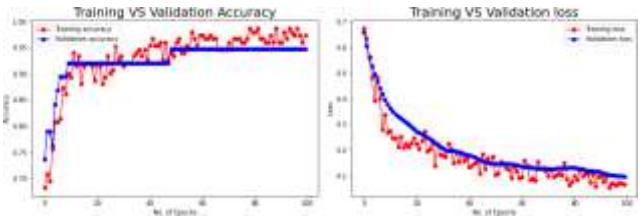

Fig. 7 Training vs Validation Accuracy and Loss of ANN 2

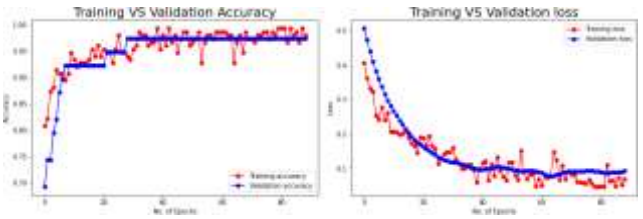

Fig. 8 Training vs Validation Accuracy and Loss of ANN 3

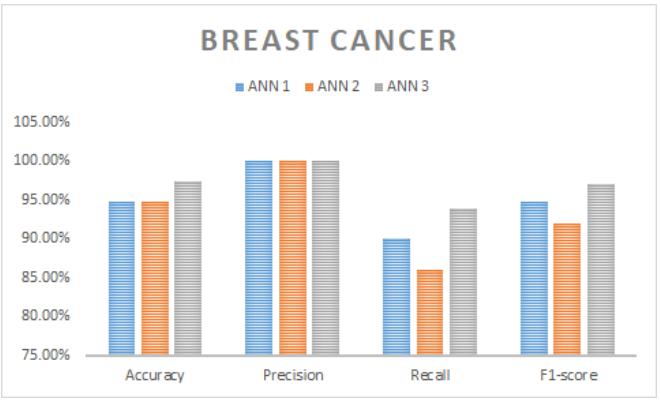

Fig. 9 Comparison between the 3 Models

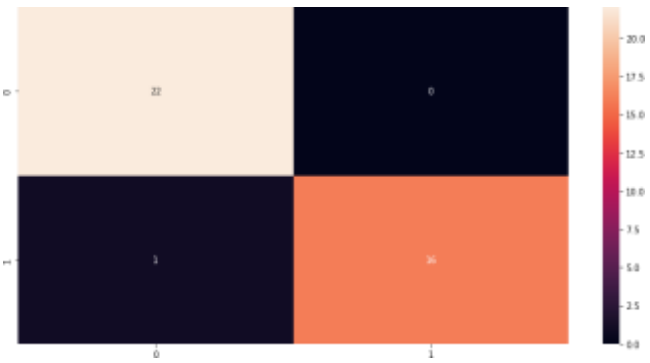

Fig. 10 Confusion Matrix of the third ANN

obtained from each model. Figure 15 represents a confusion matrix of the performance of model that made predictions on a set of 67 instances.

The matrix has three rows and three columns. The first row corresponds to the instances that belong to the ”low” class, the second row corresponds to the instances that belong to the ”medium” class, and the third row corresponds to the instances that belong to the ”high” class. The first column represents the instances that were predicted as ”low” by the model, the second column represents the instances that were predicted as ”medium”, and the third column represents the instances that were predicted as ”high”.

In this case, the confusion matrix shows that:

There are 16 instances that truly belong to the ”low” class, and the model correctly predicted all of them (i.e., true positives for the ”low” class). There are 23 instances that truly belong to the ”medium” class, and the model correctly predicted all of them (i.e., true positives for the ”medium” class). There are 27 instances that truly belong to the ”high” class, and the model correctly predicted all of them (i.e., true positives for the ”high” class). There is 1 instance that truly belongs to the ”medium” class, but the model predicted it as ”low” (i.e., false negative for the ”medium” class). There are no instances that truly belong to the ”low” or ”high” classes but were predicted as ”medium” or ”low/high” (i.e., false positives).

Table 3 Lung Cancer Results

	Accuracy	Precision	Recall	F1-score
ANN 1	95.52%	90%	100%	95%
ANN 2	92.54%	91%	100%	95%
ANN 3	98.51%	94%	100%	97%

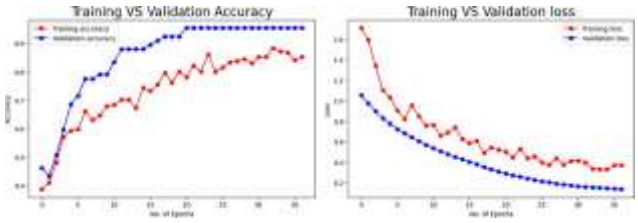

Fig. 11 Training vs Validation Accuracy and Loss of ANN 1

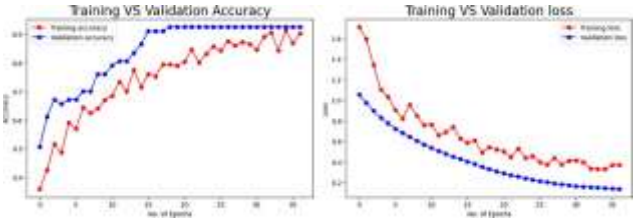

Fig. 12 Training vs Validation Accuracy and Loss of ANN 2

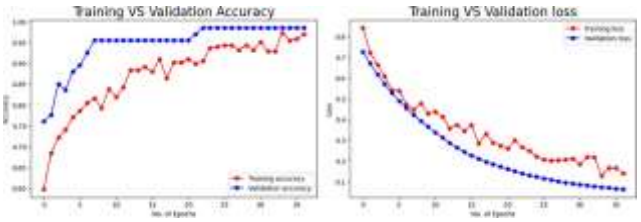

Fig. 13 Training vs Validation Accuracy and Loss of ANN 3

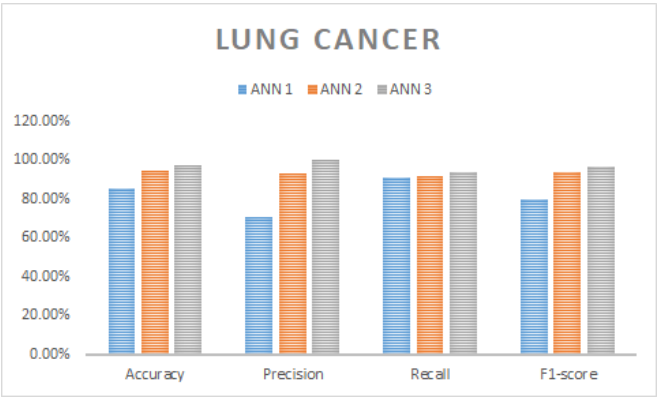

Fig. 14 Comparison between the 3 Models

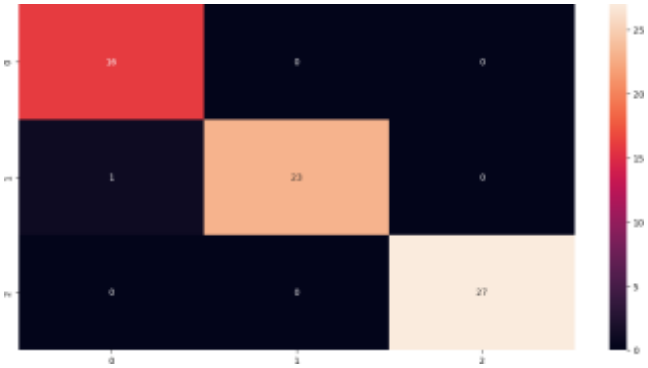

Fig. 15 Confusion Matrix of the third ANN

5.3 Diabetes

Table 4 represents the results obtained by the three ANN models trained using the Diabetes dataset in terms of Accuracy, Precision, Recall, and F1-score. Figures 16, 17, and 18 represent graphic curves of the Training vs Validation Accuracy and loss for each ANN. Figure 19 represents a comparison between the results obtained from each model. Figure 20 represents a confusion matrix of the performance the model that made predictions on a set of 35 instances.

The matrix has two rows and two columns. The first row corresponds to the instances that belong to the positive class and the second row corresponds to the instances that belong to the negative class. The first column represents the instances that were predicted as positive by the model and the second column represents the instances that were predicted as negative.

In this case, the confusion matrix shows that: There are 16 instances that truly belong to the positive class, and the model correctly predicted all of them (i.e., true positives). There are 18 instances that truly belong to the negative class, and the model correctly predicted all of them (i.e., true negatives). There is 1 instance that truly belongs to the positive class, but the model predicted it as negative (i.e., false negative). There are no instances that truly belong to the negative class but were predicted as positive (i.e., false positives).

Table 4 Diabetes Results

	Accuracy	Precision	Recall	F1-score
ANN 1	85.71%	71%	91%	80%
ANN 2	94.73%	93%	92%	94%
ANN 3	97.14%	100%	94%	97%

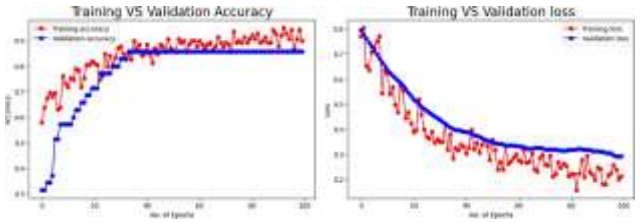

Fig. 16 Training vs Validation Accuracy and Loss of ANN 1

The comparison of the global models created by the third hospital against the local models created by hospitals 1 and 2 reveals that our methodology is indeed valuable. The results show that the global models significantly outperformed all of the local models in terms

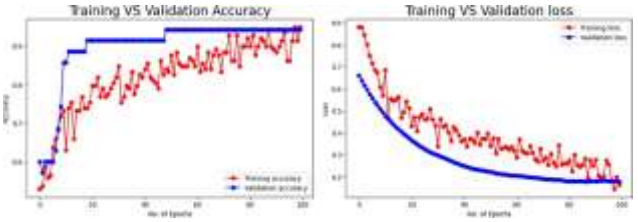

Fig. 17 Training vs Validation Accuracy and Loss of ANN 2

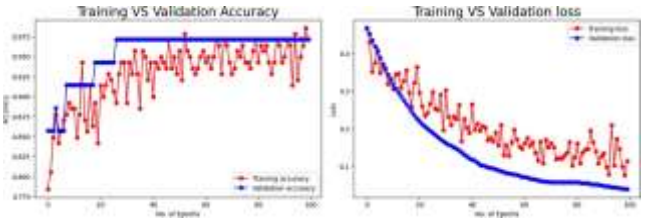

Fig. 18 Training vs Validation Accuracy and Loss of ANN 3

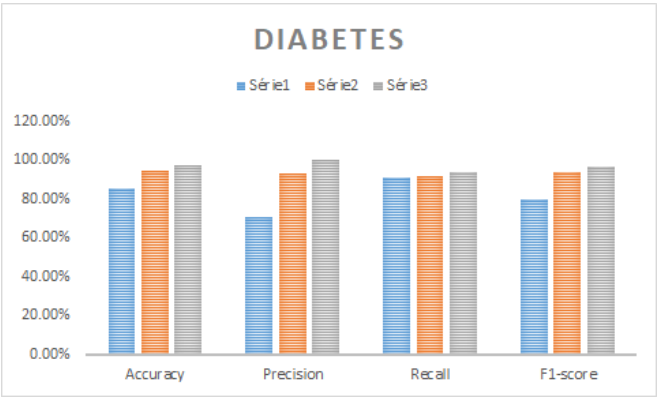

Fig. 19 Comparison between the 3 Models

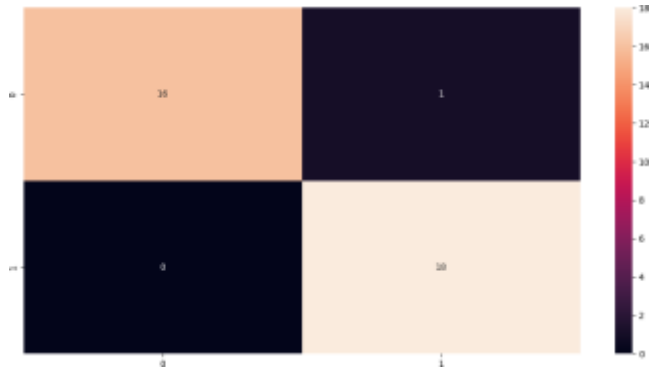

Fig. 20 Confusion Matrix of the third ANN

of accuracy, precision, recall, and F1-score for each disease. This indicates that our approach of combining models from multiple sources and using them to create a single global model has led to a significant improvement in the accuracy and efficacy of disease diagnosis.

The results also suggest that hospitals can benefit significantly by adopting our methodology of creating global models. This would enable hospitals to leverage the collective knowledge and expertise of multiple institutions and achieve higher accuracy and efficacy in disease diagnosis. Furthermore, the adoption of such a methodology can help to reduce the cost and time required for data collection and analysis, while improving the quality and accuracy of the diagnosis and maintaining the privacy of the patient data.

6 Application Overview

After the third hospital finishes training the global models it creates the multi-diagnosis application based on these models. Another important note that the scalers that were fitted on the dataset should also be saved so that they can fit to new data for prediction.

The application is created using the Streamlit library, which allows for simple and easy creation of interactive data applications. The code loads pre-trained machine learning models and scalers for each disease type. The application allows users to select which disease they would like to predict from a select box and then provides a form for users to input the values for each feature relevant to that disease. Once the user submits the form, the input data is preprocessed using the corresponding scaler, and the preprocessed data is then fed into the pre-trained ANN model to predict the likelihood of the disease. Finally, the application displays the prediction output to the user. Figures 21, 22, 23 and 24 represent an overview of the application interfaces and features.

Figure 21 shows how the selection of the disease can be done.

Fig. 21 Selection of the disease

Figure 22 shows the interface of the selected disease with the corresponding form.

Fig. 22 Interface of the selected disease

Figure 23 shows the interface after inputting the symptoms of the patient that will be used in the prediction after clicking the predict button.

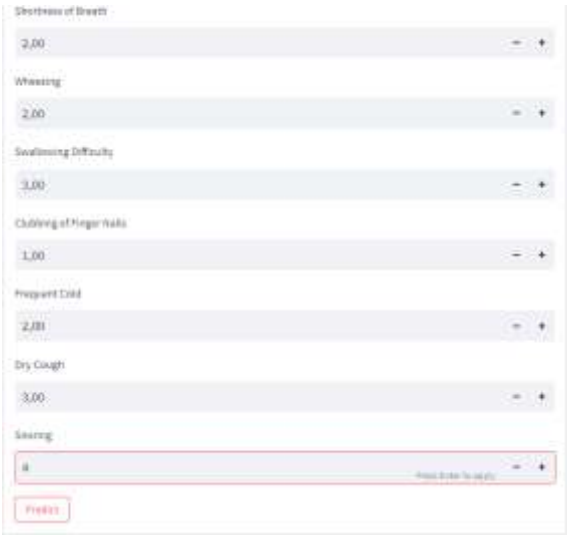

Fig. 23 Inputting data

After clicking the predict button the prediction result will be shown in the bottom of the interface as represented in Figure 24.

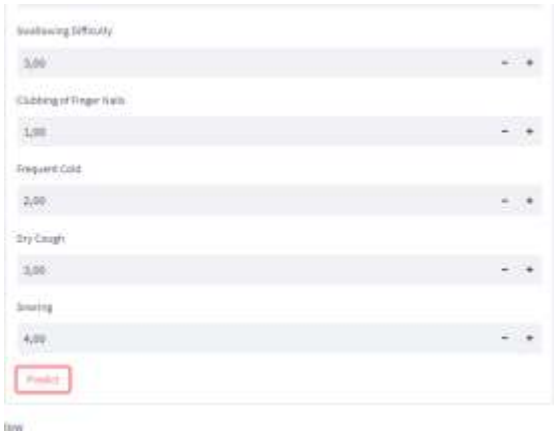

Fig. 24 Showing the result

7 Conclusion and future work

In this paper, we presented a Deep Learning application that can diagnose multiple diseases simultaneously. To construct this application, we utilized blockchain and IPFS technologies, which ensured secure communication between healthcare providers. This was essential to solve the sensitive healthcare data privacy issue. We employed the Model ensembling technique, which involved combining multiple models from different sources to create highly accurate final models. This technique resulted in models with excellent performances (97.44% accuracy for Breast cancer, 98.51% accuracy for Lung cancer, and 97.14% accuracy for Diabetes), outperforming the individual models obtained from each source. The final models were then used to develop the application. To validate our approach, we suggest conducting additional tests on sizable datasets in future research. Such tests would allow us to further evaluate the performance of the application and refine the models to ensure even higher accuracy rates.

In summary, our approach of utilizing blockchain and IPFS technologies for secure communication between healthcare providers and employing Model ensembling technique has shown promising results for the development of a Deep Learning application that can diagnose multiple diseases. This application has the potential to significantly improve healthcare outcomes and reduce the burden on healthcare providers. Further research is needed to continue refining our approach and to validate its efficacy.

8 Statements and Declarations

Competing interests: The authors have no competing interests to declare that are relevant to the content of this article.

9 Research Data Policy and Data Availability Statements

The datasets used in this study were obtained from publicly available sources. The breast cancer dataset is available on UCI Machine Learning Repository [7] and Kaggle [8], the Lung cancer dataset is available on data.world [9] and Kaggle [10], and the Diabetes dataset is available on UCI Machine Learning Repository [11] and Kaggle [12].

References

1. Huru Hasanova, Muhammad Tufail, Ui-Jun Baek, Jee-Tae Park, and Myung-Sup Kim. A novel blockchain-enabled heart disease prediction mechanism using machine learning. *Computers and Electrical Engineering*, 101:108086, 2022.
2. Misha Abraham, AH Vyshnavi, Chungath Srinivasan, and PK Namboori. Healthcare security using blockchain for pharmacogenomics. *Journal of International Pharmaceutical Research*, 46:529–533, 2019.
3. Randhir Kumar, Prabhat Kumar, Rakesh Tripathi, Govind P Gupta, AKM Najmul Islam, and Mohammad Shorfuzzaman. Permissioned blockchain and deep-learning for secure and efficient data sharing in industrial healthcare systems. *IEEE Transactions on Industrial Informatics*, 2022.
4. Eric Appiah Mantey, Conghua Zhou, Vinodhini Mani, John Kingsley Arthur, and Ebuka Ibeke. Maintaining privacy for a recommender system diagnosis using blockchain and deep learning. *Human-centric computing and information sciences*, 2022.
5. Tien-En Tan, Ayesha Anees, Cheng Chen, Shaohua Li, Xinxing Xu, Zengxiang Li, Zhe Xiao, Yechao Yang, Xiaofeng Lei, Marcus Ang, et al. Retinal photograph-based deep learning algorithms for myopia and a blockchain platform to facilitate artificial intelligence medical research: a retrospective multicohort study. *The Lancet Digital Health*, 3(5):e317–e329, 2021.
6. Hemang Subramanian, Susmitha Subramanian, et al. Improving diagnosis through digital pathology: Proof-of-concept implementation using smart contracts and decentralized file storage. *Journal of medical Internet research*, 24(3):e34207, 2022.
7. William H Wolberg, W Nick Street, and Olvi L Mangasarian. Breast cancer wisconsin (diagnostic) data set. *UCI Machine Learning Repository [http://archive. ics. uci. edu/ml/]*, 1992.
8. UCI Machine Learning Repository and Kaggle. Breast cancer wisconsin (diagnostic) data, 2017.
9. Cancer Data Health Program. Lung cancer data. https://data.world/cancerdatahp/lung-cancer-data, 2016.

10. Cancer Data Health Program. Lung cancer prediction: Air pollution, alcohol, smoking & risk of lung cancer. https://www.kaggle.com/datasets/thedevastator/cancer-patients-and-air-pollution-a-new-link, 2022.
11. Alimuddin Ahmed, Tanzila Ali, and Atiya Khanum. Early stage diabetes risk prediction dataset. https://archive.ics.uci.edu/ml/datasets/Early+stage+diabetes+risk+prediction+dataset, 2014.
12. Ahmad Alakaaay. Diabetes uci dataset. https://www.kaggle.com/datasets/alakaaay/diabetes-uci-dataset, 2020.
13. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
14. Jason Brownlee. *Ensemble learning algorithms with Python: Make better predictions with bagging, boosting, and stacking*. Machine Learning Mastery, 2021.
15. Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press, 2016.
16. Don Tapscott and Alex Tapscott. *Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world*. Penguin, 2016.
17. Michael Crosby, Prateek Pattanayak, Sanjeev Verma, and Vignesh Kalyanaraman. Blockchain technology: beyond bitcoin. *Applied Innovation*, 2(6-10):71–81, 2016.
18. Andreas M Antonopoulos and Gavin Wood. *Mastering ethereum: building smart contracts and dapps*. O'reilly Media, 2018.
19. Peter E Morrison. Ipfs: The interplanetary file system for decentralized web applications. *Journal of Web Engineering*, 20(3):245–265, 2021.