IRSTI 28.23.29 https://doi.org/10.26577/1i32jpcsit2308

N. Abdrazakuly*, L. Cherikbayeva

Al-Farabi Kazakh National university, Almaty, Kazakhstan
*e-mail: nurasil2242@gmail.com

CREATION OF AN EFFECTIVE IMAGE PROCESSING ALGORITHM BASED ON AN ENSEMBLE APPROACH

Abstract. According to the World Health Organization, more than 17 million people die annually in the world from diseases of the circulatory system, half of them die from coronary heart disease and cerebral stroke. Stroke is a structurally complex disease based on various pathogenetic mechanisms. Taking into account the multicomponent nature of this pathology, as well as its complex structure, the medical community has developed various evaluation algorithms based on the recognition of various symptoms. Determining the effectiveness of these algorithms is recognized as the most important. Incorrect symptoms appear as a result of inaccuracies made by the radiologist in the process of manual annotation of computed tomography images. A convolutional neural network is used to perform image classification in a collection of brain stroke data. Since the data set is small, training the entire neural network does not give good results, therefore, to obtain more accurate results, model training uses the concept of transfer learning. Transfer learning is a method in which a model for a specific task is used as a starting point for another task. In particular, the Inception V3 model with Imagenet scales is used for the current task. The developed neural network was developed using the Tensorflow library of the Python programming language. Using machine learning, a data set with computed tomographic images of 2,515 normal and stroke-damaged areas of the brain was obtained. The task of a neural network is to classify a given image, that is, to determine whether it is normal or damaged. Using this algorithm, accuracy increased from 65 percent to 99.2 percent, and losses decreased from 7,532 to 0.756 percent. Key indicators: accuracy 99.6%, recall 99.2%, F1-the price was 99.1%.

Key words: CT Images, CNN, Deep learning, Medical imaging, Stroke detection, ResNet-50, Segmentation, VGG-19.

1. Introduction

Advances in computer vision and machine learning have paved the way for significant improvements in image classification tasks. As these technologies continue to evolve, researchers seek innovative approaches to enhance the accuracy and reliability of such tasks. In this context, the concept of ensemble learning has emerged as a powerful strategy to capitalize on the strengths of multiple individual models, mitigating their weaknesses and ultimately achieving superior performance.

This study delves into the utilization of an ensemble comprising the VGG-19, CNN, and ResNet-50 neural network models to tackle a prominent challenge in computer vision: accurate image classification. The motivation behind creating this ensemble was to harness the distinctive qualities of each model, capitalizing on their varied capabilities to arrive at more precise and reliable image classification outcomes. Each neural network model—VGG-19, CNN, and ResNet-50 – possesses inherent strengths and limitations stemming from their architectural intricacies. The objective of this study was to judiciously exploit these attributes by synergistically integrating the models within an ensemble framework. By meticulously adjusting and fusing the individual model predictions, substantial enhancements in classification accuracy were achieved. The ensemble approach afforded diverse avenues for classifying images, thereby enabling more informed and accurate decision-making.

This paper provides an intricate exposition of the methodology deployed for the conception and utilization of the ensemble. Our methodology encompassed a sequence of pivotal stages: initial creation of discrete models for each individual neural network, meticulously tailored to conform to the distinct architectural blueprints of CNN, VGG-19, and ResNet-50. These models were subsequently fine-tuned, incorporating the distinctive training patterns of each network. Next, forecasts were generated by passing test data through each model, yielding a comprehensive set of predictions. The pivotal phase involved combining these predictions, a task accomplished through a voting mechanism.

The voting mechanism employed a simple majority principle to merge predictions from all three models. The class with the highest number of votes across the models was selected as the ensemble's consensus prediction. This approach effectively amalgamated the diverse insights provided by each individual model, thereby producing a more robust final prediction. However, it is crucial to acknowledge that while voting was employed in this study, alternate methods for combining predictions, such as mean or weighted voting, could also be explored based on the models' reliability and data characteristics. The employment of an ensemble approach necessitates additional resources and time for development and implementation. Nonetheless, the tangible benefits in terms of accuracy and reliability validate the investment. By leveraging the collective wisdom of multiple neural network models, ensembles present an influential strategy to bolster the efficacy of machine learning models. This approach effectively mitigates the impact of stochastic errors and embraces the multifaceted nature of the data.

This research aims to contribute to the discourse on ensemble learning's potential for refining accuracy and reliability in machine learning tasks. The outcomes corroborate the effectiveness of combining the VGG-19, CNN, and ResNet-50 models, surpassing the performance of each individual model and achieving remarkable classification accuracy.

2. Literature review

As outlined in reference [1], diverse machine learning approaches have the capability to forecast individuals at a heightened risk of experiencing a stroke. The study employs three machine learning models – RF, DT, and NB. Following an assessment of each technique, the prediction hinges on the patient's medical history as a feature within each model. Notably, the RF method showcases the highest accuracy, reaching 94.781%. It is trailed by the DT method with an accuracy of 91.906%, while the NB method demonstrates the lowest accuracy at 89.976%. The findings underscore the RF method's superior accuracy compared to the other approaches. In the context of [2], the authors explore the effectiveness of multiple ML algorithms – RF, LR, K-NNs, SGD, DT, stacking, and majority voting – by utilizing various attributes from participant profiles to identify the optimal algorithm for predicting stroke cases. The stacking classification approach yields highly favorable outcomes exceeding 98%. Consequently, the stacking technique emerges as a proficient method in identifying individuals with an elevated likelihood of experiencing a stroke.

The researchers in [3] developed and trained four distinct models using machine learning techniques and various physiological parameters to accurately predict strokes. Among these models, the Random Forest (RF) demonstrated the highest performance with an accuracy of around 96%. As described in [4], the researchers employed two datasets for their study. The initial dataset consisted of medical evaluations, environmental changes, and bodily changes. To balance the dataset, the SMOTE method was applied, missing values were replaced using KNN Imputer values, and outliers were eliminated. Additionally, diabetes and obesity features were generated based on corresponding feature values. The final step involved employing five machine learning algorithms – SVM, KNN, Decision Tree (DT), Random Forest (RF), and Multi-Layer Perceptron (MLP) – to analyze these features. The second dataset comprised MRI scans, and to enhance the images and reduce noise, an average filter was utilized.

The dataset was made more balanced through the application of data augmentation techniques in order to prevent the data from being overly tailored to the training set. The training and validation phases utilized approximately 80% of the dataset, while the remaining 20% was reserved for testing purposes. Deep features were extracted from the data using both the AlexNet model and the SVM algorithm. Surprisingly, the deep learning model, specifically AlexNet, exhibited poorer performance compared to a hybrid algorithm that combined deep learning and traditional machine learning approaches. The AlexNet and SVM models achieved exceptional results, with accuracy rates of 99.9%, sensitivity of 100%, specificity of 99.80%, and an AUC of 99.86%.

In a study cited as [5], researchers demonstrated the capacity of different machine learning models to effectively forecast occurrences of stroke using physiological indicators. Among the alternate algorithms, the Naive Bayes (NB) Classification technique exhibited superior performance, achieving an accuracy level of 82%.

As outlined in reference [6], the authors put forth a comprehensive strategy for evaluating patient characteristics within the digital health record. They conducted a systematic analysis to investigate diverse attributes. Employing feature correlation and stepwise analysis, they identified the most suitable features. Additionally, a principal component

analysis (PCA) was executed, revealing that principal components were essential for explaining a greater amount of variance. Subsequently, a combination of different features and principal component arrangements were employed to evaluate three machine-learning algorithms: neural network (NN), decision tree (DT), and random forest (RF). Their findings indicated that for the NN, the optimal feature combination included A, HD, HT, and AG, resulting in an accuracy of 78% and a miss rate of 19%.

In reference [7], the authors employed a stroke prediction algorithm along with an enhanced RF ensemble technique to identify risk factors. The predictive model displayed an error rate of 0.03% and achieved an accuracy of 96.97%.

In reference [8], a hybrid framework for predicting cerebral stroke disease was proposed by the authors, combining classification and clustering methodologies. They utilized an improved hierarchical clustering technique for clustering and subsequently implemented several classifiers: logistic regression (LR), RF, support vector machine (SVM), NN, and XGBoost. All classifiers yielded satisfactory results in terms of accuracy and area under the curve (AUC), with the RF classifier standing out at 97% accuracy, representing the highest performance. The literature analysis of recent studies indicates that there has been no precise classification of cerebral stroke achieved so far. Nonetheless, our suggested fine-tuned ensemble RXLM exhibited notable classification rates: 95.29%, 99.13%, 96.38%, 94.36%, 95.35%, 90.59%, and 90.63% for different parameters.

3. Source

The exploration and refinement of optimal ensemble algorithms for image processing were undertaken by the individual. Diverse techniques were employed to ascertain the most efficacious combinations of algorithms and ascertain the optimal parameters for their setup. This meticulous approach culminated in the achievement of superior outcomes.

The best image processing algorithms were chosen and their parameters were adjusted to achieve maximum processing accuracy as part of my work. Machine learning techniques were employed to aid in the identification of optimal ensemble algorithms, which were then tested on specific images to assess their effectiveness. The resulting dataset comprises a collection of data generated during the testing using computed tomography, acquired as an outcome of the diagnostic procedures conducted at the National Research Hospital of the Republic of Bangladesh. Within this dataset, a total of 2501 distinct brain tomography images were acquired. These images were subsequently partitioned into separate sets for training, testing, and validation (refer to Figure 1). Each of these sets encompasses both brain images unaffected by stroke and brain images indicative of stroke presence.

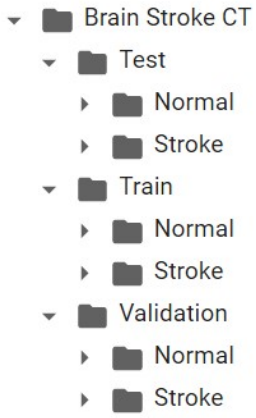

Figure 1 – Dataset hierarchy

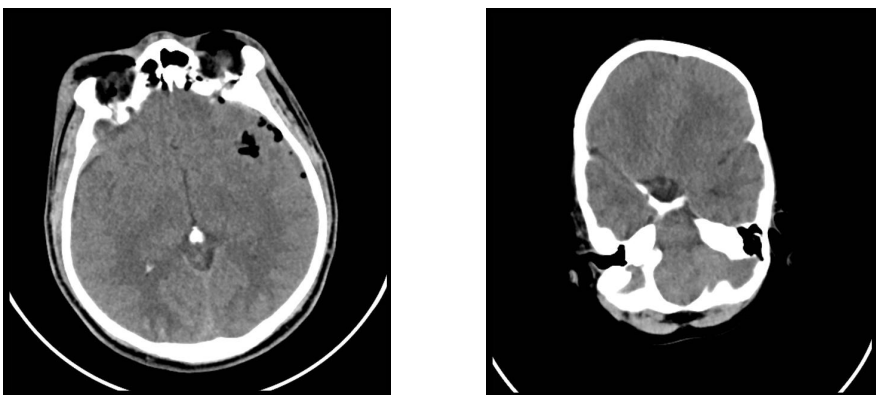

Figure 2 – Computed tomography images of a healthy and stroke-affected brain

Dataset: computed tomography images. Data size: dataset consists of 2501 images. Of these, 1,551 are healthy brains and 950 are brain images that have suffered a stroke (Figure 2). The source weighs 70 MB. Data type: all images presented in .jpg format. Target variable: the target variable is represented by classes of 2 types of brain image: healthy brain images, stroke brain images. Data preprocessing: in this dataset, data preprocessing is not necessary, as not all images have both pictorial and missing values.

Division into training, verification and test samples: the dataset was divided into a training sample (74% of the total), a verification sample (17%) and a test sample (9%) (Figure 3). Data issues: no data issues such as missing values or emissions were found in the data set. Link to dataset: https://www.kaggle.com/datasets/afridirahman/brain-stroke-ct-image-dataset. Dataset standard: the dataset is designed according to the international DICOM standard. Digital imaging and communication in medicine (DICOM) is an international standard relating to the sharing, storage and transmission of digital medical images. Before the advent of this format, there was no standardized method for transferring medical images. While 16-bit DICOM images (with values in the range -32768 and 32767), other 8-bit images in a gray image store values from 0 to 255.

The DICOM standard was developed by the National Association of electronic equipment manufacturers (National Electrical Manufacturers Association). The standard covers the functions of creating, storing, transmitting and printing individual image frames, series of frames, Patient Information, research, equipment, facilities, medical personnel conducting the examination and the like.

Two Information levels have been defined by the DICOM standard:

- file level-DICOM File (DICOM file) – an object file with a tag Organization for displaying an image frame (or series of frames) and accompanying or control information;
- network (communication)-DICOM Network Protocols (network DICOM — protocol)-for transferring DICOM files and DICOM control commands over networks that support TCP/IP.

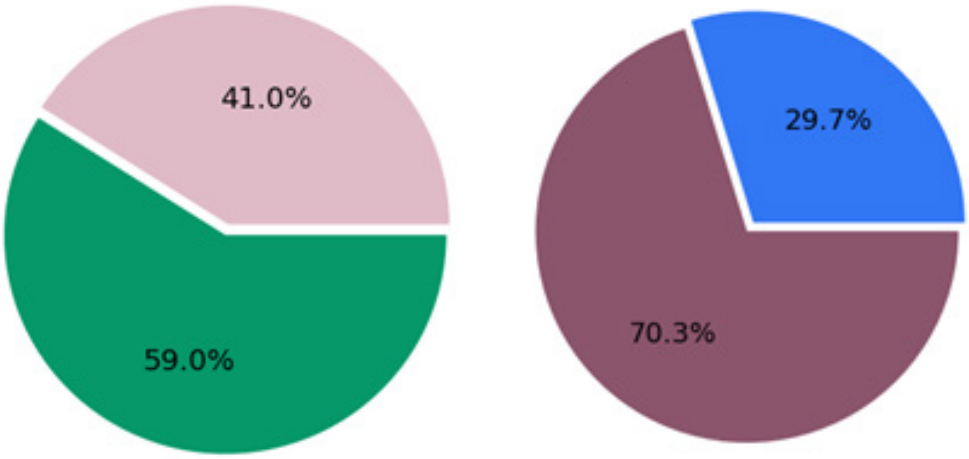

Figure 3 – Image ratio in training and testing packages

4. Diagnosis of stroke by computed tomography

Computed tomography with a stroke allows you to distinguish bleeding (hemorrhagic stroke) from a heart attack and start proper treatment in a timely manner, which allows you to limit damage and prevent the development of complications [9].

The symptom of increased arterial density is an early indirect sign of ischemic stroke. In this case, acute occlusion of the cerebral artery manifests itself in the form of an increase in its image density.

Numerous authors attribute the diminishment of differentiation between gray and white matter, alongside the smoothing of cortical gyri, to the preliminary indicators of hemispherical ischemic stroke. In cases of cerebral hemorrhage, a characteristic image manifests on CT scans, characterized by the presence of an intensified

density region within the cerebral tissue, appearing as a whitish spot. Intracerebral hematomas stemming from stroke tend to reside in the inner recesses of cerebral tissue, while traumatic hematomas are more commonly situated along the periphery (refer to figure 4). Within cerebral hemorrhages, the evaluation of the extent to which hemorrhagic elements disperse into the ventricular lumen and subarachnoid spaces is optimally achieved (as depicted in figure 5).

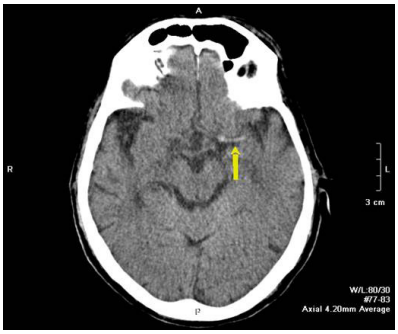

Figure 4 – Symptom of increased left middle artery density

Figure 5 – Decreased gray and white differentiation flatness of the left cortical gyrus cerebral hemisphere

Within this project, the utilization of a convolutional neural network (CNN) was pivotal for image classification within the Brain Stroke dataset. Nonetheless, owing to the dataset's limited scale, training the complete neural network failed to yield desirable outcomes. Consequently, a remedy was sought through the implementation of transfer learning—an approach that capitalizes on pre-existing knowledge—to fine-tune the model's performance and attain heightened levels of accuracy.

Trained on a diverse range of images within the ImageNet dataset, this model is especially suitable for our intended objectives. Leveraging this pre-trained model and adapting it to align with the distinct characteristics of the Brain Stroke dataset led to substantial improvements in accuracy. This stands in contrast to the outcomes attainable through the endeavor of training a new model from scratch.

By employing transfer learning and initializing with the InceptionV3 model pretrained on the ImageNet dataset, a high-precision model was successfully trained for image classification within the Brain Stroke dataset. This strategy not only conserved time and computational resources but also yielded more accurate results compared to alternative approaches. Within the realm of deep learning, convolutional neural networks (CNNs) constitute a category of artificial neural networks tailored for visual image analysis. Diverging from customary matrix multiplications, CNNs incorporate a convolution operation within at least one of their layers (as depicted in Figure 6). Their design optimally caters to pixel data processing, rendering them indispensable for image recognition and manipulation tasks. Their versatility spans domains such as image recognition, recommendation systems, image classification, image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and the analysis of financial time series data [10].

Stroke classification using convolutional neural networks is a common application of deep learning in the analysis of medical images (Figure 7). CNN is a type of neural network that is particularly effective at detecting patterns in images, making them ideal for tasks such as detection.

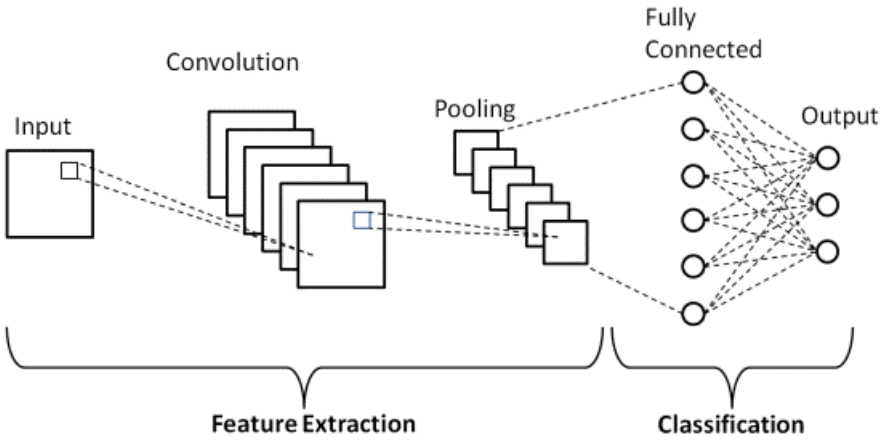

Figure 6 – Convolutional neural network architecture

Figure 7 – CNN neural network training progress

5. VGG neural network

The full name of VGG is a visual geometry group belonging to the Department of Science and engineering at the University of Oxford. The company has produced a series of convolutional network models ranging from VGG16 to VGG19 that can be used for face recognition and image classification. The original purpose of VGG convolutional network depth research is to understand how the depth of convolutional networks affects the accuracy of classification and recognition of large-scale images. – Deep-16 CNN uses a small 3x3 convolution core of all layers to increase the number of network layers and avoid too many settings [11].

The VGG input is set to a 224X244 RGB image. The average RGB is calculated for all images in a set of images, and then the image is entered as input into the VGG convolutional network. A 3x3 or 1x1 filter is used and the stacking step is fixed. . There are 3 fully connected VGG layers, which can range from VGG11 to VGG19, depending on the total number of convolutional layers + fully connected layers. The minimal VGG11 has 8 convolutional layers and 3 fully connected layers. The maximum VGG19 has 16 convolutional layers. + 3 fully connected layers (Figure 8). In addition, after the VGG network, there is no unifying layer after each convolutional layer, or a total of 5 unifying layers distributed under different convolutional layers [12].

The VGG16 contains 16 floors and the VGG19 contains 19 floors. The VGG series is exactly the same in the last three fully integrated layers. The overall structure includes a set of 5 convolutional layers and then MaxPool.

VGGNet has 2 to 4 convoluted operations at each convolution level. Convolution core size 3x3, convolution step size 1, join nucleus 2x2, and step size 2(Figure 9). The most obvious improvement of VGGNet is to reduce the size of the convolution core and increase the number of convolution layers (Fig. 10).

During the training of the VGG neuron network (Figure 11), the following was done:

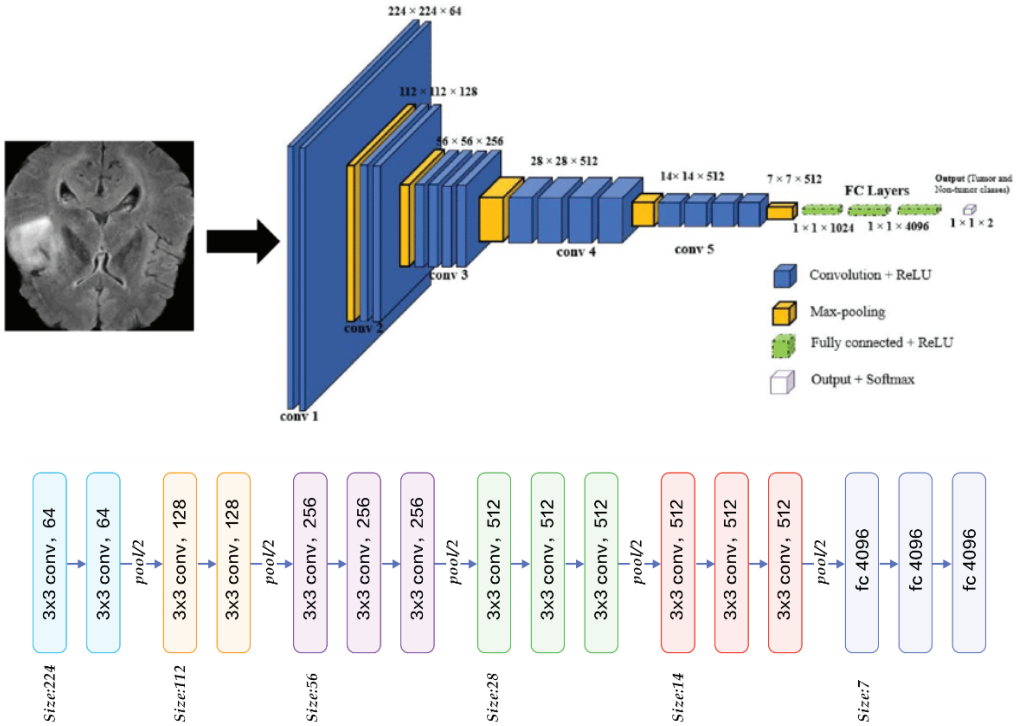

Figure 8 – VGG – 19 neural network architecture

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

Figure 9 – VGG – 19 neural network architecture

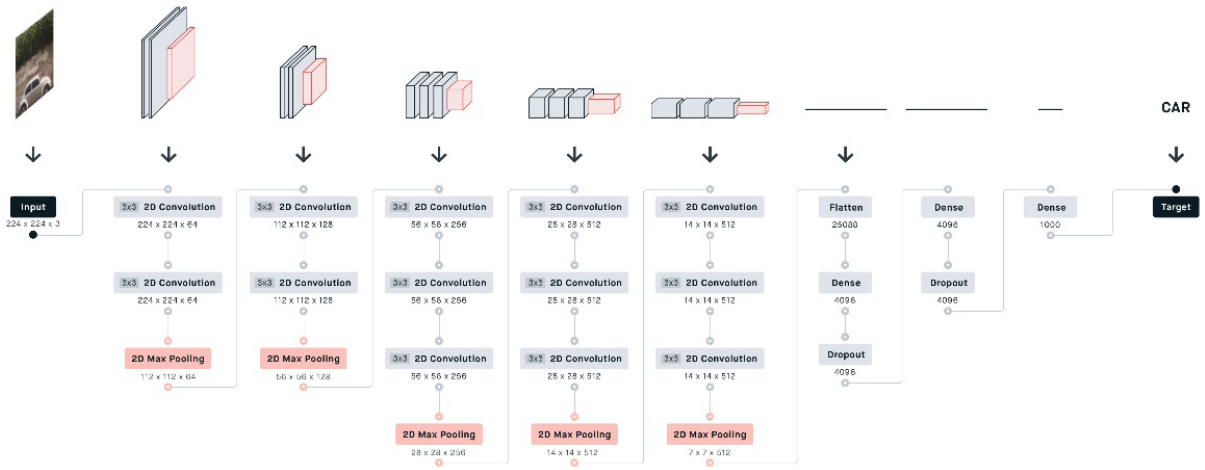

Figure 10 – VGG Neuron Network Operating Algorithm

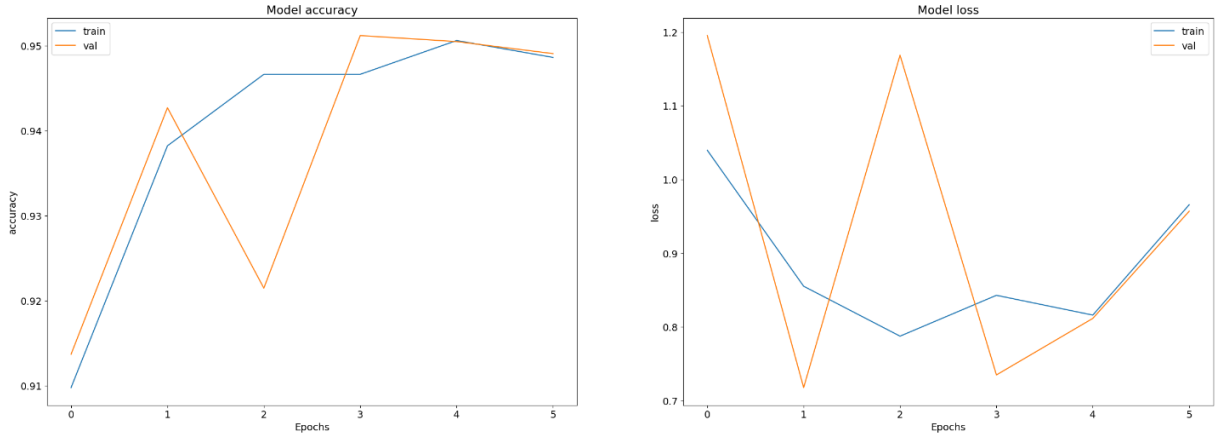

Figure 11 – VGG Neuron Network Training

1. Data Preparation: The data set has been collected and prepared. The data was divided into training and test samples.

2. Load pre-prepared weights: Pre-prepared weights loaded from available sources, such as Keras or pytorch libraries, have been used.

3. Model ingress: The architecture of the VGG neuron network was created using the selected version of the model.

4. Model training: using the training model and loss function (e.g. cross-entropy), the model was trained. In the learning process, an optimizer was used to adjust the network weight in each learning era. Reading options such as read speed and package size were chosen based on specific requirements.

5. Model assessment: After the completion of the training, the model was evaluated in the test model. To evaluate the quality of the model, accuracy indicators (accuracy) and other indicators were calculated.

6. Configuration and improvement: If the results do not meet the requirements, the model configuration methods have been applied. This may include changing model settings, adjusting data, or applying enhancements.

7. Use of Model: After successful training, the trained VGG model can be used to classify new images, predictions, or other tasks that it has been developed.

On the one hand, the use of multiple convolution layers with smaller convoluted nuclei instead of a larger convolution layer with convoluted nuclei may lower the parameters and the author believes that this is equivalent to a non-linear map, which increases the chance of a match (Figure 12)

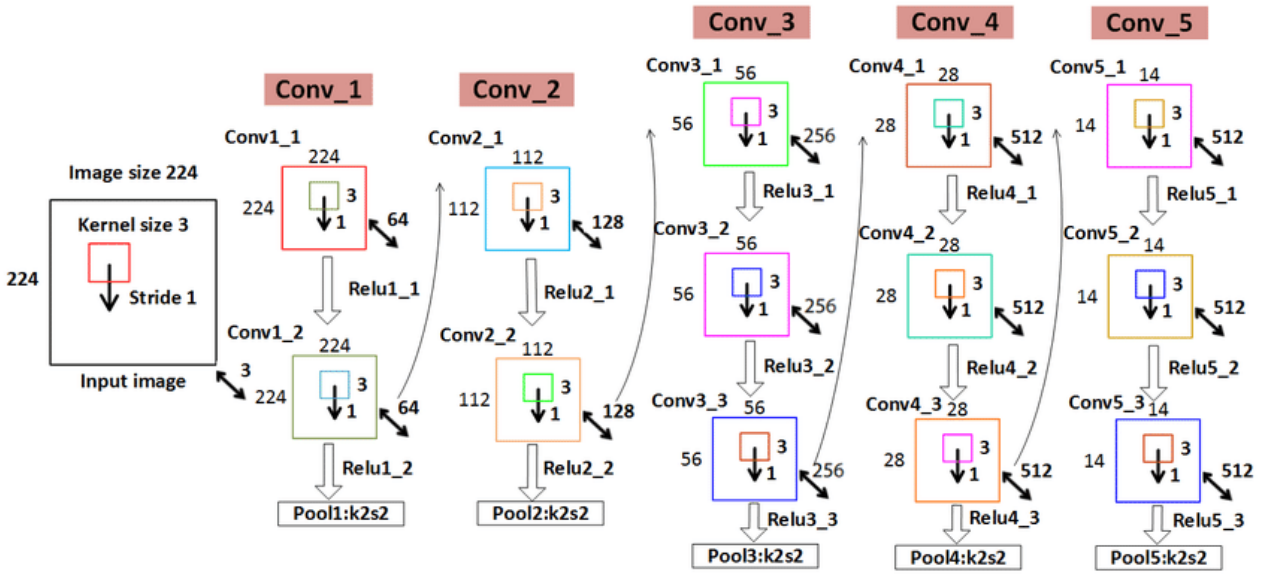

Figure 12 – The operating algorithm for the VGG neuron network

A two-row convolution measuring 3x3 equals a sensitive field measuring 5x5, and three equals 7x7. The advantages of using three 3x3 convolutions instead of one 7x7 are twofold: first, adding three ReLU layers instead of one makes the decision-making function more understandable; second, it reduces the parameters. The 1x1 convolution layer is mainly designed to increase the linear magnificence of the decision-making function without affecting the sensitive field of the convolution layer. Although the 1x1 convolution operation is linear, ReLU adds non-linear.

These networks adhere to the same design principles, but differ in depth. This image was used during the introduction of VGG16.

This is a relative table of 6 networks (Figure 13). The network from A to E deepens, with several layers added to test the effect.

Each column describes the structure of each network in detail.

This is the correct way to conduct experiments, that is, to use the simplest method of solving the problem, and then gradually optimize taking into account the problems that arise.

The Keras library also furnishes a readily available template that facilitates the retrieval of pre-trained model weights, which can be applied for diverse functionalities such as image classification, shape recognition, and object detection. The approach involves accessing the model architecture offered by the library, subsequently incorporating the requisite weights into the respective layers.

ResNet-50 constitutes a convolutional neural network characterized by a depth comprising 50 layers (see Figure 14). To leverage its capabilities, a pre-trained variant of this neural network is accessed from the Imagenet database, having been trained on an extensive corpus of over a million images. This pre-trained neural network is equipped to categorize images across a spectrum of 1,000 object classes, spanning items like keyboards, mice, pencils, and numerous animals. This renders the neural network adept at discerning intricate object representations across a diverse array of images. The neural network's input image dimensions are set at 224 by 224 pixels. Further information regarding pre-trained neural networks can be sourced from the dedicated section on pre-prepared deep neural networks in MATLAB. The process of image classification using the ResNet-50 Model mirrors the steps outlined for GoogLeNet, with the substitution of GoogLeNet by ResNet-50.

The original architecture of ResNet was ResNet-34 with 34 floors. This provided a new way to add more convolutional layers to CNN without having to face the disappearing gradient problem using the concept of short connectivity. A fast connection" skips" some levels, turning a normal network into a residual network. The conventional network was based on VGG neural networks (VGG-16 and VGG-19) — each convolutional network had a 3×3 filter. However, ResNet has fewer filters and is less complex than VGGNet. The performance of a 34-layer ResNet can reach 3.6 billion flops, while a smaller 18-layer ResNet can reach 1.8 billion flops, which is significantly faster than a VGG-19 network with 19.6 billion flops. The ResNet architecture

follows two basic design rules. First, the number of filters in each layer is the same, depending on the size of the output map of objects. Secondly, if the size of the object Map is halved, then the number of filters is doubled to maintain the time complexity of each layer. ResNet-50 has an architecture based on the model described above, but with one important difference. The 50-layer resnet network uses a density design for the building block. The Bottleneck residual block uses 1×1 convolutions called" bottlenecks", which reduces the number of parameters and matrix multiplications. This allows you to significantly speed up the reading of each layer. It consists not of two layers, but of three.The ResNet level 50 architecture includes the following elements, as shown in the table below:

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 13 – VGG network configurations.

• 7×7-dimensional kernel convolution, with 64 other cores in 2-dimensional increments.

• The maximum joining layer is in 2 dimensional increments.

• Another 9 layers are convolutions with 3×3.64 cores, the second with 1×1.64 cores, and the third with 1×1256 cores. This 3 layer is repeated 3 times.

• Another 12 layers with 1×1,128 cores, 3×3,128 cores, and 1×1,512 cores were repeated 4 times.

• Another 18 layers with 1×1256 and 2 3×3,256 and 1×1,1024 cores were repeated 6 times.

• Another 9 layers with cores 1×1512, 3×3512 and 1×12048 were repeated 3 times.

• A fully connected level of 1000 nodes using the Softmax activation function [13].

In the project aimed at constructing an ensemble to address the challenge of image classification, a variety of neural networks were employed, notably encompassing convolutional neural networks (CNN), such as VGG-16 and ResNet-50. VGG-16 is a convolutional neural network architecture featuring 16 layers, originating from the Visual Geometry Group (VGG) initiative at Oxford University in 2014. It was meticulously designed to tackle image classification predicaments and has garnered considerable prominence in the scientific domain owing to its high degree of accuracy.

ResNet-50 is a convolutional neural network developed by Microsoft Research in 2015. This neural network consists of 50 layers and differs from other convolutional neural networks by using residual blocks to accelerate learning. In the process of ensemble construction, a fusion of all three neural networks – CNN, VGG-16, and ResNet-50 – was harnessed. This strategy was adopted with the aim of enhancing classification accuracy, as each neural network can detect distinct patterns within the images. Following individual training on a dedicated training dataset, these neural networks are amalgamated into a unified ensemble. This ensemble configuration is subsequently leveraged for the purpose of classifying novel images [14].

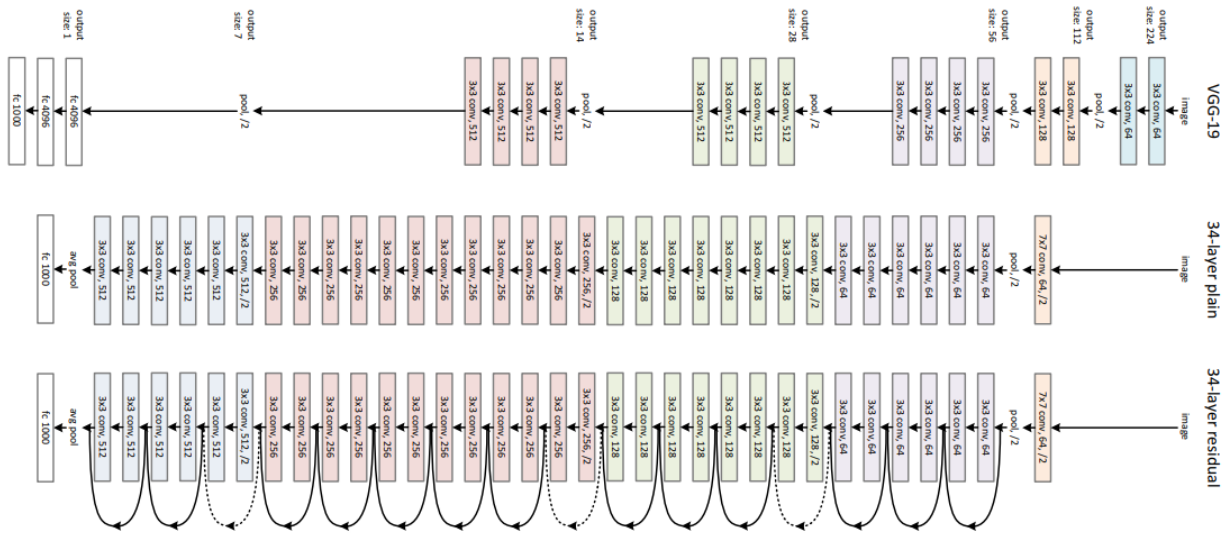

Figure 14 – ResNet-50 neural network architecture

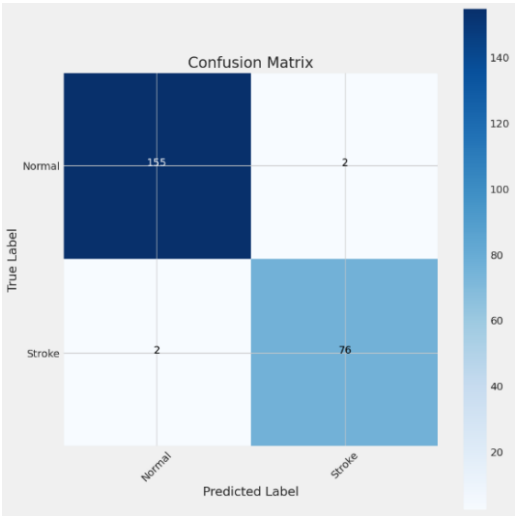

Figure 15 – Confusion matrix of convolutional neural network

	precision	recall	f1-score
0	0.88	0.92	0.90
1	0.93	0.90	0.92
accuracy			0.91
macro avg	0.91	0.91	0.91
weighted avg	0.91	0.91	0.91

Figure 16 – Results of the CNN neural network

The Confusion matrix (figures 15-19) obtained in the process of learning a convolutional neural network. True Positive – 155, False Positive – 2, False negative – 2, True negative – 76. therefore, the convolutional neural network shows an error of only 0.017%.

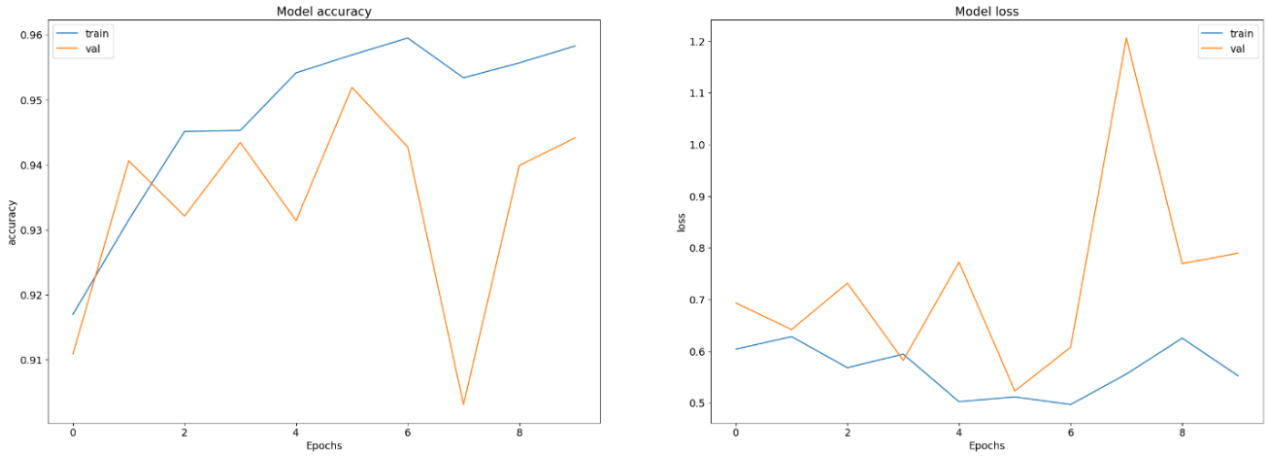

Figure 17 – ResNet – 50 neural network training

	precision	recall	f1-score
0	0.94	0.94	0.94
1	0.95	0.96	0.95
accuracy			0.95
macro avg	0.95	0.95	0.95
weighted avg	0.95	0.95	0.95

Figure 18 – Results of the VGG-19 neural network

	precision	recall	f1-score
0	0.95	0.94	0.94
1	0.95	0.96	0.95
accuracy			0.95
macro avg	0.95	0.95	0.95
weighted avg	0.95	0.95	0.95

Figure 19 – Results of the ResNet-50 neural network

In the context of classification tasks, metrics play a pivotal role in evaluating model performance. Four key metrics that are swiftly employed include Accuracy, F1-score, Precision, and Recall. This review will delve into each metric, elucidating their principal characteristics and significance.

1. Accuracy indicator measures measurements that are classified to a certain extent in relation to the total number of measurements. This shows how accurately the Model classifies data and is the simplest and most understandable metric. Accuracy improvement formula:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

Where TP (True positive) is the number of correctly predicted positive classes, TN (True Negative) is the number of correctly predicted negative classes, FP (False Positive) is the number of incorrectly predicted positive classes, FN (False Negative) is the number of incorrectly predicted negative classes.

2. F1-score – harmonic mean between score – precision and recall. It takes into account both the precision and recall of the model's forecasts. The F1-score works best if the balance between accuracy and recovery is important. F1-score calculation formula-dimensions:

$$F1 = 2 * (precision * recall) / (precision + recall)$$

3. Precision measures the proportion of correctly predicted positive classes relative to all predicted positive classes. This indicator is aimed at minimizing false positive results. Accuracy is useful in cases where false positive predictions can have serious consequences. The formula for calculating accuracy:

$$Precision = TP / (TP + FP)$$

4. Recall measures the proportion of correctly predicted positive classes relative to all real positive classes. This indicator is aimed at minimizing false negative results. Recall is useful in cases where false negative forecasts can lead to serious consequences. Recall calculation formula [15]:

$$Recall = TP / (TP + FN)$$

5. Each of these metric values has its advantages and limitations, and the choice of a specific metric depends on the specific classification task and its context.

• Accuracy is useful when all classes are equal and the total number of correctly classified examples is important. However, this may not be obvious if the classes are unbalanced, even if minor errors are significant.

• F1-score takes into account both accuracy and completeness and is a useful metric if the balance between these two aspects is important. It is widely used in reports with unbalanced classes, where accuracy and recall are equally important.

• Precision aims to reduce false positive forecasts. This is important in cases where false positive results have serious consequences, such as medical diagnoses.

• Recall aims to reduce false negative forecasts. This is important in cases where false negative results can have serious consequences, for example in the detection of diseases.

• In general, the choice of metric depends on the specific task, its priorities and context. To get a complete idea of the performance of the classification model, it is recommended to analyze several indicators together.

To reduce detection errors, a deep transfer ensemble model is proposed by combining the results of different independently trained neural networks, as shown in Figure 20.

A set of data transmission training networks can be a reliable approach by minimizing errors. This will allow you to get the optimal result from combined networks with minimal errors. After pre-processing the data, the convolutional neural network architecture is built using pre-prepared models. The main components of the models used are described in the previous sections.

• The ensemble of neural networks VGG-19, ResNet-50 and CNN was built as follows:

• The input is fed to the input of each of the VGG-19, ResNet-50, and CNN networks.

• Each of the networks processes the input data and generates a different set of symbols.

• Each of the sets of signs is combined into a single vector.

This vector is passed to the input of a fully connected network, which is used to combine the signs and make a final decision. Such an ensemble is called soft rule (soft voting). Unlike hard voting (hard voting), which makes decisions based on a simple majority vote, soft voting takes into account the probabilities predicted by each network. This approach allows you to combine different models and use their strengths to achieve better performance than using a private network.

Fully connected layers (SW) are in full contact with neurons. For SW, the input is multiplied by the SW weight matrix to obtain the multiplication result. The presented architecture uses a fully connected dense layer with a softmax activation function for multi-class classification.

The output neurons are connected to regulate convolutional neural networks by randomly setting the output neurons to 0 in hidden layers at each reading iteration. The released neurons do not contribute in any way to direct passage or reverse propagation during the learning phase.

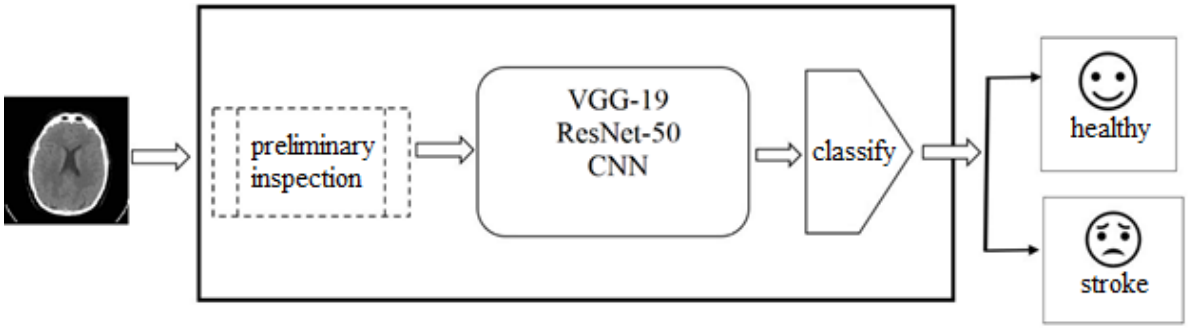

Figure 20 – Neural network operation algorithm

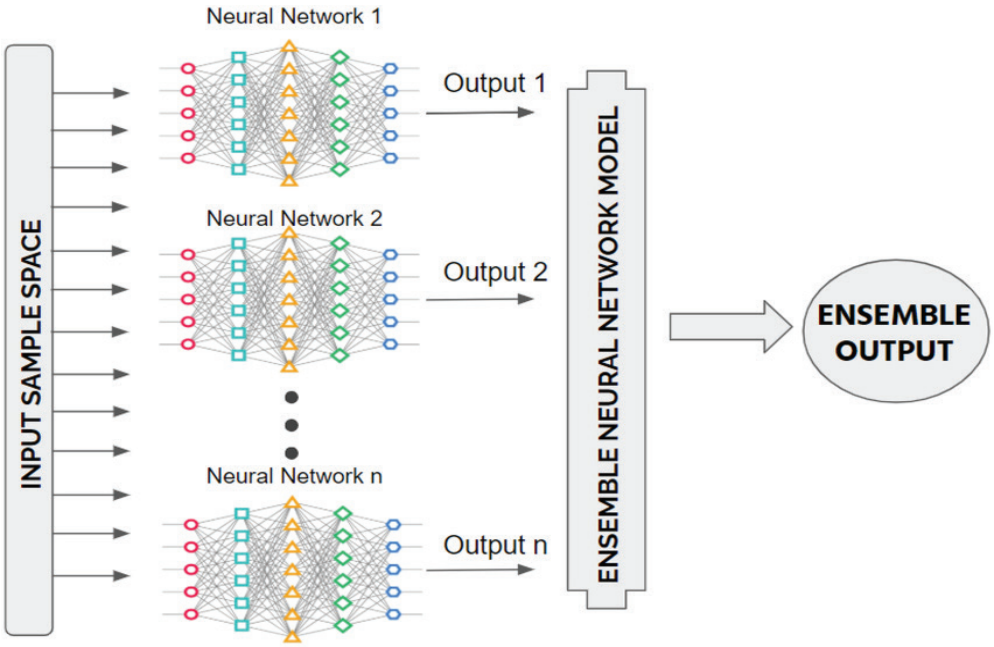

Figure 21 – Structural scheme of the ensemble learning model of neural networks

6. Stacking

In the pursuit of a robust solution, a stacked ensemble strategy was employed, leveraging the collaborative potential of multiple heterogeneous weak learners, including the VGG-19 and ResNet-50 neural networks. The diverse outputs of these individual models were harnessed to generate forecasts that collectively integrated the insights from the weak models, resulting in heightened predictive capability (Fig 22).

In navigating this approach, a training set served as the foundational input for all participating weak forecasters. Each weak model, encompassing the VGG-19 and ResNet-50 neural networks, generated forecasts based on this input. Subsequently, the forecasts were amalgamated within the scope of a final model, referred to as a mixer, meta-student, or meta-model. This concluding model harmonized the manifold forecasts, culminating in an encompassing prediction that synthesized insights from the underlying weak models.

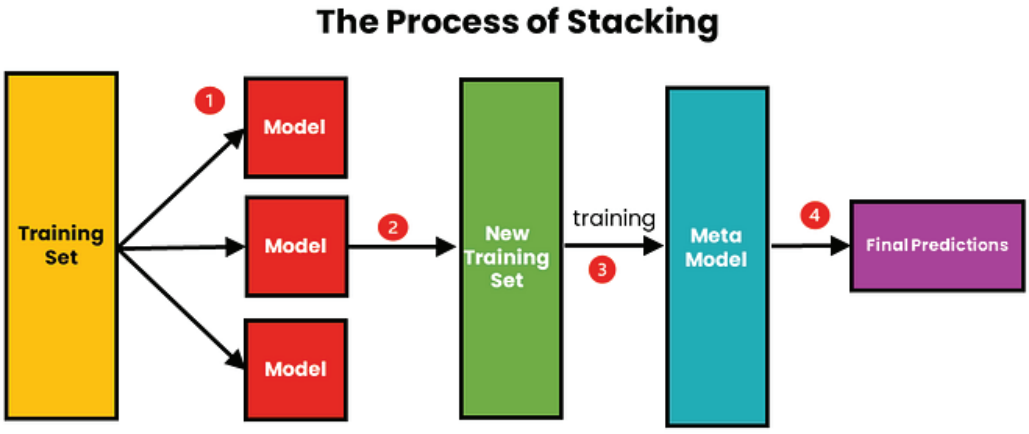

Figure 22 – The process of Stacking

Stacking is a machine learning strategy that combines the predictions of numerous basic models, also known as first-level models or basic training programs, to produce a final prediction. This entails training many basic models on the same training dataset, and then transferring their predictions to a higher-level model, also known as a meta-model or a second-level model, to obtain a final forecast. The basic idea of stacking is to combine the predictions of different underlying models to get a more outstanding forecasting performance than using a single model.

Stacking, referred to as "Stacking Generalization," is a holistic approach in machine learning that integrates multiple models to enhance the overall model performance. The fundamental concept behind stacking involves leveraging predictions from various underlying models to feed into an upper-level model, often called a meta-model or blender. This upper-level model harmonizes these predictions to generate a ultimate prediction.

The composite training set, enriched with predictions from multiple perspectives, was then presented to the meta-model. Equipped with this robust input, the meta-model embarked on a learning journey, refining its predictive capabilities to generate outputs derived from the amalgamated insights garnered from the ensemble's diverse weak models.For the first dense layer, neurons form 64 instead of 128 for the binary classes classification problem. To obtain highlights, a pre-prepared model with several layers is used (Figure 21). In a dense layer, the softmax activation function is used to solve the two-class classification problem. However, sigmoid activation is used for binary classification. Models are trained over 30 eras, the batch size is 16. The Adam Optimizer is used to fine-tune the models. Regulation is achieved through the early termination criterion to prevent overtraining.

Adam optimizer is an extended version of stochastic gradient capture that can be implemented in various deep learning applications such as computer vision and natural language processing in the future. The man was first introduced in 2014. It is an optimization algorithm that can be an alternative to the stochastic gradient capture process. This name comes from the price of adaptation of the moment. The Optimizer is called Adam because it uses the scores of the first and second points of the gradient to adapt the reading speed for each weight of the neural network. The Optimizer is called Adam; this is not an abbreviation. Adam is presented as the most effective stochastic optimization, requiring only First-Order gradients, where the memory requirements are too small.

The aforementioned transfer training classifiers have led us to propose the ensemble stacked classifier. A deep CNN set can be a powerful way to achieve good results, as it is based on the concept of combining solutions from different models. Given the stochastic nature of deep CNN, the architecture of each neural network explores some unique patterns of other neural networks. The ensemble method provides high accuracy of improvement and the ability to highlight features.

	Precision	recall	f1-score
0	1.00	0.97	0.99
1	0.06	0.97	0.11
total	1.00	0.97	0.99

Figure 23 – The result of the created ensemble

Good results were achieved through the utilization of an ensemble comprising the VGG-19, CNN, and ResNet-50 models (figure 22). The decision was made to amalgamate these models into an ensemble, thereby enhancing the accuracy and dependability of image classification within the realm of computer vision.

Distinct strengths and weaknesses are associated with each individual model, and the objective was to harness the optimal attributes of each. Following meticulous adjustments and amalgamation of the models' predictions, a noteworthy enhancement in results was realized. The ensemble yielded diverse approaches to image classification, affording the capacity to render more precise decisions.

Figure 23 illustrates that the combined ensemble of CNN, VGG-19, and ResNet-50 neural networks proves to be more efficacious compared to the individual performance of each network. The consolidation of the CNN, VGG-19, and ResNet-50 neural networks into a singular ensemble was executed in the subsequent manner:

Individual models were formulated, with distinct models being generated for each of the neural networks. The construction of each model adhered to the respective architectures of CNN, VGG-19, and ResNet-50. Specific weights were assigned to each model in correspondence with the training data for each iteration.

Forecast generation involved the passage of the test data through each individual model, facilitating the generation of predictions for each respective

network. Following the data processing phase for each model, a collection of forecasts was derived.

The aggregation of forecasts was accomplished by employing a voting mechanism, which facilitated the fusion of predictions stemming from each distinct model. A straightforward majority-based voting strategy was employed, wherein the class garnering the highest count of votes among all models was selected. This approach enabled each model to contribute its individual preference towards a specific class.

The ultimate prediction of the ensemble was derived by utilizing the amalgamated predictions obtained from the preceding steps.

7. Conclusion

In conclusion, the final prediction of the neural network ensemble was achieved through the voting process, wherein the class attaining the highest number of votes was identified as the ultimate prediction.

Importantly, the integration of neural networks into an ensemble can be done in different ways, and voting is just one of them. Depending on the specific requirements and characteristics of the data, other methods can be used to combine predictions, such as mean or weighted voting, based on the reliability of the models.

Employing the ensemble necessitates supplementary resources and time investment for both exploration and implementation. Nevertheless, the outcomes justify these costs, as the attained accuracy and dependability were satisfactory. The neural network ensemble emerges as a potent instrument for enhancing the precision and trustworthiness of machine learning models. The amalgamation of predictions from diverse models facilitates the mitigation of stochastic errors and the consideration of varied facets of the data.

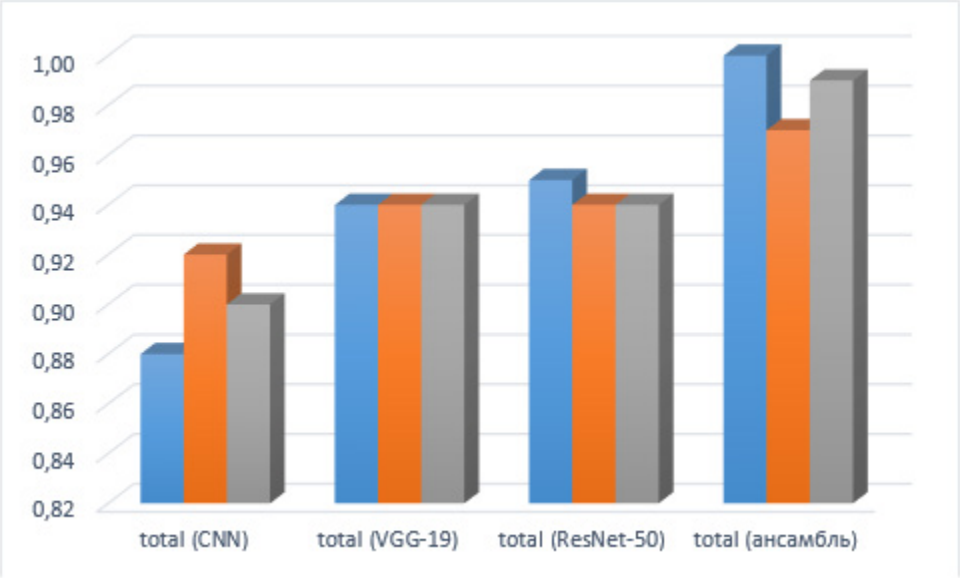

Figure 24 – Accuracy of classification of neural networks

References

1. Adi, N.; Farhany, R.; Ghina, R.; Napitupulu, H. Stroke Risk Prediction Model using Machine Learning. In Proceedings of the International Conference on Artificial Intelligence and Big Data Analytics, Bandung, Indonesia, 27–29 October 2021.
2. Dritsas, E.; Trigka, M. Stroke Risk Prediction with Machine Learning Techniques. Sensors 2022, 22, 4670.
3. Tazin, T.; Alam, M.; Dola, N.; Bari, M.; Bourouis, S.; Khan, M. Stroke Disease Detection and Prediction using Robust Learning Approaches. J. Healthc. Eng. 2021, 2021, 7633381.
4. Al-Mekhlafi, Z.; Senan, E.; Rassem, T.; Mohammed, B.; Makbol, N.; Alanazi, A.; Almurayziq, T.; Ghaleb, F. Deep learning and machine learning for early detection of stroke and haemorrhage. Comput. Mater. Contin. 2022, 72, 775–796.
5. Sailasya, G.; Kumari, G. Analyzing the Performance of Stroke Prediction using ML Classification Algorithms. Int. J. Adv. Comput. Sci. Appl. 2021, 12, 539–545.

6. Dev, S.; Wang, H.; Nwosu, C.; Jain, N.; Veeravalli, B.; John, D. A predictive analytics approach for stroke prediction using machine learning and neural networks. Healthc. Anal. 2022, 2, 100032.
7. Bandi, V.; Bhattacharyya, D.; Midhunchakkravarthy, D. Prediction of Brain Stroke Severity using Machine Learning. Rev. D'intelligence Artif. 2020, 34, 753–761.
8. Alhakami, H.; Alraddadi, S.; Alseady, S.; Baz, A.; Alsubait, T. A Hybrid Efficient Data Analytics Framework for Stroke Prediction. Int. J. Comput. Sci. Netw. Secur. 2020, 20, 240–250.
9. Anna Dobshik, Andrey Tulupov, Vladimir Berikov: Weakly supervised semantic segmentation of tomographic images in the diagnosis of stroke.
10. Karaev Nıkıta Mıhaılovıch: Analız ızobrajenıı s ıspolzovanıem ansamblıa al-gorıtmov klasternogo analıza
11. Kirill Kalmutskiy, Andrey Tulupov, Vladimir Berikov: Recognition of Tomographic Images in the Diagnosis of Stroke
12. Sandro Vega-Pons, Jyrko Correa-Morris & José Ruiz-Shulcloper: Weighted cluster ensemble using a kernel consensus function
13. Germain Forestier, Cedric Wemmert: Semi-supervised learning using multiple clusterings with limited labeled data
14. D. Withey, Z. Koles : A Review of Medical Image Segmentation: Methods and Available Software
15. Zhouming Ren, Xinzheng Fu: Stroke Risk Factors in United States: An Analysis of the 2013–2018 National Health and Nutrition Examination Survey