Balkan Journal of Electrical and Computer Engineering

BAJECE

Volume 1, Number 1, 2023

INESEG

CONTENTS

Designations used by companies to distinguish their products are often claimed as trademarks. All brand names and product names used in this journal are trade names, service marks, trademarks or registered trademarks of their respective owners. The Publisher is not associated with any product or vendor mentioned in this journal.

This publication is designed to provide accurate and authoritative information in regard to the subject matter covered. It is sold on the understanding that the Publisher is not engaged in rendering professional services. If professional advice or other expert assistance is required, the services of a competent professional should be sought.

Balkan Journal of Electrical and Computer Engineering (BAJECE)

Scope: Balkan Journal of Electrical and Computer Engineering (BAJECE) was established in 2023. It stands as a peer-reviewed international journal catering to the interests of individuals engaged in research across various domains closely tied to Electrical Engineering disciplines. BAJECE aims to deliver a remarkably comprehensible and valuable contribution to existing literature, poised to be an essential reference for years to come.

Encompassing a broad spectrum, the journal welcomes both new theoretical and experimental discoveries within the Engineering realm and its closely associated areas. Furthermore, it actively solicits the submission of critical review articles that delve into recent advancements in these fields, alongside technical notes. BAJECE aspires to be a vital platform fostering growth and collaboration within these disciplines.

The scopes include:

Electrical & Electronics Engineering

Computer Engineering

Biomedical Engineering

EDITORIAL BOARD MEMBERS

Publisher Of Journal

INESEG

ETHICS and POLICIES

Balkan Journal E.and Com.Engin. is committed to following the Code of Conduct and Best Practice Guidelines of COPE (Committee on Publication Ethics) . It is a duty of our editors to follow Cope Guidance for Editors and our peer-reviewers must follow COPE Ethical Guidelines for Peer Reviewers We expect all prospective authors to read and understand our Ethics Policy before submitting any manuscripts to our journals.

Please note that submitted manuscripts may be subject to checks using the iThenticate service, in conjunction with CrossCheck, in order to detect instances of overlapping and similar text.

The iThenticate software checks submissions against millions of published research papers, documents on the web, and other relevant sources. If plagiarism or misconduct is found, consequences are detailed in the policy.

The chief goal of our policy is threefold: to provide advice for our authors, to maintain the scholarly integrity of our journals and their content, and to detail the ethical responsibilities of BAJECE, our editors and authors.

We expect all authors to read and understand our ethics policy before submitting to any of our journals. This is in accordance with our commitment to the prevention of ethical misconduct, which we recognise to be a growing problem in academic and professional publications. It is important to note that most incidents of plagiarism, redundant publication, copyright infringement or similar occur because of a lack of understanding, and not through fraudulent intent. Our policy is one of prevention and not persecution.

If you have any questions, please contact the relevant editorial office, or BAJECE.

Authors' Responsibilities

Authors should:

- Ensure that all researched work submitted is original, fully referenced and that all authors are represented accurately. The submission must be exclusive and not under consideration elsewhere.
- Provide accurate contact details for a designated corresponding author, who shall be deemed by the publisher and editor as fully responsible for the authorship of the paper and all communications concerning the ethical status and originality of the paper. This includes any queries or investigations that may arise, pre- or post publication.
- Openly disclose the source of all data and third party material, including previously unpublished work by the authors themselves. Anything that could compromise the originality of the submission should be expressly avoided and/or discussed with the editorial office in the first instance.
- Identify any third party material that they intend to include in their article, and obtain written permission for re-use in each instance from the relevant copyright holders. Such permissions should be submitted once the manuscript is accepted, or requires small changes to be accepted. For further guidance on seeking permission to use 3rd party material please see the Rights and Permissions section.
- Openly disclose any conflict of interest - for example, if publication were to benefit a company or services in which the author(s) has a vested interest.

- Expect to formally agree publication terms which defines the author and the publishers rights for the work. Visit our website for further information.
- Expect the editor to scan submissions using plagiarism detection software at iThenticate to check a paper's originality before sending out for review.
- Fully correspond and comply with the editor and publisher in any requests for source data, proof of authorship or originality in a timely manner, providing reasonable explanation for discrepancies or failures to disclose vital information.
- Fully co-operate with any consequent investigations if the editor and/or publisher are dissatisfied with the evidence available or the explanations provided.
- Expect transparency, efficiency and respect from the publisher and the editor during the submissions process.
- Remain in good communication with both the publisher and the editor.
- When necessary, submit corrigenda in a timely and responsible fashion.
- Co-operate fully with the publication of errata and with the retraction of articles found to be unethical, misleading or damaging.
- Remain in good communication with the editor(s), the publisher and any co-authors.

Editors' Responsibilities

Editors should:

- Read and understand COPE guidelines as well as BAJECE's ethics policy, and follow them during all editorial processes.
- Protect the reputation of their journal(s) and published work by only publishing content of the highest quality and relevance in a timely and responsible manner.
- Carry out thorough, objective and confidential peer review for original article submissions that pass the initial quality check and editorial assessment, in adherence with COPE guidelines and BAJECE' ethics policy.
- Detail and justify any article types which will not be peer reviewed (e.g. editorials, opinion pieces etc.).
- Provide a transparent review and publication process as far as is possible, with full respect and care paid to the author(s).
- Provide advice and give reasonable explanation and updates to authors during the submissions process and once a decision has been made.
- Allow authors the right to appeal any editorial decision.
- Only accept papers based on the original merit, quality and relevance of their content.
- Support authors in queries concerning the originality of their submissions and request the support of BAJECE if necessary.
- Advise the publisher of any third party material which has been included for which they do not believe sufficient permission has been cleared.

- Be ready and prepared to publish corrections, corrigenda, errata when necessary, as well as retract articles that (the editor and BAJECE) deem unethical, misleading or damaging.
- Remain in good communication with both the publisher and the author(s).

Reviewers' Responsibilities

Reviewers should:

- Adhere to BAJECE's policy of confidential peer review of their journals. This includes, but is not restricted to, keeping their identity hidden from authors and not externally distributing any work that is passed to them for their eyes only.
- Only accept invitations to review work that is relevant to their own expertise and speciality.
- Review submitted work in a responsible, impartial and timely manner.
- Report any suspected ethical misconduct as part of a thorough and honest review of the work.
- Avoid the use of unnecessarily inflammatory or offensive language in their appraisal of the work.
- Accept the commitment to review future versions of the work and provide 'follow up' advice to the editor(s), if requested.
- Seek advice from the editor if anything is unclear at the time of invitation.
- Remain in good communication with both the publisher and the editor.

BAJECE's Responsibilities

BAJECE will:

- Protect the reputation of our journals and published work by only publishing content of the highest quality and relevance in a timely and responsible manner.
- Provide detailed information concerning both our understanding of publication ethics and our implementation of the same. Emphasise a desire for prevention, not eventual detection, of ethical misconduct.
- Uphold our COPE membership (or of such similar organisations) and keep our editorial offices, publishing staff and society partners up-to-date with their guidelines and policies, adapting our own where appropriate (and publicising any update).
- When necessary, request proof of originality/accuracy from the corresponding author of any work submitted to any of our journals.
- Use plagiarism detection software when necessary for any submission to any journal at any stage of the submissions and publication process.
- Provide a transparent submissions and publication process, with full respect and care paid to the author. This includes detailed and dedicated instructions to authors for each journal, outlining referencing style, accepted article types and submission processes.
- Investigate thoroughly any suggestion of ethical misconduct detected during any stage of the submissions process. This can include, but is not restricted to, the following: plagiarism, redundant publication, fabrication or misuse of data and authorial disputes.

- When necessary, retract articles that we deem to be unethical, misleading or damaging.
- When necessary, publish errata, corrigenda and retractions in a timely and responsible fashion, detailing the decision online in an open access format and publishing in print as soon as possible.
- Remain in good communication with editors, authors, reviewers and society partners (where applicable).

Further reading

- Authorship of the paper: Authorship should be limited to those who have made a significant contribution to the conception, design, execution, or interpretation of the reported study.
- Originality and plagiarism: The authors should ensure that they have written entirely original works, and if the authors have used the work and/or words of others that this has been appropriately cited or quoted.
- Data access and retention: Authors may be asked to provide the raw data in connection with a paper for editorial review, and should be prepared to provide public access to such data.
- Multiple, redundant or concurrent publication: An author should not in general publish manuscripts describing essentially the same research in more than one journal or primary publication. BAJECE do not view the following uses of a work as prior publication: publication in the form of an abstract; publication as an academic thesis; publication as an electronic preprint. Information on prior publication is included within each BAJECE and its journal Guideline for Authors.
- Acknowledgement of sources: Proper acknowledgment.
- Disclosure and conflicts of interest: All submissions must include disclosure of all relationships that could be viewed as presenting a potential conflict of interest.
- Fundamental errors in published works: When an author discovers a significant error or inaccuracy in his/her own published work, it is the author's obligation to promptly notify the journal editor or publisher and cooperate with the editor to retract or correct the paper.
- Reporting standards: Authors of reports of original research should present an accurate account of the work performed as well as an objective discussion of its significance.
- Hazards and human or animal subjects: Statements of compliance are required if the work involves chemicals, procedures or equipment that have any unusual hazards inherent in their use, or if it involves the use of animal or human subjects.
- Use of patient images or case details: Studies on patients or volunteers require ethics committee approval and informed consent, which should be documented in the paper.

BAJECE has also accessed and learned from the existing policies of other publishers and leading experts as well as open access articles that detail and define ethical misconduct.

- 'Plagiarism and the law', Joss Saunders, Learned Publishing, 23:279-202: http://www.ingentaconnect.com/content/alpsp/lp/2010/00000023/00000004/art00002
- iThenticate Plagiarism Resources: http://www.ithenticate.com/resources/6-consequences-of-plagiarism

Balkan Journal of Electrical and Computer Engineering (BAJECE)
Volume 1 Number 1, September 2023

Development of a Human-Robot Interaction System for Industrial Applications

Mustafa Can Bingol[1] and Omur Aydogmus[2]

1 Department of Electrical-Electronic Engineering Burdur Mehmet Akif Ersoy University, Burdur, Turkey,(e-mail: mcbingol@mehmetakif.edu.tr).

2 Department of Mechatronic Engineering Firat University, Elazig, Turkey, (e-mail: oaydogmus@firat.edu.tr).

Manuscript received Jan 12, 2023; accepted August 16, 2023.

Abstract— **The use of robots is on the rise, and this study focuses on developing manufacturing-assistant robot software for small production plants involved in non-mass production. The primary objective is to address the challenges of hiring expert robot operators by creating user-friendly software, thus enabling non-experts to operate robots effectively. The software comprises three main modules: the convolutional neural network (CNN), process selection-trajectory generation, and trajectory regulation. To initiate operations within these modules, operators record the desired process and its trajectory through hand gestures and index finger movements, captured in a video. The recorded video is then separated into images. These images undergo classification by the CNN module, which also calculates the positions of landmarks, such as joints and index finger's fingernail. Out of eight different pre-trained CNN structures tested, the Xception structure yielded the best result, with a test loss of 0.0051. Using the CNN's output data, the desired process is determined, and its trajectory is generated. The trajectory regulation module identifies the connection points between the generated trajectory and the object, subsequently eliminating unnecessary trajectory segments. The regulated trajectory, along with desired tasks like welding or sealing, is simulated using an industrial robot within a simulation environment. In conclusion, the developed software empowers non-expert operators to program industrial robots, particularly beneficial for companies with non-standardized production lines, where hiring expert robot operators might be challenging.**

Index Terms—**Classification and localization, Fingertip detection, Human-robot interaction, Welding process, Sealing process.**

I. Introduction

Robots can be classified according to the location (mobile and fixed), power systems (pneumatic, hydraulic, and electric), locomotion methods (stable, wheeled, legged, and others), or application areas (industrial and non-industrial) [1]. An industrial robot, as defined by the Robotic Institute of America [2], is a programmable mechanical device used to perform dangerous or repetitive tasks with high accuracy, replacing human labour. Collaborative robots accounted for 5.37%, 6.59%, and 7.54 % of the installed industrial robots between 2019 and 2021, respectively [3]. These installed industrial robots are used in various manufacturing processes such as machine tending, welding, and assembling, as shown in Figure 1. The aim of the current study is to transform a traditional industrial robot into a modern collaborative robot capable of performing welding and sealing processes.

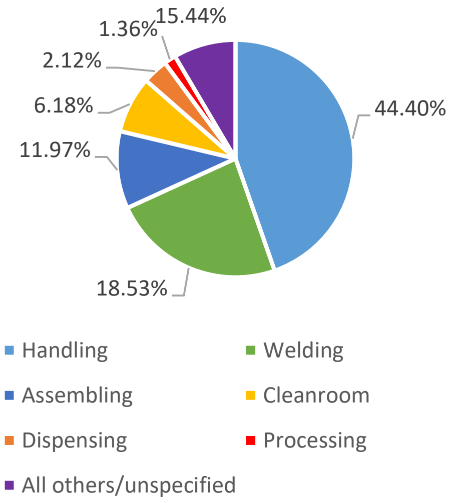


Fig. 1. Task distribution of industrial robots installed in 2022 [3]

Human-robot interaction (HRI) plays a crucial role in enabling human-robot collaboration (HRC). At least one communication channel, such as vision or speech should be used in order to occur HRI. Vision involves the interpretation of images captured by sensors such as cameras. Numerous studies in the literature have utilized vision to establish HRI in literature [4]–[6]. For instance, Hamabe et al. trained a lightweight robot for the assembly process using vision communication channels [7]. In another work, Ding et al. developed robot software to ensure safe manufacturing using vision [8]. In the current study, vision-based HRI software was developed.

Hand gestures recognition and fingertip position determination are commonly used methods for human-computer interaction and HRI [9]–[12]. Raheja et al. calculated to fingertip position by using skin colour of hand [9]. In another study, sequential mathematical operations were employed to obtain the fingertip position after subtracting the color-based hand image from the main image [13]. Other studies have also explored similar approaches using colour-based method [11], [12]. Another method employed for in hand gesture identification and fingertip position detection involves obtaining information from depth images captured by RGB-D sensors [14], [15]. Shin and Kim achieved an air writing process utilizing an RGB-D sensor and fingertip detection [16]. Similarly, another study successfully detected with successful by using RGB-D sensor [17]. Huang et al. has achieved fingertip detection by using a cascaded convolutional neural network (CNN) and RGB images, employing a different approach than the aforementioned studies [18]. In another study, air writing has been performed using color separation and faster R-CNN structures together [10]. In the robotic field, one notable example involves determining the orientation of the robot using an image from a sensor worn on the operator's wrist, along with hand gesture recognition and fingertip detection [19]. Many other studies have also utilized specialized sensors for detecting hand gestures or fingertips [20], [21]. In the current study, hand gestures and fingertip position were determined by processing the RGB images obtained from the environment using a single CNN.

The designed CNN structure incorporates a pre-trained CNN, and a transfer learning method was employed to train this designed CNN. Transfer learning is a skill that people often unwittingly use to apply an acquired ability to another similar task. It has been widely used in various applications, ranging from fault detection [22] to time series forecasting [23]. Li et al. used transfer learning to classify text data [24]. In another study, transfer learning has been utilized in order to process hyperspectral image [25]. Moreover, a robot has been successfully developed to detect damaged ropes on bridges using transfer learning [26]. Similarly, in another project, a robot employed transfer learning to distinguish objects from underground images [27]. In the current study, eight different pre-trained CNN architecture was trained for pre-defined task by using transfer learning and CNN structure that was obtained best result was chosen.

In the current study, a robot software was developed to assist in welding and sealing processes based on HRI. The developed software intended for use in small scale plants with non-mass production lines. Firstly, the desired task and a trajectory were defined according to the operator’s hand gestures and positions of the fingertip. Next, the relationship between the defined trajectory and the metal object was searched. Finally, we implemented the obtained trajectory onto the robot in the simulation environment. As a result, the user could command the robot to perform the desired task without the need for manual programming. This study offers a user-friendly approach, allowing the robot to perform similar tasks without requiring any specific knowledge of robotics.

II. Material and Methods

A. Structure of Developed Software

Developed software consists of CNN, process selection-trajectory generation, and trajectory regulation modules. The modules of developed software and data traffic between modules are given in Figure 2.

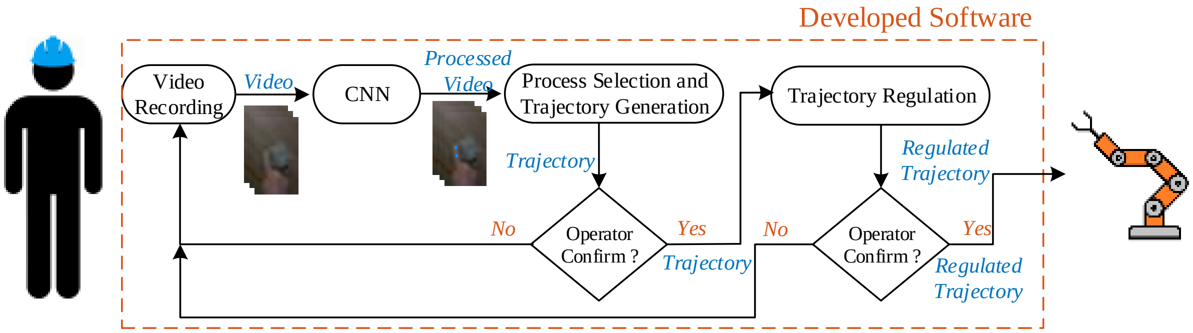


Fig. 2. Block diagram of developed software

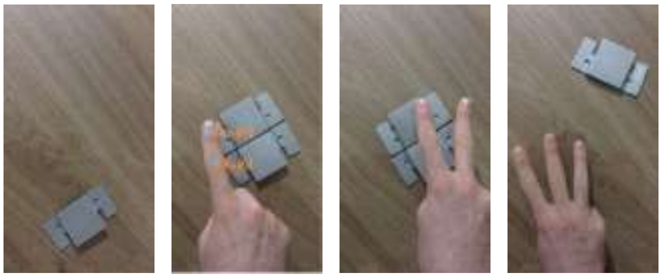
A camera was used to be the robot aware of its environment and perceive the desired process of the robot. The camera’s task was to record the hand movement of the operator in the robot's environment. The operator starts the video recording process before the operation, and the video recording is stopped by the operator when the defined process demonstration is finished. After the recorded video is separated into images, the images are sent to the CNN structure. CNN classifies the images according to hand gestures. If classifying result is calculated as index finger, middle joint, and fingernail positions of the index finger is produced by CNN. In the process selection and trajectory generation module, the type and start-end times of the operation such as welding or sealing are determined from hand motions. The index finger class between the start and end times determines the trajectory of the process. Then, the obtained process and trajectory are submitted for operator approval. If the operator does not confirm the process or trajectory, the program returns to the video recording stage. If the operator confirms the action and the trajectory, the trajectory regulation is applied to the trajectory. After the regulated trajectory is confirmed by the operator again, the determined process is carried out by the KUKA KR Agilus KR6 R900 sixx robot located in the simulation environment along the arranged trajectory. After the process is complete, the program returns to the first step.

1) CNN Module

Training of CNN, which is a part of developed software, consist of three steps as dataset forming, train dataset augmenting, and model training.

Fifty-three videos with a total size of 1.54GB were recorded in the experimental environment in order to create a dataset. Forty-five of these videos were used for the training and validation dataset. Rest of these videos were used for the test dataset. Total 8000 images, that were 180×320px size, were obtained from training and validation videos. 512 images, that were according to homogeneous each class, were randomly separated from this dataset for the validation dataset. Rest of these images was used as training dataset. Total 128 images, that were 180×320px size, were obtained from test videos to generate test dataset. Formed datasets were contained of four class as *Zero*, *One*, *Two*, and *Three*. Each class label was typified finger count of hand gestures. The data contained in *One* labelled class has 4 location information $(x_J, y_J$ and $x_T, y_T)$, as well as class labels. *J* and *T* letters were symbolized joint and fingernail of index finger, respectively. Class label and position data of datasets was given in Figure 3.

Data augmentation is an operation that artificial images, which are generated from training dataset images, are incorporated into the training dataset to increase the performance of CNN. The images were rotated 180° and the brightness of the rotated images were modified ratio of ±25% in order to increase the variety of images in the training data set. After these processes, obtained artificial images was added in the training dataset.



(a) *(b)* *(c)* *(d)*

Fig. 3. Classes in the datasets; (a) Zero, (b) One, (c) Two, (d) Three

After the data sets were created, CNN structure was trained using the block diagram in Figure 4.

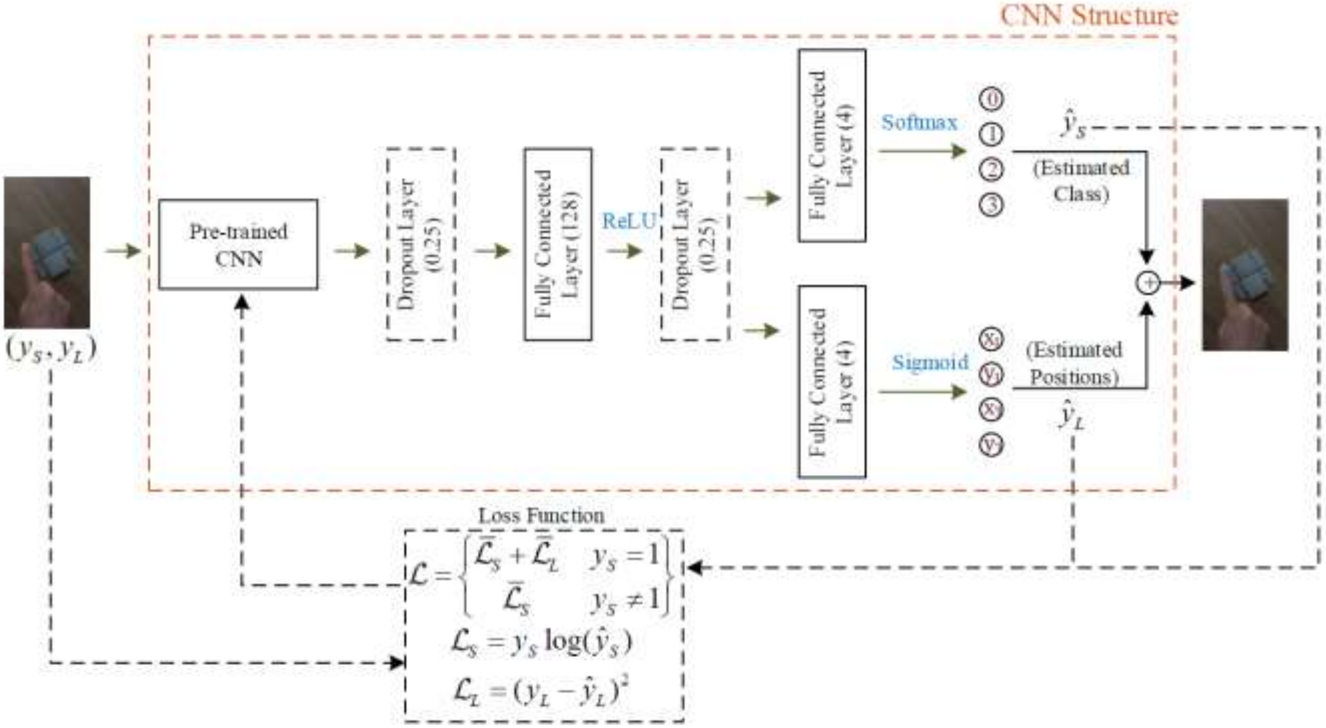

CNN Structure
(y_S, y_L)
Pre-trained CNN
Dropout Layer (0.25)
Fully Connected Layer (128)
ReLU
Dropout Layer (0.25)
Fully Connected Layer (4)
Softmax
0
1
2
3
$\hat{y}_S$
(Estimated Class)
Fully Connected Layer (4)
Sigmoid
x_J
y_J
x_T
y_T
(Estimated Positions)
$\hat{y}_L$
Loss Function
$\mathcal{L} = \begin{cases} \bar{\mathcal{L}}_S + \bar{\mathcal{L}}_L & y_S = 1 \\ \bar{\mathcal{L}}_S & y_S \neq 1 \end{cases}$
$\mathcal{L}_S = y_S \log(\hat{y}_S)$
$\mathcal{L}_L = (y_L - \hat{y}_L)^2$

Fig. 4. Block diagram of CNN structure

y_S and y_L were shown class label and positions data of image in Figure 4, respectively. Eight different CNN model was tried as pre-trained CNN. $\hat{y}_S$ and $\hat{y}_L$ were symbolized predicted class and position data of CNN structure, respectively. Total, classification, and localization loss functions were presented as L, L_S, and L_L symbols, respectively. Cross-Entropy loss function was used as classification loss function and squared error was utilized as localization loss function. If the class of the input image is One, the total loss function is calculated by using a formula that sums the mean of L_S and L_L. Otherwise, the total loss function equals the mean of the classification loss function. The black dashed lines in Figure 4 show the blocks used only during the training of the CNN architecture. The dropout layer is also a structure consisting of dashed lines. The dropout layer ratio was chosen as 0.25. The black solid-lined blocks show the blocks used in both training and testing phases. In the fully connected layer, one of the black solid-lined blocks, there are 128, 4, and 4 neurons, respectively. ReLU, Softmax, and Sigmoid are activation functions found in network outputs. Detailed information about the functions of the blocks was mentioned in [28]–[30]. Also, the CNN structure was trained during 10 epochs by using the Adam optimization algorithm. The learning rate was chosen as 10^{-3} in the first 5 epochs and the learning rate was adjusted as 10^{-4} for the rest of the training process. Mini-batch size was chosen as 32. Weights of the pre-trained CNNs were set up as ImageNet and update of pre-trained CNNs weights were continued during the training process. The training process was shortened by using transfer learning.

2) *Process Selection and Trajectory Generation Module*

Process selection and trajectory generation were realized in this part of developed software by depending on class and position data obtained from CNN structure. Process selection operation was performed before trajectory generation. Firstly, noised class labels need to remove from obtained class labels to choose the process. The noised labels are formed by blurred images that occur when the operator's hand enters or quits on video. It is inspired by the exponential weight averages formula presented in Equation 1 to eliminate noised labels.

$$V_k = \beta V_{k-1} + (1-\beta) Q_k, k = 2,3,\dots,N \tag{1}$$

In Equation 1, V, β, Q, N, and k refer to mean value, mean coefficient, current measure, total sample size, and discrete time index, respectively. How many samples will be averaged with β is determined by using as flows:

$$T_s = \frac{1}{1-\beta} \tag{2}$$

Total sample size is shown as T_s in Equation 2. Equation 1 was not used because the class output of CNN wasn't numerical value. Also, bias coefficient was not used owing to the same reason. Equation 3 that formed by inspiring Equation 1 was used to filter class labels.

$$fL_k = \begin{cases} Q_k & k \le T_s \\ mod(fL_{k-T_s}, fL_{k-T_s+1}, \dots, fL_{k-1}, Q_k\,) & k > T_s \end{cases} \tag{3}$$

fL is typified filtered label data in Equation 3. Raw and filtered label data that belong to the sealing process were given in Figure 5.

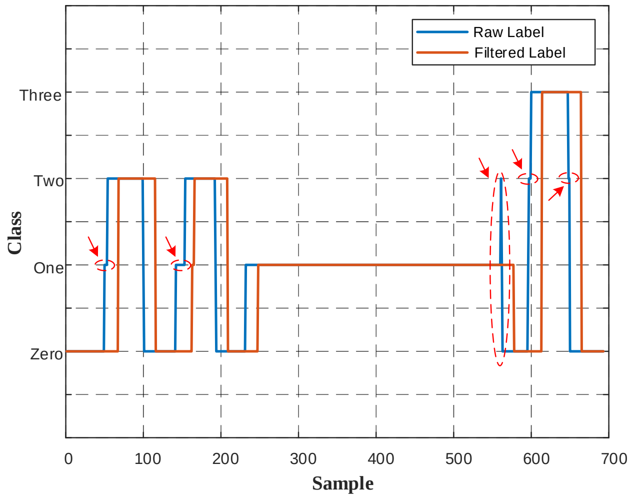


Fig. 5. Label data for sealing process

The red arrows in Figure 5 were shown noised label data. The detected noised labels were eliminated by using Equation 3. In the current study, β coefficient was chosen as 0.97. β coefficient is range of between 0 and 1 as can be understood from Equation 2. Also, while the noise increases as the β approaches 0, the inertia of the system increases as it approaches 1.

After filtering the label data, the process is determined by using the flowchart presented in Figure 6.

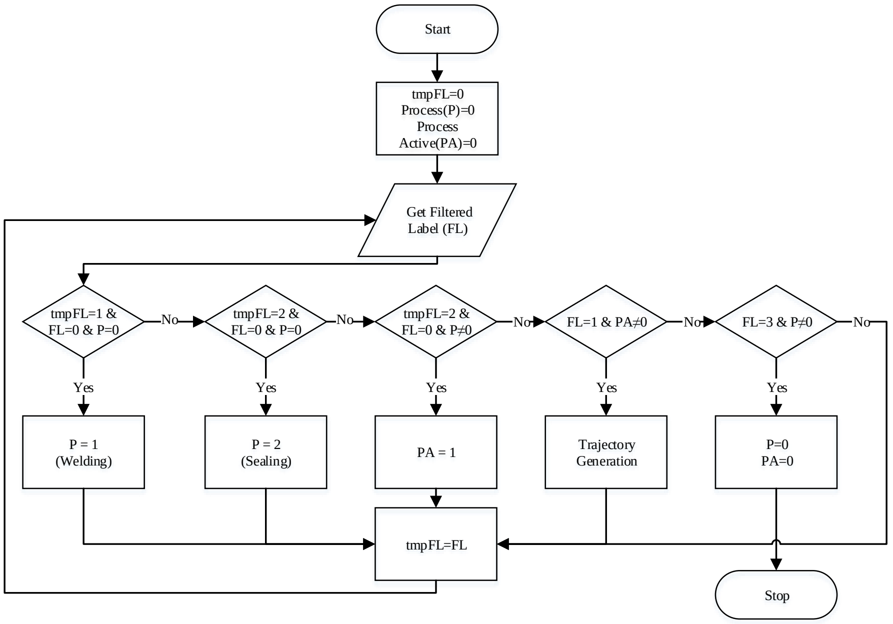


Fig. 6. Process determination flowchart

Firstly, the process is determined in Figure 6. After the defined process is activated, the trajectory generation is started when the filtered label is *One*. The trajectory generation process is continued until the filtered label is *Three*. The position and orientation of the index finger are calculated by using Equation 4-7 and the joint and fingernail position of the index finger. The reason for calculating the positions of the joint and fingernail of the index finger is that the operator can also determine the orientation of the manipulator in orientation-dependent operations such as welding.

$$\alpha = tan^{-1}(\frac{y_T - y_J}{x_T - x_J}) \qquad (4)$$

$$d = \sqrt{(x_T - x_J)^2 + (y_T - y_J)^2} \qquad (5)$$

$$x_F = x_T + \frac{d}{4}\sin(\alpha) \qquad (6)$$

$$y_F = y_T + \frac{d}{4}\cos(\alpha) \qquad (7)$$

α refers to the orientation angle of the index finger in Equation 4. d represents the distance between the joint and fingernail of the index finger in Equation 5. (x_F, y_F) are typified fingertip positions of the index finger on X and Y axes, respectively. The hand quickly moves towards the metal object during the trajectory generation process. The distance between two points was calculated using Equation 5 with finger position information obtained from two sequential images in order not to create a trajectory during this orientation process. If the calculated distance is lower than 7px, obtained (x_F, y_F) was included trajectory. Otherwise, (x_F, y_F) was not incorporated trajectory.

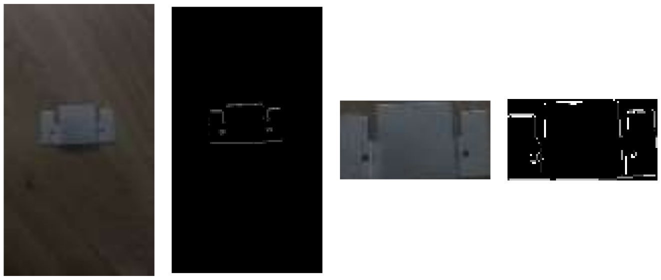
$$tx_k = \delta x_{F_k} + \emptyset\overline{(x_{F_{k-1}}, x_{F_{k-2}}, \dots, x_{F_{k-N}})} \qquad (8)$$

$$ty_k = \delta y_{F_k} + \emptyset\overline{(y_{F_{k-1}}, y_{F_{k-2}}, \dots, y_{F_{k-N}})} \qquad (9)$$

Trajectory of $trj_{2\times k} = [tx, ty]$ was obtained utilizing Equation 8 and 9. tx and ty refer to trajectory position on X and Y axes, respectively. $\delta, \emptyset,$ and N represent last measure coefficient, mean coefficient, and count of elements to be averaged, respectively. These value of the coefficient were chosen as 0.5, 0.5, and 5, respectively. Noises on the trajectory were partially cleared by using Equations 8 and 9.

3) *Trajectory Regulation Module*

Trajectory regulation module was formed to establish relationship between generated trajectory and metal object and decrease noise on trajectory. An image that is not consist operator's hand was taken from the video to regulate trajectory. Initially, an image without operator's hand was taken from the video to regulate trajectory. The image was converted greyscale image and Sobel filter was applied to the greyscale image. Edges of object that is in the image was roughly calculated with the method as can be seen in Figure 7b. After this step, section of object in image was cropped by help of object edges. The cropped image was converted to grayscale image and blurred, respectively. Lastly, Sobel filter was applied on the blurred image and Figure 7d, that shows more clearly edges of object, was obtained. Blur filter was not used in first step because undesired edges were occurred in image.



(a) *(b)* *(c)* *(d)*

Fig. 7. Obtaining edge images process; (a) Original image, (b) Rough edge image, (c) Cropped image, (d) Edge image

Distance between each of the edge points in the obtained edge image and each of the trj points was measured by using Equation 5. trj points that were 15px away from edge points were removed from the trajectory after the measured distances. $mtrj$ trajectory that was not contained in the 15px away points was formed. Finally, the trajectory regulating process was carried out by applying a 20 × 1 dimensional median filter to the positions of the X and Y axis in $mtrj$. Figures 11 and 12 can be examined for a better understanding of the trajectory and regulated trajectory difference.

B. *Simulation of Developed Software*

In this study, the KUKA KR Agilus KR6 R900 sixx robot with 6 axes and an Euler wrist was used in our laboratory. The maximum payload and reach of this robot are 6kg and 901mm respectively. Also, the position repeatability of this robot is 0.03mm. Developed software was simulated on CoppeliaSim program that is a simulation environment. Before the simulation scene was not formed, a 3D solid model of the robot was drawn on the SolidWorks program. The formed 3D solid model was converted to URDF (Unified Robotic Description Format) with URFD exporter [31]. Then the URDF file was added to the designed scene as presented in Figure 8.

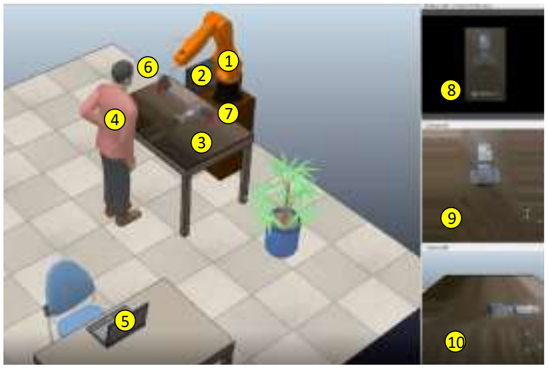


Fig. 8. Simulation environment

Manipulator (1) and robot PC (2) are components of the robot in Figure 9. Work plane, operator, and the PC that the software will run represents as (3), (4), and (5) respectively. Cameras (6-7) were added to the simulation environment to watch to the work plane from different angles. Video streams from the cameras were shown (9-10) windows, respectively. It is assumed that a camera is placed at the endpoint of the manipulator that records the hand movements of the operator and the video stream of this camera is presented on the screen (8). ManyCam program [32] was used to input video from outside to the simulation environment while creating the screen. The manipulator in the simulation environment was moved using MATLAB package program and the inverse kinematic solution of the simulation program. The simulation program was run using the Newton dynamic engine with 50ms step size.

III. Results

A part of the developed software is the CNN structure. The most important building block of this CNN architecture is pre-trained CNN structures. In the current study, CNN structure that using 8 different pre-trained CNN was trained. After the training process, training, validation, and test performance were presented in Table 1.

TABLE I
LOSS FUNCTION VALUES

Algorithms	PS[33] (MB)	TT (s)	L_{train}	L_{valid}	L_{test}
ResNet50[34]	98	1030	0.712	0.429	2.396
VGG16[35]	528	2460	1.539	1.515	1.412
DenseNET121[36]	33	2120	0.961	0.886	0.709
InceptionResNetV2 [37]	215	3820	0.016	0.003	0.533
EfficientNetB0[38]	29	2570	0.031	0.092	0.407
MobileNetV2[39]	14	1930	0.019	**0.002**	0.155
InceptionV3[40]	92	1720	0.017	0.230	0.080
Xception[41]	88	2200	**0.011**	0.003	**0.005**

Bold numbers indicate the best results.

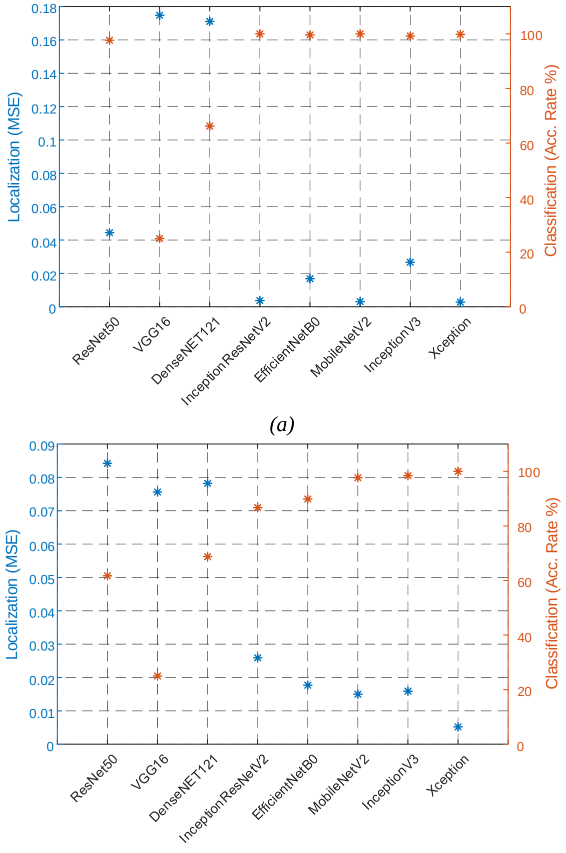
Parameter size and training time were shown as PS and TT in Table 1, respectively. The CNN training process was performed on the Google Colab platform. PC components used on this platform; GPU: Nvidia P100-16GB, CPU: Intel Xeon-2.30GHz, RAM: 25.51GB, Disk memory: 68.40GB. Since the best test result was obtained from the Xception algorithm, the loss values of Xception and other algorithms were compared by using multivariate Tukey comparison test and the Tukey test results are presented in Table 2.

TABLE II
XCEPTION AND OTHER METHODS COMPARISON

Algorithms	$\bar{L} \pm SD$	p (According to Xception)
ResNet50	1.1795±0.8683	**<0.05***
VGG16	1.4889±0.0552	**<0.05***
DenseNET121	0.8522±0.1057	0.238
InceptionResNetV2	0.1846±0.2468	0.999
EfficientNetB0	0.1771±0.1649	0.999
MobileNetV2	0.0592±0.0685	1.000
InceptionV3	0.1096±0.0894	1.000
Xception	0.0067±0.0003	-

**** Statistically significant difference. $\bar{L} \pm SD$ represents the mean of loss and standard deviation.***

The Xception algorithm was found to have statistical differences with the ResNET50 and VGG16 algorithms as can be seen in Table 2. There is no statistical difference between other algorithms and the Xception algorithm. In addition, after the training process was completed, all data in the validation and test datasets were classified by the CNN architecture and the relevant classes were located. These classification and localization results are shown in Figure 9.

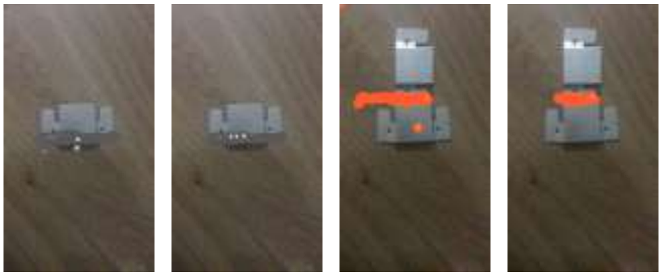


(a)

(b)

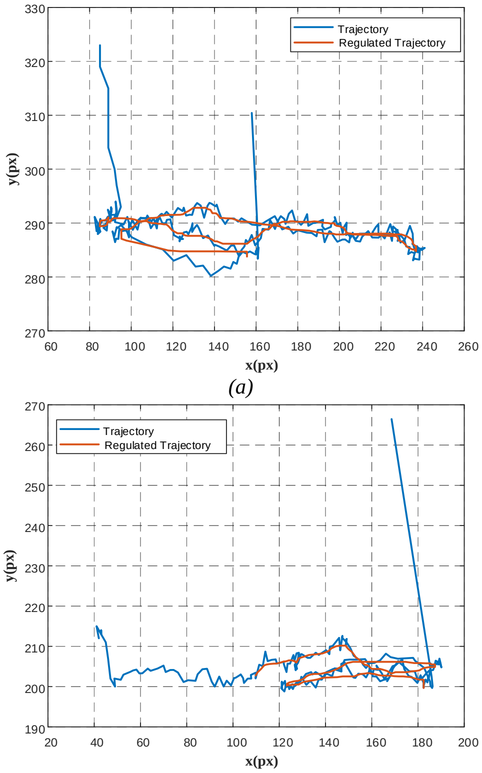
Fig. 9. Classification and localization results; (a) Results of the validation dataset, (b) Results of the test dataset

The Xception algorithm was used in the application as the best performance was obtained using the algorithm. Firstly, the operator recorded various sealing and welding process videos. These recorded videos were processed by CNN as seen in Appendix 1. Sealing process, welding process, and joint-fingernail of index finger were shown with black, red, and blue colour, respectively in Appendix 1. Then, generated trajectories were regulated and regulated trajectories were given in Figure 10 and 11.



(a) *(b)* *(c)* *(d)*

Fig. 10. Visual results of trajectory regulation; (a) Sealing process trajectory, (b) Regulated sealing process trajectory, (c) Welding process trajectory, (d) Regulated welding process trajectory



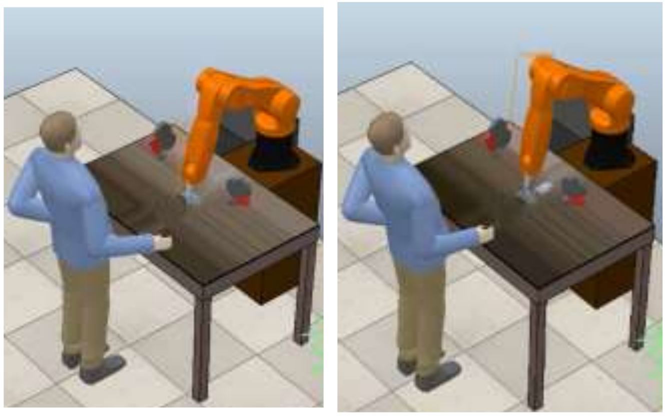
(a)

(b)

Fig. 11. Graphical results of trajectory regulation; (a) Sealing process trajectory, (b) Welding process trajectory

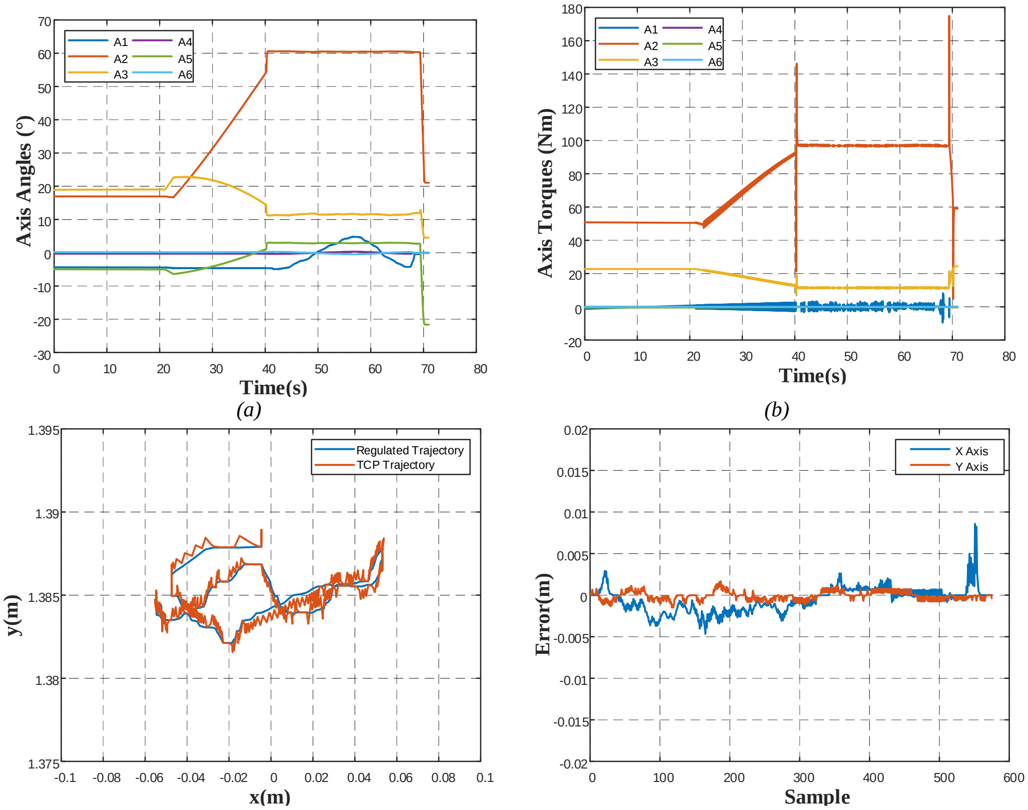
The regulated trajectories were sent to the robot that in the simulation environment. The desired tasks were simulated as seen in Figure 12.

Axis angles, axis moments, tool centre point (TCP) position and trajectory tracking error occurring during the sealing and welding process are presented in Figures 13 and 14.



(a) *(b)*

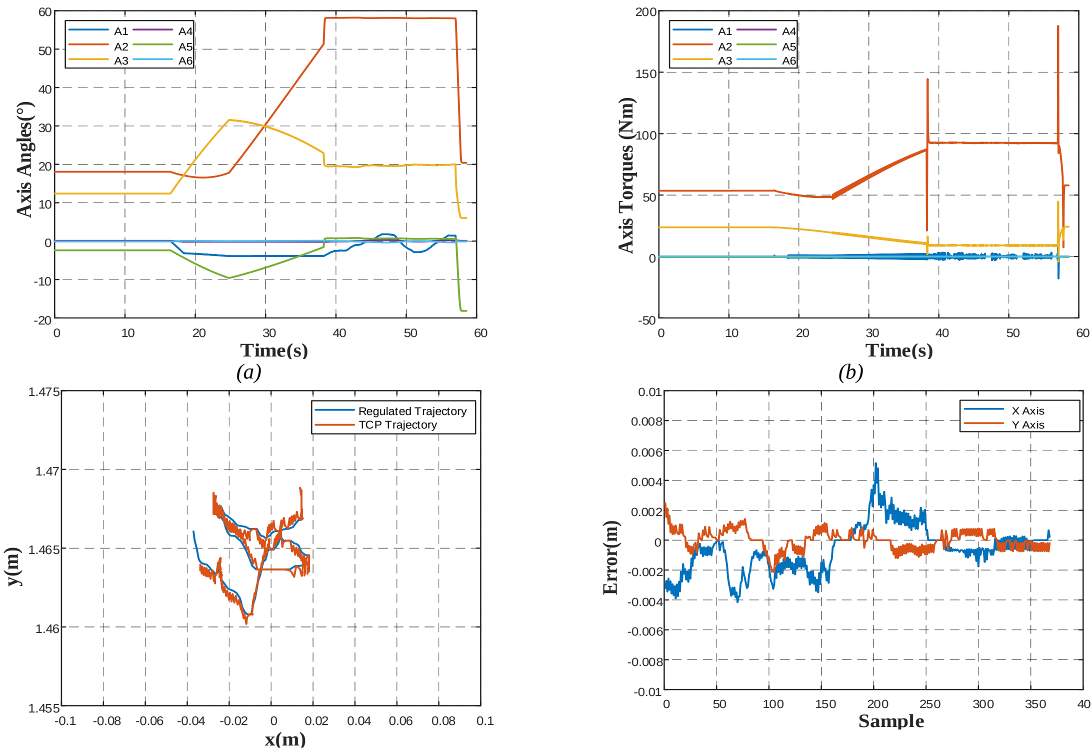
Fig. 12. Simulated desired tasks; (a) Sealing process, (b) Welding process



(a) *(b)*

(c) *(d)*

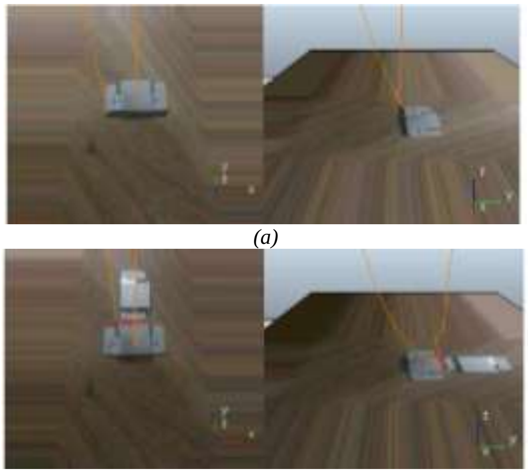
Fig. 13. Values occurring during the sealing process; (a) Axis angles, (b) Axis Moments, (c) TCP trajectory, (d) Error values during trajectory tracking



(a) *(b)* *(c)* *(d)*

Fig. 14. Values occurring during the welding process; (a) Axis angles, (b) Axis Moments, (c) TCP trajectory, (d) Error values during trajectory tracking

After sealing and welding processes, metal object in work plane was given in Figure 15.



(a)

(b)

Fig. 15. Processed metal object; (a) Sealing process, (b) Welding process

The desired task by the operator was performed by the robot as it can be seen in Figure 15. The sealing process, welding process, and robot movements were given as videos in the [42] and [43], respectively.

IV. Discussion

Fingertip location was calculated by using skin colour in some studies when studies in the literature that fingertips detection were examined [11]–[13]. Depth images that were obtained from RGB-D sensors were used to detect fingertip position in other fingertip detection studies [14]–[16]. Skin colour and depth images were not used in this study. Also, when fingertip detection studies that were based on CNN structures were investigated, using cascade CNN structure was seen [18]. A single CNN was used as different from the study. In addition, when fingertip detection studies that were in the robotic field were researched, special sensors were developed to perceive hand gestures [19], [20]. In the current study, a standard camera was used to sense hand gestures.

In the current study, solving of classification and localization problem was implemented to hand gestures recognition and fingertip position detection. In this way, two different problems were solved with a single structure. This study has some limitations. In this study, the most important restriction of the current study is that the thicknesses of the parts to be machined were predefined and a standard depth was worked on. Another limitation is the CNN architectures used. Pre-trained CNN architectures were used to increase the accuracy performance by reducing the training time with the transfer learning method.

V. Conclusion

In this study, a robot software capable of performing processes such as sealing and welding was developed for small-scale plants without mass production capabilities. Operators without any prior robot education/knowledge can program the robot using finger movements through the developed robot software. This programmability capability was achieved through the integration of the CNN, process selection-trajectory generation, and trajectory regulation modules. The CNN structure consisted of a pre-trained CNN, fully connected layers, and activation functions connected in series. Eight pre-trained CNNs were trained on formed datasets and subsequently tested, with the Xception algorithm yielding the best result (L_{test}=0.0051). The CNN structure was used to classify image data and determine the positions of the robot's joints and the index finger's fingernail. With the classification data, the process selection and trajectory generation module detected the desired task, and the same module created the trajectory based on the positions data. Furthermore, a special algorithm was developed within the process selection and trajectory generation module to reduce any noise that may occur during video processing. The generated trajectory was then regulated by the trajectory regulation module to ensure proper alignment with the objects. Following this step, the robot performed the desired process within the simulation environment. In future work, an additional module will be developed to predict trajectories based on the objects and will be incorporated into the software. Subsequently, the software will undergo testing on a real robot. In addition, the developed software will become more improved by using other deep learning architectures such as LSTM.

References

[1] A. Dobra, “General classification of robots. Size criteria,” in *2014 23rd International Conference on Robotics in Alpe-Adria-Danube Region (RAAD)*, IEEE, 2014, pp. 1–6.

[2] “Defining The Industrial Robot Industry and All It Entails.” https://www.robotics.org/robotics/industrial-robot-industry-and-all-it-entails (accessed Sep. 28, 2020).

[3] IFR, *World Robotics 2022*. 2022. [Online]. Available: https://ifr.org/downloads/press2018/2022_WR_extended_version.pdf

[4] S. M. M. Rahman, Z. Liao, L. Jiang, and Y. Wang, “A regret-based autonomy allocation scheme for human-robot shared vision systems in collaborative assembly in manufacturing,” in *IEEE International Conference on Automation Science and Engineering*, 2016, pp. 897–902. doi: 10.1109/COASE.2016.7743497.

[5] H. Ding, M. Schipper, and B. Matthias, “Collaborative behavior design of industrial robots for multiple human-robot collaboration,” in *2013 44th International Symposium on Robotics, ISR 2013*, 2013. doi: 10.1109/ISR.2013.6695707.

[6] S. M. M. Rahman, Y. Wang, I. D. Walker, L. Mears, R. Pak, and S. Remy, “Trust-based compliant robot-human handovers of payloads in collaborative assembly in flexible manufacturing,” in *IEEE International Conference on Automation Science and Engineering*, 2016, pp. 355–360. doi: 10.1109/COASE.2016.7743428.

[7] T. Hamabe, H. Goto, and J. Miura, “A programming by demonstration system for human-robot collaborative assembly tasks,” in *2015 IEEE International Conference on Robotics and Biomimetics, IEEE-ROBIO 2015*, 2015, pp. 1195–1201. doi: 10.1109/ROBIO.2015.7418934.

[8] H. Ding, J. Heyn, B. Matthias, and H. Staab, “Structured collaborative behavior of industrial robots in mixed human-robot environments,” in *IEEE International Conference on Automation Science and Engineering*, 2013, pp. 1101–1106. doi: 10.1109/CoASE.2013.6653962.

[9] J. L. Raheja, K. Das, and A. Chaudhary, “Fingertip Detection: A Fast Method with Natural Hand,” *Int. J. Embed. Syst. Comput. Eng. Local Copy*, vol. 3, no. 2, pp. 85–88, 2012, [Online]. Available: http://arxiv.org/abs/1212.0134

[10] S. Mukherjee, S. A. Ahmed, D. P. Dogra, S. Kar, and P. P. Roy, “Fingertip detection and tracking for recognition of air-writing in videos,” *Expert Syst. Appl.*, vol. 136, pp. 217–229, 2019, doi: 10.1016/j.eswa.2019.06.034.

[11] S. K. Kang, M. Y. Nam, and P. K. Rhee, “Color based hand and finger detection technology for user interaction,” in *2008 International Conference on Convergence and Hybrid Information Technology, ICHIT 2008*, 2008, pp. 229–236. doi: 10.1109/ICHIT.2008.292.

[12] G. Wu and W. Kang, “Vision-Based Fingertip Tracking Utilizing Curvature Points Clustering and Hash Model Representation,” *IEEE Trans. Multimed.*, vol. 19, no. 8, pp. 1730–1741, 2017, doi: 10.1109/TMM.2017.2691538.

[13] G. Wu and W. Kang, “Robust Fingertip Detection in a Complex Environment,” *IEEE Trans. Multimed.*, vol. 18, no. 6, pp. 978–987, 2016, doi: 10.1109/TMM.2016.2545401.

[14] J. Yang, X. Ma, Y. Sun, and X. Lin, “LPPM-Net: Local-aware point processing module based 3D hand pose estimation for point cloud,” *Signal Processing: Image Communication*, vol. 90. p. 116036, 2021. doi: 10.1016/j.image.2020.116036.

[15] C. Wang, Z. Liu, M. Zhu, J. Zhao, and S. C. Chan, “A hand gesture recognition system based on canonical superpixel-graph,” *Signal Processing: Image Communication*, vol. 58. pp. 87–98, 2017. doi: 10.1016/j.image.2017.06.015.

[16] J. Shin and C. M. Kim, “Non-Touch Character Input System Based on Hand Tapping Gestures Using Kinect Sensor,” *IEEE Access*, vol. 5, pp. 10496–10505, 2017, doi: 10.1109/ACCESS.2017.2703783.

[17] J. L. Raheja, A. Chaudhary, and K. Singal, “Tracking of fingertips and centers of palm using KINECT,” in *Proceedings - CIMSim 2011: 3rd International Conference on Computational Intelligence, Modelling and Simulation*, 2011, pp. 248–252. doi: 10.1109/CIMSim.2011.51.

[18] Y. Huang, X. Liu, L. Jin, and X. Zhang, “DeepFinger: A Cascade Convolutional Neuron Network Approach to Finger Key Point Detection in Egocentric Vision with Mobile Camera,” in *2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, 2016, pp. 2944–2949. doi: 10.1109/SMC.2015.512.

[19] F. Chen *et al.*, “WristCam: A Wearable Sensor for Hand Trajectory Gesture Recognition and Intelligent Human-Robot Interaction,” *IEEE Sens. J.*, vol. 19, no. 19, pp. 8441–8451, 2019, doi: 10.1109/JSEN.2018.2877978.

[20] G. Shi, C. S. Chan, W. J. Li, K. S. Leung, Y. Zou, and Y. Jin, “Mobile human airbag system for fall protection using mems sensors and embedded SVM classifier,” *IEEE Sens. J.*, vol. 9, no. 5, pp. 495–503, 2009, doi: 10.1109/JSEN.2008.2012212.

[21] L. Peternel, N. Tsagarakis, and A. Ajoudani, “A human-robot co-manipulation approach based on human sensorimotor information,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 7, pp. 811–822, 2017, doi: 10.1109/TNSRE.2017.2694553.

[22] C. Li, S. Zhang, Y. Qin, and E. Estupinan, “A systematic review of deep transfer learning for machinery fault diagnosis,” *Neurocomputing*, vol. 407, pp. 121–135, 2020, doi: 10.1016/j.neucom.2020.04.045.

[23] R. Ye and Q. Dai, “Implementing transfer learning across different datasets for time series forecasting,” *Pattern Recognit.*, vol. 109, 2021, doi: 10.1016/j.patcog.2020.107617.

[24] Z. Li, B. Liu, and Y. Xiao, “Cluster and dynamic-TrAdaBoost-based transfer learning for text classification,” in *ICNC-FSKD 2017 - 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, 2018, pp. 2291–2295. doi: 10.1109/FSKD.2017.8393128.

[25] S. Mei, X. Liu, G. Zhang, and Q. Du, “Sensor-specific Transfer Learning for Hyperspectral Image Processing,” in *2019 10th International Workshop on the Analysis of Multitemporal Remote Sensing Images, MultiTemp 2019*, 2019. doi: 10.1109/Multi-Temp.2019.8866896.

[26] S. Hou, B. Dong, H. Wang, and G. Wu, “Inspection of surface defects on stay cables using a robot and transfer learning,” *Autom. Constr.*, vol. 119, 2020, doi: 10.1016/j.autcon.2020.103382.

[27] G. A. Atkinson, W. Zhang, M. F. Hansen, M. L. Holloway, and A. A. Napier, “Image segmentation of underfloor scenes using a mask regions convolutional neural network with two-stage transfer learning,” *Autom. Constr.*, vol. 113, 2020, doi: 10.1016/j.autcon.2020.103118.

[28] M. C. Bingol and O. Aydogmus, “Practical application of a safe human-robot interaction software,” *Ind. Rob.*, vol. 47, no. 3, pp. 359–368, 2020, doi: 10.1108/IR-09-2019-0180.

[29] M. C. Bingol and O. Aydogmus, “Performing predefined tasks using the human–robot interaction on speech recognition for an industrial robot,” *Eng. Appl. Artif. Intell.*, vol. 95, 2020, doi: 10.1016/j.engappai.2020.103903.

[30] M. C. Bingol and Ö. Aydoğmuş, “İnsan-Robot Etkileşiminde İnsan Güvenliği için Çok Kanallı İletişim Kullanarak Evrişimli Sinir Ağı Tabanlı Bir Yazılımının Geliştirilmesi ve Uygulaması,” *Fırat Üniversitesi Müh. Bil. Derg.*, vol. 31, no. 2, pp. 489–495, 2019, doi: 10.35234/fumbd.557590.

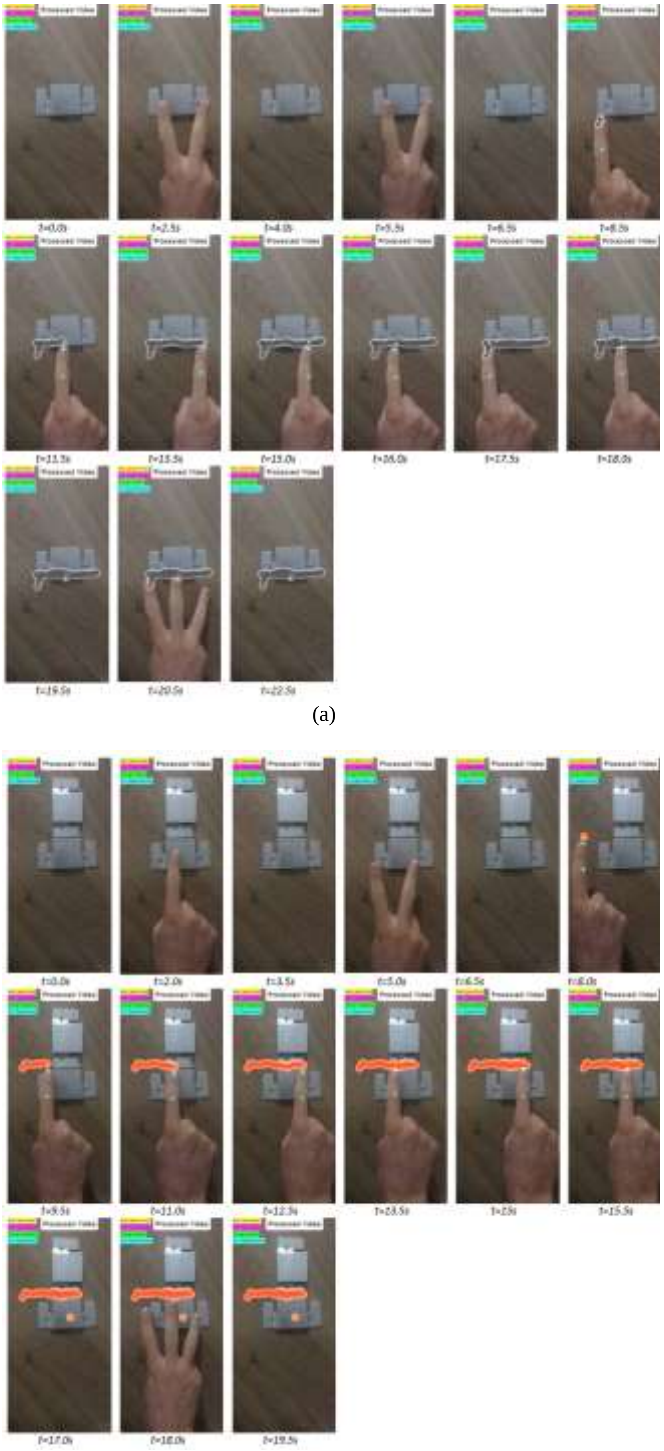
[31] OSRF, “SolidWorks to URDF Exporter,” 2020. http://wiki.ros.org/sw_urdf_exporter (accessed Oct. 17, 2020).

[32] “ManyCam Main Page.” https://manycam.com/ (accessed Oct. 17,

2020).
[33] TensorFlow, “Keras Applications,” 2020. https://keras.io/api/applications/%0A (accessed Oct. 15, 2020).
[34] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” 2015.
[35] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” 2014.
[36] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” 2016.
[37] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” 2016.
[38] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” 2019.
[39] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” 2018.
[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” 2015.
[41] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” 2016.
[42] M. C. Bingol and O. Aydogmus, “Sealing Process,” 2020. https://drive.google.com/file/d/1BbK_wv2Q76ItVLSbViC1Vb7YpxlN-tEa/view?usp=sharing (accessed Nov. 16, 2020).
[43] M. C. Bingol and O. Aydogmus, “Welding Process,” 2020. https://drive.google.com/file/d/1Y5jE8uPoiJMKLEHeG-wj1OqVRs1bO48-/view?usp=sharing (accessed Nov. 16, 2020).

APPENDICES

Appendix 1. Processing video images, (a) Sealing process, (b) Welding process



(a)

(b)

Balkan Journal of Electrical and Computer Engineering (BAJECE)
Volume 1 Number 1, September 2023

Vision Transformer Based Photo Capturing System

Abdulkadir Albayrak

1 Department of Computer Engineering, Dicle University, Diyarbakir, Turkey,(e-mail: abdulkadir.albayrak@dicle.edu.tr).

Manuscript received Feb 1, 2023; accepted August 18, 2023.

Abstract— **Portrait photo is one of the most crucial documents that many people need for official transactions in many public and private organizations. Despite the developing technologies and high resolution imaging devices, people need such photographer offices to fulfil their needs to take photos. In this study, a Photo Capturing System has been developed to provide infrastructure for web and mobile applications. After the system detects the person's face, facial orientation and facial expression, it automatically takes a photo and sends it to a graphical user interface developed for this purpose. Then, with the help of the user interface of the photo taken by the system, it is automatically printed out. The proposed study is a unique study that uses imaging technologies, deep learning and vision transformer algorithms, which are very popular image processing techniques in several years. Within the scope of the study, face detection and facial expression recognition are performed with a success rate of close to 100% and 95.52%, respectively. In the study, the performances of Vision Transformer algorithm is compared with the state of art algorithms in facial expression recognition.**

Index Terms— ***Deep learning, facial expression recognition, photo capturing system, single shot detection, vision transformer***

1. INTRODUCTION

At the present time, with the rapid development of mobile technologies, most of the persons have digital photo camera and can capture high-quality photos. However there is still a lack of applications that are useful for capturing portraits for passports and other legal documents. In this study, a sophisticated real time portrait capturing system that combines hand crafted image processing techniques with state of art deep learning approaches is proposed.

The proposed system involves automatic detection of frontal face, determining the face orientation through detected landmark points and facial expression analysis.

Face detection and determination of face orientation are two basic steps that should be performed for various computer vision applications. These tasks are also critical for the proposed portrait capturing system since they constitute a precondition for the subsequent facial expression analysis step. In order to develop a robust face detection system, Single-Shot-Multibox detector with ResNet-10 is used as a backbone architecture [1]. In the literature, face detection is generally focused on finding 68 points using the distinctive textural features of the face[2-3] Facial landmarks are found in order to localize eyes, nose, contour of the face and mouth. Landmark points are exploited for determining whether the eyes and mouth are open or closed. Histogram of Oriented G radients (HOG) and Support Vector Machine (SVM) are employed for selecting the images with opened eyes, while the proportion of the width and height of the mouth is calculated for determining weather the mouth is opened or closed. Frontal face images with open eyes and closed mouth are than processed for facial expression analysis.

Facial expression is one of the most effective channels of human communication, and therefore, automatic facial expression analysis systems can take place in various applications related to human-computer interaction. The success of the deep learning methods in modeling complex systems lead researchers to apply various deep learning approaches in this challenging task. Convolutional Neural Networks (CNN) based systems have proven their success in facial expression recognition problem [4]. However, facial expressions have high variability due to the nature of the human. This reality has demanded improvements in CNN-based systems, which require large amounts of training data to build a reliable model. Another drawback of the CNN based models is their relatively fragile structure against variant backgrounds and head poses [5]. Lozoya et.al aimed to improve generalization of their CNN based system by learning from mixed instances taken from different databases [6]. In [7], researchers employed CNN with Rectified Adam Optimizer in order to improve generalization.

Success of the Transformers in natural language processing pave the way for attempts to adapt transformers to computer vision problems. One of the key ideas of Transformer models is being pre-trained on a large corpus and fine-tuning on the target task with a smaller dataset [8]. Naseer et.al. compared CNN and Vision Transformer networks and stated that Vision Transformers are robust to occlusion and pose variant [9]. For all that reasons, in this study, a Vision Transformer based system is used for facial expression recognition. Proposed automatic portrait capturing system selects the neutral faces for further processing.

Selected proper portrait photographies (Frontal and neutral face images with opened eyes and closed mouth) are than post-processed for cleaning small speckles on the face and then the photographies are sent to the printer according to the preferences of the users.

The main contributions of the proposed system are given below:

- At the first time in the literature, a real time portrait capturing system that combines handcrafted image processing techniques with deep learning approaches is proposed.
- Performance of Vision Transformers in facial expression recognition task is evaluated.
- A comparison between the performance of Vision Transformer and state of art methods is done in facial expression recognition.

The rest of the paper is structured as follows: The proposed system is presented in Section 2. The experimental results are discussed in Section 3. The performance of the algorithms presented in this study is discussed in Section 4. Finally, the paper is concluded with Section 5.

2. MATERIALS AND METHODS

2.1 Performing Face Detection and Finding Face Orientation

First step of the this proposed system is to focus on face region in an image. Because one of the most basic conditions that must be met in passport photos or official documents is that the face should be detected and centered symmetrically. Most of the methods suggested in the literature try to find a total of 68 points belonging to the face including eyes, chin, nose and eyebrows. In this study, Single Shot Detection (SSD) method has been applied to detect face regions. The orientation of the face was tried to be calculated by using the positions of these points according to a line that passes through the points of nose vertically. Figure 1 shows a representation of the line dividing the face exactly in half from the vertical. The distance of this line to points 0 and 16 separately is calculated. The ratio of these distances to each other should be approximately 1. The acceptable range of face orientation is set to 0.9 and 1.1, but this range can be narrowed if this value is desired to be more precise. Equation 1 expresses the distance from point 0 to point 27 shown in Figure 1 at the nose level.

Equation 2 expresses the distance from point 16 to point 27 shown in Figure 1 at the nose level.

$$dist\_0\_27 = \sqrt{(x_0 - x_{27})^2 + (y_0 - y_{27})^2} \qquad (1)$$

Here x_0 and $y_0$represent the x and y coordinates of the point 0. x_{27} and y_{27} represent the x and y coordinate information of point 27. *dist_0_27* shows the distance between these two points.

Equation 2 expresses the distance between point 16 in Figure 1 and point 27 at nose level.

$$dist\_16\_27 = \sqrt{(x_{16} - x_{27})^2 + (y_{16} - y_{27})^2} \qquad (2)$$

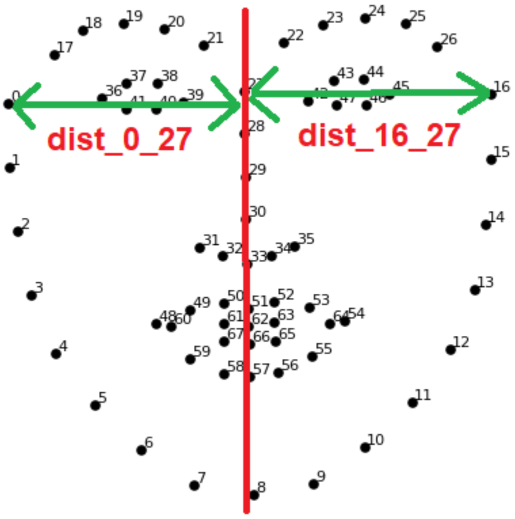


Figure 1. Sample image with 68 points used in face detection

Here, x_{16} and y_{16} represent the x and y coordinates of the point 16 on the front. x_{27} and y_{27} represent the x and y coordinates of the point 27 on the front. *dist_16_27* shows the distance between these two points.

$$dist = \frac{dist\_0\_27}{dist\_16\_27}, \qquad 0.9 \leq dist \leq 1.1 \qquad (3)$$

The ideal value of the dist value obtained in Equation 3 should be 1. In this study, the value range was chosen between 0.9 and 1.1. Thus, the system is able to take pictures in small value ranges without being bound by a very strict rule. Figure 2 shows the whole flowchart of the proposed system from turning on the camera to getting output from the system.

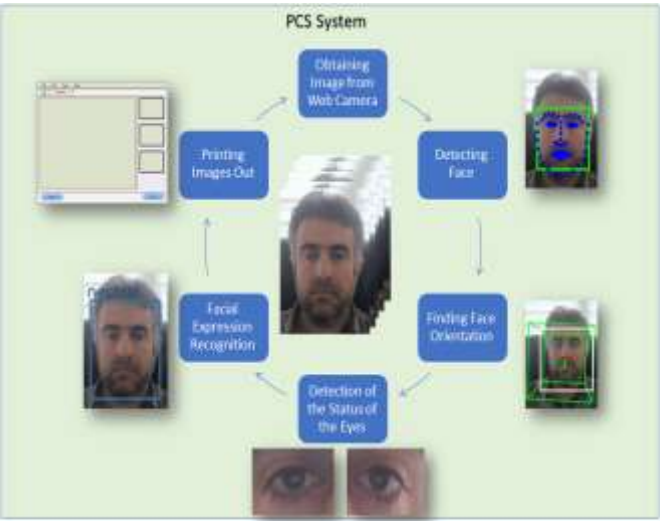

PCS System
Obtaining Image from Web Camera
Detecting Face
Finding Face Orientation
Detection of the Status of the Eyes
Facial Expression Recognition
Printing Images Out

Figure 2. Process steps followed in the Photo Capturing System(PCS) developed within the scope of the study

2.1.1 Single Shot Detection (SSD) Deep Learning Algorithm

SSD is a feed-forward convolutional network-based deep learning method used to detect objects in images. The SSD approach makes a score estimation of the proportions of the boxes surrounding the objects that are intended to be detected. This approach does not make an estimation about the whole image like the methods in the early studies of deep learning, but it is used to determine in which part of the image the object to be classified is located. SSD architecture consists of 3 parts (parts):

Base Convolutions: Networks such as VGG, ResNet, which are suggested for image classification, are the name given to the part as the base.

Auxiliary convolutions: It is the part that is placed as the backbone to obtain higher level features.

Prediction Convolutions: The attributes of the object to be detected are classified in this section. It is the part that makes predictions about the location and score of the object in the image.

The deep learning model applied for face detection within the scope of this study is the SSD model, which is trained with 300x300 image sizes and 140000 iterations. This SSD model in OpenCV's DNN library uses ResNET-10 architecture as backbone.

2.2 Detection of the Status of Eyes

One of the conditions that must be met in passport photos (or passport photos) is to have eyes open. No matter how accurately the eye lines are determined with facial landmark detection, there is no control such as whether the eyes are open or not. In order to carry out this process, the Histogram of Oriented Gradients (HOG) algorithm, which is one of the traditional image processing methods, was used. A total of 200 eye picture systems, 100 closed and 100 open, were trained with HOG.

2.2.1 Histogram of Oriented Gradients (HOG)

Histogram of Oriented Gradients was first proposed by Dalal and Triggs for pedestrian detection [10]. HOG is used to obtain shape-based features of the regions whose attributes are desired to be extracted. It has been applied in many different areas of computer vision, particularly pedestrian detection [11]. In the HOG algorithm, the orientation of all the pixels in the image is calculated and the focus is on determining the silhouette of the object (or region) that is desired to be distinguished. The basic processing steps applied in the HOG algorithm are as follows:

First, edge information is obtained by applying a sobel filter on the horizontal and vertical axis of the image. For this, the following formula referred in Equation 4 is used:

$$I_x = I * S_x, \qquad I_y = I * S_y \tag{4}$$

Here, I denotes the input image, S_x the vertically applied sobel filter and Sy the horizontally applied sobel filter. I_x and I_y show the output images obtained after applying sobel vertically and horizontally, respectively. The I_x and $I_y$images are then used in the formula below to calculate the magnitude values.

$$|G| = \sqrt{{I_x}^2 + {I_y}^2} \tag{5}$$

Here $|G|$ value is expressed as a gradient and is calculated using the square root value of the sum of the squares of the I_x and $I_y$values specified in the previous formula. Finally, the magnitude value is |G| are obtained by calculating the arctan of the values. As a result of these operations, the orientation of the object or region is calculated and the process of distinguishing it from other objects or regions is performed.

2.3 Facial Expression Recognition with Vision Transformer

Neutral facial expression is one of the conditions that must be handled in portrait or passport photos. The vision transformer algorithm, which is inspired by the transformer algorithm, which has been very popular in the field of natural language processing in recent years, has been applied in automatic facial expression detection.

2.3.1 Vision transformer (ViT)

Transformer is a deep learning model that adopts self-attention mechanism. It can be expressed as the calculation of the relationship of each word of the data given as input in the attention mechanism with all the other words. It is primarily used in the fields of natural language processing (NLP) tasks such as machine translation and text summarization which has sequential input data. However, Transformers do not necessarily process data sequentially like Long Short Term Memory(LSTM) and Recurrent Neural Network(RNN). Instead, the attention mechanism provides context for any position in the input data. Inspired by the Transformer scaling achievements in NLP, we attempted to apply a standard Transformer directly to images with the least possible modification. To do this, we split an image into patches and provide the linear embedding order of those patches as an input to a Transformer. Image patches are treated in the same way as tokens (words) in an NLP application. We train the model on image classification in a supervised manner. Figure 3 shows the processing steps of the vision transformer method used in the proposed system for facial expression recognition.

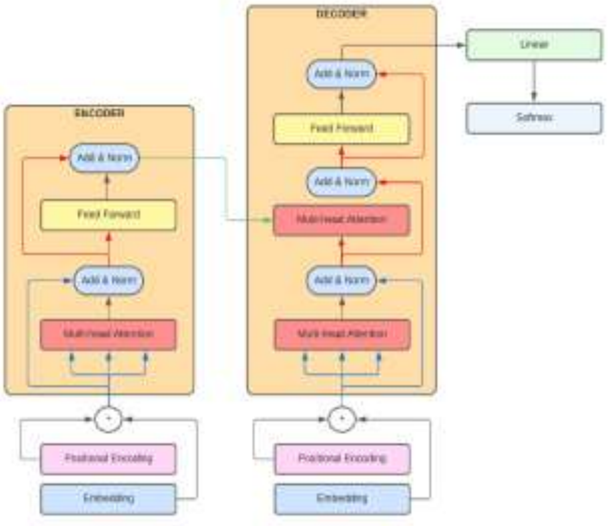


Figure 3. Vision Transformer

The classification header is implemented by an MLP with a hidden layer and a single linear layer of fine tuning at pre-training time. The MLP consists of two fully connected layers with an occasional GeLU nonlinear activation. The most effective part to highlight in the Transformer model is the attention mechanism. The attention mechanism looks at the input sequence and selects parts of the sequence that remain important at each step, preserving knowledge of which parts of the sequence are important. The attention mechanism instantly takes into account several other input data and assigns different weights to these inputs, providing prioritization. All encoder layers use an attention mechanism for each input that measures the fitness of all other inputs and retrieves the appropriate information to produce the output. Then, as a result of the attention mechanism, it takes the weights sent as output and the encoded string as input. These networks consist of repeated multi-headed attention blocks and feedforward layers. Multi-headed attention runs the processes in the attention mechanism in parallel and combines the results. Thus, it is ensured that different relationships are learned.

2.4 Graphical User Interface (GUI)

After the image processing steps were completed, a design that could be output to the printer was realized with the help of the user interface developed for the users. Face detection, face orientation, eye and mouth opening, facial expression recognition operations are performed in "CaptureFace" option which located in the top menu in GUI. When all the determined rules are fulfilled, 10 pictures are saved to a folder named for a specific user/customer. One of the captured image is given as input to the system using the GUI. Then, possible noises are cleaned by applying *3x3* median filter. Finally, after the user is asked whether he wants a digital copy, the interface page where the information is entered comes. On this page, the images is sent to the printer after the user fills all the contact information. Photos are printed from the printer by clicking the Confirm button. Figure 4 shows the graphical user interface (GUI) developed to proceed further processes and to print out the final images after billing.

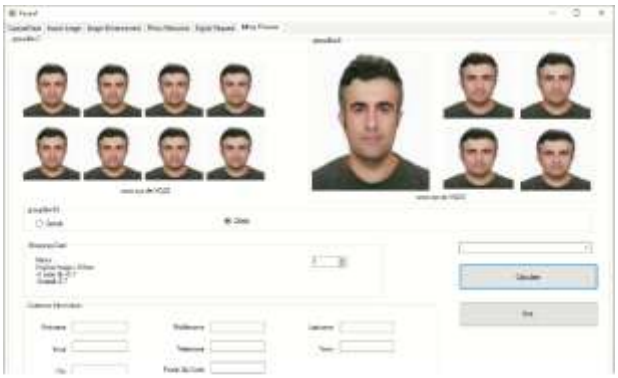
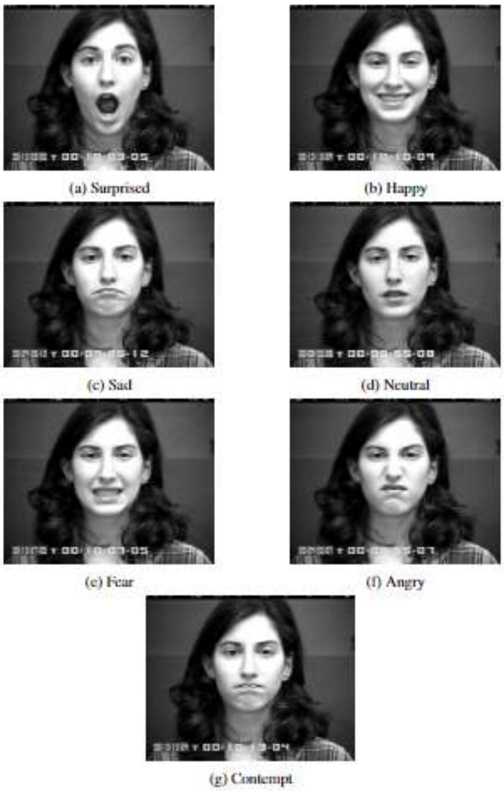


Figure 4. Graphical user interface (GUI) developed to proceed further processes and to print out the final image.

3. EXPERIMENTAL RESULTS

In this part of the study, face detection, recognition of facial expression, openness of the eyes and openness of the mouth were analyzed. While the publicly available facial expression recognition data set available in the literature was used within the scope of the study, the system performance was tried to be increased with a data set created within the scope of the study for eye opening/closed status.

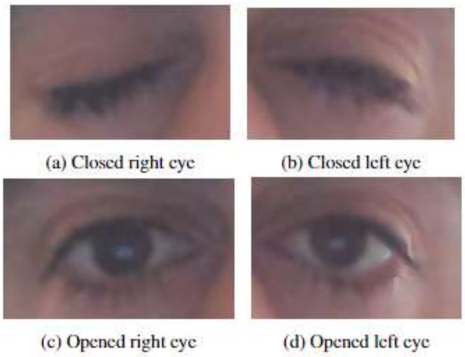
<u>CK+ data set for facial expression recognition</u>: The CK+(Cohn and Kanade) is an publicly available data set for facial expression recognition [12]. There are 7 classes in total in the data set: neutral, happy, disgusted, surprised, sad, angry and afraid. The data set includes 593 sequences from 123 individuals. These sequences begin with a neutral facial expression and end with the expression belonging to each class. Figure 5 shows sample facial expressions from the data set.



(a) Surprised (b) Happy (c) Sad (d) Neutral (e) Fear (f) Angry (g) Contempt

Figure 5. Sample images obtained from the CK+ dataset used for facial expression recognition within the scope of the study. The system is required to take pictures with neutral facial expressions.

<u>Eye Status (ES) Data set</u>: The points around the eyes obtained in the detection of facial regions are not sufficient to understand whether the eye is open or closed alone. In order to solve this problem, another data set was created within the scope of the study to evaluate the open/closed status of the eyes in the frames obtained from the camera (See Figure 6). A data set consisting of a total of 280 images, including 140 open-eye image sections and 140 closed-eye image sections, was created.



(a) Closed right eye (b) Closed left eye (c) Opened right eye (d) Opened left eye

Figure 6. (a), (b) represent the close eyes and (c), (d) represents the opened eye image crops obtained from the data sets.

Evaluation: In order for the proposed system to be used successfully in real time, face detection must be performed with high success in the first stage. For this, the SSD network, which is frequently used in object detection in the literature, is used. At this stage of the study, the facial regions in all of the randomly shot sample videos were detected with SSD. Since the ambient lighting must be very good in passport photos, it is seen that the facial regions are easily detected. In all the trials conducted within the scope of the study, it was observed that face detection was detected in all images. However, face detection may not be performed when the face area is rotated up to a certain angle. This situation is not directly related to the ambient effect.

After the face detection was performed, it was tried to decide whether the image to be obtained was appropriate by taking into account the orientation of the head. An image taken with a certain angle or tilt is not valid. Therefore, in the proposed system, it is necessary to guide the user in images that do not comply with Equations 1,2 and 3. Provided that the face area is at the level of the camera, the user needs to adjust the angle and tilt of the person. Otherwise, the system will not automatically take a picture. If the user abides by the determined rules, the orientation phase will be successfully passed for the system to take a picture along with other conditions.

After the orientation phase is passed without any problems, the stage of determining the appropriate image according to the opening/closing status of the eyes and mouth is started. 140 images belonging to each class in the dataset were separated as training and test sets with 5-fold cross-validation method. A total of 28 test images belonging to each class were classified with HOG method and SVM (rbf kernel) with a success of 93%. While the open eye success rate was 96%, the detection success of the closed eye was 85%. When the picture was taken, the condition of having an eye opening ratio above 95% was accepted as successful. Since the success rate was high enough at this stage, it was not necessary to try alternative methods in order to determine the openness / closure of the eye. It has been seen that the HOG algorithm performs well in detecting the eye opening/closure state. A model was not trained for the aperture and closure of the mouth, but the points obtained by facial landmark detection were used. Since these points for the eyes did not show a significant difference, the data set was created, but the landmark points were sufficient within the scope of the study, since the points changed more significantly according to the shape of the mouth.

Finally, the facial expression recognition phase was carried out. Since the image accepted by the official authorities must be neutral, if the facial expression is not neutral, the picture is not taken. The method used for facial expression detection is the ViT method. The achievements after applying the ViT model to the CK+ dataset are shown in TableXX. As seen in the table, the expression recognition success of the ViT network is 95.52%. In the literature, the successes obtained in the CK+ dataset using traditional image processing techniques are relatively low. It is seen that deep learning networks, which have been performing successfully in many fields recently, have carried this success over 90%. It has been observed that the ViT model applied within the scope of the study gives as successful results as other deep learning methods. When all these conditions are fulfilled, the system automatically takes 10 images and these images are sent to the printer with the help of the user interface prepared within the scope of the study.

Table 1. The classification success obtained with the ViT model applied to the CK dataset and its comparison with the results obtained in the literature

Method	**Accuracy(%)**
3D SIFT [13]	81.35
LBP-TOP [14]	88.99
ITBN [15]	86.3
CERT [16]	87.21
MCF [17]	89.4
MSR [18]	91.4
TMS [19]	91.89
STM [20]	91.13
AUDN [21]	93.70
BDBN [22]	**96.7**
CNN [23]	92.73
Proposed ViT	95.52

4. DISCUSSION

For the necessary processes in public or private institutions and organizations, taking a passport photo has an important place in real life. People often go to places specialized for this process and passport photos are taken with high resolution cameras. Today, many technological devices have cameras capable of taking high-resolution pictures. By processing the images obtained with these devices, it is possible to obtain passport images with the same quality and similar standards. In this study, an autonomous system that takes passport photos with people's own devices or systems that can be installed in public environments is designed. The system focuses on major aspects such as face detection, detection of facial regions, eye and mouth condition, and facial expression. Since deep learning methods perform quite successfully in face detection processes, they can be used in such systems. The process of finding the orientation of the face is similarly possible.It may not be possible to determine whether the eye is open or not by using the points located around the eye. By using the coordinates of the points around the eyes, the image sections with the eyes were cropped and the status of the eyes was determined more easily with the help of the model trained with shape based HOG algorithm. A similar situation could be considered for the mouth, but since it is not as solid as the eye, the opening and closure of the mouth could be determined. Facial expression is very important in the process of taking passport photos. The CK+ dataset, which is one of the frequently used facial expression recognition datasets in the literature, was used for model training in this study. Since the expression recognition performance with traditional image processing methods is somewhat limited compared to deep learning methods, deep learning methods can be preferred. In this study, as an alternative to deep learning methods, the ViT model, which is inspired by the transformer method, which has been very successful in natural language processing, has been used. The use of the convolution layer in training the data in deep learning methods considerably extends the training time.

The absence of a training phase in ViT models enables data to be modeled in a relatively faster training period. The biggest handicap of ViT models is that they need larger resources to achieve greater success. Since the GPU used in this study has 8 GB of ram, the success is limited to 95.22%. It is thought that the success can be increased by increasing the RAM resource. Because after the image is divided into sections, the calculation of the relationship of each section with each other and with all other sections directly contributes to the success of classification.

5. CONCLUSION

In this study, a system that takes passport photos is proposed for use in official documents or in public and private institutions. The system is designed to combine traditional image processing methods and deep learning methods and have the capacity to perform real-time processing. In the study, the transformer method, which has recently shown a very high performance in natural language processing, has been applied for facial expression recognition. When the results obtained were evaluated, the success rates of facial detection, eye opening rate and facial expression recognition were obtained as 100%, 96% and 95.22%, respectively. The system works in real time and can take dozens of pictures depending on certain rules. Users can then select one of the saved images and take a printout with the help of a developed user interface. The system has a very important place in that it combines traditional image processing methods, deep learning methods and ViT models and works in real time. In future studies, it can be developed as a system where people can perform similar operations using their own mobile devices and the resulting image can be sent to users via mail. For this, designing it with a logic that will work on users' mobile devices will improve this system a little more.

ACKNOWLEDGMENT

The authors state no conflicts of interest and have nothing to disclose.

REFERENCES

[1] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in European conference on computer vision. Springer, 2016, pp. 21–37.

[2] S. Ren, X. Cao, Y. Wei, and J. Sun, "Face alignment at 3000 fps via regressing local binary features," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1685–1692.

[3] O. Déniz, G. Bueno, J. Salido, and F. De la Torre, "Face recognition using histograms of oriented gradients," Pattern recognition letters, vol. 32, no. 12, pp. 1598–1603, 2011.

[4] I. M. Revina and W. S. Emmanuel, "A survey on human face expression recognition techniques," Journal of King Saud University-Computer and Information Sciences, vol. 33, no. 6, pp. 619–628, 2021.

[5] H. Li, M. Sui, F. Zhao, Z. Zha, and F. Wu, "Mvt: Mask vision transformer for facial expression recognition in the wild," arXiv preprint arXiv:2106.04520, 2021.

[6] S. M. González-Lozoya, J. de la Calleja, L. Pellegrin, H. J. Escalante, M. Medina, A. Benitez-Ruiz et al., "Recognition of facial expressions based on cnn features," Multimedia Tools and Applications, vol. 79, no. 19, pp. 13 987–14 007, 2020.

[7] D. O. Melinte and L. Vladareanu, "Facial expressions recognition for human–robot interaction using deep convolutional neural networks with rectified adam optimizer," Sensors, vol. 20, no. 8, p. 2393, 2020.

[8] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," ACM Computing Surveys (CSUR), 2021.

[9] M. M. Naseer, K. Ranasinghe, S. H. Khan, M. Hayat, F. Shahbaz Khan, and M.-H. Yang, "Intriguing properties of vision transformers," Advances in Neural Information Processing Systems, vol. 34, 2021.

[10] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in Proceedings of the 15th ACM international conference on Multimedia, 2007, pp. 357–360.

[11] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," IEEE transactions on pattern analysis and machine intelligence, vol. 29, no. 6, pp. 915–928, 2007.

[12] Z. Wang, S. Wang, and Q. Ji, "Capturing complex spatio-temporal relations among facial muscles for facial expression recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2013, pp. 3422–3429.

[13] G. Littlewort, J. Whitehill, T. Wu, I. Fasel, M. Frank, J. Movellan, and M. Bartlett, "The computer expression recognition toolbox (cert)," in 2011 IEEE International Conference on Automatic Face & Gesture Recognition (FG). IEEE, 2011, pp. 298–305.

[14] S. W. Chew, S. Lucey, P. Lucey, S. Sridharan, and J. F. Conn, "Improved facial expression recognition via uni-hyperplane classification," in 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012, pp. 2554–2561.

[15] R. Ptucha, G. Tsagkatakis, and A. Savakis, "Manifold based sparse representation for robust expression recognition without neutral subtraction," in 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). IEEE, 2011, pp. 2136–2143.

[16] S. Jain, C. Hu, and J. K. Aggarwal, "Facial expression recognition with temporal modeling of shapes," in 2011 IEEE international conference on computer vision workshops (ICCV workshops). IEEE, 2011, pp. 1642–1649.

[17] M. Liu, S. Shan, R. Wang, and X. Chen, "Learning expressionlets on spatio-temporal manifold for dynamic facial expression recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1749–1756.

[18] M. Liu, S. Li, S. Shan, and X. Chen, "Au-inspired deep networks for facial expression feature learning," Neurocomputing, vol. 159, pp. 126–136, 2015.

[19] P. Liu, S. Han, Z. Meng, and Y. Tong, "Facial expression recognition via a boosted deep belief network," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1805–1812.

[20] X. Sun, M. Lv, C. Quan, and F. Ren, "Improved facial expression recognition method based on roi deep convolutional neutral network," in 2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII). IEEE, 2017, pp. 256–261

Balkan Journal of Electrical and Computer Engineering (BAJECE)
Volume 1 Number 1, September 2023

African Vultures Optimization Algorithm-Based Selective Harmonic Elimination for Multi-level Inverter

Yasin Bektas

1 Department of Electrical-Electronic Engineering Aksaray University, Aksaray, Turkey,(e-mail: yasinbektas@aksaray.edu.tr)

Manuscript received Apr 10, 2023; accepted August 26, 2023.

***Abstract*—In recent years, multilevel inverters have gained significant attention due to their advantages, such as improved output waveform quality and reduced harmonic distortion. However, harmonics in multilevel inverter systems continue to be a persistent issue. Researchers have addressed this problem using the Selective Harmonic Elimination-Pulse Width Modulation (SHE-PWM) technique. However, the SHE equations formulated for eliminating or reducing the selected harmonics involve complex and computationally intensive calculations, encompassing nonlinear and transcendental equations. Various optimization techniques have been developed to tackle these intricate and demanding calculations. This article presents a new approach that utilizes the relatively newly developed African Vultures Optimization (AVO) algorithm to solve the SHE equations in multilevel inverters. The AVO-based SHE-PWM technique is tested on a three-phase cascade multilevel inverter (CHB-MLI) with 7, and 11 levels. The proposed algorithm demonstrates the ability to locate adequate solutions within the modulation index range of 0.1 to 1.0. It is shown that the modulation index range of 0.5 to 1.0 allows for the successful elimination of the selected harmonics and precise control of the fundamental voltage with an error of less than 0.5%.**

Index Terms—**African Vultures Optimization Algorithm, Multi-level inverter, Selective Harmonic Elimination, Optimization.**

I. Introduction

Multilevel inverters find applications in various fields, including renewable energy systems (such as solar and wind energy integration), electric vehicles, high-voltage direct current (HVDC) transmission systems, and medium-voltage motor drives [1-6]. Their ability to generate high-quality output waveforms makes them suitable for applications that require solid power quality. Alongside the circuit structures of multilevel inverters, control techniques play a crucial role. Among the most popular modulation techniques used for multilevel inverters are pulse width modulation (PWM), selective Harmonic elimination (SHE-PWM), and space vector PWM (SV-PWM) [7-9]. These techniques enhance the efficiency of the inverter while also lowering the amount of harmonic distortion that is present in the waveform that is produced. SHE-PWM stands out as the most effective due to its direct harmonic elimination capability. SHE-PWM is a modulation technique that selectively eliminates or reduces lower-order harmonics in the output waveform. When SHE-PWM is used, the output waveform is improved with smoother characteristics, and the total harmonic distortion (THD) value is reduced. This minimizes the effect of unwanted harmonic components and yields a cleaner output waveform.

Various optimization methods have been developed to address the computational complexity of SHE. These optimization techniques aim to find optimum switching angles that satisfy the desired harmonic elimination conditions while minimizing the computational effort. Genetic Algorithms (GA) [10], Particle Swarm Optimization (PSO) [11], Differential Evolution (DE) [12], Artificial Bee Colony (ABC) [13], Red Deer Algorithm (RDA) [14], and other metaheuristic algorithms have been applied in SHE optimization. These algorithms explore the solution space to find the best solutions, considering factors such as convergence speed, computational efficiency, robustness, and the ability to handle different modulation indices. Utilizing optimization methods in selective harmonic elimination enables the efficient implementation of multilevel inverters, achieves high-quality output waveforms, and reduces harmonic distortion. Researchers and engineers can adapt the performance of multilevel inverters to meet specific application requirements, optimize power conversion efficiency, and adhere to power quality standards by selecting an appropriate optimization algorithm. Ongoing research in this field aims to enhance the efficiency and accuracy of optimization algorithms and their application in real-time control systems, thus paving the way for the widespread adoption of multilevel inverters in various power electronics applications.

This study presents a new approach that utilizes the African Vultures Optimization (AVO) algorithm [15] to solve the SHE equations in multilevel inverters. The results show that the proposed algorithm works well at finding reasonable solutions within a specific range of modulation indices, allowing the elimination of certain harmonics and precise control of the fundamental voltage. This research contributes to the advancement of optimization techniques in multilevel inverters and offers promising possibilities for improving power quality in various applications.

II. Material And Method

A. *Cascaded Multilevel Inverters*

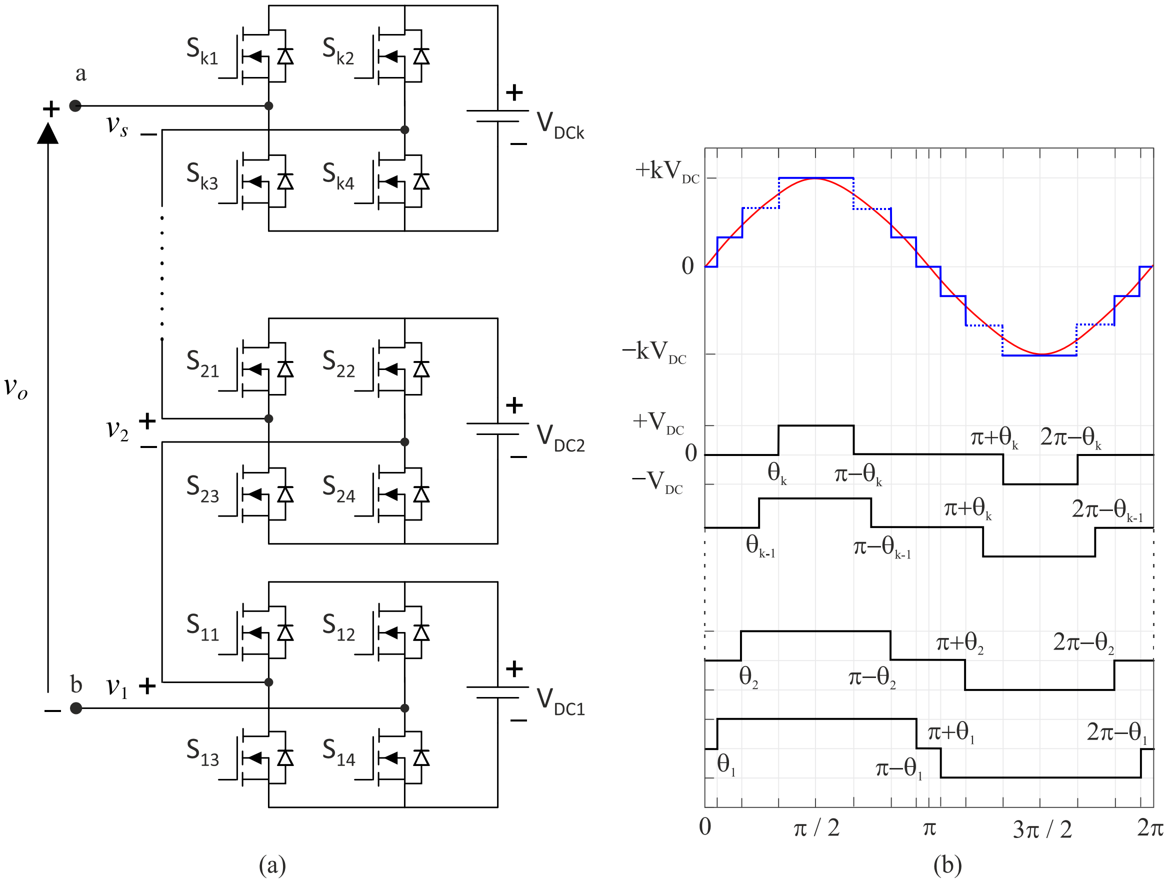
Multilevel inverters come in three main topologies: diode-clamped, capacitor-clamped, and cascaded. Among these structures, cascade multi-level inverter has different advantages compared to other inverter types. The cascaded H-bridge inverter (CHB) structure with distinct DC sources is the topology that needs the fewest number of components overall. Because of its modular design and flexible circuit configuration, it is well suited for use in applications requiring high voltage and high power. A cascaded H-bridge multilevel inverter, also known as a CHB-MLI, is created by joining a number of single-phase H-bridge inverters in a series configuration, as seen in Fig. 1(a) for an N-level inverter. N=2m+1 describes how many different levels of output phase voltage the CHB-MLI can produce, where m refers to the number of H-bridges used in each phase. Each H-bridge is fed by its own individual DC source. The three distinct voltage levels of $+V_{DC}$, 0 V_{DC}, and $-V_{DC}$ may be achieved by choosing various combinations of the switches labelled S_1, S_2, S_3, and S_4 located inside each H-bridge. In order to make the synthesized AC voltage waveform identical to the sum of all the voltages produced by the cascaded H-bridge cells, the outputs of the H-bridge switches are connected in series with one another.

Depending on the application, a three-phase circuit structure is obtained by either star or delta connection. This study uses three-phase circuit structures with 7, and 11 levels. For seven levels, three H-bridge structures are required for each phase, while four H-bridge structures are needed for nine levels, and five H-bridge structures are required for eleven levels.

B. *Mathematical Model of SHE-PWM Technique*

Selective Harmonic Elimination (SHE) is a PWM (pulse width modulation) technique commonly used at low frequencies. Its purpose is to control the output voltage while adjusting the fundamental harmonic to the desired value and eliminating selected harmonics to achieve a sinusoidal AC output voltage waveform. After applying SHE-PWM, the remaining insignificant higher-order harmonics can be eliminated using a small passive filter. In the SHE-PWM technique, a set of nonlinear equations, usually denoted by m, is solved to find the optimum switching angles for different modulation indices.

One of the equations equals the fundamental harmonic value, while the others are set to zero to eliminate the selected harmonics. Subsequently, the switching angles that satisfy these equations are calculated using appropriate methods. The nonlinear harmonic equations required to obtain the optimal switching angles can be expressed using the Fourier series expansion of the output voltage. The expression for the output voltage, which includes all harmonic components, can be defined as shown in Eq. (1).



(a) (b)

Fig.1. Single-phase N-level CHB-MLI (a) circuit structure (b) output voltage waveform

$$V_{ab}(\omega t) = \sum_{n=1,3,5,7,\dots}^{\infty} \frac{4V_{DC}}{n.\pi} * [\cos(n\theta_1) + \cos(n\theta_2) + \cdots + \cos(n\theta_{k-1}) + \cos(n\theta_k)] * [\sin(nwt)] \quad (1)$$

where, V_{DC} is the input voltage for each H-bridge inverter. θ_1, θ_2, ..., and θ_k are the switching angles, and due to quarter-wave symmetry, the switching angles should satisfy the condition in Eq.(2).

$$0 \le \theta_1 < \theta_2 < \dots < \theta_{k-1} < \theta_k \le \frac{\pi}{2} \quad (2)$$

k represents the number of switching angles, and n represents the degree of the harmonic. For the 7-level, and 11-level inverters, the required number of discrete DC sources is three, four, and five, respectively. In a balanced three-phase system, there are no harmonics that are multiples of three, so the harmonics that are multiples of three are not considered. Typically, in the context of *k* switching angles, a single switching angle is employed to regulate the intended fundamental output voltage V_1, while the remaining (*k*-1) switching angles are utilised to elimination or diminish the low-order harmonics (*k*-1). The Eq.(1) provides the formula for the fundamental output voltage V1, expressed in relation to the switching angles.

$$V_1 = \frac{4V_{DA}}{\pi}\begin{pmatrix}\cos(\theta_1) + \cos(\theta_2) + \\ \cdots + \cos(\theta_{k-1}) + \cos(\theta_k)\end{pmatrix} \quad (3)$$

The modulation index controls the fundamental voltage in the SHE technique. M, can be defined as the ratio of the peak value (V1p) of the desired base voltage given in (4) to the total DC input voltage [16].

$$M = \frac{V_{1p}}{kV_{DA}} \quad (4)$$

The determination of the modulation index and switching angles that result in the generation of an AC waveform with minimal total harmonic distortion (THD) necessitates the resolution of k transcendental nonlinear equations, commonly referred to as SHE equations, that describe the chosen harmonics.

The seven-level inverter utilizes three h-bridge structures. The SHE equations for the 7-level inverter are as follows:

$$\begin{aligned} V_1 &= \cos(\theta_1) + \cos(\theta_2) + \cos(\theta_3) = Mk\pi/4 \\ V_5 &= \cos(5\theta_1) + \cos(5\theta_2) + \cos(5\theta_3) = 0 \\ V_7 &= \cos(7\theta_1) + \cos(7\theta_2) + \cos(7\theta_3) = 0 \end{aligned} \quad (5)$$

The eleven-level inverter utilizes five h-bridge structures. The SHE equations for the 11-level inverter are as follows:

$$\begin{aligned}
V_1 &= \cos(\theta_1) + \cos(\theta_2) + \ldots + \cos(\theta_5) = Mk\,\pi/4 \\
V_5 &= \cos(5\theta_1) + \cos(5\theta_2) + \ldots + \cos(5\theta_5) = 0 \\
V_7 &= \cos(7\theta_1) + \cos(7\theta_2) + \ldots + \cos(7\theta_5) = 0 \\
V_{11} &= \cos(11\theta_1) + \cos(11\theta_2) + \ldots + \cos(11\theta_5) = 0 \\
V_{13} &= \cos(13\theta_1) + \cos(13\theta_2) + \ldots + \cos(13\theta_5) = 0
\end{aligned} \tag{6}$$

For the correct solution, the switching angles for all inverter states must satisfy the condition $0 \leq \theta_1 < \theta_2 < \ldots \leq \theta_k$. In this study, two different total harmonic distortion values will be calculated. The first is %THD, and the other is % THDe. The %THD limit value is usually infinite but will be considered up to the 50th harmonic. THDe represents the total value of the harmonics to be eliminated. When calculating the THDe [16] value, the maximum harmonic value to be eliminated is considered. The maximum harmonic value for the seven-level is 7th, 11th , and 13thfor the eleven-level.

$$\%THD = \frac{\sqrt{V_5^2 + V_7^2 + V_{11}^2 + \cdots V_{49}^2}}{|V_1|} \tag{7}$$

$$\%THD_e = \frac{\sqrt{V_5^2 + V_7^2 + \ldots}}{|V_1|} \tag{8}$$

C. *African Vultures Optimization Algorithm*

African Vulture Optimization (AVO) is an optimization algorithm developed by drawing inspiration from the feeding behavior of vultures in natural habitats. Vultures efficiently locate their prey over a wide area through their social interactions and observational abilities. Mimicking these natural strategies, the AVO algorithm is used to solve complex optimization problems. It employs a population-based approach and utilizes a set of candidate solutions (individuals) to solve the problem. The algorithm moves each individual through steps that imitate the vultures' prey-finding behavior. Individuals search to explore the potential solution space and move towards better solutions. This process is combined with a competitive selection mechanism to determine the best solutions among individuals.

The AVO algorithm has demonstrated its effectiveness in solving various optimization problems, as evidenced by several studies [15], [17-25]. It has been successfully applied to complex real-world problems inspired by nature, as well as function optimization, data mining, machine learning, artificial neural networks, and other domains [15]. AVO offers several key advantages, including computational efficiency, the ability to perform global searches, scalability, and high solution accuracy. This section provides a summary of the general steps of the African Vultures Optimization (AVO) algorithm. For more comprehensive details and in-depth explanations, readers are encouraged to refer to [15]. This optimization is capable of resolving major optimization issues and has a low computational complexity of O(P X (M + Mb). Here, 'P' stands for population size, 'M' for the maximum number of iterations, and 'b' for the dimensions of the problem.

The behavior of vultures in nature while foraging is depicted in Fig. 2. When many vultures congregate around one food source, it can cause severe conflicts over food acquisition. At such times, physically powerful vultures prefer not to share food with other vultures, as shown in Fig. 2(a). Vulture rotational motion is modeled using spiral motion in Fig. 2(b). The distance between the vulture and one of the two best vultures is first calculated in this method. Moreover, a spiral between the vulture and one of the best vultures is created. As seen in Fig. 2(c), when vultures are hungry and have low energy, vulture gathers together in search of food. Vultures compete in this pursuit. Vultures are aggressive in this search.

In general, the steps of the algorithm are as follows.

Step 1.

Create the initial population by considering the fitness function.

Step 2.

Choose the two best solutions as the best vultures of the first group and the second-best solution as the best vultures of the second group.

Step 3.

Move the other vultures to the best vultures using equations 9 and 10.

$$E(i) = \begin{cases} \text{Vulture}_{Best1} & if \;\; k_i = \beta_1 \\ \text{Vulture}_{Best2} & if \;\; k_i = \beta_2 \end{cases} \tag{9}$$

In Eq. (9), $\text{Vulture}_{\text{Best1}}$ represents the position vector of the best vulture in the first group in the current iteration, and $\text{Vulture}_{\text{Best2}}$ represents the position vector of the best vulture in the second group. One of the top vultures in the current iteration is represented by the position vector E(i). This vector is selected based on the given selection operator in Eq. (10). β_1 and β_2 are parameters measured before the search process. The values of these parameters range between zero and one, and the sum of both parameters is equal to one.

$$k_i = \frac{F_i}{\sum_{i=1}^{n} F_i} \tag{10}$$

Step 4.

Use equations (11) and (12) to search for food.

$$F = (2 \times rand_1 + 1) \times m_1 \times \left(1 - \frac{\text{it}_i}{\text{max}_{\text{it}}}\right) + t \tag{11}$$

$$t = m_2 \times \left(\sin^{m_3}\left(\frac{\pi}{2} \times \frac{\text{it}_i}{\text{max}_{\text{it}}}\right) + \cos\left(\frac{\pi}{2} \times \frac{\text{it}_i}{\text{max}_{\text{it}}}\right) - 1\right) \tag{12}$$

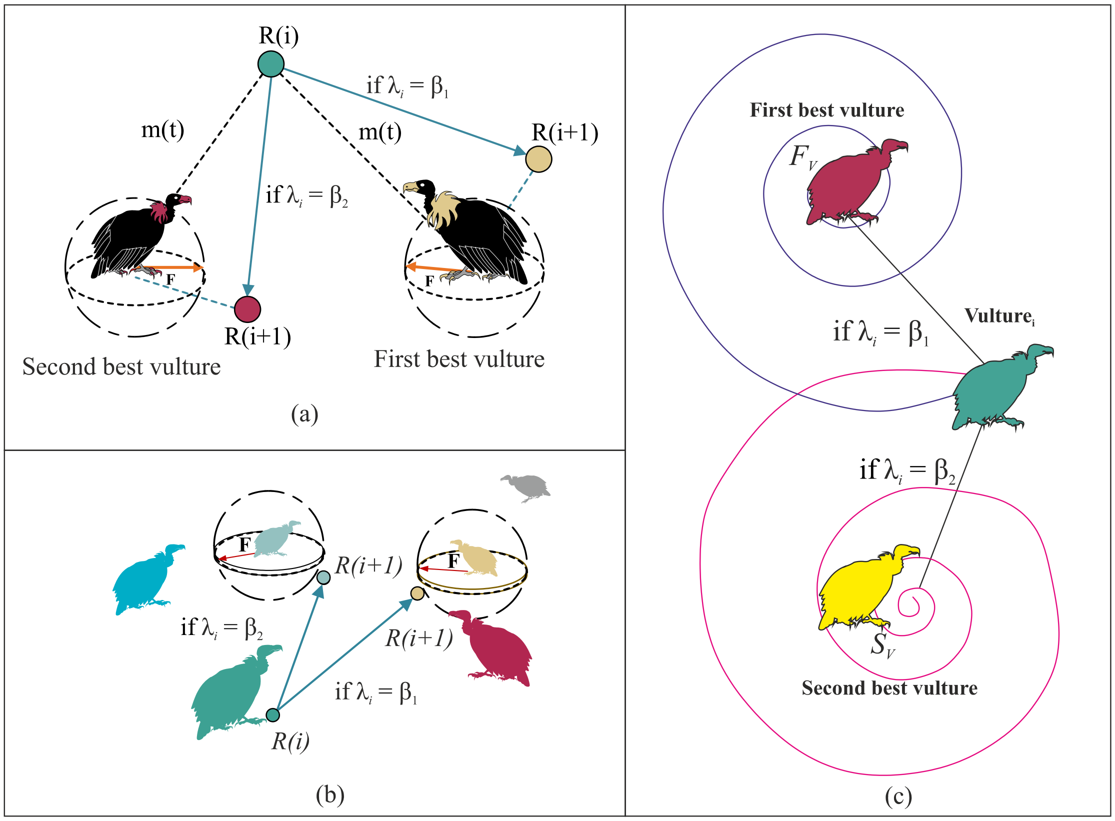


Fig.2. Natural behavior of vultures while foraging a) competition for food, b) rotating flight of vultures, c) aggressive competition for Food.

The F value given in Eq. (11) shows the saturation rate of vultures. it_i represents the valid number of iterations, max_{it} represents the maximum number of iterations. The rand1, m_1, m_2 parameters take random values in the ranges [0,1], [-1,1] and [2,-2], respectively. The m_3 parameter is a constant number. Increasing this value increases the probability of entering the exploration phase towards the maximum iteration.

|F| If the value is greater than or equal to 1, the exploration phase begins, and the vultures forage for the best vulture location at random locations. In order to benefit from different exploration strategies, the k_1, and $rand_{k1}$ parameters, which take random values in the [0,1] range, are compared. Eqs. 13 and 14 are used if k_1 is greater than or equal to randk1. If k1 is less than randk1, Eq.15 is used.

$$K(i+1) = E(i) - D(i) \times F \tag{13}$$

$$D(i) = | 2 \times rand_2 \times E(i) - K(i) | \tag{14}$$

$$K(i+1) = E(i) - F + rand_3 \times \left((s_2 - s_1) \times rand_d + s_1 \right) \tag{15}$$

In equation, K(i) refers to the vulture's current position vector, and K(i+1) is the position vector in the following iteration. The lower and upper limits of the search space are represented by s_1 and s_2, respectively. The parameters $rand_2$, $rand_3$, and $rand_4$ are assigned a random value between 0 and 1.

$$K(i+1) = D(i) \times \left(F + rand_4 \right) - m(t) \tag{16}$$

$$m(t) = E(i) - K(i) \tag{17}$$

$$K(i+1) = E(i) - \left(S_1 + S_2 \right) \tag{18}$$

$$\begin{aligned} S_1 &= E(i) \times \left(\frac{rand_5 \times K(i)}{2\pi} \right) \times \cos(K(i)) \\ S_2 &= E(i) \times \left(\frac{rand_6 \times K(i)}{2\pi} \right) \times \sin(K(i)) \end{aligned} \tag{19}$$

|F| when the value is less than 0.5, the k_3 and $rand_{k3}$ parameters are compared. If k_3 is equal to or greater than $rand_{k3}$, when using Eq.20 and Eq.21; If k_3 is less than $rand_{k3}$, 21 and 22 are used in the equation. Thus, the accumulation and aggressive bickering of vultures around the food source is modeled.

$$K(i+1) = \frac{B_1 + B_2}{2} \tag{20}$$

$$\begin{aligned} B_1 &= \text{Vulture}_{Best1}(i) - \frac{\text{Vulture}_{Best1}(i) \times K(i)}{\text{Vulture}_{Best1}(i) - K(i)^2} \times F \\ B_2 &= \text{Vulture}_{Best2}(i) - \frac{\text{Vulture}_{Best2}(i) \times K(i)}{\text{Vulture}_{Best2}(i) - K(i)^2} \times F \end{aligned} \tag{21}$$

$$K(i+1) = E(i) - | d(t) | \times F \times Levy(d) \tag{22}$$

$$L(z) = 0.01 \times \frac{m_4 \times \sigma}{|m_5|^{\frac{1}{\lambda}}},$$

$$\sigma = \left(\frac{\Gamma(1+\lambda) \times \sin\left(\frac{\pi\lambda}{2}\right)}{\Gamma(1+\lambda) \times \lambda \times 2^{\left(\frac{\lambda-1}{2}\right)}} \right)^{\frac{1}{\lambda}} \qquad (23)$$

In Eq. (23) λ is a constant number, parameters m_4 and m_5 take random values between 0 and 1. $\Gamma(z) = (z-1)!$. The flow chart of AVOA is given in Fig. 3.

D. Application of AVO Algorithm to SHE-PMW equations

The AVO algorithm implemented in MATLAB was utilized to solve the SHE equations (5), and (6) for the targeted harmonics in the 7, and 11-level inverters, respectively. Three scenarios were considered, with a population size of 100 and 100 iterations. The solutions were obtained by incrementing the modulation index M from 0 to 1 in steps of 0.01. The calculations were performed on a personal computer equipped with an Intel(R) Core (TM) i7-10870H CPU @ 2.20GHz, 16.0 GB RAM, and a GeForce RTX 2060 NVIDIA graphics card. At each step, the obtained solution was evaluated using a fitness function. The objective was to determine the switching angles that either eliminate or reduce the selected low-order harmonics to an acceptable level while achieving the desired voltage value. Eq. (24) defines the fitness function used for each set of solutions. The goal of this optimization process was to find the optimal set of switching angles that minimizes the distortion caused by the harmonics, ensuring the output voltage closely matches the desired waveform. The AVO algorithm, with its population-based approach and imitation of vulture behavior, efficiently searched the solution space to find the best configuration of switching angles for each modulation index.

$$f = \min_{\theta_i} \left\{ \left| V_{ref} - V_{1p} \right| + \left(\frac{4V_{DA}}{n\pi} . \left(\sum_{i=2}^{k} \cos(n\theta_i) \right) \right)^2 \right\} = 0 \qquad (24)$$

In Eq. (24) V_{ref} represents the maximum value of the desired base voltage, while V_{1p} represents the maximum value of the base voltage obtained at the inverter output applying the calculated switching angles to the inverter. The fundamental frequency of the fundamental voltage is 50Hz. Total source voltage values are selected to be a maximum of 311 volts. For the seven-level inverter, all sources are 311/3 volts, and all sources for the eleven-level inverter are 311/5 volts. An RL load was used as the load for both seven-level, and eleven-level CHB-MLI. The value of the resistance R is 10 ohms, and the value of the inductance L is 5 mH. The load connected to the inverter output affects the harmonic formation [26]. Here, harmonic analysis has been made only for RL load. The simulation results are explained in the next section.

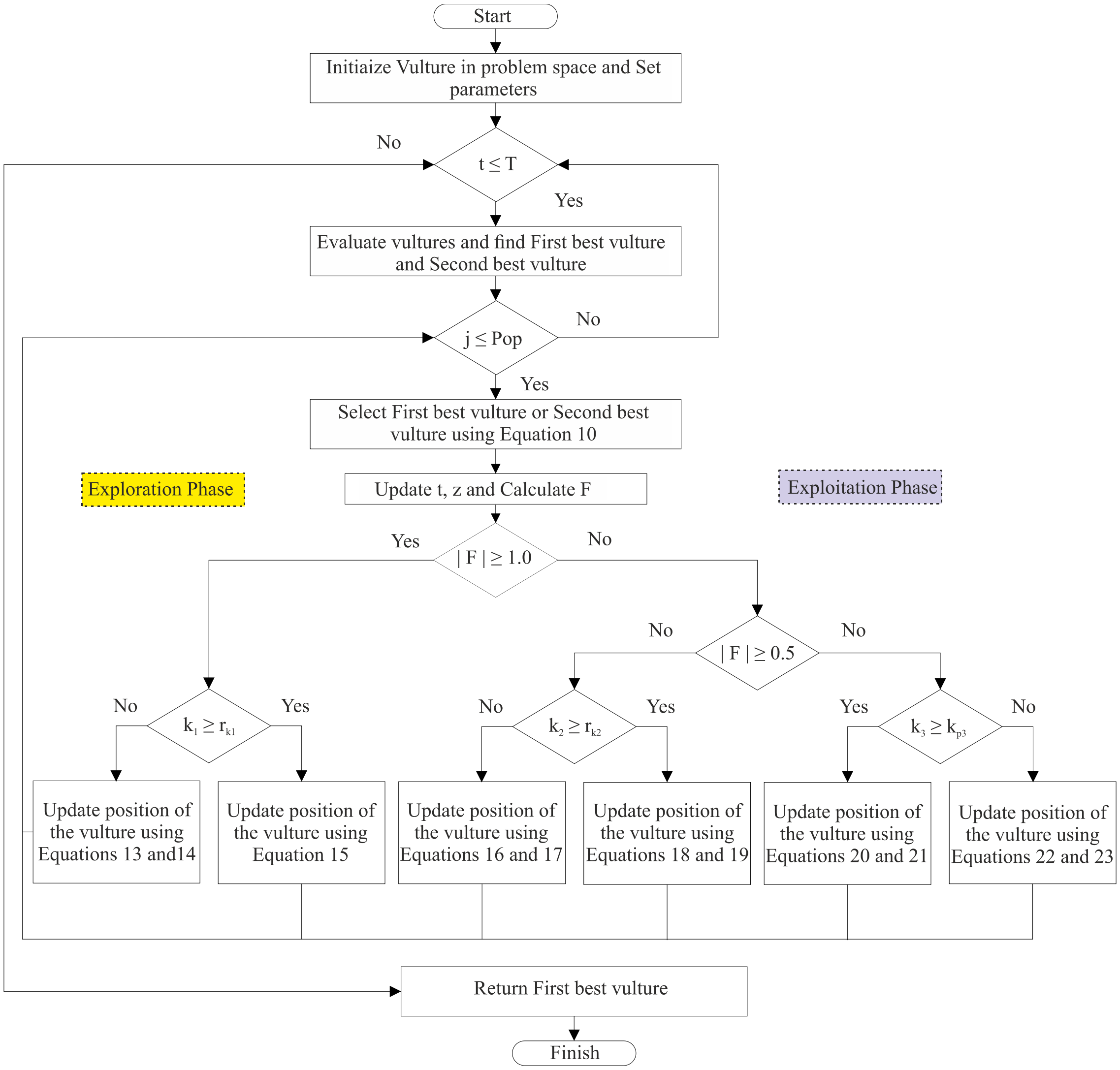

Start
Initiaize Vulture in problem space and Set parameters
No
$t \le T$
Yes
Evaluate vultures and find First best vulture and Second best vulture
No
$j \le Pop$
Yes
Select First best vulture or Second best vulture using Equation 10
Exploration Phase
Update t, z and Calculate F
Exploitation Phase
Yes
$|F| \ge 1.0$
No
No
$|F| \ge 0.5$
No
No
$k_1 \ge r_{k1}$
Yes
No
$k_2 \ge r_{k2}$
Yes
Yes
$k_3 \ge k_{p3}$
No
Update position of the vulture using Equations 13 and14
Update position of the vulture using Equation 15
Update position of the vulture using Equations 16 and 17
Update position of the vulture using Equations 18 and 19
Update position of the vulture using Equations 20 and 21
Update position of the vulture using Equations 22 and 23
Return First best vulture
Finish

Fig.3. AVOA [15]

III. Results And Discussion

To assess the effectiveness of the AVOA optimization in solving the SHE-PWM equations, the results obtained from applying the AVOA algorithm in the MATLAB Simulink environment for the 7-, and 11-level inverters are discussed below.

TABLE I
PARAMETER SETTING OF AVOA

Parameter	Value
β_1	0.8
β_2	0.2
w	2.5
k_1	0.6
k_2	0.4
k_3	0.6

Table I displays the parameters utilized for AVOA at levels 7 and 11. The w operator mentioned in Table II is a parameter that influences the extent of disruption during the exploration and exploitation phases.

A. *Simulation results for seven-level MLI*

The graph in Fig. 4(a) depicts the fitness function values for each set of switching angles across the range of modulation index values Fig. 4(b) displays the switching angles determined for the corresponding modulation index values, providing insight into the optimal configuration for the 7-level MLI. In Fig. 4(c), the plots showcase the THD, THDe, and magnitudes of the 5th and 7th harmonics. These metrics are evaluated for modulation index values, highlighting the system's harmonic performance. Notably, the graph indicates that solutions within the M range of 0.1 to 0.4 fail to meet the IEEE-519 standard [27], while acceptable solutions are found in the M range of 0.5 to 1.0.

The switching angles calculated by AVO versus different modulation indices for 7-level MLI are shown in Table II. The switching angles calculated in Table II were applied to the 7-

level inverter in the Matlab environment, and the simulation output values are shown in Table III.

As shown in Table III, the AVO algorithm has successfully found suitable solutions for the 7-level inverter within the modulation index range of 0.1 to 1.0. The desired fundamental voltage is controlled with an error of less than 0.5% compared to the reference value. The 0.5 to 1.0 index values modulation range effectively suppresses the selected harmonics. Examining this range, both the THDe value and the magnitudes of the 5th and 7th harmonics are below 0.09%. Figure 5 illustrates the load voltage waveforms for modulation indices of 0.4, 0.6, and 1.0. The THD and THDe analysis of these waveforms is presented in Figure 6 and Figure 7, respectively.

For a modulation index of M=0.4, the voltage waveform of the load exhibits a THD value of 16.94%. The maximum value of the fundamental voltage, Van(max), is measured as 124.2V, and the rms value, Van(rms), is calculated as 87.83V (Figure 6(a)). For a modulation index of M=0.6, the THD of the load voltage waveform is determined as 12.41%. The maximum value of the fundamental voltage, Van(max), is found to be 186.2V, and the rms value, Van(rms), is calculated as 131.7V (Figure 6(b)). Finally, for a modulation index of M=1.0, the THD of the load voltage waveform is reduced to 7.83%. The maximum value of the fundamental voltage, Van(max), is measured as 311.1V, and the rms value, Van(rms), is determined as 220.1V (Figure 6(c)).

Figure 7 illustrates the extent to which the selected harmonics are eliminated. For M=0.4 modulation index, the calculated values are THDe=%6.35, h5=%3.99 (V5=4.95V), and h7=%3.99 (V7=6.14V). For M=0.6 modulation index, the calculated values are THDe=%0.09, h5=%0.02 (V5=0.03V), and h7=%0.15 (V7=0.08V). Lastly, for M=1.0 modulation index, the calculated values are THDe=%0.04, h5=%0.03 (V5=0.01V), and h7=%0.01 (V7=0.03V).

The analysis indicates that the AVO algorithm precisely controls the fundamental voltage and effectively suppresses the selected harmonics within the modulation index range of 0.5 to 1.0. The obtained results demonstrate the capability of the AVO algorithm to achieve desired voltage control and harmonic suppression for the 7-level inverter.

B. *Simulation results for eleven-level MLI*

The graph in Fig. 8(a) depicts the fitness function values for each set of switching angles across the range of modulation index values Fig. 8(b) displays the switching angles determined for the corresponding modulation index values, providing insight into the optimal configuration for the 11-level MLI. In Fig. 8(c), the plots showcase the THD, THDe, and magnitudes of the 5th, 7th,11th and 13th harmonics. These metrics are evaluated for modulation index values, highlighting the system's harmonic performance. Notably, the graph indicates that solutions within the M range of 0.1 to 0.4 fail to meet the IEEE-519 standard, while acceptable solutions are found in the M range of 0.5 to 1.0.

The switching angles calculated by AVO versus different modulation indices for 11-level MLI are shown in Table IV. The switching angles calculated in Table IV were applied to the 11-level inverter in the Matlab environment, and the simulation output values are shown in Table V.

TABLE II
SWITCHING ANGLES CALCULATED WITH AVO (FOR 7 LEVELS)

Modulation Index	**Switching Angles (Radians)**		
M	**θ_1**	**θ_2**	**θ_3**
0.1	1.3329	1.5708	1.5708
0.2	1.0801	1.5708	1.5708
0.3	0.8907	1.4928	1.5708
0.4	0.7787	1.3380	1.5708
0.5	0.7115	1.1488	1.5597
0.6	0.6882	1.0224	1.4504
0.7	0.6691	0.9414	1.2908
0.8	0.5102	0.9503	1.1254
0.9	0.3056	0.7512	1.1196
1.0	0.2029	0.5370	1.0181

TABLE III
SIMULATION RESULTS (FOR 7 LEVELS)

M	**Vref(peak)**	**V_{ref}(rms)**	**V(rms)**	**Error (%)**	**THD (%)**	**THD_e (%)**	**h_5**	**h_7**
0.10	31.1	22	21.96	**0.18**	109.14	99.42	79.03	60.32
0.20	62.2	44	43.95	**0.11**	37.77	28.44	27.09	8.68
0.30	93.3	66	65.83	**0.26**	32.95	10.37	3.65	9.71
0.40	124.4	88	87.83	**0.19**	16.94	6.35	3.95	4.95
0.50	155.5	110	109.8	**0.18**	17.44	**0.06**	**0.01**	**0.06**
0.60	186.6	132	131.7	**0.23**	12.41	**0.09**	**0.02**	**0.08**
0.70	217.7	154	153.6	**0.26**	12.26	**0.07**	**0.02**	**0.06**
0.80	248.8	176	175.6	**0.23**	10.72	**0.04**	**0.01**	**0.00**
0.90	279.9	198	197.4	**0.30**	11.81	**0.03**	**0.00**	**0.01**
1.00	311.0	220	220.1	**0.05**	7.84	**0.04**	**0.01**	**0.03**

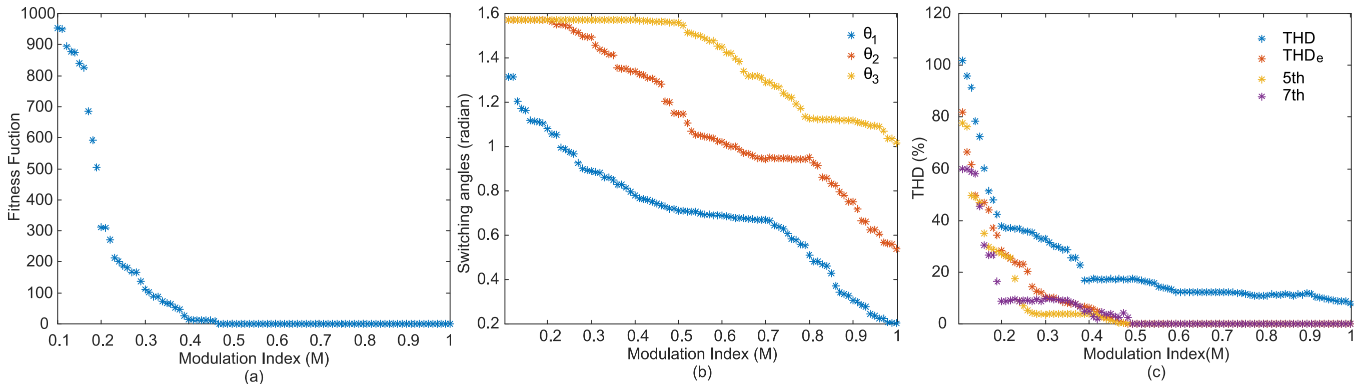


Fig.4. Analysis of the 7-level MLI inverter in terms of (a) fitness function, (b) switching angles, and (c) THD, THDe, 5th , and 7th harmonic characteristics as a function of M.

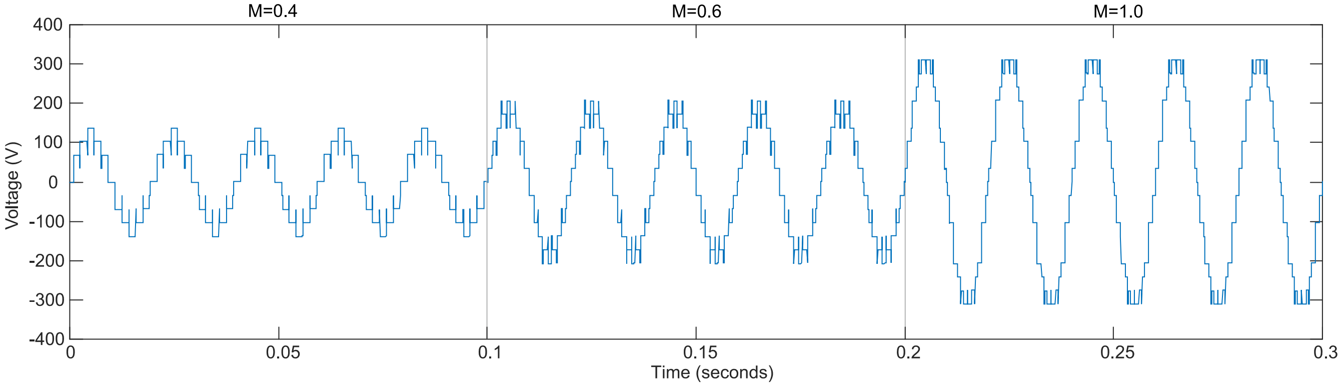


Fig.5. Load voltage waveforms for different modulation indices (7-level): (a) M=0.4, (b) M=0.6, (c) M=1.0

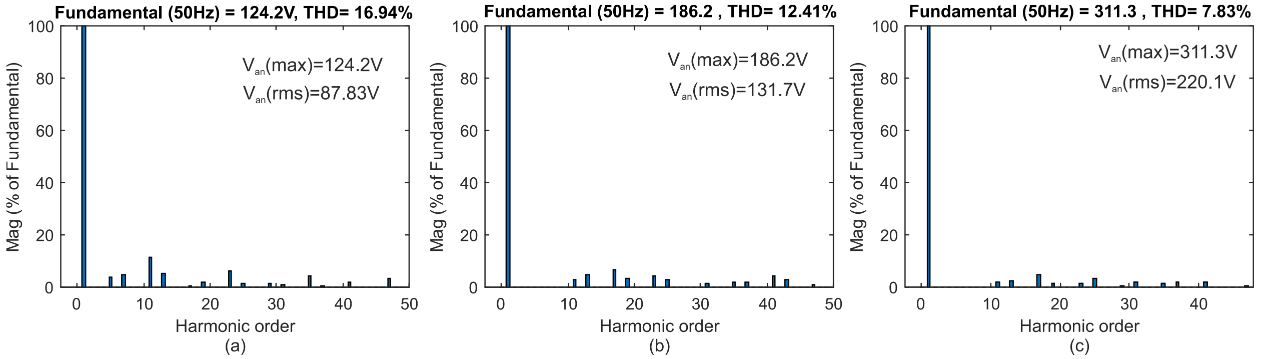


Fig.6. THD values for different modulation indices (7-level) (a) M=0.4, (b) M=0.6, (c) M=1.0

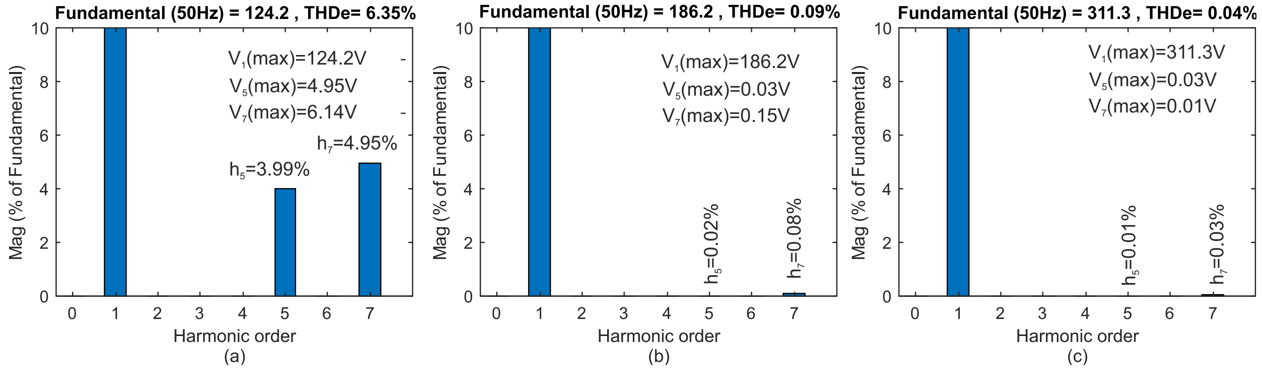


Fig.7. THDe values for different modulation indices (7-level) (a) M=0.4, (b) M=0.6, (c) M=1.0

TABLE IV
SWITCHING ANGLES CALCULATED WITH AVO (FOR 11 LEVELS)

Modulation Index	Switching Angles (Radians)				
M	**θ_1**	**θ_2**	**θ_3**	**θ_4**	**θ_5**
0.1	1.1672	1.5708	1.5708	1.5708	1.5708
0.2	0.7575	1.5119	1.5708	1.5708	1.5708
0.3	0.7706	1.1356	1.5317	1.5708	1.5708
0.4	0.6859	0.9827	1.3266	1.5704	1.5708
0.5	0.6409	0.8684	1.1454	1.4738	1.5708
0.6	0.6168	0.8196	1.0224	1.2672	1.5331
0.7	0.3426	0.6797	0.9852	1.1093	1.5395
0.8	0.3898	0.6855	0.9197	1.0352	1.2386
0.9	0.1334	0.4805	0.7115	0.9169	1.2745
1.0	0.1332	0.3363	0.5102	0.8244	1.1008

TABLE V
SIMULATION RESULTS (FOR 11 LEVELS)

M	Vref(peak)	Vref(rms)	V(rms)	Error (%)	THD (%)	THDe (%)	h_5	h_7	h_{11}	h_{13}
0.10	31.1	22	21.91	**0.41**	59.35	55.17	46.09	11.48	0.31	17.01
0.20	62.2	44	43.84	**0.36**	27.64	18.27	12.34	2.88	12.38	2.19
0.30	93.3	66	65.74	**0.39**	19.34	8.12	4.32	3.26	0.01	6.06
0.40	124.4	88	87.66	**0.39**	12.77	3.10	2.32	0.71	1.89	0.39
0.50	155.5	110	109.2	**0.73**	9.05	**0.95**	**0.47**	**0.25**	**0.70**	**0.34**
0.60	186.6	132	131.5	**0.38**	6.48	**0.09**	**0.05**	**0.04**	**0.02**	**0.02**
0.70	217.7	154	153.3	**0.45**	8.10	**0.05**	**0.02**	**0.02**	**0.03**	**0.01**
0.80	248.8	176	175.2	**0.45**	6.72	**0.03**	**0.01**	**0.00**	**0.02**	**0.01**
0.90	279.9	198	197.2	**0.40**	6.31	**0.05**	**0.02**	**0.00**	**0.01**	**0.02**
1.00	311.0	220	219.8	**0.09**	4.49	**0.03**	**0.02**	**0.01**	**0.01**	**0.01**

As shown in Table V, the AVO algorithm has successfully found suitable solutions for the 11-level inverter within the M range of 0.1 to 1.0. It is observed that the desired fundamental voltage is controlled with an error of less than 0.5% compared to the reference value. M range of 0.6 to 1.0 is identified as the effective range for suppressing selected harmonics. Within this range, both the THDe load value and the magnitudes of the 5th, 7th, 11th, and 13th harmonics are below 0.09%.

In Fig. 9, the voltage waveforms of the load are shown for modulation indices of 0.4, 0.6, and 1.0. The THD and THDe analyses of the waveforms are presented in Figures 10 and 11, respectively. For a modulation index of 0.4, the THD of the load voltage is calculated to be 12.27%, with a maximum value of Van(max) = 124V and an RMS value of Van(rms) = 87.71V (Fig. 10(a)). For a modulation index of 0.6, the THD of the load voltage is calculated to be 6.84%, with a maximum value of Van(max) = 185.9V and an RMS value of Van(rms) = 131.5V (Fig. 10(b)). Finally, for a modulation index of 1.0, the THD of the load voltage is determined to be 4.48%, with a maximum value of Van(max) = 310.8V and an RMS value of Van(rms) = 219.8V (Fig. 10(c)).

The degree of suppression of the selected harmonics is shown in Fig. 11. For a modulation index of 0.4, the THDe is calculated to be 3.10%, with harmonic magnitudes of $h_5 = 2.32$ ($V_5 = 2.88$V), $h_7 = 0.71$ ($V_7 = 0.89$V), h11 = 1.89 ($V_{11} = 2.34$V), and $h_{13} = 1.89$ ($V_{13} = 2.34$V). For a modulation index of 0.6, the THDe is determined to be 0.09%, with harmonic magnitudes of h5 = 0.05 ($V_5 = 0.08$V), $h_7 = 0.04$ (V7 = 0.07V), $h_{11} = 0.02$ (V_{11}

= 0.04V), and h_{13} = 0.02 (V_{13} = 0.03V). Finally, for a modulation index of 1.0, the THDe is found to be 0.04%, with harmonic magnitudes of h5 = 0.02 (V_5 = 0.05V), h_7 = 0.01 (V_7 = 0.03V), h_{11} = 0.01 (V_{11} = 0.02V), and h_{13} = 0.01 (V_{13} = 0.04V).

IV. Conclusion

This study utilized the recently developed AVOA (Artificial Virus Optimization Algorithm) to tackle the SHE-PWM (Selective Harmonic Elimination Pulse Width Modulation) issue in multilevel inverters. The effectiveness of AVOA in resolving SHE-PWM was showcased through MATLAB simulations conducted on 7 and 11 level cascade H-bridge inverters. The algorithm has demonstrated its capability to find suitable solutions within the M range of 0.1 to 1.0. The desired reference voltage has been effectively controlled with an error of less than 0.5%. Furthermore, the selected harmonics have been efficiently suppressed within the modulation range of 0.6 to 1.0, as indicated by the THDe values and the magnitudes of the 5th, 7th , 11th , and 13th harmonics, all of which are below 0.09%.

The waveform analysis of the output voltage for different modulation indices (0.4, 0.6, and 1.0) has shown that the THD values decrease as the modulation index increases. The maximum values of the fundamental voltage and its RMS value have also been determined for each modulation index, providing important insights into the performance of the system. Overall, the AVO algorithm has proven its effectiveness in achieving accurate voltage control and harmonic suppression in multilevel inverters, thereby contributing to the improvement of power quality and efficient operation of the system.

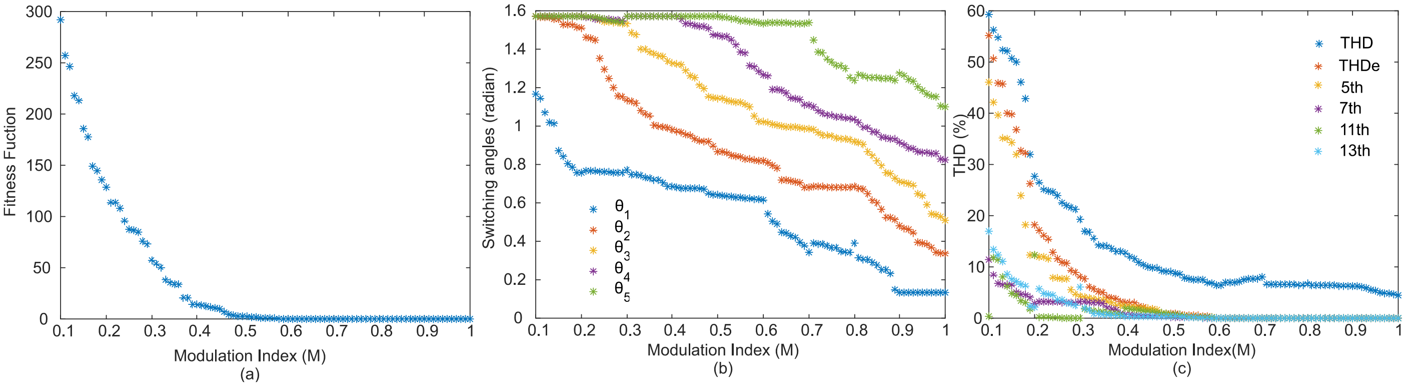


Fig.8. Analysis of the 11-level MLI inverter in terms of (a) fitness function, (b) switching angles, and (c) THD, THDe, 5th , 7th, 11th and 13th harmonic characteristics as a function of M.

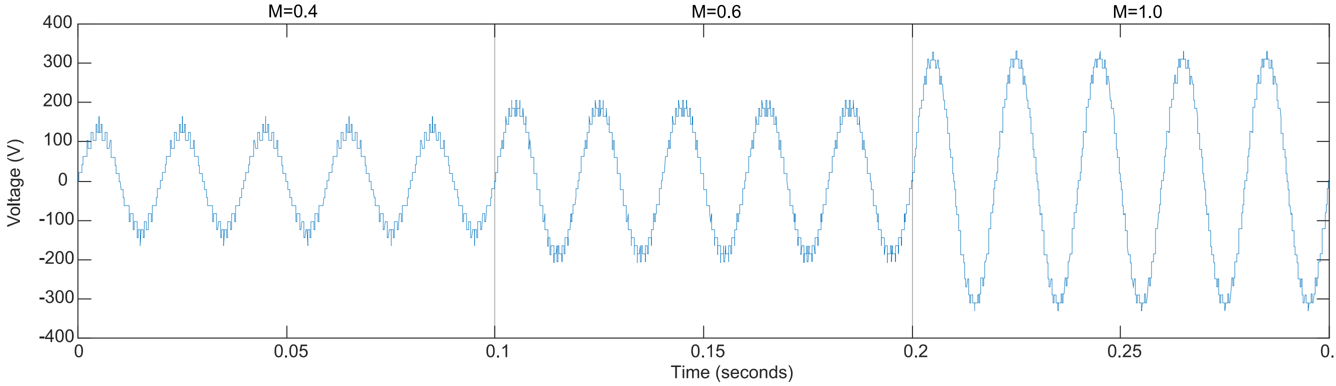


Fig.9. Load voltage waveforms for different modulation indices (11-level) (a) M=0.4, (b) M=0.6, (c) M=1.0

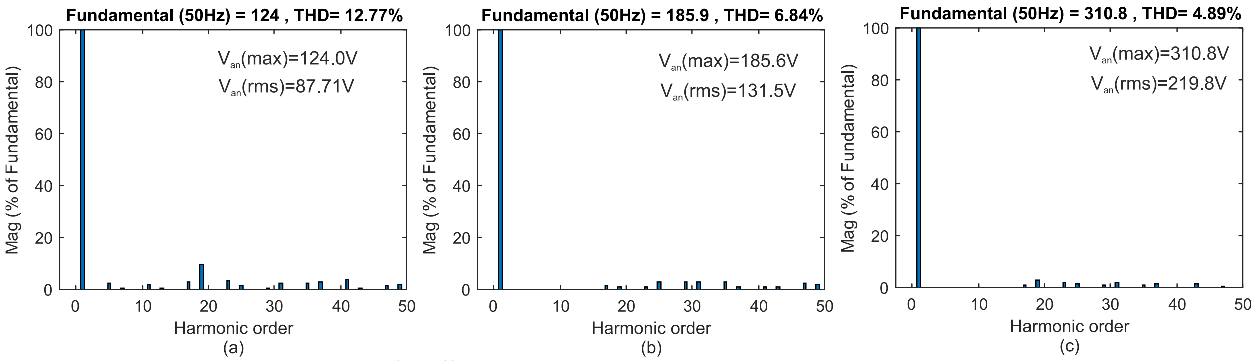


Fig.10. THD values for different modulation indices (11-level) (a) M=0.4, (b) M=0.6, (c) M=1.0

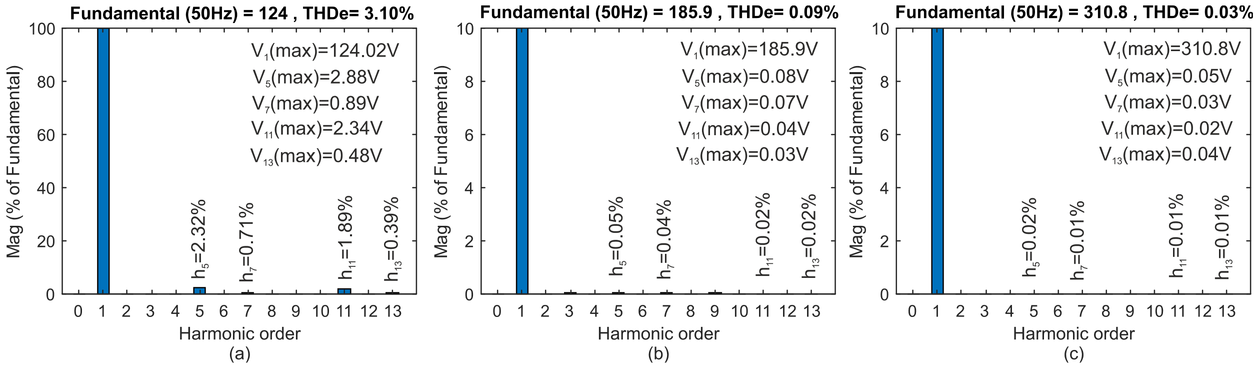


Fig.11. THDe values for different modulation indices (11-level) (a) M=0.4, (b) M=0.6, (c) M=1.0

References

[1] A. Raki, Y. Neyshabouri, M. Aslanian, & H. Iman-Eini, "A Fault-Tolerant Strategy for Safe Operation of Cascaded H-Bridge Multilevel Inverter under Faulty Condition". IEEE Transactions on Power Electronics.

[2] B. Karthikeyan, B. Swetha, C. Shahana, B. Vijay, & M. Swathi, "Switched Capacitor Based Multilevel Inverter for\newline PV System". In 2023 9th International Conference on Electrical Energy Systems (ICEES) (pp. 471-475). IEEE. March, 2023.

[3] N. Padmamalini, P. Deepa, T. Joel, S. Loganayagi, M. F. Banu, & S. Gomathi, "Control of Diode Clamped Multilevel Inverter based STATCOM for Reactive Power Compensation using H-bridge Topology". In 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1860-1865). IEEE.

[4] H. I. Hazim, K. A. Baharin, C. K. Gan, A. H. Sabry, & A. J. Humaidi, Review on optimization techniques of PV/inverter ratio for grid-Tie PV systems. Applied Sciences, 13(5), 3155, 2023.

[5] C. Mayet, D. Labrousse, R. Bkekri, F. Roy, & G. Pongnot, "Energetic Macroscopic Representation and Inversion-Based Control of a Multi-Level Inverter with Integrated Battery for Electric Vehicles". In 2021 IEEE Vehicle Power and Propulsion Conference (VPPC) (pp. 1-6). IEEE, October, 2021.

[6] C. Pisani, G. Bruno, H. Saad, P. Rault, & B. Clerc, "Functional validation of a real VSC HVDC control system in black start operation", In 2019 AEIT HVDC International Conference (AEIT HVDC) (pp. 1-6). IEEE. May, 2019.

[7] H. Lin, R. Chen, R. Li, L. Zhu, H. Yan, & Z. Shu , "A flexible and fast space vector pulse width modulation technique for multilevel converters", In 2019 22nd International Conference on Electrical Machines and Systems (ICEMS) (pp. 1-4). IEEE, August, 2019.

[8] S. A. Azmi, A. A. Shukor, & S. R. A. Rahim, "Performance evaluation of single-phase H-bridge inverter using selective harmonic elimination and sinusoidal PWM techniques", In 2018 IEEE 7th International Conference on Power and Energy (PECon) (pp. 67-72). IEEE, December, 2018.

[9] S. E. Arslan, F. E. Uzun, K. Gürkan, A. Acar, İ. O. Yildirim, , U. Güven, ... & S. Yarman, "Realization of SV-PWM motor control algorithm using ARM Cortex-M4 based microcontroller", In 2016 National Conference on Electrical, Electronics and Biomedical Engineering (ELECO) (pp. 282-285). IEEE. December, 2016.

[10] E. Bektas, & H. Karaca, "GA based selective harmonic elimination for multilevel inverter with reduced number of switches: an experimental study", Elektronika ir Elektrotechnika, 25(3), 10-17, 2019.

[11] A. Pourdadashnia, M. Farhadi-Kangarlu, B. Tousi, & M. Sadoughi ,. "SHM-PWM technique in a cascaded H-bridge multilevel inverter with adjustable DC-link for wide voltage range applications", International Journal of Circuit Theory and Applications, 51(5), 2228-2246, 2023.

[12] S. Pani, N. Guru, D. Puhan, & A. K. Barisal, "Comparative Performance analysis of Multilevel Inverter through meta heuristics", In 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 129-135). IEEE, May, 2022.

[13] B. G. Babu & M. S. Kalavathi, "Hardware Implementation of Multilevel Inverter using NR, GA, Bee Algorithms" In 2021 International Conference on Sustainable Energy and Future Electric Transportation (SEFET) (pp. 1-6). IEEE. January, 2021.

[14] Y. Bektaş, & H. Karaca, "Red deer algorithm based selective harmonic elimination for renewable energy application with unequal DC sources" Energy Reports, 8, 588-596, 2022.

[15] B. Abdollahzadeh, F. S. Gharehchopogh, & S. Mirjalili, African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems", Computers & Industrial Engineering, 158, 107408, 2021.

[16] H. Karaca, & E. Bektas, "Selective Harmonic Elimination Technique Based on Genetic Algorithm for Multilevel Inverters", In Transactions on Engineering Technologies: World Congress on Engineering and Computer Science 2015 (pp. 333-347). Springer Singapore, 2017

[17] M. YEŞİLBUDAK, "Extraction of photovoltaic cell and photovoltaic module parameters using african vultures optimization algorithm" Gazi University Journal of Science Part C: Design and Technology, 9(4), 708-725.

[18] M. E. Bento, "Design of a Resilient Wide-Area Damping Controller Using African Vultures Optimization Algorithm", In 2021 31st Australasian Universities Power Engineering Conference (AUPEC) (pp. 1-6). IEEE, September ,2021.

[19] H. E. ALOUACHE, S. SAYAH, & A. HAMOUDA, "Africa vultures optimization algorithm for optimal power flow solution including SVC devices", In 2022 19th International Multi-Conference on Systems, Signals & Devices (SSD). IEEE. May, 2022

[20] R. Liu, T. Wang, J. Zhou, X. Hao, Y. Xu, & J. Qiu, “Improved African vulture optimization algorithm based on quasi-oppositional differential evolution operator”, IEEE Access, 10, 95197-95218, 2022.

[21] D. Gürses, P. Mehta, S. M. Sait, & A. R. Yildiz, African vultures optimization algorithm for optimization of shell and tube heat exchangers”, Materials Testing, 64(8), 1234-1241. 2022.

[22] J. Zhang, M. Khayatnezhad, & N. Ghadimi, “Optimal model evaluation of the proton-exchange membrane fuel cells based on deep learning and modified African Vulture Optimization Algorithm”. Energy Sources, Part A: Recovery, Utilization, and Environmental Effects, 44(1), 287-305. 2022.

[23] M. Ahmed, M. Khamies, G. Magdy, & S. Kamel, “Designing Optimal PI λ D μ Controller for LFC of Two-Area Power Systems Using African Vulture's Optimization Algorithm”, In 2021 22nd International Middle East Power Systems Conference (MEPCON) (pp. 430-437). IEEE, December, 2021.

[24] S. Mil’shtein, & D. N. Asthana, “Brief Comparison of Tandem and Cascaded Solar Cells”, In 2021 IEEE 48th Photovoltaic Specialists Conference (PVSC) (pp. 2260-2263). IEEE, June, 2021.

[25] F. S. Gharehchopogh, “An improved Harris Hawks optimization algorithm with multi-strategy for community detection in social network” Journal of Bionic Engineering, 20(3), 1175-1197, 2023.

[26] T. Sonmezocak, O. Akar, & U. K. Terzi, “HIGH PERFORMANCE ADAPTIVE ACTIVE HARMONIC FILTER DESIGN FOR NON-LINEAR LED LOADS” Light & Engineering, 30(1), 29-38, 2022.

[27] A. Demirci, O. Akar, U. K. Terzi, & T. Sönmezocak, “Investigation of International Harmonic Standards in Power Systems”, In 4th International Mardin Artuklu Scientific Research Congress (pp. 7-8), 2020, August.

Balkan Journal of Electrical and Computer Engineering (BAJECE)
Volume 1 Number 1, September 2023

A Real-time Face Recognition Based on MobileNetV2 Model

Vafaa Sukkar[1] and Ergun Ercelebi[2]

1,2 Department of Electrical and Electronics Engineering, University of Gaziantep, 27310 Gaziantep, Turkey

Manuscript received Jan 22, 2023; accepted August 12, 2023.

Abstract—**Facial recognition technology is one of the fastest developing technologies. It is the most widespread technology compared to other biometric ones. Technologies based on deep learning and neural networks have demonstrated superior efficiency and speed when compared to traditional approaches for recognizing persons. In this work, we introduce a light system for real-time facial recognition with an improved recognition time. It relies on today's latest convolution neural networks algorithms. The database is built based on photos from a collection of both known people and some VggFace dataset celebrities. The system pipeline is divided into several phases, beginning with detecting faces in input images using the MTCNN algorithm, then aligning and preprocessing them, extracting facial characteristics vectors for each face using a pre-trained mobileNetV2 model, and finally classifying faces using the SoftMax classifier layer. Evaluating its performance on some samples, the system achieved an average accuracy of 92.67% with an execution time of 10 milliseconds to process an image.**

Index Terms—**Deep Learning, Face Recognition, MobileNetV2, MTCNN, Real-time**

I. Introduction

Face recognition technology has sparked much attention as an effective and secret tool that provides a doorway into massive amounts of data and enables the identification of people without their feeling or knowledge.

Because of its simplicity, low cost, flexibility of use, and immediate results providence, facial recognition is the most appropriate way in the majority of cases to identify people compared to the prevailing methods, such as fingerprints, which require that the person's finger be clean and place it on the scanner, or as DNA that requires samples to be analyzed and this it takes time and money, or such as the eye print that requires being very close to the reading camera, or the voice print that requires shouting sometimes and approaching the microphone, or the RF technology that the person who may already doubt his credibility has to take the RFID card out of his pocket and touch the surfaces that may be contaminated.

Often there is confusion between facial identification and facial authentication, but they are different techniques and usually designed for different purposes. Facial authentication or verification systems verify the person by matching his photo with a specific identity. We see such systems in mobile phones that are being unlocked after checking and verifying the image of the owner's face. It is a 1:1 matching process. Face identification systems identify a person by comparing the image of his face with the pre-defined people in the database and determining the identity of this person. An example of this is the system in some universities; once the student arrives at the university, the system recognizes him/her after analyzing his/her facial image, and the gate automatically opens then. Face identification is a 1: N process.

The pipeline of facial recognition systems is generally composed of sequential steps that help capture, analyze, compare, and match the captured face to a precompiled database of images. Face Detection is the first stage where the image is scanned and the face is distinguished from the rest of the existing objects. Followed by Alignment stage where facial landmarks are detected and localized precisely and the face is normalized to be homogenous with the dataset. The following stage is Feature Extraction in which the face is analyzed and unique data is extracted and converted into vectors for comparison. Classification is the final stage where the final extracted data is compared with the stored database of pre-defined people, Fig. 1.

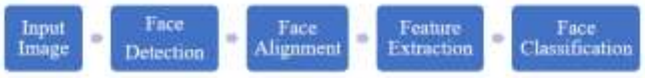


Fig.1. Face Recognition building blocks.

II. Literature review

A. Face Recognition Methods

Face detection is a kind of object detection and the primary step in face recognition systems. It is used to determine the existence of human faces and returns their location and sizes in the images if they are present, and then detect and mark each of them with a bounding box. Face Detection identifies the required components of the image for creating a faceprint. Algorithms of face detection fall under four main categories:

1) *Knowledge-Based Methods (Rule-Based Methods)*

In these methods, human faces are described based on rules related to the structure of the human face, as well as the relationship and arrangement between the facial characteristics in a typical human face.

2) *Template Matching Methods*

These approaches compare input images to predefined stored template images by finding similar features and correlating the two to detect the presence of a face based on correlation values. These methods have problems with pose, shape, and scale variations.

3) *Appearance-Based Methods*

These methods are based on the templates learned from the training images that capture the representative variability of the face appearance.

4) *Feature-Based Methods*

They locate faces by searching for invariant structural features of the face and extracting them. These algorithms work even with the angle and pose variations.

B. Face Recognition Methods

The facial image is obtained from the input image and converted into numerical expression. After that, the geometry of the face is scanned by a facial recognition algorithm to dig up particular distinctive details and identify the key points of the face. The extracted data of a certain face is called a face template. Face templates are used to distinguish faces from each other by calculating and comparing the distances between the data of input and stored faces.

1) *Local features approach*

It is a part-based approach that focuses on extracting the local features in the face like mouth, nose, and eyes and determining their locations. The histogram of oriented gradients (HOG)[1], Local Binary Pattern (LBP) [2,3] and Scale Invariant Feature Transform (SIFT)[4] algorithms fall under this field. These techniques partially overcome the variation in illumination, pose, and facial expression. On the other hand, they deal with high dimensional feature space, leading to computational complexity.

2) *Holistic approach*

It deals with the entire face. The image face is described by feature vectors that are converted from the matrix of pixels representing the global information and characteristic features of faces. Dimension reduction is the key benefit of this approach, but there is a stability issue with rotations and translation cases. Independent Component Analysis (ICA)[5], Eigenfaces[6] and Principal Component Analysis (PCA) [7] are the most common techniques of this category.

3) *Hybrid approach*

It is a combination of both local features and holistic algorithms. Methods in this category utilize the strengths of the local feature approach in overcoming problems of recognition and the holistic approach in the reduction of dimensionality and complexity.

Although good results are often achieved under standard conditions, the identification of people is sometimes mistaken. This is due to several reasons, including blurred images, lack of proper illumination, different facial expressions, and the position of the face in the images. All this adversely reflects on the performance and efficiency of algorithms and prevents proper analysis and precise results, especially in real-time recognition.

C. Deep Learning and Convolutional Neural Network

Facial recognition techniques have undergone great changes and evolutions throughout the years, especially in the last decade, resulting in a rapid expansion of use in commercial applications. Classical face recognition techniques focus on using geometry-based methods and statistical subspaces. However, due to variations caused by view angles, background clutter, and occlusions, these methods reflect some failures in representing faces. Thanks to deep learning, facial recognition has become more powerful, and less affected by varying conditions.

Deep learning (DL) is a categorization of the Machine Learning classification that falls under Artificial Intelligence. It is centered on the creation of algorithms and models that can learn and make intelligent judgments without human intervention. A deep learning system is a self-learning system that relies primarily on deep neural networks and learns through passing, processing, and filtering data within the networks. Convolutional neural network (ConvNet, or CNN) is one of the deep learning methods. They are similar in shape to artificial neural networks (ANN), which are the backbone of deep learning, as they are hierarchical. CNN, like ANN, has learnable weights and biases. What distinguishes CNN is that it is primarily utilized for images classification and facial recognition issues, with an image as the input in this case. The CNN transforms the image into a simpler form with fewer dimensions without losing its properties, making it easier to analyze and process.

The ability to train CNN-based models with vast data sets to learn the best data representation features is the primary benefit of the deep learning-based approaches. Also, the availability of large data sets of diverse faces images of people has contributed to the superiority of these methods.

Convolutional neural networks[8] consist of several layers:

1) *Input layer*

The input image is converted into a matrix of numbers that represent its pixels.

2) *Convolutional layers*

These layers extract low-level and high-level features of the image in phases until all the characteristics are extracted. The first convolutional layer could be confined to extracting low-level features such as colors and edges.

3) *Pooling layer*

This layer sits between the convolutional and FC layers to reduce the spatial size of the image and hence decrease the computational cost. A pooling layer might be one of two kinds: average pooling or max pooling.

4) *Fully connected layers*

These layers usually represent the last layers of CNN. The input to these layers is the output of the last layer of the convolution layer or pooling output (if it exists) after it is flattened. Through training, a fully connected layer collects extracted data from previous layers and feeds them into the Softmax layer, which is the classification layer, Fig. 2.

With the spread and popularity of deep learning methods, researches on facial recognition accelerated, and CNNs are used to deal with many other issues such as objects detection, handwritten character recognition, translations, question answering, analysis of facial expressions, and others.

Deep learning-based methods are now the most successful among facial recognition techniques; they provide the best results compared to all other algorithms, especially with the significant development in convolutional neural network architectures such as R-CNN, Fast R-CNN, VGG16 and ResNet50. Despite these algorithms being among the most commonly used algorithms for face recognition, they have main problem related to poor processing speed which make them unapplicable in real-time cases or problems that require fast outputs.

D. Related Work

Authors in research[9] proposed a robust face recognition system based on CNN. Viola-Jones algorithm was used for detecting the face. Then contrast enhancement using Histogram Equalization proceeded to the input face. They implemented their work on the Extended Yale B and CMU PIE face databases. Their work achieved a recognition rate of 97.23% and 98.38% on both VGG16 and ResNet50 architectures, respectively.

Other researchers[10] designed a FR system that takes the attendance automatically using deep learning technology. The maximum recognition rate of their system was 70%. Face detection was carried out using Haar cascade method, whereas face recognition was performed using LBPH.

In research[11], the authors designed an attendance system using computer vision and machine learning technology. They used a DNN-based detector for detection and LDA and PCA methods for feature extraction. Their FR method achieved a real-time accuracy of 56% for MLP and SVM classifiers and about 89% for the CNN classifier.

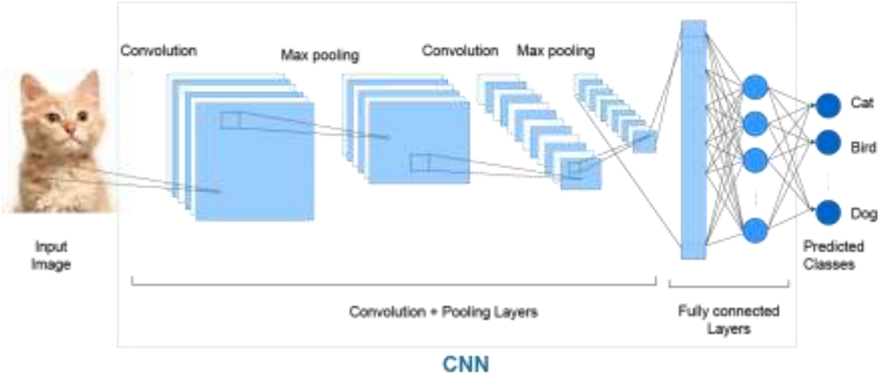


Fig.2. An example of a Convolutional Neural Network.

III. Methodology

This work aims to design a real-time facial recognition system that is robust, fast, and light at the same time using predefined CNN algorithms. The fundamental reason for this is that it is more practical to integrate a system into any type of device, whether it is an embedded system, a mobile device, or a computer without GPU, regardless of its capabilities.

The proposed face recognition system begins with the input image, where the face is detected and located using the MTCNN algorithm. After that, the image is aligned, and the face is cropped from it. The deep CNN model MobileNetV2 is applied to extract features from the cropped face, and the classification process is performed using the SoftMax layer classifier.

A. Face Detection

We used the Multi-task Cascaded Convolutional Neural Network (MTCNN) algorithm to efficiently search for faces in the image, detect them and recognize their facial marks such as eyes, lips, eyebrows, etc. MTCNN is a robust algorithm presented to perform both face detection and alignment. It detects faces with high speed and accuracy, and it is more potent than other algorithms in encountering the challenges that negatively affect the detector's efficiency, including the conditions in which the image was taken and changes in the face.

MTCNN is made up of three separate cascade stages of networks. They are P-Net, R-Net, and the O-Net. The output of one stage is the input to the next stage. Each of these networks returns three information: a face bounding rectangle, the probability that a particular rectangle contains a face, and five landmarks.

For the detector to be able to detect faces of all sizes, copies of the image at different scales are created as a first step, resulting in an image pyramid.

The overall three stages of MTCNN, Fig. 4, are as follow:

1) *Proposal network (P-Net)*

It is a Fully Convolutional Network (FCN) utilized to fast analyze the image and returns many candidate windows with corresponding boundary box regression vectors. These are then filtered using the Non-Maximum Suppression (NMS) technique to downsize the candidate windows and obtain the best boundary boxes out of the overlapping boundary boxes.

2) *Refine Network (R-Net)*

It is a CNN since the dense layer exists in the architecture of this network. The network further filters the predicted candidate windows from the previous P-Net and provides more credible and accurate boundary boxes accomplished with the confidence level of each of them. The NMS is then applied again to clear out those boundary boxes of low confidence.

3) *The Output Network (O-Net)*

It is more complicated CNN than R-Net, and it is the slowest network of the three cascade networks since it aims to get more facial features and returns locations of the five facial landmarks, including right and left eyes, nose, right and left corners of the mouth. After this stage, only one boundary box should remain for each face in the image.

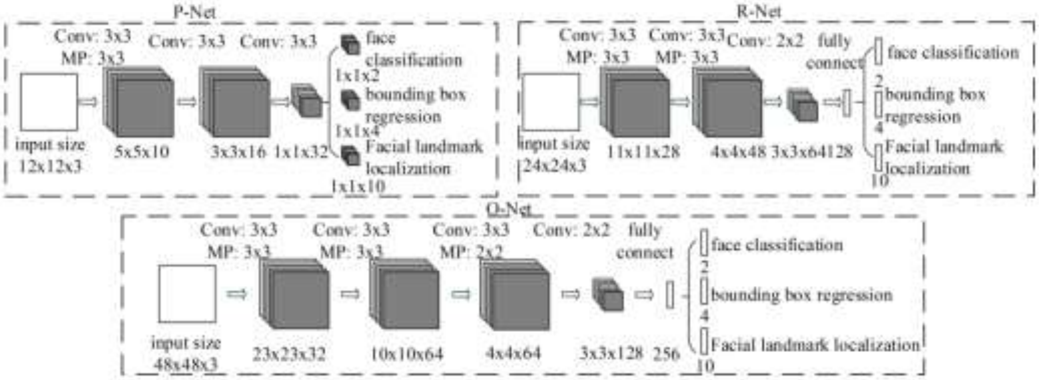


Fig.3. MTCNN stages.

B. Face Alignment

Facial alignment is essential as it improves the overall performance of the recognition system and provides higher accuracy. After the face has been discovered in the picture, it must be centered, Fig. 4. So, the next step is to locate the face and identify the facial markers like eyes and the nose. Fortunately, the MTCNN algorithm locates the face and its components; all that's left is to:

1. Use the coordinates of these facial components and analyze their positions to estimate the required rotation angle.
2. Rotate the face so that the eyes are at the same horizontal level.
3. Center the face in the image.
4. Cut it and change its size to fit the classification network input size.

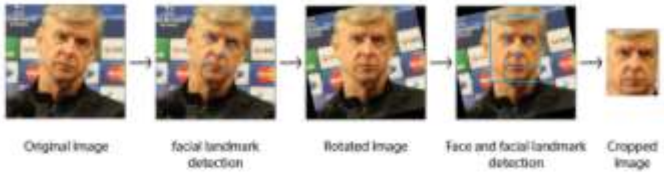


Fig.4. Face detection and alignment.

C. Feature Extraction and Face Classification

MobileNetV2[12]is one of the CNN models that are used for image recognition. This algorithm has high effectiveness, performance, and speed in extracting the features. Moreover, it is light and applicable for mobile devices and devices of low computational power.

Compared to the standard CNN algorithms with similar depth, MobileNet[13] has significantly less model size and uses a smaller number of parameters. It uses Depthwise separable convolution, which conduces to less computational cost since it reduces the multiplication and addition operations. Fig. 5 and Fig. 6 show the standard convolution and the separable depthwise convolution, respectively

Depthwise separable convolution is split into two operations:

1) Depthwise Convolution

Unlike in the normal convolutions where the convolution is applied to all or multiple input channels at a time, in depthwise convolution, the convolution of a kernel is performed over a single channel. The output is then shaped by stacking the outputs of these channels.

2) Pointwise Convolution

It is a 1x1 convolution applied at each point on the M channels to change the size of the depthwise convolution output. The kernel's channels are equal to the number of input channels.

Authors of MobileNet paper show that the ratio of the total computational cost of depthwise separable convolution compared to normal convolution is: $1/N + 1/D_k \times D_k$. When N is larger, as in normal cases, the total cost of depthwise separable convolution will be around ten times cheaper in computational cost.

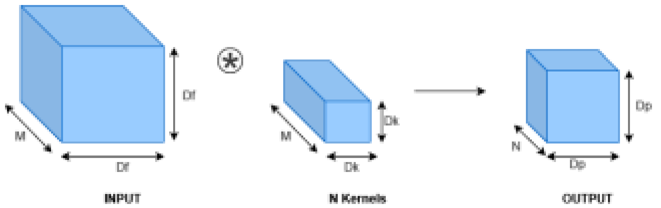


Fig.5. The standard convolution.

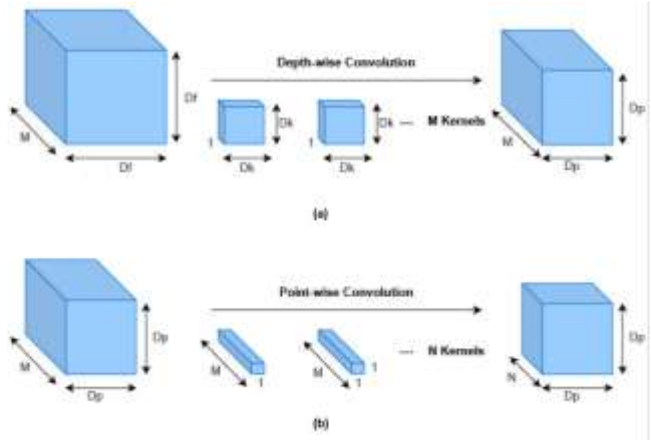


Fig.6. Separable Depthwise convolution: (a) Depthwise convolution (b) Pointwise convolution.

MobileNetV2 architecture is introduced on the basis of MobileNetV1 to increase the accuracy and reduce the computational cost more. The residual connections and the expansion layer are two new features applied to mobileNetV2 architecture. MobileNetV2 is based on two types of Bottleneck blocks. Each block has three different convolutional layers: 1x1 Convolution with Relu6, Depthwise Convolution, and 1x1 Convolution.

MobileNetV2 network is built of 53 convolutional layers and 19 blocks. Table 1 [12] shows the model structure, with conv2d denoting a standard 2D convolution layer, Bottleneck representing a bottleneck residual block, and AvgPool denoting the average pooling layer. The convolution is applied to the input firstly using 32 filters, then the feature extraction task is performed using the middle layers, and the classification is accomplished using the last convolution layer. *s* refers to the stride, *n* is the repeats, *t* is the expansion factor, and *c* is the number of the output channels.

TABLE I
ABSORBENCY FACTOR FOR 3% CNFs FILLER LAYER

Input H x W	number of Input channels	Operator	s	t	n	c
224x224	3	conv2d	2	-	1	32
112x112	32	Bottleneck	1	1	1	16
112x112	16	Bottleneck	2	6	2	24
56x56	24	Bottleneck	2	6	3	32
28x28	32	Bottleneck	2	6	4	64
14x14	64	Bottleneck	1	6	3	96
14x14	96	Bottleneck	2	6	3	160
7x7	160	Bottleneck	1	6	1	320
7x7	320	Conv2d	1	-	1	1280
7x7	1280	AvgPool	-	-	1	-
1x1	1280	conv2d	-	-	1	K

D. *Transfer Learning*

It is quite challenging to have that much data to train an entire CNN network from scratch. Using a model that has previously trained on a large dataset as the starting point for training on a new problem is more efficient than consuming time, effort, and power on training all of the model's layers using randomly initialized weights. The transfer learning technique is used in this face recognition system to save training time and avoid overfitting by leveraging a pre-trained model that has already learned the characteristics. The model was pre-trained on the ImageNet dataset[14], which contains more than 14 million images and over 22000 distinct categories. The model then serves as the base of our custom model for recognizing faces.

IV. EXPERIMENT AND RESULTS

Experiments were done utilizing the proposed face recognition procedure to assess the performance of the system constructed using the proposed technique. The work was split into a training stage and a testing stage. In the training stage, the mobileNetV2 model that was pre-trained on the ImageNet dataset was used as the feature extractor. The training process was performed on the classification layers newly added to the model after omitting the existing ones. To fine-tune it, we set some layers as trainable. After the model was trained and learned features, we saved it with its weights for implementation in the testing stage, Fig. 7.

A. *Software and Hardware*

For the implementation of face detection MTCNN and feature extraction MobileNetv2 algorithms, Python3 Programming language with TensorFlow v2.5 platform, Keras library, and Jupyter notebook were used. The training of the face recognition model was carried out on the PRO version of Google colaboratory of 12GB RAM and NVIDIA Tesla K80 GPU. The testing was then performed on a local machine: HP Laptop of Intel® Core™ i7, 2.60 GHz processor, 8.00 GB RAM, 64-bit operating system, and HP TrueVision HD Webcam with a resolution of 1280×720 (0.922MP) for real-time face recognition.

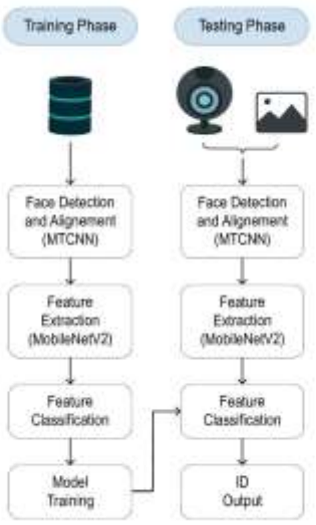

Training Phase
Testing Phase
Face Detection and Alignement (MTCNN)
Face Detection and Alignement (MTCNN)
Feature Extraction (MobileNetV2)
Feature Extraction (MobileNetV2)
Feature Classification
Feature Classification
Model Training
ID Output

Fig.7. Block diagram of the training and testing phases.

B. *Preparation of Dataset*

The dataset prepared for training the system model was a mix of known people and some celebrities extracted from the VGGFace dataset. The final database consists of 1500 good-quality faces images extracted from a set of 1532 images that contain only one face. The sizes of faces extracted from the images must be at least 60 px in width and 70px in hight, and every face smaller than this size is rejected and not taken into account. The faces images belong to 15 distinct persons grouped in 15 folders labeled by the person's name. The images were taken in different conditions, with different facial expressions, poses, angles, backgrounds, and lighting conditions. The number of images for each person was fixed to 100.

C. *System Implementation*

The faces were detected, aligned, cropped, preprocessed, and then split into three sets; training set, validation set, and test sets with weights of 80%, 10%, and 10. MobileNetV2 model was then initialized by the pre-trained weights. For the first phase, the training process was performed on the newly added classifier layers to the model after truncating the pre-trained ones and freezing the feature extractor layers. In fine-tuning phase, some layers of the base model were unfrozen and the model was trained again using a smaller learning rate, Fig. 8.

For computing the loss, categorical_crossentropy loss function was used since we deal with a multi-class classification model. Evaluating the performance of the model was done using the accuracy function which measures the accuracy of the model and evaluates its performance. For optimization, we employed the RMSprop (Root Mean Square Propagation) optimizer.

The total training epochs performed on the model is 57; 20 are the training stage epochs, and 37 are fine-tuning epochs. The total training epochs performed on the model is 57; 20 are the training stage epochs, and 37 are fine-tuning epochs.

Fig. 9 shows the plots of model accuracy and loss over epochs of both training and fine-tuning stages, and they are separated by a green line, and Table 2 summarizes the results of the stages.

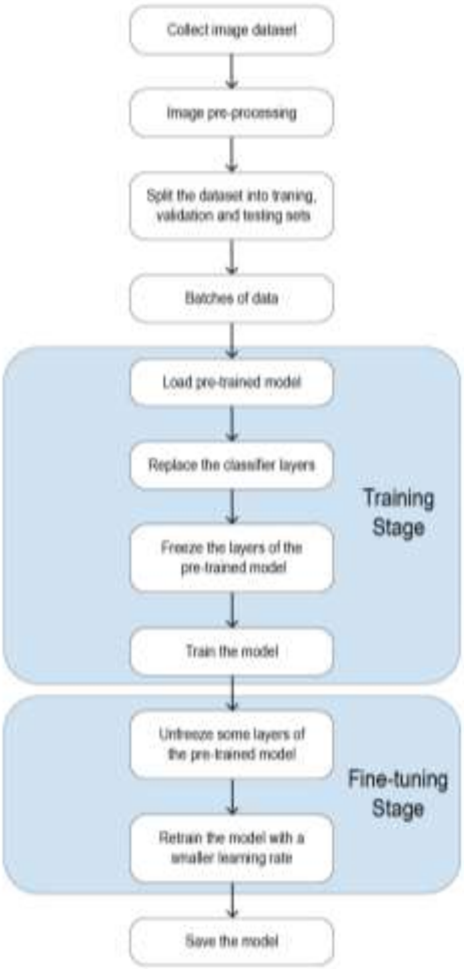

Collect image dataset
Image pre-processing
Split the dataset into traning, validation and testing sets
Batches of data
Load pre-trained model
Replace the classifier layers
Freeze the layers of the pre-trained model
Train the model
Training Stage
Unfreeze some layers of the pre-trained model
Retrain the model with a smaller learning rate
Fine-tuning Stage
Save the model

Fig.8. Proposed training work.

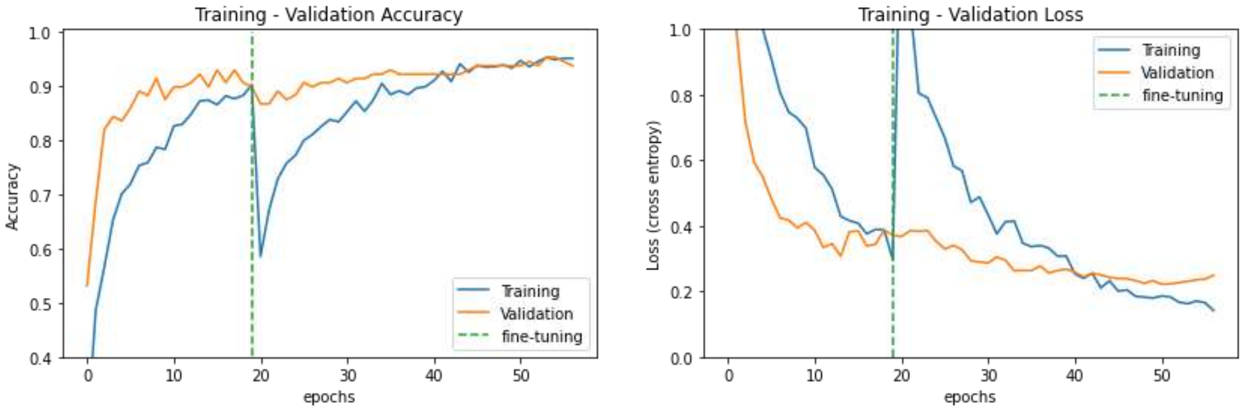


Fig.9. Model Accuracy and Loss after training and fine-tuning process.

TABLE II
Summary of Training and Fine-tuning Stages of MobileNetV2 model

		Training stage	Fine-tuning stage
Epochs		40	40
Actual epochs		20	37
Trainable parameters		32,383,503	34,244,943
Non-trainable parameters		2,257,984	396,544
Training set	Accuracy	90.24%	95.12%
	Loss	0.2980	0.1414
Validation set	Accuracy	89.84%	93.75%
	Loss	0.34	0.2492
Testing set	Accuracy	90.67%	92.67%
	Loss	0.32	0.25

The metrics used to evaluate the performance of the trained model on the testing dataset are:

$$\text{Precision}(\text{PRE}) \;= \frac{TP}{TP \,+\, FP}$$

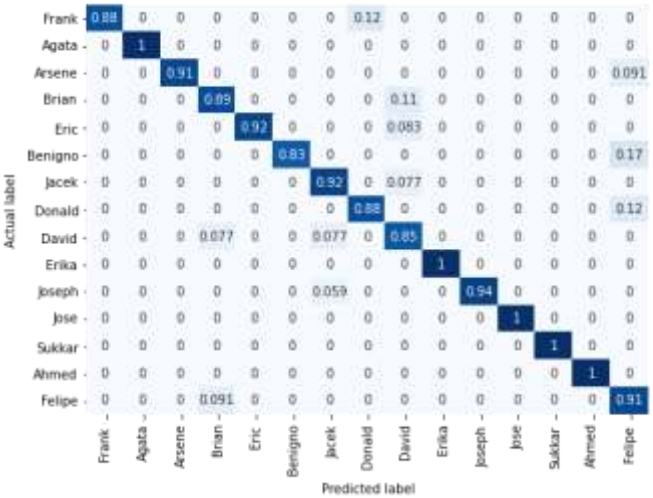
$$\text{Recall}(\text{RE}) \;= \frac{TP}{TP \,+\, FN}$$

$$\text{F1}-\text{Score(F1)} \ = \frac{2*PRE*RE}{PRE+\ RE}$$

$$\text{Accuracy(AC)} \ = \frac{TP+TN}{TP+TN+FP+FN}$$

Where TP is true positive, FP is false positive, TN is true negative and FN is the false negative.

The summary of prediction outcomes on our classification model is displayed in a confusion matrix that represents the number of correct and incorrect predictions in a counted way and separates them by each class, as illustrated in fig. 10.



Actual label \ Predicted label	Frank	Agata	Arsene	Brian	Eric	Benigno	Jacek	Donald	David	Erika	Joseph	Jose	Sukkar	Ahmed	Felipe
Frank	0.88	0	0	0	0	0	0	0.12	0	0	0	0	0	0	0
Agata	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Arsene	0	0	0.91	0	0	0	0	0	0	0	0	0	0	0	0.091
Brian	0	0	0	0.89	0	0	0	0	0.11	0	0	0	0	0	0
Eric	0	0	0	0	0.92	0	0	0	0.083	0	0	0	0	0	0
Benigno	0	0	0	0	0	0.83	0	0	0	0	0	0	0	0	0.17
Jacek	0	0	0	0	0	0	0.92	0	0.077	0	0	0	0	0	0
Donald	0	0	0	0	0	0	0	0.88	0	0	0	0	0	0	0.12
David	0	0	0	0.077	0	0	0.077	0	0.85	0	0	0	0	0	0
Erika	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Joseph	0	0	0	0	0	0	0.059	0	0	0	0.94	0	0	0	0
Jose	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Sukkar	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Ahmed	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Felipe	0	0	0	0.091	0	0	0	0	0	0	0	0	0	0	0.91

Fig.10. Normalized confusion matrix for our classification problem.

Fig. 11 displays some correct and incorrect predicted images where the incorrect prediction label is presented in red.

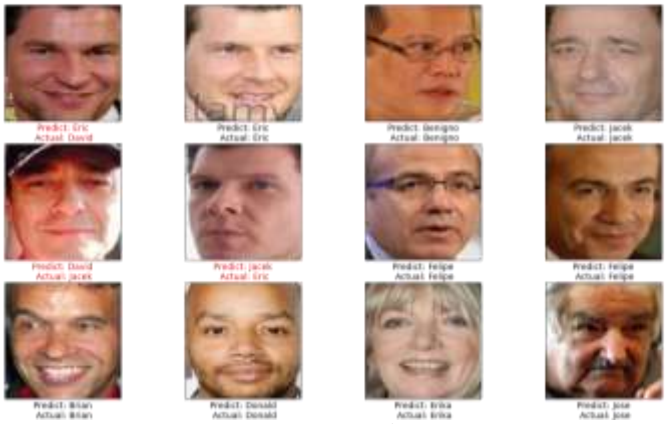


Fig.11. Some predicted images of the testing dataset.

D. System Testing in Real-time

The system received the loaded picture or video frame as input. Each face of the input image was detected by the MTCNN algorithm, resized to the specific size of 224x224 pixels to fit the input of MobileNetV2 model. The model.predict() function mapped and predicted the test data based on the learned labels by feeding the array into the model, determining the prediction value, and returning the label with the highest probability. The prediction value refers to the similarity ratio between the input face and the persons in the dataset. The person with the max prediction, which must be above the threshold of 85%, is the most likely to be the true identity. If the max prediction is less than that threshold, the input person image is considered as unknown and belongs to no one. Fig.12 depicts a flowchart for the real-time. And Fig. 13 is a screenshot from our system output when tested in real-time.

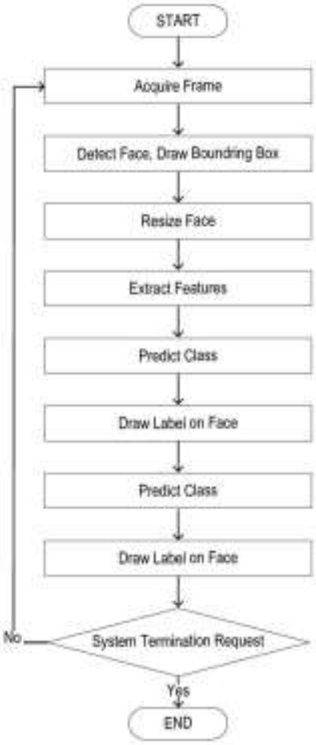


Fig.12. Flowchart of the Real-time System.

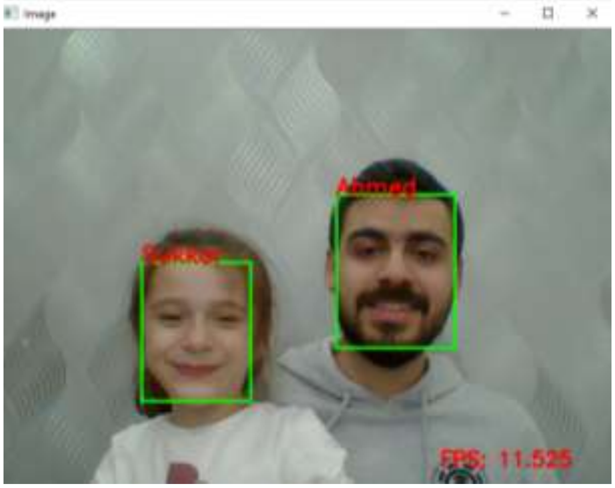


Fig.13. Recognition test in real-time.

To assess and evaluate the effectiveness of our proposed system, we compared it to VGG16 and ResNet-50 algorithms by training them on the same dataset. Using the exact procedure followed for MobileNetV2, we trained and fine-tuned the selected models. The summaries of training processes of

VGG16 and ResNet50 models are shown in tables 3 and 4, respectively.

TABLE III
Summary of Training and Fine-tuning Stages of VGG16 model

	Training stage	Fine-tuning stage
Epochs	40	40
Actual epochs	37	8
Total parameters	27,830,607	27,830,607
Trainable parameters	13,115,919	20,195,343
Non-trainable parameters	14,714,688	7,635,264
Output Accuracy	97.33%	98.67%
Output Loss	0.08	0.04

TABLE IV
Summary of Training and Fine-tuning Stages of ResNet50 model

	Training stage	Fine-tuning stage
Epochs	40	40
Actual epochs	39	27
Total parameters	75,238,799	75,238,799
Trainable parameters	51,651,087	68,866,575
Non-trainable parameters	23,587,712	6,372,224
Output Accuracy	94.00%	97.33%
Output Loss	0.023	0.09

The time it took to train and fine-tune the models were close to each other because the training processes were carried out on GPU. Resnet50 and Vgg16 models achieved higher accuracy and less loss than MobileNetV2 model. However, all the algorithms achieved more than 90% accuracy in both training and testing stages. In terms of speed and fps, MobileNetV2 surpassed the other models; it was about two times faster than Resnet-50 and three times faster than VGG16. Due to their large sizes and number of parameters, VGG16 and Resnet-50 took a long time to analyze and recognize faces, table 5.

TABLE V
Summary of Models performance in terms of execution time and fps

Model	Execution time	fps
Vgg16	0.33 s	3.011
Resnet50	0.21 s	4.707
MobileNetV2	0.09 s	11.677

V. Discussions

We trained the classifier layers first with 20 epochs and obtained an accuracy of 90.67%. Then we fine-tuned it with another 37 epochs to improve it, and we could reach 92.67% accuracy. Further increasing of epochs' number could not improve accuracy, but instead, the model could fall in overfitting. To enhance the system and avoid overfitting, we increased our dataset by applying some data augmentation on the existing images. Following face detection, we employed face alignment to rotate and center faces in pictures as needed; this stage aids in obtaining better results and a more accurate representation of extracted facial features. For speeding up the training process, we cropped all faces and extracted them from images before feeding them to the model.

Using the Fast MTCNN algorithm in real-time testing provided better results in terms of speed, where in the normal MTCNN the average fps (frames per second) was 10, whereas in Fast MTCNN, it increased four times. Previous tables reflect the robustness of our proposed work over the other analyzed models. Comparing the speed of our system with the other state-of-art models, ours recognize faces with higher fps. Considering accuracy, the results were very close to each other.

VI. Conclusion

The work presented in this paper uses a combination of two deep convolutional neural networks, MTCNN for face detection and MobileNetV2 for feature extraction, to perform a real-time facial recognition system. Using the transfer learning technique, pre-trained model weights of MobileNetV2 on the ImageNet dataset were utilized as initial values for feature extraction layers after removing its classifier and adding new classifier layers. The model then continued learning from that point on our dataset, consisting of 1500 images for 15 classes. The proposed system was tested on images and live videos and achieved an accuracy of 92.67%, with an average fps of 11.68.

References

[1] Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, **I**. https://doi.org/10.1109/CVPR.2005.177

[2] Ojala, T., Pietikäinen, M., & Harwood, D. (1994). Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. *Proceedings - International Conference on Pattern Recognition*, **3**, 582–585. https://doi.org/10.1109/ICPR.1994.576366

[3] Bouwmans, T., Silva, C., Marghes, C., Zitouni, M. S., Bhaskar, H., & Frelicot, C. (2018). On the role and the importance of features for background modeling and foreground detection. *Computer Science Review*, **28**, 26–91. https://doi.org/10.1016/j.cosrev.2018.01.004

[4] Lenc, L., & Král, P. (2015). Automatic face recognition system based on the SIFT features. *Computers and Electrical Engineering*, **46**, 256–272. https://doi.org/10.1016/j.compeleceng.2015.01.014

[5] Bartlett, M. S., Movellan, J. R., & Sejnowski, T. J. (2002). Face recognition by independent component analysis. *IEEE Transactions on Neural Networks*, **13(6)**. https://doi.org/10.1109/TNN.2002.804287

[6] Turk, M. A., & Pentland, A. P. (1991). *Face recognition using eigenfaces*. doi: 10.1109/CVPR.1991.139758

[7] Sirovich, L., & Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, **4(3)**, 519. https://doi.org/10.1364/josaa.4.000519

[8] Albawi, S., Mohammed, T. A. M., & Alzawi, S. (2017). Understanding of a Convolutional Neural Network. *Ieee*, 16.

[9] Ilyas, B. R., Mohammed, B., Khaled, M., & Miloud, K. (2019). Enhanced Face Recognition System Based on Deep CNN. *Proceedings - 2019 6th International Conference on Image and Signal Processing and Their Applications, ISPA 2019, January*. https://doi.org/10.1109/ISPA48434.2019.8966797

[10] Harikrishnan, J., Sudarsan, A., Sadashiv, A., & Remya Ajai, A. S. (2019). Vision-Face Recognition Attendance Monitoring System for Surveillance using Deep Learning Technology and Computer Vision. *Proceedings - International Conference on Vision Towards Emerging Trends in Communication and Networking, ViTECoN 2019*, 1–5. https://doi.org/10.1109/ViTECoN.2019.8899418

[11] Damale, R. C. (2018). *Face Recognition Based Attendance System Using Machine Learning Algorithms*. *Iciccs*, 414–419.

[12] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

[13] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. http://arxiv.org/abs/1704.04861

[14] Fei-Fei, L., Deng, J., & Li, K. (2010). ImageNet: Constructing a large-scale image database. *Journal of Vision*, **9(8)**, 1037–1037. https://doi.org/10.1167/9.8.1037

Balkan Journal of Electrical and Computer Engineering (BAJECE)
Volume 1 Number 1, September 2023

Designing Effective Models for COVID-19 Diagnosis through Transfer Learning and Interlayer Visualization

Cuneyt Ozdemir[1]

1 Department of Computer Engineering University of Siirt University, Siirt, Turkey,(e-mail: cozdemir@siirt.edu.tr).

Manuscript received Jun 1, 2023; accepted August 6, 2023.

Abstract— **Creating an optimal model for a specific dataset can be challenging and time-consuming. This article presents an innovative approach to model creation by leveraging transfer learning models and employing the interlayer visualization method. The study aims to overcome the complexities of designing a new model specifically for the COVID-19 dataset. Transfer learning models, which are pre-trained models, offer a practical solution due to their adaptability. However, not all layers in these models may be suitable for a given dataset, emphasizing the importance of identifying and removing unnecessary layers for successful model performance.**

Experimental studies were conducted using transfer learning models, including Densenet201 and InceptionV3, on the COVID-19 dataset. The interlayer visualization method was utilized to identify irrelevant layers, resulting in the creation of new models. The evaluation metrics demonstrated that the derived models outperformed the original transfer learning models. The Mixed3 model derived from InceptionV3 achieved a higher accuracy of 98.6%, compared to the original model's accuracy of 98.3%. Similarly, the Pool4 model derived from DenseNet201 achieved an accuracy of 98.8%, surpassing the original model's accuracy of 98.19%. Moreover, the sensitivity for detecting images belonging to the Covid-19 class reached 98.9% and 99.7% for the Mixed3 and Pool4 models, respectively.

The new models obtained through the interlayer visualization method offer several advantages, including lightweight design, faster training processes, and improved performance. This study highlights the effectiveness of leveraging transfer learning models with the interlayer visualization method for creating robust models tailored to specific datasets. The proposed approach serves as a valuable solution to the challenges associated with model creation, particularly in the context of COVID-19 diagnosis using medical imaging data.

Index Terms—**Covid-19, DenseNet201, InceptionV3, Interlayer visualization, Model pruning, Transfer learning models**

I. Introduction

Transfer Learning is a method in machine learning, utilizes pre-trained models to perform new tasks. When designing a new deep neural network architecture, transfer learning models are commonly employed [1]. This approach allows the adaptation of a pre-trained model's general features to a new task, proving particularly useful when data scarcity or large datasets hinder training a model from scratch. Transfer learning leverages features learned from one dataset for another, leading to superior results with reduced training data [2].

Transfer learning finds extensive applications in various domains, including agriculture, healthcare, image processing, natural language processing, and speech recognition [3-7]. However, not all transfer learning models are suitable for every dataset, and some models may contain unnecessary layers. Therefore, selecting the most appropriate transfer learning model for a specific dataset and identifying the optimal layers within that model are crucial for achieving successful outcomes.

The size and characteristics of the dataset influence the selection of an appropriate model. Larger datasets benefit from complex and deep models, while smaller datasets are better suited for simpler models. Hence, the choice of transfer learning models should align with the specific characteristics of the dataset.

Despite the advantages of transfer learning, there are certain drawbacks to consider. Firstly, not all layers in a pre-trained model are necessary for a new task, and some layers may introduce redundancy or contribute to overfitting depending on the dataset's characteristics [8]. Therefore, it is essential to remove or modify unnecessary layers to enhance the performance of the pre-trained model. Adapting pre-trained models to match the size and complexity of the dataset is crucial, and having a clear understanding of which layers to modify or remove is essential for optimizing model performance.

In some cases, employing all layers of a pre-trained model can result in longer training times or reduced model efficiency. Thus, pre-trained models should be adaptable to the dataset's size and complexity. Nevertheless, transfer learning models require less time and computational power compared to training a model from scratch for a new task. Moreover, pre-trained models typically capture more general features due to their training on larger datasets. These features can be fine-tuned for the new task, resulting in a more accurate and efficient model.

To address unnecessary or redundant information in transfer learning models, the inter-layer visualization technique, known as model pruning, is employed. Model pruning aims to eliminate unnecessary parameters or layers from a model. It identifies redundant layers by analyzing the model's layers or filters, eliminating similar or weak activations. Model pruning reduces the model's size, shortens training time, and improves its generalization ability.

Zeiler et al. [9] utilized the AlexNet deep learning model to visualize how each layer learns its features and represents each feature map. This visualization process aimed to comprehend the combination of lower-level features learned by certain layers in higher-level layers, leading to the production of the final output. The study investigated the impact of removing specific layers on the model's performance, revealing a significant drop in performance when the first two layers were removed. This result demonstrated that the initial layers of the model learn lower-level features, while higher-level layers learn higher-level features through their integration.

Urban et al. [10] investigated the effect of removing layers in deep learning models on the model's performance. The study showed that, in certain cases, the number of layers in deep learning models could be reduced, and some models achieved high performance even with a few layers.

Bau et al. [11] introduced an approach to comprehend the features of layers within deep learning models. This approach introduced the concept of "neuron activation" to quantify the relevance of each feature to a specific object or concept.

Li et al. [12] developed a method for visualizing the loss landscape of various deep learning models. This method enables an understanding of the impact of removing or modifying layers on model performance by visually demonstrating the effects of different layers in the model.

This study aims to create new models and enhance their performance by conducting inter-layer visualization on transfer learning models. The inter-layer visualization method can identify layers that carry less information by visualizing the outputs of each layer in the model. When a model is provided with input, inter-layer visualization visualizes the activations generated by each layer of the model. These visualizations assist in determining the importance of each layer and which layers contribute more significantly to the model's performance.

COVID-19 X-ray images were employed to examine the performance of model pruning operations on transfer learning models. COVID-19, a viral disease caused by SARS-CoV-2, was initially identified in Wuhan, China. While the initial cases were reported in Wuhan, the precise origins and transmission of the virus remain subjects of ongoing scientific research and investigation.

COVID-19 primarily affects the respiratory system and belongs to the coronavirus family. It is a novel virus for humans, posing a significant risk, particularly to elderly individuals and those with chronic illnesses. COVID-19 can induce various symptoms, including fever, cough, fatigue, shortness of breath, muscle aches, headaches, and loss of taste or smell, among others. The outbreak of this disease rapidly escalated into a major worldwide health crisis, profoundly impacting healthcare systems and economies of numerous countries. Consequently, COVID-19 continues to be a globally-relevant research topic [13-14].

The health effects of COVID-19 are not limited to the respiratory system. The virus can cause brain inflammation, psychological disorders, cardiovascular diseases, kidney failure, and damage to other organs. PCR or antigen tests are commonly employed for COVID-19 diagnosis. With the COVID-19 pandemic becoming a global health issue, artificial intelligence techniques, such as deep learning, have started to play a crucial role in addressing this problem [15,16].

Ertugrul et al. [17] proposed a machine learning-based automatic diagnosis system that utilizes X-ray images to detect COVID-19-related diseases. The system employed robust texture features, such as Histogram of Oriented Gradients, Law's Tissue Energy Measure, Gabor Wavelet Transform, Gray Level Co-occurrence Matrix, and Local Binary Pattern, to train a random neural network. This approach facilitated a rapid and robust diagnostic process for COVID-19 by extracting identifying indicators from the two-dimensional space of X-ray images from virus patients.

Kaya et al. [18] presented a novel approach for COVID-19 detection using X-ray images. They introduced the Angle Transform (AT) method, which captures the angle information between pixels in the images. The AT method generated eight different images per dataset image, which were then used to train a hybrid deep learning model combining GoogleNet and LSTM. The proposed approach achieved a high classification accuracy of 98.97% and demonstrated success in COVID-19 detection using chest X-ray images.

This study involves the creation of new models through model pruning operations on transfer learning models using the COVID-19 dataset. Experimental studies have been conducted using these newly obtained models, and the performance of transfer learning models and the new models derived from them through model pruning methods has been observed.

II. DATASET AND PREPROCESSING

A. Dataset

A group of researchers from Doha, Qatar and Dhaka University, Bangladesh have shared a dataset that contains images. These images consist of chest X-rays for COVID-19 positive cases, Normal images, and Viral Pneumonia images. Table I provides further details regarding the dataset [19].

TABLE I
Distribution of images according to classes

Image Class	Number
COVID 19	3616
Normal	10192
Viral Pneumonia	1345

The distribution of visual data among different classes exhibits an imbalance, as illustrated in Table I. This imbalance can lead to incorrect learning by the model and subsequently generate inaccurate results. In an effort to mitigate this issue, only 3500 normal images from the dataset were utilized. The distribution of the dataset is depicted in Figure 1, encompassing a total of 8461 images.

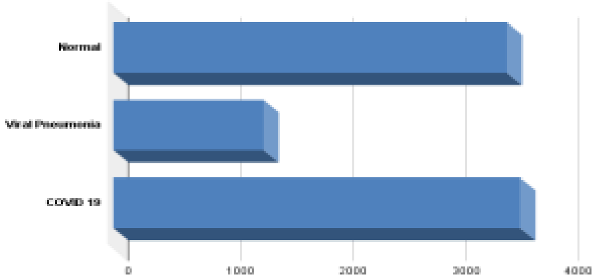
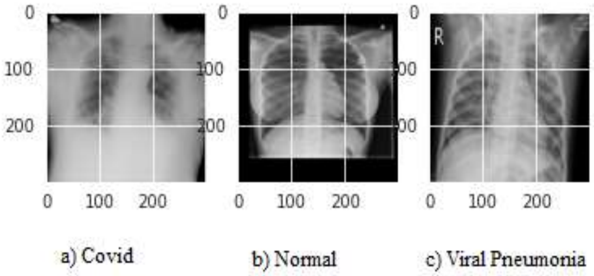


Fig.1. Dataset distribution

Figure 2 displays image examples from three distinct classification clusters within the dataset. These image samples were obtained following the application of data augmentation techniques on the images.



a) Covid b) Normal c) Viral Pneumonia

Fig 2. Examples of 3 different image classes

Several studies have been conducted in the literature using this dataset:

Aslan et al. [20] conducted a study where they utilized features extracted from various well-known Convolutional Neural Network (CNN) models, such as AlexNet, Inceptionresnetv2, ResNet18, Inceptionv3, ResNet50, MobileNetv2, Densenet201, and GoogleNet. These extracted features were then used for classification using different machine learning algorithms, including SVM, k-NN, Naive Bayes, and Decision Tree. The DenseNet + SVM method achieved a classification accuracy of 96.29%.

Sohan et al. [21] employed ResNet-18 and VGG16 transfer learning models on the same dataset, achieving an accuracy of 97% in their study.

Sahlol et al. [22] proposed a hybrid classification approach that utilized the Marine Predators algorithm to select a swarm-based feature from the InceptionV3 transfer learning model. In their study, they achieved an accuracy of 98.7% and an F-score of 98.2%.

Abdollahi et al. [23] achieved an accuracy of 97.99% in their study using the VGG16 transfer learning model.

Abdrakhmanov et al. [24] employed few-shot learning techniques to classify images with a small amount of training data and achieved a classification accuracy of 97.7%.

These studies demonstrate the effectiveness of various transfer learning models and approaches in achieving high accuracies on the dataset.

B. Pre Processing

Before being inputted into the model, the images underwent pre-processing steps. During this stage, the image dimensions were resized to 224x224 pixels. To prevent overfitting and enhance the diversity of the dataset for improved model performance, data augmentation techniques were employed. The data augmentation process included applying a 15% rotation, 10% shifting to the right and left, horizontal flipping, and a 20% zoom in and out of the images.

Following the pre-processing and data augmentation steps, the images were split into three distinct sets: training, validation, and testing. Specifically, 15% of the images were allocated for testing, while the remaining 85% were assigned to the training set. Within the training set, an additional 15% of the images were set aside for validation purposes.

III. METHODOLOGY

In order to address the challenges and time constraints associated with building a model from scratch for our specific dataset, we turned to the powerful technique of transfer learning. Transfer learning involves utilizing the knowledge and features learned by a pre-trained model on a different task or dataset and applying it to a new task or dataset, aiming to achieve improved performance.

Transfer learning brings several advantages to the table, as it allows us to leverage the wealth of information captured by pre-trained models and adapt them to our specific task. By employing transfer learning, we aimed to overcome the difficulties inherent in starting from scratch and enhance the performance of our model on our dataset.

In our experimental studies, the primary objective was to identify the models that exhibited the best performance on our dataset. To this end, we employed transfer learning models, and the performance metrics of these models when applied to our dataset can be found in Table II. To optimize the training process, we utilized the Adam optimization function with a fixed learning rate of 0.0003. Furthermore, we implemented the ReduceLROnPlateau and EarlyStopping methods, which dynamically reduce the learning rate and automatically halt the training of the model when the accuracy fails to improve over a certain number of epochs.

The utilization of transfer learning, along with the aforementioned optimization techniques, enabled us to effectively leverage pre-existing knowledge and adapt it to our specific task. This approach not only accelerated the training process but also allowed us to achieve superior results on our dataset. The detailed performance metrics of the transfer learning models applied in our experimental studies are presented in Table II, providing insights into the effectiveness of our approach.

TABLE II
PERFORMANCE OF TRANSFER LEARNING MODELS

	Accuracy	**Precision**	**Recal**	**F1-Score**

Xception	0.9779	0.9784	0.9779	0.9779
InceptionV3	**0.9832**	0.9832	0.9832	0.9832
ResNet50V2	0.9799	0.9799	0.9799	0.9799
DenseNet201	**0.9819**	0.9819	0.9819	0.9819

According to Table II, the InceptionV3 and DenseNet201 transfer learning models achieved the best results for the dataset. After identifying these models through initial experimental studies, model pruning was conducted on them.

Model pruning involves removing unnecessary layers from transfer learning models that are not essential for the dataset. Various methods can be employed for model pruning. In this study, the inter-layer visualization or data visualization method was utilized. The inter-layer visualization method generates visual representations of the output produced by each layer involved in processing the input image of an image classification model. By analyzing the output of each layer, we gain insights into the features learned by the model at each stage and how these features are propagated to subsequent layers. For example, it helps us understand that the initial layers generally learn object contours or colors, while deeper layers focus on more specific object features. This information assists in identifying the layers where learning occurs and those where it does not.

In the inter-layer visualization method, the appearance of an image in all layers of the DenseNet201 and InceptionV3 models trained on the dataset was examined. Through this examination, the layers in which learning occurred were determined in these two models, and the layers where learning did not occur were removed from the models. Figure 3 illustrates the image appearance at different layers of the InceptionV3 model for the dataset. As depicted in Figure 3, learning did not occur after the mixed3 layer.

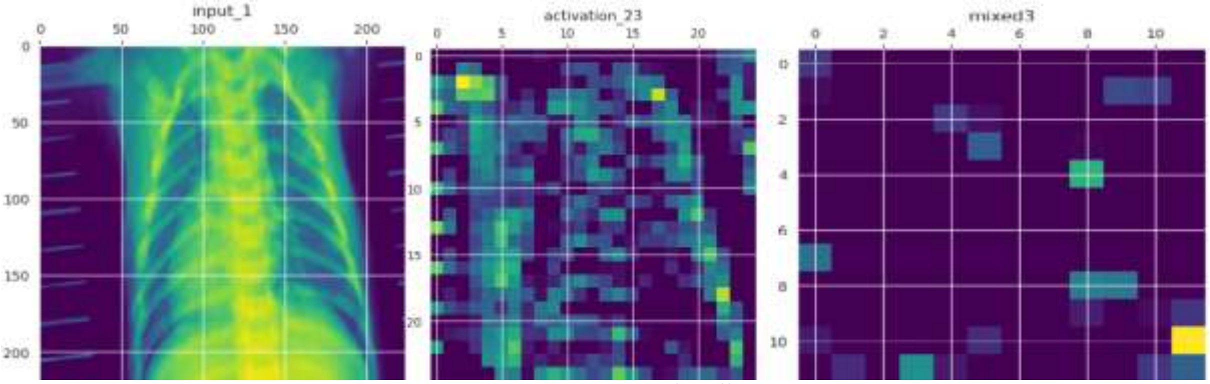


Fig 3. Examples of 3 different image classes

As a result of the process, it was observed that learning occurred up to the mixed3 layer in the InceptionV3 model and up to the pool4 layer in the DenseNet201 model. Based on this observation, new and modified models were created from the transfer learning models.

In the Mixed3 model, only the layers up to the mixed3 layer of the InceptionV3 model were utilized, and the remaining layers were discarded. Similarly, in the Pool4 model, only the layers up to the pool4 layer from the DenseNet201 model were retained, and the rest were removed to obtain a new model.

Figure 4 illustrates the structure of the new Convolutional Neural Network (CNN) model obtained through inter-layer image visualization from the InceptionV3 transfer learning model. The mixed3 model was used for feature extraction in the new model, which were subsequently fed into an artificial neural network. The neural network consists of two layers with 256 and 128 neurons, respectively. The output layer of the artificial neural network consists of three classes, and the Softmax activation function was employed.

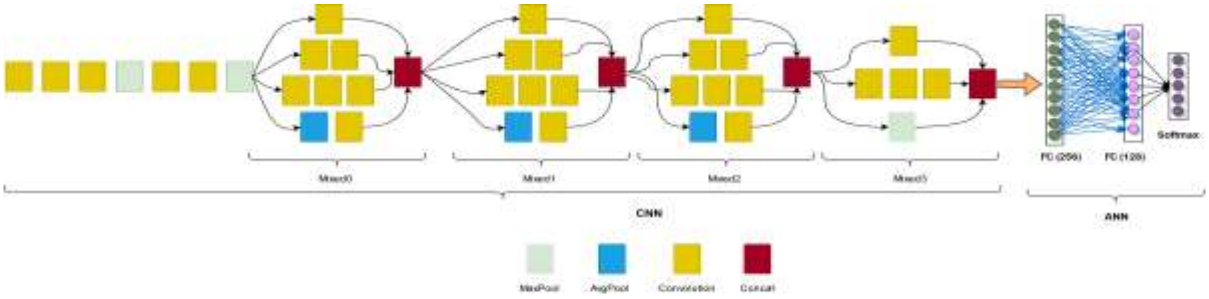


Fig 4. Mixed3 CNN model

After the new models were obtained, experimental studies were performed. The results of the experimental studies with Mixed3 and Pool4 models are shown in Table III. The train and validation, loss and accuracy graphs of the Mixed3 model are shown in Figure 5. The confusion matrix of the Mixed3 model is shown in Figure 6 and the classification report is shown in Table IV.

TABLE III
ACCURACY RESULTS OF NEW MODELS

	Train Acc	Val Acc	Test Acc
Mixed3	0.998	0.979	0.986
Pool4	**0.999**	**0.988**	**0.988**

.

TABLE IV
Mixed3 classification report

	Precision	Recall	F1-score
Normal	0.98	0.986	0.983
Viral Pneumonia	0.99	0.985	0.988
Covid	0.989	0.986	0.989
accuracy			0.986
macro avg	0.987	0.986	0.986
weighted avg	0.986	0.986	0.986

TABLE V
Pool4 classification report

	Precision	Recall	F1-score
Normal	0.975	0.997	0.986
Viral Pneumonia	1	0.963	0.981
Covid	0.997	0.989	0.993
accuracy			0.988
macro avg	0.991	0.983	0.987
weighted avg	0.988	0.988	0.988

The evaluation of classification performance was conducted using various metrics, including accuracy, precision, recall, and F1 score for each class. Additionally, the classification report provides average precision, recall, and F1 score, offering an overall assessment of the model's performance across all classes. Table IV demonstrates that the Mixed3 model derived from the InceptionV3 model achieved superior results. While the InceptionV3 model attained an accuracy rate of 98.3% on the same dataset, the Mixed3 model achieved a success rate of 98.6%. Notably, images belonging to the Covid-19 class were detected with a sensitivity of 98.9%. The classification report for the Pool4 model derived from the DenseNet201 model is displayed in Table V. The DenseNet201 model achieved an accuracy rate of 98.19%, while the Pool4 model achieved 98.8% accuracy. Remarkably, images belonging to the Covid-19 class were detected with 99.7% accuracy.

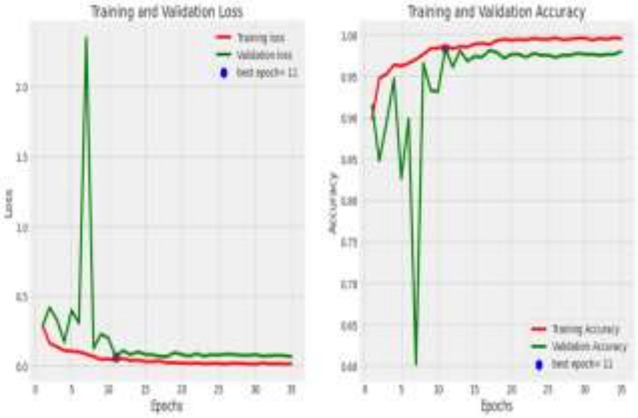


Fig 5. Loss and accuracy graph for Mixed3 model

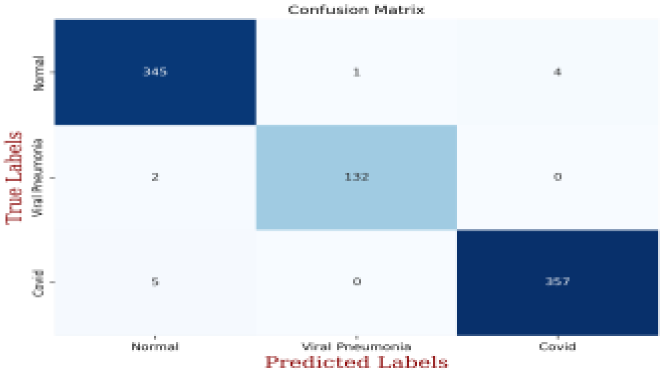


Fig 6. Confusion matrix of the Mixed3 model

Figure 7 illustrates the train and validation loss and accuracy graph of the Pool4 model, providing insights into the model's performance during the training process. Furthermore, Figure 8 presents the confusion matrix for the Pool4 model, allowing for a visualization of the model's performance in terms of correct and incorrect predictions for each class.

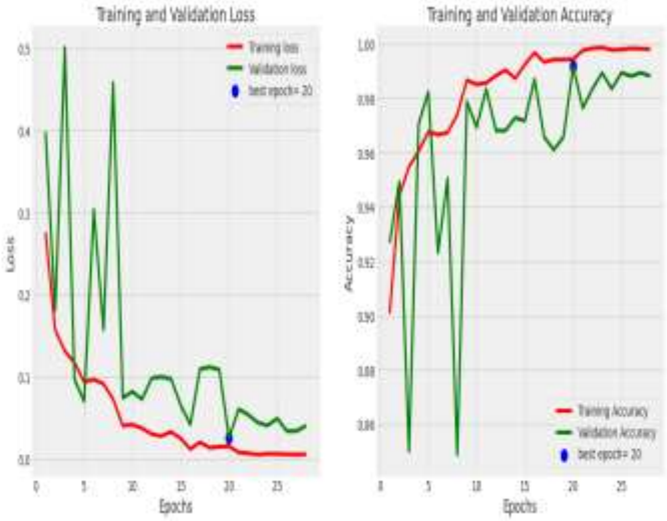
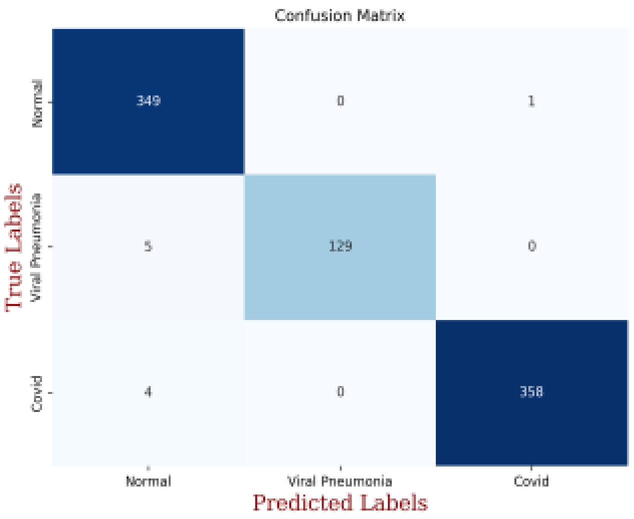


Fig 7. Loss and accuracy graph for Pool4 model



Fig

Fig 8. Confusion matrix of the Pool4 model

The experimental studies demonstrated that the new models derived from transfer learning models yielded more successful results. The InceptionV3 model achieved an accuracy of 98.32%, while the Mixed3 model achieved an even higher accuracy of 98.6%. Similarly, the DenseNet201 model achieved an accuracy of 98.19%, while its derived Pool4 model achieved a higher success rate of 98.8%. These results indicate that lightweight and fast models obtained through the interlayer image visualization method can yield more successful outcomes.

IV. CONCLUSIONS

This study proposes an approach that utilizes the interlayer visualization method from model pruning techniques to derive new models from transfer learning models, instead of designing a new model specifically for the COVID-19 dataset. The aim is to address the challenges associated with creating a new model, which is often time-consuming and laborious. By leveraging transfer learning models, the focus is on adapting them to the dataset and creating a new model. It is crucial to identify the layers in transfer learning models that may not extract relevant features from the dataset, as this significantly impacts the model's success and complexity.

To evaluate the performance, transfer learning models were employed on the COVID-19 dataset, rather than building a model from scratch. In these experimental studies, the Densenet201 and InceptionV3 transfer learning models achieved the highest scores. The interlayer visualization method was utilized to identify unnecessary layers in the transfer learning models. The experimental studies conducted with the derived new models demonstrated more successful results.

The new models created using the interlayer visualization method are lightweight, fast, and have fewer parameters. They facilitate faster training processes and yield improved performance in terms of model accuracy. The InceptionV3 model achieved an accuracy rate of 98.3%, while the Mixed3 model derived from it achieved a higher accuracy of 98.6%. Moreover, images belonging to the Covid-19 class were detected with a sensitivity of 98.9%. Similarly, the DenseNet201 model achieved an accuracy rate of 98.19%, and the Pool4 model derived using the interlayer visualization method achieved an even higher accuracy of 98.8%. Images belonging to the Covid-19 class were detected with a sensitivity of 99.7%.

This study demonstrates that the interlayer visualization method can effectively design the most suitable CNN model for a given dataset by creating new models from transfer learning models. Leveraging the interlayer visualization method provides an effective approach to overcome the challenges associated with creating a new model.

REFERENCES

[1] H. E. Kim, A. Cosa-Linan, N. Santhanam, M. Jannesari, M. E. Maros, and T. Ganslandt, ‘Transfer learning for medical image classification: a literature review’, BMC Med. Imaging, vol. 22, no. 1, p. 69, Apr. 2022.

[2] S. Atasever, N. Azginoglu, D. S. Terzi, and R. Terzi, ‘A comprehensive survey of deep learning research on medical image analysis with focus on transfer learning’, Clin. Imaging, vol. 94, pp. 18–41, Feb. 2023.

[3] I. Atas, C. Ozdemir, M. Atas, and Y. Dogan, ‘Forensic dental age estimation using modified deep learning neural network’, arXiv [eess.IV], 21-Aug-2022.

[4] Y. Dogan and H. Y. Keles, ‘Iterative facial image inpainting based on an encoder-generator architecture’, Neural Comput. Appl., vol. 34, no. 12, pp. 10001–10021, Jun. 2022.

[5] M. Ataş, C. Özdemir, İ. Ataş, B. Ak, and E. Özeroğlu, ‘Biometric identification using panoramic dental radiographic images withfew-shot learning’, TURK. J. OF ELECTR. ENG. COMPUT. SCI., vol. 30, no. 3, pp. 1115–1126, Jan. 2022.

[6] Y. Dogan and H. Yalim Keles, ‘Stability and diversity in generative adversarial networks’, in 2019 27th Signal Processing and Communications Applications Conference (SIU), Sivas, Turkey, 2019.

[7] C. Ozdemir, M. A. Gedik, and Y. Kaya, ‘Age estimation from left-hand radiographs with deep learning methods’, Trait. Du Signal, vol. 38, no. 6, pp. 1565–1574, Dec. 2021.

[8] M. Iman, K. Rasheed, and H. R. Arabnia, ‘A review of Deep Transfer Learning and recent advancements’, arXiv [cs.LG], 18-Jan-2022.

[9] M. D. Zeiler and R. Fergus, ‘Visualizing and understanding convolutional networks’, in Computer Vision – ECCV 2014, Cham: Springer International Publishing, 2014, pp. 818–833.

[10] G. Urban et al., ‘Do deep convolutional nets really need to be deep and convolutional?’, arXiv [stat.ML], 17-Mar-2016.

[11] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, ‘Network dissection: Quantifying interpretability of deep visual representations’, in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017.

[12] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, ‘Visualizing the loss landscape of neural nets’, arXiv [cs.LG], 28-Dec-2017.

[13] H. Harapan et al., ‘Coronavirus disease 2019 (COVID-19): A literature review’, J. Infect. Public Health, vol. 13, no. 5, pp. 667–673, May 2020.

[14] J. Elliott, M. Whitaker, B. Bodinier, O. Eales, S. Riley, H. Ward, ... & P. Elliott, Predictive symptoms for COVID-19 in the community: REACT-1 study of over 1 million people. PLoS medicine, 18(9), e1003777., 2021.

[15] U. Jain, Effect of COVID-19 on the Organs. Cureus, 12(8)., 2020

[16] D. L. Weiner, V. Balasubramaniam, S. I. Shah, J. R. Javier, and Pediatric Policy Council, ‘COVID-19 impact on research, lessons learned from COVID-19 research, implications for pediatric research’, Pediatr. Res., vol. 88, no. 2, pp. 148–150, Aug. 2020.

[17] Ertuğrul, Ö. F., Emrullah, A. C. A. R., Öztekin, A., & Aldemir, E. (2021). Detection of Covid-19 from X-ray images via ensemble of features extraction methods employing randomized neural networks. European Journal of Technique (EJT), 11(2), 248-254.

[18] Kaya, Y., Yiner, Z., Kaya, M., & Kuncan, F. (2022). A new approach to COVID-19 detection from X-ray images using angle transformation with GoogleNet and LSTM. Measurement Science and Technology, 33(12), 124011

[19] Kaggle. COVID-19 Radiography Database. https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database. Accessed 18 January 2023

[20] M. F. Aslan, K. Sabanci, A. Durdu, and M. F. Unlersen, ‘COVID-19 diagnosis using state-of-the-art CNN architecture features and Bayesian Optimization’, Comput. Biol. Med., vol. 142, no. 105244, p. 105244, Mar. 2022.

[21] M. F. Sohan, A. Basalamah, and M. Solaiman, ‘COVID-19 detection using machine learning: a large scale assessment of x-ray and CT image datasets’, J. Electron. Imaging, vol. 31, no. 04, Mar. 2022.

[22] A. T. Sahlol, D. Yousri, A. A. Ewees, M. A. A. Al-Qaness, R. Damasevicius, and M. A. Elaziz, ‘COVID-19 image classification using deep features and fractional-order marine predators algorithm’, Sci. Rep., vol. 10, no. 1, p. 15364, Sep. 2020.

[23] J. Abdollahi and L. Mahmoudi, ‘An artificial intelligence system for detecting the types of the epidemic from X-rays : Artificial intelligence system for detecting the types of the epidemic from X-rays’, in 2022 27th International Computer Conference, Computer Society of Iran (CSICC), Tehran, Iran, Islamic Republic of, 2022.

[24] R. Abdrakhmanov, M. Altynbekov, A. Abu, A. Shomanov, D. Viderman, and M.-H. Lee, ‘Few-shot learning approach for COVID-19 detection from X-ray images’, in 2021 16th International Conference on Electronics Computer and Computation (ICECCO), Kaskelen, Kazakhstan, 2021.

Balkan Journal of Electrical and Computer Engineering (BAJECE)
Volume 1 Number 1, September 2023

Intelligent Video Surveillance Systems: A Survey

Olayemi Olaniyi[1], Shefiu Ganiyuo[2] and S. J. Akam[3]

1 Department of Computer Engineering, Federal University of Technology, Minna, Nigeria, (e-mail: mikail.olaniyi@futminna.edu.ng)
2 Department of Information Technology, Federal University of Technology, Minna, Nigeria, (e-mail: shefiu.ganiyu@futminna.edu.ng)
3 Department of Computer Engineering, Federal University of Technology, Minna, Nigeria, (e-mail: sundayjames115@gmail.com)

Manuscript received Dec 20, 2022; accepted August 27, 2023.

Abstract— **Over the years, the need for intelligent video surveillance has increased in other to enhance security and safety in the society. Government, private organization and individual need to make sure there properties are kept safe from intruders and as such intelligent video surveillance plays a key role in ensuring that this is achieved. Intelligent video surveillance is embedded with the capability of providing real time intelligent surveillance and also automatically provide analysis of video and image data without human operation. In the development of such systems, computer vision, machine learning and deep learning plays a vital role in achieving this. Therefore, this paper presents a survey on intelligent video surveillance system, overview of background concept and discussion on object detections and classification, tracking and deep networks. Also, this paper presents an efficient and faster object detection and classification techniques for intelligent video surveillance.**

Index Terms— **CNN, Deep learning, Detection, Video, Surveillance**

INTRODUCTION

The need for day to-day security cannot be over emphasized. Surveillance is a very important aspect of security and plays a vital role in ensuring that lives and properties are kept safe. In the early days, surveillance system relied on humans for its operations [1]. With the advent of video surveillance systems, governments, individuals and various organizations across society use this system to keep track of various activities for the sole aim of security and safety [2]. In today's smart cities, video surveillance is used for inventory control in retail outlets, security on corporate and educational campuses, and both security and demand monitoring on homes and rapid transit networks. Intelligent video surveillance systems are interdisciplinary systems that include electronic (sensing devices), pattern recognition and computer vision, networking, artificial intelligence, and communication [2].

Manual monitoring by human operators is an inefficient or even impractical solution because human resources are expensive and have limited capabilities. The goal of an intelligent video surveillance system is to automatically monitor people, property, and the environment without the need for human intervention [2]. As a result, this monitoring task entails automatically detecting and classifying objects (either humans or household pets), as well as performing additional analysis and taking actions. Image processing and artificial intelligence (deep learning) techniques are important in the development of intelligent video systems [3].

With advancements in deep learning, particularly Convolution Neural Network (CNN) and in computer vision applications, the accuracy of object detection and classification has improved dramatically for intelligent video surveillance. [4]. Neural network algorithms which offer state-of-the-art performance in classification and object detection are widely used in intelligent video surveillance for intrusion detection. This paper presents a review of literature for intelligent video surveillance and general overview of efficient and accurate method object detection and classification.

I. Background of Related Work

This section provides a review of the literatures on different object detection and classification techniques as well as the advancements in neural network architectures. Recently, object detection and classification have drawn the attentions of many researchers into deep learning and its techniques. Several deep learning techniques based on CNN for real-time classification and recognition in computer vision have lately been proposed. Their performances, however, is dependent on the scenarios in which they are used [4].

A. Object detection and classification

Object classification refers to the task of assigning a label to an image. It includes the following techniques: k-nearest neighbors (KNN), Neural networks (NN), Naive Bayes classifier and CNN [2]. Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, animals, or cars) in digital images and video object. It also serves as a means of focusing attention on an object. It is a well-researched domains of which include face detection and pedestrian detection. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance. The ability to automatically detect and classify object is one of key component in intelligent video surveillance system. For a machine (computer), detecting object like human is a hard job due to wide range of possible appearance as result of changing articulated pose, clothing, lighting and background [3]. Deep convolutional neural networks (DCNNs) have proven very effective for computer vision in object detection and classification tasks [6]. Furthermore, several deep learning techniques were recently proposed based CNN for real-time detection and classification in computer vision [1]. Computational complexity and object resolution requirements of CNNs limit their applicability in wide-view video surveillance settings where objects are small [4].

B. Supervised Classification

Supervised classification is a classification of digital images where the classes of objects on Earth's surface are known priory in certain limited areas of the image (areas that are called test areas or sites). These areas fall into patterns and then rules are developed which will be extended to parts unknown in the image. Supervised classification can work with several types of algorithms, the average minimum distance algorithm (minimum distance to means), algorithm parallelepiped (Multi-level slicing) and algorithm Gaussian of maximum similarity (maximum likelihood), [7]

$$y_i(x) = Inp(w_i) - \frac{1}{2\text{Inp}\,|\Sigma_i\,|} - 1/2(x - m_i)^{y} \sum\nolimits_i^{-1} (x - m_i) \qquad (1)$$

C. Unsupervised Classification

Unsupervised classification of digital images requires the creation of groups of pixels that represent geographic features, without previously knowing what is classified, and subsequently verifying the meaning of the pixels in the digital image researched. It is based on mathematical algorithms K-means and algorithm Iterative Self Organizing Data Analysis (ISODATA), in the present study having used ISODATA algorithm [5].

$$SD_{xyc} = \sqrt{\sum_{i=1}^{n} \left(\mu_{ci} - X_{xyi}\right)^2} \qquad (2)$$

D. Object Tracking

Object tracking is a deep learning application in which the program takes an initial set of object detections and creates a unique identification for each of the initial detections before tracking the detected objects as they move around frames in a video [6]. Object tracking has been an interesting field of research due to its challenges and importance. It is the main aim of intelligent video surveillance system. Recently, tracking by detection methods had emerged as immediate effect of deep learning with remarkable achievements in object detection. For example, [7] used CNN features for people tracking by detection. They used a model that is based on simple Euclidean distance. The result obtain shows that simple minimum Euclidean distance association performs well compared to SNN in most scenes. Fig 1 shows an example of the result obtained.

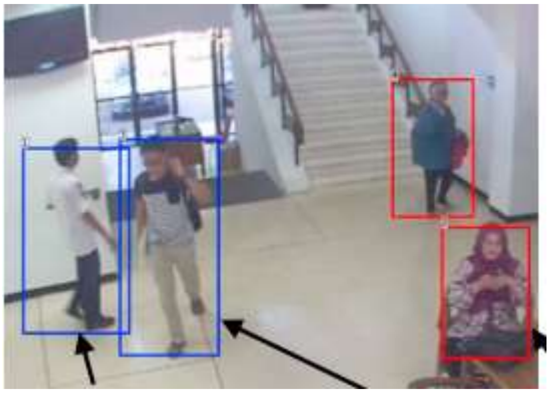


Fig.1. People tracking by detection [8]

II. Deep learning TECHNIQUES

A. Convolutional Neural Network

Convolutional neural network (CNN, convNet), is a class of deep neural network, widely applicable to image analysis. It was inspired by visual system's structure. The first computational models based on this local connectivity between neurons and on hierarchically organized transformations of the image are found in Neoncognition, it describes that when neurons with the same parameters are applied on patches of the previous layer at different locations, a form of translational invariance is acquired [32] The CNN-based architecture is built up on ConvNets with several layers of convolutional filters, RELU layer and pooling algorithms [33].

$$y(x,y) * g(x,y) = \sum_{i=-\infty}^{\infty} \sum_{j=\infty}^{\infty} f(i,j) * g(x-i, y-j) \qquad (3)$$

CNN consist of three main types of neural layers; (i) convolutional layers, (ii) pooling layers, and (iii) fully connected layers.

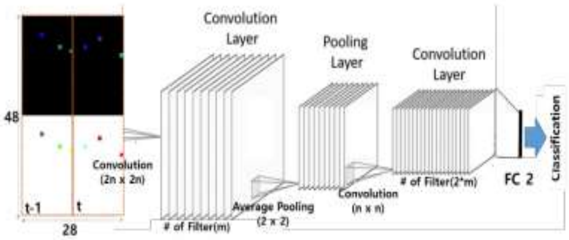


Fig.2. CNN Architecture [9].

B. Fully Connected Layer

Neurons in a fully connected layer have full connections to all activation in the previous layer, as their name implies. Their activation can hence be computed with a matrix multiplication followed by a bias offset. Fully connected layers eventually convert the 2D feature maps into a 1D feature vector. The derived vector either could be fed forward into a certain number of categories for classification or could be considered as a feature vector for further processing [34]

$$y^{\beta} = \delta CWy^{(d-1)} + b) \tag{4}$$

If the input to d−1 convolutional layer is of dimension $N \times N$ and the receptive field of units at a specific plane of convolutional layer d is of dimension $m \times m$, then the constructed feature map will be a matrix of dimensions $(N - m + 1) \times (N - m + 1)$. Specifically, the element of feature map at (i, j) location will be;

$$Y_{ij}{}^{\beta} = \delta CX_{ij}{}^{(d)} + b) \tag{5}$$

C. *Convolutional Layers*

Convolutional layers are considered the core building blocks of CNN architectures. The Figure 3 illustrates, convolutional layers transform the input data by using a patch of locally connecting neurons from the previous layer. The layer will compute a dot product between the region of the neurons in the input layer and the weights to which they are locally connected in the output layer.

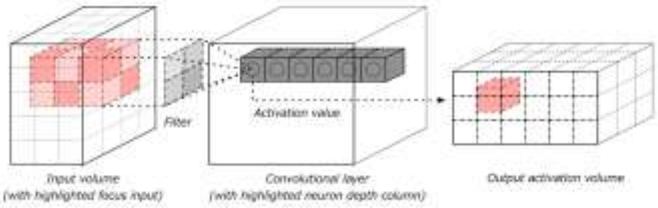


Fig.3. Convolution layer with input and output [10].

D. *Pooling Layer*

After the convolutional layer, a new layer called a pooling layer is introduced. After a nonlinearity (ReLU) has been applied to the feature maps produced by a convolutional layer. The largest pool is used for maximum pooling, The goal of maximum pooling is to reduce the size of an input image by down sampling it [35].

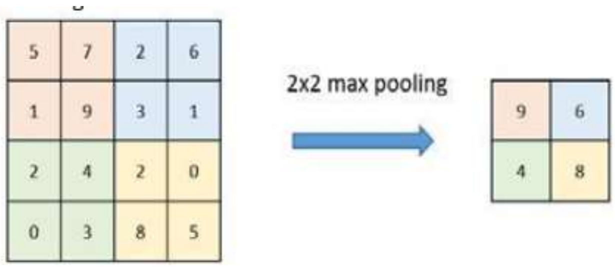


Fig.3. Pooling layer [35].

E. *Long Short-Term Memory networks (LSTM)*

The LSTM departed from typical neuron-based neural network architectures and instead introduced the concept of a memory cell. The memory cell can retain its value for a short or long time as a function of its inputs, which allows the cell to remember what's important and not just its last computed value.

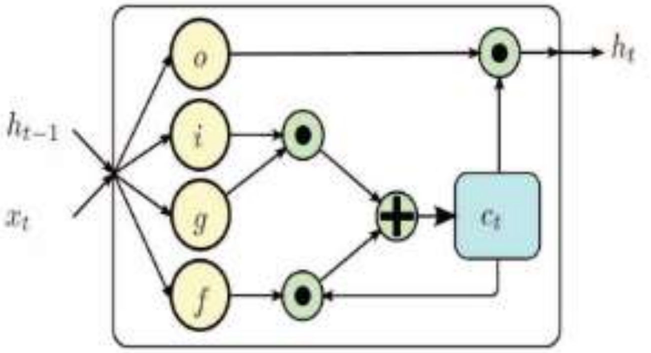


Fig.4. LSTM networks [12].

F. *R Self-organized maps (SOM)*

Self-organized map (SOM) was invented by Dr. Teuvo Kohonen in 1982 and was popularly known as the Kohonen map. SOM is an unsupervised neural network that creates clusters of the input data set by reducing the dimensionality of the input. SOMs vary from the traditional artificial neural network in quite a few ways (Jones, 2017).

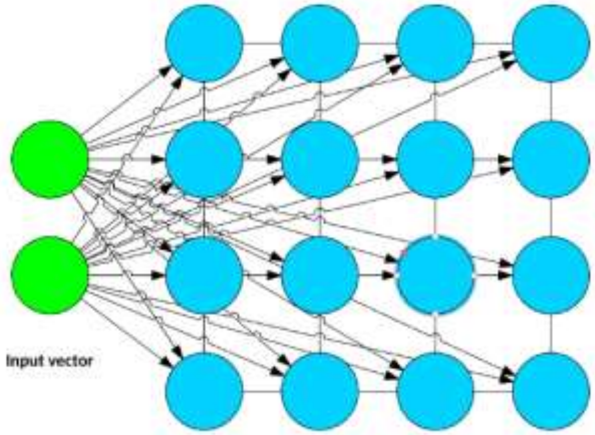


Fig.5. Self-organized maps [11].

III. Review of Related works

Several researches have been carried out on intelligent video surveillance and object detection, classification, and tracking. Authors in [13] developed an object detector using multi regional convolutional neural network (RCNN) detectors. Object detection and tracking which plays a very important role in surveillance for traffic control, counting and public security field. The system adopted multi-detector model based on faster RCNN, a combination of CNN and Amulet is use to extract the raw feature from image, region proposal network (RPN) is use to predict the expected Region of interest (ROI). Multiple detection is used to detect the image. However, the system performed better when detection only vehicles.

Similarly, [14] designed and implemented an intelligent surveillance system with smartphone enabled. The system uses a passive infrared (PIR) sensor and a microcontroller (MCU) attached to a smartphone through the MCU for motion detection. When motion is detected, video is captured and the footage is sent to the user via short message service (SMS). The surveillance record is stored in a cloud and the link to the record is also sent to the user via email. The developed system ensures efficient use of memory by storing the record in a cloud. It is cost-effective and also offers efficient energy use as the camera is only activated when motion is detected by the PIR sensor. However, the developed system cannot efficiently differentiate radiation changes between humans, household pets, or other animals.

Also, [15] developed an intelligent surveillance system for low-cost convolutional neural network (CNN) design. The developed system makes use of hardware accelerator known as Neural Compute Stick (NCS) with ROCK64 for high-speed calculation of images. A lightweight MobileNet network is use to extract the features and the classify images. The authors used the NCS to load a single shot multibox detector (SSD) network for human detection. Also, the Darknet architecture of You Only Look Once (YOLO) is used for extraction and classification of images and combine with SSD to create bounding box for the region of interest of the detected images. A simple mail transfer protocol was used to send email to deliver the detected object. However, the system has low human recognition accuracy and therefore cannot be used for other intelligent surveillance applications

Furthermore, [16] developed enhanced background subtraction algorithm for smart surveillance system using adaptive gaussian mixture technique. The smart system can efficiently detect motion and detect object by means of background subtraction with illumination change. However, the developed system cannot different between human and home pet.

In addition, [17] developed a Real-Time Action Detection in Video Surveillance using Sub Action Descriptor with Multi-CNN. The system presented a novel real-world surveillance video dataset and a new approach to real-time action detection in video surveillance system. The joint space of the sub-action descriptor was not considered. Also, more powerful temporal feature methods, such as a skin-color MHI or optical flow, and other deep architectures of CNNs are not considered.

Similarly, [10] developed an activity recognition using temporal optical flow convolutional features and multi multiplayer LSTM. The activity recognition framework for industrial systems proposed with a trained map CNN model help to select only the salient region that are activated for persons in the video frame which reduce verboseness and ambiguity of information in video frame. However, the system focuses on the activity recognition of a single person in a video frame.

Surveillance video analysis for store-base using deep learning techniques proposed by [36]. A skeleton recognition algorithm is adopted in place of object detection algorithm to conquer occlusion problem for gathering sufficient customer information and realizing crowd counting and density map drawing. For human tracking and counting, multiple human tracking algorithm and human re-identification (ReID) technology are adopted. However, the system was only trained to track only human object in the area of surveillance

Also, [3] developed people tracking system Using CNN Features. They represented each person with 4096 Faster-RCNN feature vectors, and the Euclidean distance method was used to calculate the distance between two feature vectors of each input pair. A pair is considered the same person if their Euclidean distance is a minimum. This is due to the assumption that convolutional features of similar objects generated by Faster-RCNN should be quite similar compared to features of dissimilar objects.

Furthermore, [19] suggested A General-Purpose Intelligent Surveillance System for Mobile Devices using Deep Learning. The developed system was divided into two: a detection and a classification module. The detection module combined background subtraction techniques, optical flow and recursively estimated density. The classification module is based on a convolutional neural network (CNN) used to classify objects into one of the seven predefined categories using a pre-trained CNN. However, the dataset is enormous for the targeted mobile device and so it become a problem to process in a real time.

In addition, [20] designed and developed an Edge Intelligence-Assisted Smoke Detection in Foggy Surveillance Environments. The system was developed using the architecture of convolutional neural network (CNN) for detecting smoke in video streams. Pre-trained MobileNet model was trained on ImageNet dataset which focus on trying to achieve accuracy and eliminating rate of false alarm in Foggy Surveillance Environments. However, the method can only be applied for smoke detection.

Similarly, [22] presented the use of Adaboost and CNN in crowded surveillance environment for people counting based on head detection. In this system three module were used to achieve people counting. The module includes: Two off-line training and one online detection stage. The first off-line training, Adaboost algorithm is adopted to learn a fast-cascaded head detector with Histogram of oriented gradients (HOG) feature. In the other off-line training, a CNN is trained using a new dataset gotten from the detection result in applying the cascaded head detector to the original dataset. Then in the online detection stage, the cascaded head detector is applied to the test image to get head proposal then they post-process. However, the method used is prone to uniform noise. Weak classifiers being too weak can lead to low margins and overfitting

In addition, [23] developed a Smart Surveillance as an Edge Network Service: from Harr-Cascade, SVM to a Lightweight CNN. The system uses histogram Oriented Gradient (HOG) and Support vector Machine (SVM) algorithm for fast and accurate human detection. The system also uses Harr cascade, Harr-like feature made up of three shapes: two rectangular features, three rectangular features and four rectangular features alongside Lightweight CNN as the classifier trained with keras dataset. However, the Harr cascade has high rate of complexity, result obtain are highly tricky. The Haar Cascaded, HOG+SVM, GoogleNet and L-CNN required a lot of processing power and can make the system quite slow for video surveillance.

Also, [24] implemented a video structured description technology-based intelligence analysis of surveillance videos for public security applications. A pre-trained CNN architecture was adapted for tracking and re-identification of people and they analyze the result with CUHK03 dataset. Finally provided both manually cropped images and automatically detected bounding boxes with DPM detector, which respectively contains 13,164 images of 1360 pedestrians captured by six surveillance cameras. However, the summarization of the video stream is limited to 18 fps processing rate and cannot be combine with spectrum sensing technologies for smarter surveillance.

Furthermore, [11] presented an Efficient CNN based summarization of surveillance videos for resource-constrained devices. The study investigated deep features for shot segmentation and intelligently divide the video stream into meaningful shots. Deep features were extracted from two consecutive frames to determine whether the underlying frames belong to the same or different shot. Features were extracted from the fully connected layer (FC7) of CNN model which is trained using MobileNet architecture (version 2) on ImageNet dataset. However, the summarization of video stream is limited to 18 fps processing rate and cannot be combine with spectrum sensing technologies for smarter surveillance.

In addition, [25] developed a Kernel ELM and CNN based Facial Age Estimation. They introduced a two-level system for apparent age estimation from facial images. Then first classify samples into overlapping age groups. Within each group, the apparent age is estimated with local repressors, whose outputs are then fused for the final estimate. They use a deformable parts model-based face detector, and features from a pre-trained deep convolutional network. Kernel extreme learning machines are used for classification. However, the system cannot handle real and apparent age estimation task, and only uses a pre- trained convolutional neural network and cannot train a convolutional neural network by itself

Also, [26] designed and developed a surveillance system using CNN for face recognition with object, human and face detection. They developed a surveillance system using convolutional neural network (CNN). Region of object they considered in an entire image is picked by object detection and discriminate whether the area is human or not human or human face and then analyze his movement if the detected object is a human. However, among several frames, there are successful and unsuccessful ones. It is difficult to judge why object disappear.

Similarly, [27] implemented a vegetable category recognition system using Deep Neural Network. They implemented a caffe framework based on convolutional neural network (CNN) for the system classification and used Deep Neural Network (DNN) for the vegetable category recognition. However, as the number of iterations increases the performance of the system decreases.

Also, [28] presented an adaptive feature learning CNN for behavior recognition in crowd scene. A 3D scale convolutional neural network (3DSCNN) is implemented on crowd video scene, the 3D-CNN was used in a large-scale supervised crowd dataset which optimized convolutional architectures settings. The outcomes from 3DS-CNN captured information related to objects, scenes, and actions in a video, making them useful for different applications that do not fine tune the architectural setup. However, the system cannot handle action recognition with available dataset with variation in temporal and scale information.

In addition, [29] designed HOG-CNN based on real time face recognition. They used HOG-CNN model to recognize faces. Histogram of Oriented Gradient (HOG) was used as the feature extractor, also for detecting all the faces in the image and the CNN is used as the training algorithm for classifying the images. However, the system made consideration to only face detection of humans.

Furthermore, [30] designed and implemented an engineering vehicles detection based on faster R-CNN for power grid surveillance. CNN methods were divided into two categories, one is the two-stage methods based on region proposal and the other is the one-stage methods based on regression. The feature extraction part of these methods is done by the convolutional neural network. Some methods are based on region proposal such as R-CNN, Fast R-CNN and SPPnet, which adopt selective search algorithm to generate candidate boxes. Also, YOLO was used as the topmost feature map to predict confidences and bounding boxes for all categories over a fixed grid. SSD detects multiple categories by a single evaluation of the input image. However, the architecture used to train the system cannot handle complex application scenes.

Similarly, [31] presented an implementation of Machine Learning for Gender Detection using CNN on Raspberry Pi Platform. The implementation of the system is based on the architecture of convolutional neural network (CNN) and solution permits users to extract some relevant information from the visual data containing image labelling, face and landmarks detection, optical character recognition (OCR). Also REST API was used to interact with Google’s cloud vision platform. The real-time implementation of the hardware as well as software solution was done on a Raspberry Pi 3 model B+ board with Pi Camera module. However, the system cannot identify human from their movement as well as their facial properties. From the foregoing, the ability of intelligent video surveillance systems were to detect and distinguish between human intruder from home pet in the area of surveillance with accuracy and improved speed but with false alarm and failure to notify the user with the right intruder detected. This paper tries to tackle such limitations presented by [8], [2], [9], [10] by making use of raspberry pi and faster object detection and classification technique to improve video surveillance system.

IV. *Proposed Design and Methodology*

A. *Methodology*

To implement the proposed system for intelligent video surveillance, Faster R-CNN architecture will be considered the architecture is fast, accurate and suitable for detecting and classifying humans [37] object and home pets [35]. The Faster R-CNN architecture is divided into two modules: The Region Proposal Network (RPN) and a Fast R-CNN Detector. The RPN and the Fast R-CNN detector share the same convolutional layers. Faster R-CNN, by consequence, could be considered as a single and a unified network for object detection. To generate high quality object proposal, a highly descriptive feature extractor in the convolutional layers can be used. The Fast R-CNN detector uses many regions of interest (ROIs) as input. Then, the ROI pooling layer extracts a feature vector for each ROI. This feature vector will constitute the input for a classifier formed by a series of fully connected (FC) layers.

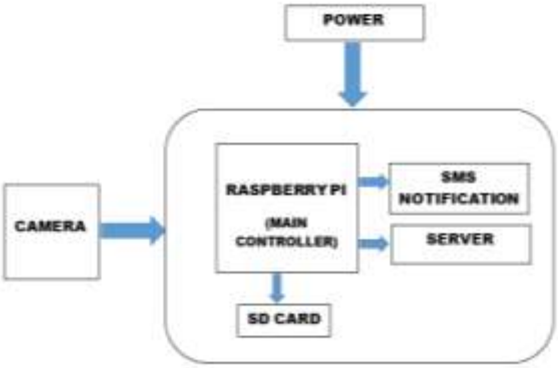


Fig.6. Proposed system block diagram

The embedded system of the proposed system, include Raspberry Pi 3B 8MP camera module, Raspberry Pi 3B as the main controller for all the object detection and programming for the whole system, SMS notification, buzzer (Alarm notification) and power supply.

The system comprises of five basic components

- Raspberry Pi 3
- Raspberry Pi 3 8MP Camera module
- SMS notification
- Power supply
- Raspberry microSD card.

The Raspberry Pi 3B is a basic module for processing images/videos, executing object detection on acquired video frames to detect objects. The Board has ARM cortex A53 clocked at 1.2GHz, 4000MHz Video Core IV multimedia GPU, 1Gb memory, power supply, HDMI, USB ports and other features.

The camera module takes in video stream then the raspberry pi 3B module implement the object detection on the captured frames. Figure 7 shows the flow diagram of the proposed system;

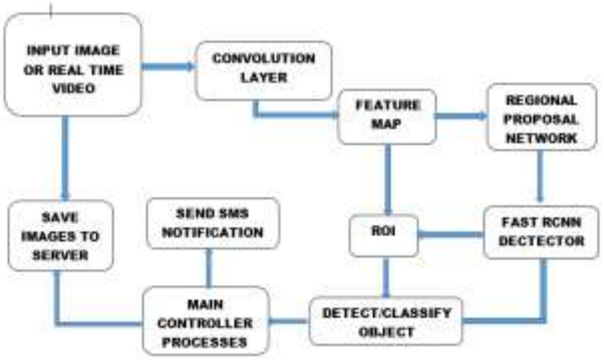


Fig.7. Flow diagram of the proposed system

The electronic components to be used are; Raspberry pi 3B, PiCamera module SMS notification and power supply. The camera and the power supply all the required inputs to the Raspberry pi 3B, while the SMS notification act as the output for the system. Once frames/video are acquired from the camera and fed into the Raspberry pi 3B controller, the image is being processed using the faster regional convolutional neural network as stored within the programmed Raspberry pi 3B.

V. *Conclusion*

Security remains a major concern to everyone, everybody wants to be protected from being attacked, and means to prevent this has been a challenge over the years. A lot of solution has already been put in place to tackle insecurity. This study intends to provide improvement on already existing motion and object detection techniques. The anticipated system shall intelligently detect object and by means of SMS notification sends alert the user the right action to be taken. The system proposed in this study is also cost effective and thereby an average citizen can afford, in other to enhance security in home and place of work.

References

[1] Y. Kurylyak, “A Real-Time Motion Detection for Video Surveillance System,” no. September, pp. 386–389, 2009.

[2] S. Ibrahim, “A comprehensive review on intelligent surveillance systems,” vol. 1, pp. 7–14, 2016.

[3] A. A. Shafie, F. Hafizhelmi, and K. Zaman, “Smart Video Surveillance System,” no. October 2018, 2010.

[4] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, “Car Detection using Unmanned Aerial Vehicles : Comparison between Faster R-CNN and YOLOv3,” 2019 1st Int. Conf. Unmanned Veh. Syst., pp. 1–6, 2019.

[5] A. Deshmukh, S. Deshmukh, A. Zalte, K. Gaware, and P. S. S. Deore, “Intelligent video surveillance system using cnn,” no. 05, pp. 483–486, 2020.

[6] F. Bousetouane and B. Morris, “Fast CNN Surveillance Pipeline for Fine-Grained Vessel Classi fi cation and Detection in Maritime Scenarios,” no. August, pp. 242–248, 2016.

[7] H. M. Valentin and M. V. Boldea, “Using mathematical algorithms for classification of LANDSAT 8 satellite images,” no. March, pp. 1–6, 2015, doi: 10.1063/1.4912899.

[8] D. Chahyati, M. I. Fanany, and A. M. Arymurthy, “ScienceDirect ScienceDirect Tracking People by Detection Using CNN Features,” Procedia Comput. Sci., vol. 124, pp. 167–172, 2018, doi: 10.1016/j.procs.2017.12.143.

[9] H. C. Shin and J. Y. Lee, “Pedestrian Video Data Abstraction and Classification for Surveillance System,” 9th Int. Conf. Inf. Commun. Technol. Converg. ICT Converg. Powered by Smart Intell. ICTC 2018, pp. 1476–1478, 2018, doi: 10.1109/ICTC.2018.8539426.

[10] A. Ullah, K. Muhammad, J. Del Ser, S. W. Baik, and V. Albuquerque, “Activity Recognition using Temporal Optical Flow Convolutional Features and Multi-Layer LSTM,” IEEE Trans. Ind. Electron., vol. PP, no. c, p. 1, 2018, doi: 10.1109/TIE.2018.2881943.

[11] T. Hussain, K. Muhammad, A. Ullah, Z. Cao, S. W. Baik, and V. H. C. De Albuquerque, “Cloud-assisted multiview video summarization using CNN and bidirectional LSTM,” IEEE Trans. Ind. Informatics, vol. 16, no. 1, pp. 77–86, 2020, doi: 10.1109/TII.2019.2929228.

[12] I. I. Conference and E. Workshops, “Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW) 2017 10-14 July 2017,” no. July, pp. 585–590, 2017.

[13] W. Tan, “Object Detection with Multi-RCNN Detectors,” pp. 193–197.

[14] A. H. Sanoob, J. Roselin, and P. Latha, “Smartphone Enabled Intelligent Surveillance System,” no. c, pp. 1–7, 2015, doi: 10.1109/JSEN.2015.2501407.

[15] L. W. Yang and C. Y. Su, “Low-cost CNN Design for Intelligent Surveillance System,” 2018 Int. Conf. Syst. Sci. Eng., pp. 1–4, doi: 10.1109/ICSSE.2018.8520133.

[16] O. M. Olaniyi, J. A. Bala, S. O. Ganiyu, and P. E. Wisdom, “A Systematic Review of Background Subtraction Algorithms for Smart Surveillance System,” vol. 8, no. 1, pp. 35–54, 2020.

[17] C. Jin, S. Li, and H. Kim, “Real-Time Action Detection in Video Surveillance using Sub-Action Descriptor with Multi-CNN,” pp. 1–29.

[18] Q. Xu, W. Zheng, X. Liu, and P. Jing, “Deep Learning Technique Based Surveillance Video Analysis for the Store,” Appl. Artif. Intell., vol. 34, no. 14, pp. 1055–1073, 2020, doi: 10.1080/08839514.2020.1784611.

[19] A. Antoniou, “A General Purpose Intelligent Surveillance System For Mobile Devices using Deep Learning,” pp. 2879–2886, 2016.

[20] K. Muhammad, S. Khan, S. Member, and V. Palade, “Edge Intelligence-Assisted Smoke Detection in,” IEEE Trans. Ind. Informatics, vol. PP, no. c, p. 1, 2019, doi: 10.1109/TII.2019.2915592.

[21] S. Hargude and M. T. It, “i-surveillance : Intelligent Surveillance System Using Background Subtraction Technique,” vol. 1.

[22] C. Gao, P. Li, Y. Zhang, J. Liu, and L. Wang, “Author ’ s Accepted Manuscript People counting based on head detection combining environment Reference : To appear in : Neurocomputing,” Neurocomputing, 2016, doi: 10.1016/j.neucom.2016.01.097.

[23] S. Y. Nikouei, Y. Chen, S. Song, R. Xu, B. Y. Choi, and T. Faughnan, “Smart surveillance as an edge network service: From harr-cascade, SVM to a Lightweight CNN,” Proc. - 4th IEEE Int. Conf. Collab. Internet Comput. CIC 2018, pp. 256–265, 2018, doi: 10.1109/CIC.2018.00042.

[24] Z. Xu, C. Hu, and L. Mei, “Video structured description technology based intelligence analysis of surveillance videos for public security applications,” 2015, doi: 10.1007/s11042-015-3112-5.

[25] H. Kaya, H. Dibeklio, and A. A. Salah, “Kernel ELM and CNN based Facial Age Estimation,” pp. 80–86.

[26] Y. Byeon and S. Pan, “A Surveillance System Using CNN for Face Recognition with Object , Human and Face Detection,” pp. 975–984, doi: 10.1007/978-981-10-0557-2.

[27] Y. Sakai, T. Oda, M. Ikeda, and L. Barolli, “A Vegetable Category Recognition System Using Deep Neural Network,” 2016, doi: 10.1109/IMIS.2016.84.

[28] A. N. Shuaibu, A. S. Malik, and I. Faye, “Adaptive Feature Learning CNN for Behavior Recognition in Crowd Scene,” pp. 357–361, 2017.

[29] H. Ahamed, I. Alam, and M. Islam, “HOG-CNNBasedRealTimeFaceRecognition,” 2018 Int. Conf. Adv. Electr. Electron. Eng., pp. 1–4, 2018.

[30] X. Xiang, N. Lv, X. Guo, S. Wang, and A. El Saddik, “Engineering vehicles detection based on modified faster R-CNN for power grid surveillance,” Sensors (Switzerland), vol. 18, no. 7, 2018, doi: 10.3390/s18072258.

[31] M. H. Gauswami, “Implementation of Machine Learning for Gender Detection using CNN on Raspberry Pi Platform,” 2018 2nd Int. Conf. Inven. Syst. Control, no. Icisc, pp. 608–613, 2018.

[32] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep Learning for Computer Vision : A Brief Review,” vol. 2018, 2018.

[33] P. Wang et al., “Regional Detection of Traffic Congestion Using in a Large-Scale Surveillance System via Deep Residual TrafficNet,” vol. 6, 2018, doi: 10.1109/ACCESS.2018.2879809s.

[34] H. C. Shin and J. Y. Lee, “Pedestrian Video Data Abstraction and Classification for Surveillance System,” 9th Int. Conf. Inf. Commun. Technol. Converg. ICT Converg. Powered by Smart Intell. ICTC 2018, pp. 1476–1478, 2018, doi: 10.1109/ICTC.2018.8539426

[35] R. Article, “ANIMAL DETECTION USING DEEP LEARNING ALGORITHM,” vol. 7, no. 1, pp. 434–439, 2020.

[36] Q. Xu, W. Zheng, X. Liu, and P. Jing, “Deep Learning Technique Based Surveillance Video Analysis for the Store,” Appl. Artif. Intell., vol. 34, no. 14, pp. 1055–1073, 2020, doi: 10.1080/08839514.2020.1784611.

[37] H. Jiang and E. Learned-miller, “Face Detection with the Faster R-CNN,” 2017, doi: 10.1109/FG.2017.82