KLIK: Kajian Ilmiah Informatika dan Komputer
ISSN 2723-3898 (Media Online)
Vol 3, No 6, Juni 2023, Hal 680-687
DOI 10.30865/klik.v3i6.855
https://djournals.com/klik

Analysis of Smart Home Security System Design Based on Facial Recognition With Application of Deep Learning

Rafly Athalla Farizan, Satria Mandala*

School of Computing, Informatics Study Program, Telkom University, Bandung, Indonesia
Email: [1]raflyathalla@student.telkomuniversity.ac.id, [2,*]satriamandala@telkomuniversity.ac.id
Correspondence Author Email: satriamandala@telkomuniversity.ac.id

Abstract−Currently, there is a growing interest in utilizing the Internet of Things (IoT) for Smart home systems. One crucial aspect of Smart home systems is their security capabilities, particularly the convenience of locking and unlocking doors or gates. However, smart home security systems have faced challenges in terms of low accuracy and image processing delays. Experiments conducted using the KNN and Decision Tree methods have shown accuracy rates of approximately 65% - 70%. To address this issue, this research proposes a Deep Learning approach that achieves an accuracy of over 80%. The methodology employed in this study involves four key steps: 1. Conducting a literature review on Smart Home Security, 2. Developing an RNN model for face detection, 3. Creating a prototype for face detection in a smart home setting, and 4. Evaluating the developed prototype for smart homes. The experimental results demonstrate that the proposed prototype achieves an accuracy of 94.3%. Additionally, the recall rate is 94.3%, the f1 score is 91.66%, and the precision is 94.8%.

Keywords: Smart Home Security; KNN; Deep Learning; RNN; CNN; Decision Tree; Prototype

1. INTRODUCTION

Smarthome is a concept that utilizes technology to automate and enhance the intelligence of a home. This involves connecting electronic devices and household systems via computer networks and the internet, enabling centralized control and monitoring. On the other hand, deep learning is a subset of machine learning that uses artificial neural networks to analyze large amounts of data and identify complex patterns. In the smarthome context, deep learning can be used to increase system intelligence and adaptability based on occupant preferences. By analyzing data collected from various sensors and devices within the smarthome, deep learning enables the system to more effectively understand occupant usage patterns and preferences. For example, it can study and recognize patterns of electricity usage, identify rush hours, habits of using certain devices, and patterns of inefficient energy consumption. By leveraging this knowledge, the system can provide recommendations to optimize energy use and reduce unnecessary consumption. Additionally, deep learning can help detect and recognize unusual or suspicious occupant behavior. By training a neural network on normal behavior data, the system can detect deviations, such as irregular sleep patterns or unexpected activity. This increases the security and reliability of the smarthome in identifying potential threats or emergency events. The integration of deep learning in smarthomes has great potential to increase the intelligence and functionality of the connected home. By analyzing and studying the data generated by smarthome systems, deep learning enables homes to be more adaptive, efficient and secure, thereby providing a better experience for their occupants. However, several studies focusing on home security systems report relatively low accuracy and performance. For example, experiments conducted by [2], [4],[6], and [7] use facial recognition methods using different machine learning techniques, including testing RNN [6] RNN [7]. However, the accuracy obtained with RNN[7] is 84.71%, while the test obtained by RNN[6] has an accuracy of 83.6%. In testing RNN [6] RNN [7] they use little data and features contained in the image data is not too much which causes the accuracy of the test is still below 85%. The main challenge in smart home security systems lies in the low accuracy observed in experiments conducted by [2] using the KNN and Decision Tree methods, achieving an accuracy rate lower than 80%. Therefore, this experiment aims to achieve several goals: 1. Design a deep learning model for home security facial recognition detection,2. Develop a prototype based on the facial recognition model created in the 1st objective for home security, and 3. Perform a performance analysis of the prototype developed in 2nd objective using the metrics of accuracy, recall, precision and f1 score.

2. RESEARCH METHODOLOGY

2.1 System Design

The research system design comprises several processes depicted in Figure 1. The system initiates with the data crawling process, which involves real-time collection of facial image data. Subsequently, the data is labeled as either a family member or a non-family member. It then proceeds to the preprocessing stage, where feature extraction using the Oriented Gradient Histogram (HOG) technique and RNN models are applied.

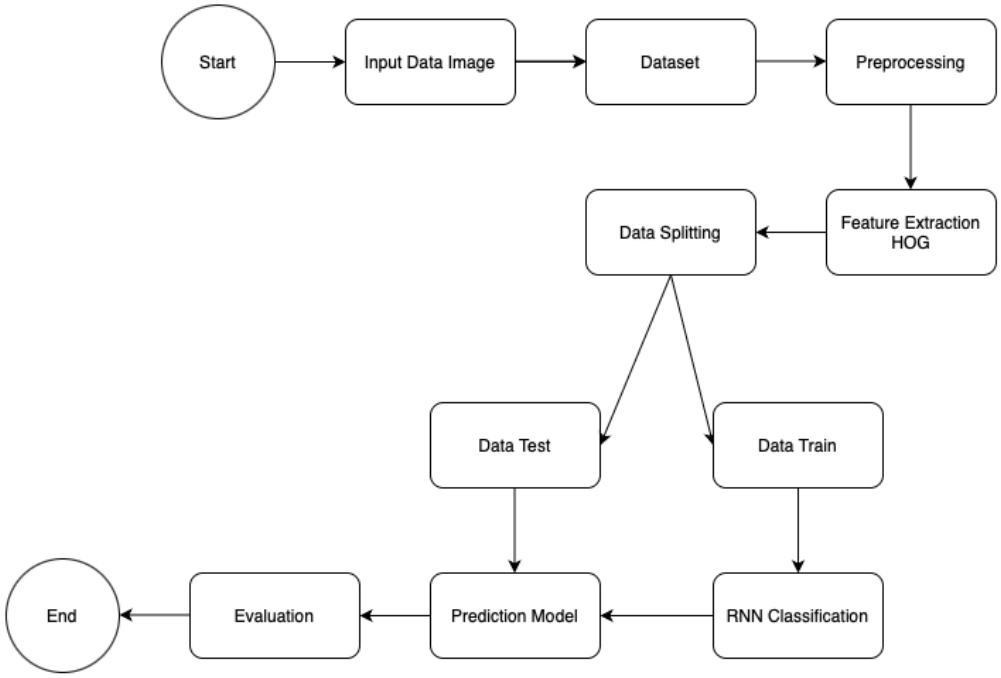


Figure 1. The flowchart of the facial recognition system based on deep learning in this experiment.

In Figure 1, the initial step involves capturing image data using the ov7670 camera tool based on Arduino. Once the images are obtained, they are stored in the dataset folder. Next, the data is processed using the Oriented Gradient Histogram (HOG) method to extract its features. Subsequently, the dataset is divided into two sets: test data and train data.Following the data splitting process, the test data is utilized for prediction, employing the RNN algorithm. The predictions are then compared to the train data, and the evaluation stage begins. During this stage, the experiment's results are examined, considering metrics such as accuracy, precision, recall, and f1 score. It is crucial for these metrics to achieve a minimum threshold of 80% to ensure satisfactory performance.

2.2 Input Data Image

Data required for this research consists of facial images of family members, each sized 250x250 pixels. These images are captured using an OV7670 camera with the assistance of an Arduino program. The collected data will be used to create a private dataset. The images will be organized into folders based on whether they belong to family members or not. For each image capture, a total of 126 images will be generated by varying the shooting positions from top, bottom, front, left side, and right side, while keeping the facial object as the focal point. As a result, the dataset will comprise 2 labels multiplied by 1,260, resulting in a total of 2,520 images. The data collection process is visualized in the image below.

2.3 Feature Extraction

Feature extraction in face recognition involves converting a face image into a numerical representation that captures the essential characteristics of the face. This process aims to filter out irrelevant information and emphasize the distinctive attributes of each face. In this particular study, feature extraction for face recognition employs the Histogram of Oriented Gradients (HOG) method. This method extracts features based on the distribution of pixel intensity gradients within a face image. The face image is divided into cells, and a gradient orientation histogram is computed for each cell. This representation of features reflects the patterns and orientations of gradients present in the face.

2.4 Recurrent Neural Network method

In addition to the description provided, the forget gate in LSTM is responsible for deciding which information from the previous time step should be forgotten or discarded. It takes as input the previous hidden state and the current input and outputs a number between 0 and 1 for each memory cell. A value of 0 means that the information should be completely forgotten, while a value of 1 means that the information should be completely retained.The input gate, on the other hand, determines which new information should be stored in the memory cells. It takes as input the previous hidden state and the current input and uses a sigmoid function to produce a number between 0 and 1 for each memory cell. This gate also controls the flow of information into the memory cells by deciding which values to update.The cell gate is responsible for updating the values in the memory cells based on the previous hidden state, the current input, and the output of the forget and input gates. It uses a tanh function to compute a new candidate value that can be added to the memory cells. The cell gate combines the candidate values with the output of the forget and input gates to update the memory cells.

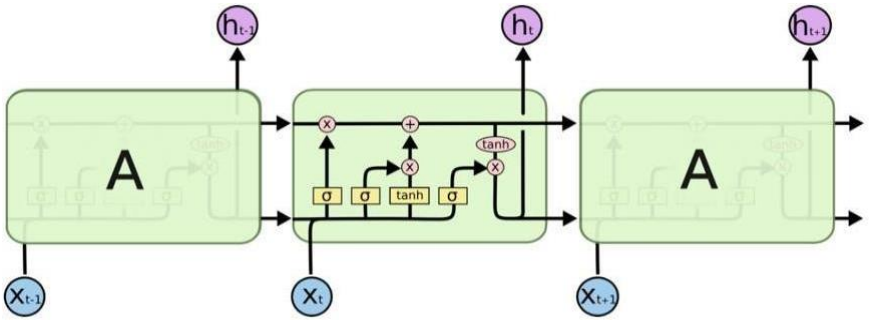


Figure 2. LSTM Model Schematic

Lastly, the output gate determines the output of the LSTM cell. It takes as input the previous hidden state and the current input, and it also considers the updated values in the memory cells. The output gate applies a sigmoid function to produce a number between 0 and 1 for each memory cell, which represents the amount of information that should be outputted.By incorporating these gates and memory cells, LSTM is able to selectively remember or forget information over long sequences, enabling it to effectively capture and understand long-term dependencies in the data. This makes LSTM particularly useful for tasks involving sequential data, such as speech recognition, natural language processing, and time series analysis.

2.5 Performance Evaluation

Once the models have been constructed, it is crucial to evaluate them in order to determine the best-performing model. Model evaluation involves utilizing metrics such as accuracy, precision, recall, F1 score, and the confusion matrix [17]. To evaluate a classification model, it is necessary to employ separate test datasets that were not used during the training phase. These test datasets provide an independent assessment of the model's performance.

Table 1. Confusion Matrix

Confusion matrix		Actual Values	
		Positive	**Negative**
Predicted Values	Positive	TP	FP
	Negative	FN	TN

Several evaluation measures can be employed to assess the performance of the TP learning model, where TP represents true positives, TN denotes true negatives, FP signifies false positives, and FN indicates false negatives. The confusion matrix serves as a metric for evaluating 16 transfer learning models in this study [11]. Additionally, the author calculates the precision score, recall score, and F1 score to evaluate the constructed model using the following formulas.

a. Accuracy is the value of the ratio of correctly predicted data compared to all data. The accuracy value is formulated as follows:

$$accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

b. Precision is the comparison value between positive values that are correctly predicted and all data that are predicted to be positive. The precision value is formulated as follows:

$$precision = \frac{TP}{TP+FP} \quad (2)$$

c. Recall is the value of the ratio of true positive prediction data compared to all existing positive actual data. The recall value is formulated as follows:

$$recall = \frac{TP}{TP+FN} \quad (3)$$

d. F1-Score is a performance metric that takes recall and precision into account. F1-score is formulated as follows:

$$F1\ score = \frac{2\ (recall\ x\ precision)}{(recall\ +\ precision)} \quad (4)$$

3. RESULT AND DISCUSSION

3.1 Input data image and image processing

The collected data consists of personal data, particularly random images of human faces. These face images were

captured using an OV7670 camera and programmed with Arduino. For each recording process, a total of 250 images were captured, where the images were selected by eliminating those that had close similarities or unclear details (blurry). This selection process aims to limit the amount of data used in the model training phase, considering the limitations of the computer specifications used for training and the absence of a GPU device. During the image capture process, the OV7670 camera was moved in different directions (right, left, up, and down) while focusing on the human face. The distance between the camera and the subject's face was approximately 1 meter. The resulting images have a resolution of 250x250 pixels and are in a 3-dimensional format (RGB). The process of collecting the personal dataset is depicted in Figure 3.

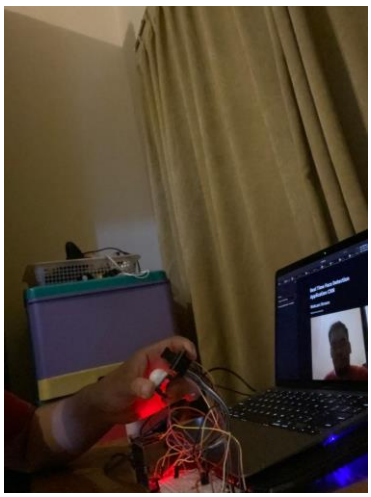
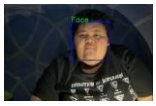


Figure 3. The process of capturing image data

The retrieval of this image data is performed in real time for the data labeled as family, while a combination of real-time and dummy data is used for the data labeled as not family.The labels or classes used in private datasets are: family members and non-family members. The following are sample images for the 2 labels or classes:

Table 2. Face Sample

Image	Label	Amount of data	Image	Label	Amount of data
	**Family members**	**252**		**Not a Family Member**	**10**
	**Family members**	**252**		**Not a Family Member**	**10**
	**Family members**	**252**		**Not a Family Member**	**10**
	**Family members**	**252**		**Not a Family Member**	**10**
	**Family members**	**252**		**Not a Family Member**	**10**

Furthermore, as depicted in Table 2, the facial images will undergo a transformation process to create a private dataset, which will then be trained using the RNN algorithm. Before training, the size of each image will be resized to 50x50x1 and 224x224x1, serving as inputs for the 17 planned experiments. Subsequently, augmentation techniques will be applied, including flipping, random rotation, and Affine, generating 10 additional images for each original image. This

augmentation process will result in an increased number of data samples for the family members class, reaching 2,520, consisting of 252 original data samples plus 10 augmented data samples for each original data sample. Similarly, for the non-family members class, the data samples will increase to 4,000, comprising 40 original data samples plus 10 augmented data samples for each original data sample. Consequently, the augmented dataset will amount to a total of 6,520 samples, while the original data samples will contribute 2,520 samples, resulting in a combined dataset of 9,040 samples. To ensure a balanced representation of each class, an equal number of data samples will be allocated to each class. Following the augmentation process, the dataset will be divided into training and testing sets, with 70% of the data (6,328 images) assigned to the training set and the remaining 30% (2,712 images) designated for the testing set.

3.2 Hardware and Software

In this study the authors used available software and hardware, hardware without a GPU. Model training only uses CPU and RAM. Detailed specifications can be seen in table 3.

Table 3. Software and Hardware Specifications

Software	Hardware
Os: Windows 10	CPU:Intel i7-10750H 2.6 GHz
Jupyter lab	RAM: 16 GB
Python 3.9.12, Arduino IDE 2.1.0	Hard Disk : 1 TB
KERAS Framework	Brand: ASUS
Anaconda Library	Webcam : OV7670 with PIR sensor

in table 3, is the hardware and software used in this experiment.Arduino is an open-source electronic platform that provides a flexible and user-friendly way to create interactive projects. It consists of both hardware and software components, with the Arduino board serving as the hardware platform. Arduino boards are equipped with microcontrollers that can be programmed using the Arduino programming language, which is based on C/C++. The boards are widely used for various applications, including robotics, automation, and Internet of Things (IoT) projects.Python, on the other hand, is a versatile and high-level programming language that is known for its simplicity and readability. It has a vast ecosystem of libraries and frameworks that make it suitable for a wide range of applications, including data analysis, web development, and machine learning. Python's syntax is designed to be easy to understand, making it an excellent choice for beginners and experienced programmers alike.The OV7670 camera is a low-cost image sensor module that is commonly used for capturing images and videos in various projects. It integrates a CMOS image sensor and signal processing circuitry, allowing it to capture images in different resolutions and formats. The OV7670 camera is often utilized in Arduino projects to capture visual data and process it for various applications, such as image recognition, computer vision, and surveillance systems.Keras is a high-level neural networks API that is built on top of the TensorFlow library. It provides a simplified interface for building and training deep learning models. With Keras, developers can easily construct complex neural networks by stacking layers and defining the connections between them. Keras supports a wide range of deep learning architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative adversarial networks (GANs). Its user-friendly and intuitive design makes it popular among researchers and practitioners in the field of machine learning and artificial intelligence.In summary, Arduino is a versatile hardware platform, Python is a powerful programming language, the OV7670 camera is an affordable image sensor module, and Keras is a user-friendly deep learning framework. Together, they provide the tools and capabilities necessary for developing a wide range of projects, from simple IoT applications to complex machine learning models

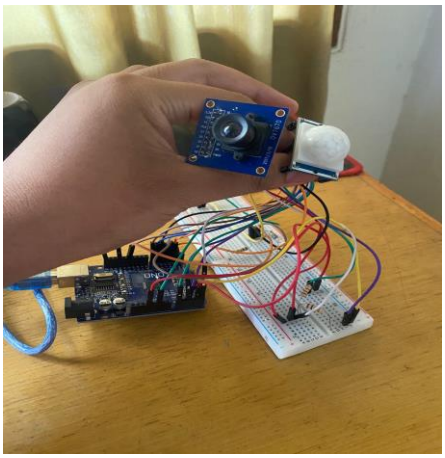


Figure 3. OV7670 cam with PIR sensor

The OV7670 Cam is connected to a PIR sensor to test from the human motion detection function, whether there is damage to the sensor or not, so testing and calibration at the beginning of the study is necessary to facilitate further research. In this explanation you can see the results as shown in the picture.

3.3 Experiment

In this experiment, the face recognition research is conducted using the RNN algorithm. RNN possesses several features that enable the study of facial images by storing long-term memory to recognize and retain information in longer sequences of data, thereby facilitating a better comprehension of facial changes and patterns. In this experiment, the results can be optimized through proper tuning of the RNN. The performance of RNN is evaluated before and after tuning, and the following is a comparison of RNN's performance before and after tuning:

Table 4. RNN comparison before tuning and after tuning

Algoritma	Information	F1 Score	Accuracy	Recall	Precision
RNN	Before Tuning	90.02%	91.12%	91.12%	92.16%
RNN	After Tuning	91.66%	94.3%	94.3%	94.8%

Based on the information provided in Table 4, comparisons were made between the RNN algorithms before tuning and after tuning in terms of various evaluation metrics including accuracy, precision, recall, and F1 score. It can be observed that there is significant variation among the algorithms. RNN before tuning achieved an accuracy rate of 91.12%, Recall 91.12%, Precision 92.16, and F1 score 90.02%, while RNN after tuning had a greater accuracy rate than RNN before tuning with 94.3%, F1 score 91.66%, Recall 94.3 %, and 94.8% precision.

3.4 Discussion

Recent advancements in facial recognition research have demonstrated significant advancements in the field through the application of the Recurrent Neural Network (RNN) method. This study highlights the utilization of RNNs within a facial recognition architecture to effectively capture and model the temporal context of facial features, including sequences of facial movements and expressions. By leveraging the RNN's short-term and long-term memory capabilities, temporal information can be incorporated into the facial recognition process, resulting in improved accuracy and stability of the outcomes. Below is a comparison of accuracy percentages, precision, recall, and f1 scores from facial recognition tests using the RNN method in contrast to other research experiments:

Table 5. Performance Comparison with Other Advance Studies on Face Detection

Author	F1 Score	Accuracy	Recall	Precision
Salama (SVM)[4]	73.7%	80.00%	66.7%	71.5%
Taiwo(KNN)[2]	78.8%	83.5%	75.8%	82.3%
RNN(Before Tuning)	90.02%	91.12%	91.12%	92.16%
RNN(After Tuning)	91.66%	94.3%	94.3%	94.8%

This study employs the RNN (Recurrent Neural Network) method for classifying facial images in the context of smart home security based on the Internet of Things. The objective of this test is to identify authorized individuals entering the house through the door. Furthermore, the experimental results of the RNN method are compared with other studies utilizing the SVM[23] and KNN[2] methods. The evaluation results documented in Table 5 reveal that the RNN achieved an accuracy of 94.3%, an f1 score of 91.66%, a recall of 94.3%, and a precision of 94.8%. These findings indicate that the RNN exhibits superior classification accuracy compared to SVM[23] (80.00% accuracy, 66.7% recall, 71.5% precision, and 73.7% f1 score) and KNN[2] (83.5% accuracy, 75.8% recall, 82.3% precision, and 78.8% f1 score).Additionally, the RNN (Recurrent Neural Network) is often regarded as superior to SVM (Support Vector Machine) and KNN (K-Nearest Neighbors) in processing image data due to several key factors. Firstly, RNN excels in capturing temporal and pattern dependencies in sequential data, making it suitable for tasks involving image sequences or time series data. In the realm of image processing, RNNs can effectively model spatial relationships and temporal changes within images or sequences of images. This proves especially valuable in tasks such as object tracking, video analysis, or facial expression recognition, where the sequence and context of data play significant roles. Secondly, RNNs possess the capability to learn and represent complex, high-level features within the data. They can automatically extract and encode hierarchical representations of visual features, resulting in more robust and discriminative image representations. In contrast, SVM and KNN typically rely on manually crafted features or distance metrics, which may not capture the intricacies and nuances present in image data as effectively as the representations learned by RNNs. Furthermore, RNNs can handle input sequences of varying lengths, providing an advantage when dealing with image data of different sizes or varying numbers of images in a sequence. SVM and KNN, on the other hand, generally require fixed-sized feature vectors or a predefined number of nearest neighbors, which can be limiting in certain image processing scenarios.It is important to note that the effectiveness of a particular method, whether it be RNN, SVM, or KNN, depends on various factors such as the specific task, the quality and quantity of training data, and appropriate implementation. In some cases, SVM or KNN may still be suitable choices for image processing tasks, particularly when dealing with simpler or less sequential data. Therefore, it is crucial to carefully consider the data characteristics and task requirements when selecting the most suitable method.

4. CONCLUSION

In this section, the author will present the outcomes and analyses of the three conducted trials, aiming to determine whether the proposed research objectives have been achieved. The results of the trials demonstrate the successful functionality of the Facial Recognition-Based Home Security System, which utilizes the Deep Learning Method. According to this study, the best model achieved an accuracy of 94.3%, an f1 score of 91.66%, a precision of 94.8%, and a recall of 94.3% using RNN. Another study conducted additional testing using SVM, which yielded satisfactory results with an accuracy of 80.0%, an f1 score of 73.7%, a recall of 66.7%, and a precision of 82.3%.The RNN method can be considered superior due to several key factors. Firstly, RNN excels in studying and modeling temporal patterns in data. Specifically in face classification, RNNs are capable of considering the spatial arrangement of facial features, such as eye movements or changes in facial expressions from frame to frame. Conversely, SVM and KNN lack the inherent ability to capture such temporal relationships. Secondly, facial data is often sequential, particularly in tasks involving emotion recognition or facial expression analysis. RNNs are explicitly designed to handle sequential data and incorporate memory cells (e.g., LSTM or GRU) that can retain information from previous instances, enabling them to comprehend contextual information for classification purposes. On the other hand, SVM and KNN assume independence among data samples, rendering them less effective in dealing with sequential data. Lastly, RNN employs complex networks and possesses the capacity to learn high-level and intricate representations of features within the data. In facial classification tasks, features like skin texture, eye shape, or complex facial expression patterns are critical for accurate classification, and RNN can effectively capture such nuanced representations. SVM and KNN, in contrast, typically employ simpler feature representations.While the tests conducted by other authors using SVM and KNN showcased favorable results for KNN over SVM, with an accuracy of 83.5%, a precision of 82.3%, a recall of 75.8%, and an f1 score of 78.8%, it should be noted that SVM and KNN assume independence among data samples and may not be as effective when dealing with sequential data. Hence, in the context of facial recognition, testing using the RNN method tends to yield superior results compared to KNN and SVM.However, it is important to consider that the effectiveness of a method depends on the dataset, the complexity of the problem, and the specific implementation. Moreover, further review of this system prototype is necessary, as the authors identified certain shortcomings in terms of prototype durability. Real-world trials may produce different results and impact the prototype's detection rate.

REFERENCES

[1] Ajao, Lukman Adewale, Jonathan Kolo Kolo, E. A. Adedokun, Olayemi Mikail Olaniyi, O. C. Inalegwu, and S. K. Abolade. "A Smart Door Security-Based Home Automation System." (2018).

[2] Taiwo, O. and Ezugwu, A. E. (2021), 'Internet of things-based intelligent smart home control system', *Security and Communication Networks* 2021.

[3] Phawinee, S., Cai, J.-F., Guo, Z.-Y., Zheng, H.-Z. and Chen, G.-C. (2021), 'Face recognition in an intelligent door lock with resnet model based on deep learning', *Journal of Intelligent & Fuzzy Systems* 40(4), 8021–8031.

[4] Salama AbdELminaam, D., Almansori, A. M., Taha, M. and Badr, E. (2020), 'A deep facial recognition system using computational intelligent algori- thms', *Plos one* 15(12), e0242269.

[5] M. Z. Alom et al., "The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches," 2018, [Online]. Available: http://arxiv.org/abs/1803.01164.

[6] Vimal, K. U., S. K. Sandij, M. Yogesh, and S. Soundarya. "Facial Emotion Recognition Using Deep Learning." In *Journal of Physics: Conference Series*, vol. 1916, no. 1, p. 012118. IOP Publishing, 2021.

[7] Kuppusamy, P., and C. Harika. "Human action recognition using CNN and LSTM-RNN with attention model." *Int. J. Innov. Technol. Explor. Eng* 8 (2019): 1639-1643.

[8] Suyanto, Machine Learning Tingkat Dasar dan Lanjut, Ed. I. Bandung, Indonesia: Informatika, 2018.

[9] R. M. Rawat, S. Garg, N. Jain, and G. Gupta, "COVID-19 detection using convolutional neural network architectures based upon chest X-rays images," Proc. - 5th Int. Conf. Intell. Comput. Control Syst. ICICCS 2021, no. Iciccs, pp. 1070– 1074, 2021, doi: 10.1109/ICICCS51141.2021.9432134.

[10] X. Li, Z. Yang, and H. Wu, "Face detection based on receptive field enhanced multi-task cascaded convolutional neural networks," IEEE Access, vol. 8, pp. 174922–174930, 2020, doi: 10.1109/ACCESS.2020.3023782.

[11] M. S. Majib, M. M. Rahman, T. M. Shahriar Sazzad, N. I. Khan, and S. K. Dey, "VGG-SCNet: A VGG Net based Deep Learning framework for Brain Tumor Detection on MRI Images," IEEE Access, vol. 9, pp. 116942–116952, 2021, doi: 10.1109/ACCESS.2021.3105874.

[12] BABU, K. P., RAO, G. S. N. and KONDABALA, R. (2021), 'A design of identity (facial) recognition system for smart home based on internet of things'.

[13] Balogh, Z., Magdin, M. and Moln´ar, G. (2019), 'Motion detection and face recognition using raspberry pi, as a part of, the internet of things', *Acta Polytechnica Hungarica* 16(3), 167–185.

[14] Majumder, A. J. and Izaguirre, J. A. (2020), A smart iot security system for smart-home using motion detection and facial recognition, *in* '2020 IE- EE 44th Annual Computers, Software, and Applications Conference (COM- PSAC)', IEEE, pp. 1065–1071.

[15] S. A. Alex, N. Z. Jhanjhi, M. Humayun, A. O. Ibrahim, and A. W. Abulfaraj, "Deep LSTM Model for Diabetes Prediction with Class Balancing by SMOTE," *Electronics (Switzerland)*, vol. 11, no. 17, Sep. 2022, doi: 10.3390/electronics11172737.

[16] Karaman, Yunus, Fulya Akdeniz, Burcu Kır Savaş, and Yaşar Becerikli. "A Comparative Analysis of SVM, LSTM and CNN-RNN Models for the BBC News Classification." In *The Proceedings of the International Conference on Smart City Applications*, pp. 473-483. Cham: Springer International Publishing, 2022.

[17] A. Negi, K. Kumar, P. Chauhan, and R. S. Rajput, "Deep neural architecture for face mask detection on simulated masked face dataset against covid-19 pandemic," Proc. - IEEE 2021 Int. Conf. Comput. Commun. Intell. Syst. ICCCIS 2021, pp. 595–600, 2021, doi: 10.1109/ICCCIS51004.2021.9397196.

[18] Yu, Rui, Xiaohua Zhang, and Minyuan Zhang. "Smart home security analysis system based on the internet of things." In *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, pp. 596-599. IEEE, 2021.

[19] Awang, Nor Fatimah, Ahmad Fudhail Iyad Mohd Zainudin, Syahaneim Marzuki, Syed Nasir Alsagoff, Taniza Tajuddin, and Ahmad Dahari Jarno. "Security and threats in the internet of things based smart home." In *International Conference of Reliable Information and Communication Technology*, pp. 676-684. Cham: Springer International Publishing, 2020.

[20] F. Landi, L. Baraldi, M. Cornia, and R. Cucchiara, "Working Memory Connections for LSTM," *Neural Networks*, vol. 144, pp. 334–341, Dec. 2021, doi: 10.1016/j.neunet.2021.08.030.

[21] Sarhan, Qusay I. "Systematic survey on smart home safety and security systems using the Arduino platform." *IEEE Access* 8 (2020): 128362-128384.

[22] Negi, A., Kumar, K., Chauhan, P. and Rajput, R. (2021), Deep neural architecture for face mask detection on simulated masked face dataset against covid-19 pandemic, in '2021 international conference on computing, communication, and intelligent systems (ICCCIS)', IEEE, pp. 595–600.