

An Efficient CNN Model for Real-Time Facial Expression Recognition

Mr. Pathipati Chandrasekhar¹, B Lakshmi sai mahima², A Vyshanavi³, B Nandini⁴

¹Assistant Professor Computer Science Engineering & Guru Nanak Institute of Technology

²Student Computer Science and Engineering & Guru Nanak institute of technology

³Student Computer science and Engineering & Guru Nanak institute of technology

⁴Student Computer Science and Engineering & Guru Nanak institute of technology

Abstract – This paper presents a lightweight convolutional neural network model for real time and bulk facial emotion detection. The objective is to improve classification accuracy while maintaining low computational cost. The proposed system uses a CNN architecture to extract local features from key facial regions such as eyes, eyebrows, and mouth. These features are combined to produce the final emotion classification. A real time vision system is developed to evaluate the effectiveness of the model. The methodology focuses on efficient feature learning and fusion to enhance recognition performance. Experimental results show that the proposed model achieves an accuracy of 98.6 percent, outperforming existing systems. The results indicate that the model is reliable for practical applications in emotion recognition and can support intelligent human computer interaction systems.

Keywords:

- **Facial Emotion Recognition (FER):** A critical aspect of human-computer interaction used to accurately classify human emotions from facial expressions, enabling systems to adapt to a user's emotional state.
- **Convolutional Neural Network (CNN):** A deep learning architecture highly effective at automatically learning spatial hierarchies and complex patterns from image data, such as edges, shapes, and textures.
- **Real-Time Vision System:** A system developed to evaluate the effectiveness of a model in live or immediate scenarios, ensuring efficient computation and minimal latency for practical deployment.
- **MDI (Mean Decrease Impurity):** A technique used in conjunction with a Gradient Boosting Regressor to enhance feature selection. It ensures that the most critical attributes are prioritized for classification, which helps reduce computational complexity while maintaining high performance.
- **SVM (Support Vector Machines):** A type of classifier used in early computational methods for facial emotion recognition. These traditional approaches fed handcrafted geometric or appearance-based features into the SVM to classify emotions.
- **VGG-16:** A deep convolutional neural network (CNN) architecture originally designed for image classification. In the context of facial expression recognition, it uses 16 layers of convolutional and fully connected layers to capture complex patterns such as textures and facial expressions.
- **ROI (Region of Interest):** An approach that identifies and focuses on key facial regions—such as the eyes, nose, and mouth—for more targeted analysis. This technique is often paired with classifiers to improve accuracy, particularly when training data is limited.
- **KNN (K-Nearest Neighbors):** A classification algorithm that determines an expression's category based on the distance between feature vectors. In the paper, it is combined with ROI extraction to address the poor generalization issues often found in deep neural networks with small datasets.
- **Feature Extraction:** The process of identifying and analyzing local facial features—such as the eyes, eyebrows, and mouth—to provide the data necessary for final emotion classification.
- **Deep Learning:** A set of representation-learning methods that use multiple levels of non-linear modules to transform raw input into higher-level, abstract representations, allowing a machine to discover patterns needed for detection or classification.
- **Emotion Classification:** The final output of the system where features are combined and processed (often via a Softmax activation function) to yield a probability distribution across specific emotion categories, such as happiness, sadness, or anger.
- **Max Pooling:** A process that reduces the spatial dimensions of an image by selecting the maximum value from a specific area (e.g., a 2x2 grid), which helps retain dominant features while reducing computational load.

1. INTRODUCTION

Facial expression recognition is a critical aspect of human-computer interaction, with applications in various fields such as security, healthcare, entertainment, and customer service. The ability to accurately classify human emotions from facial expressions enables more intelligent and responsive systems that can adapt to users' emotional states. Traditional methods of facial emotion recognition relied heavily on handcrafted feature extraction techniques, which often lacked accuracy and robustness in real-world scenarios. With the advent of deep learning, Convolutional Neural Networks (CNNs) have proven to be highly effective in learning complex patterns from image data. In this study, we propose a lightweight CNN model for real-time and bulk facial emotion detection. By leveraging deep learning techniques, our approach aims to enhance classification accuracy while ensuring efficient computation for real-time applications. The system focuses on extracting and analyzing local facial features, such as the eyes, eyebrows, and mouth, to determine the overall emotional state of an individual. The proposed model is designed to be both accurate and computationally efficient, making it suitable for deployment in real-time vision systems across various industries. Representation learning is a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification. Deep-learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level. With the composition of enough such transformations, very complex functions can be learned. For classification tasks, higher layers of representation amplify aspects of the input that are important for discrimination and suppress irrelevant variations.

2. SCOPE OF THE PROJECT

The scope of this project extends to developing a robust and efficient facial emotion recognition system using a lightweight CNN architecture. The project aims to improve classification accuracy and real-time processing capabilities, making it suitable for applications in human-computer interaction, surveillance, mental health monitoring, and customer experience analysis. The system focuses on detecting emotions such as happiness, sadness, anger, surprise, and neutrality by analyzing key facial features. It can be deployed in real-world scenarios such as smart security systems, interactive AI assistants, and emotion-aware applications in healthcare. The project also aims to make the system scalable, allowing it to process multiple facial images simultaneously for bulk emotion detection. By optimizing the CNN model, the

system ensures minimal latency and computational overhead, making it feasible for real-time applications in embedded systems, mobile devices, and cloud-based platforms.

3. OBJECTIVE

The primary objective of this project is to develop an efficient and accurate machine learning-based system for classifying sleep disorders. By utilizing the Sleep Health and Lifestyle Dataset, the system aims to analyze sleep patterns and identify key features that contribute to different sleep disorders. The study specifically focuses on comparing existing classification models, such as Random Forest, with the proposed XGBoost-based system, highlighting the advantages of gradient boosting techniques in improving predictive accuracy. Another key objective is to enhance feature selection using Gradient Boosting Regressor with the Mean Decrease Impurity (MDI) technique, ensuring that the most critical attributes are used for classification. This will help reduce computational complexity while maintaining high classification performance. The system also seeks to improve the generalization capabilities of sleep disorder classification models, ensuring robust performance across diverse datasets. Additionally, this project aims to bridge the gap between medical research and machine learning applications by providing an automated, scalable, and clinically applicable solution for diagnosing sleep disorders. The ultimate goal is to create a system that can assist healthcare professionals in making informed decisions, reducing diagnosis time, and improving treatment outcomes for individuals suffering from sleep-related conditions.

4. RELATED WORK

4.1 Traditional Approaches

Ekman and Friesen established foundational evidence for the universality of facial expressions across cultures, identifying six basic emotions: happiness, sadness, anger, fear, surprise, and disgust. Early computational methods built on this framework by extracting geometric or appearance-based features and feeding them to classifiers such as Support Vector Machines (SVM) or K-Nearest Neighbors (KNN).

4.2 Deep Learning Approaches

Barros et al. adapted the VGG-16 topology into the Face Channel network for FER, leveraging deep convolutional layers to capture complex facial patterns. Koujan et al. proposed a CNN for single-image emotion recognition, demonstrating competitive accuracy on benchmark datasets. Xiao et al. combined region-of-interest (ROI) extraction with KNN classification atop CNN features to address limited training data scenarios.

Despite their effectiveness, many existing deep models suffer from excessive parameters, slow training, overfitting on small datasets, and degradation when network depth increases without proper regularization.

4.3 Lightweight Architectures

Recent work has explored depth-wise separable convolutions (MobileNet), residual connections (ResNet), and global average pooling to reduce parameters while preserving accuracy. These strategies inform the design of the proposed model.

5. EXISTING TECHNIQUE

Facial Expression Recognition Using VGG-16: In the existing system, a common algorithm used for facial expression recognition is based on the VGG-16 network, a deep convolutional neural network (CNN) that was originally designed for image classification tasks. This model, proposed by Barros et al., adapts the VGG-16 architecture by formalizing it into a specialized model for facial expression recognition (FER). The VGG-16 network utilizes 16 layers of convolutional and fully connected layers to extract and process features from facial images, capturing complex patterns that correlate with different emotional states. The success of VGG-16 in FER tasks is attributed to its ability to extract hierarchical features that capture both low-level details (such as edges and textures) and high-level patterns (such as facial expressions).

Facial Expression Recognition Using CNN and KNN: Another existing method, proposed by Xiao et al., involves combining the K-Nearest Neighbor (KNN) algorithm with a Region of Interest (ROI) approach. This algorithm addresses the problem of poor generalization ability in deep neural networks, especially when there is limited training data. The process first identifies key regions on the face (such as eyes, nose, and mouth) and focuses on them, using KNN to classify expressions based on these features. The KNN algorithm uses the distance between feature vectors to classify the expression, making it a simple yet effective technique when paired with a CNN for feature extraction. This approach can improve classification accuracy, especially in smaller datasets, by concentrating on the most relevant parts of the face.

5.1 Existing System

The current facial expression recognition methods are mainly divided into two categories, one is the traditional manual method, and the other is the network model using deep learning. Although the traditional method is widely used, it is very limited in practical applications. Using deep learning to classify facial expressions is usually learning how to use strong supervision methods to represent the emotional features of great sample data. Barros et al. proposed a network model based on the topological structure of VGG-16 for the formalization of

the Facial Channel neural network for Facial Expression Recognition (FER). Koujan et al. proposed a CNN that recognized human emotions from a single face image.

5.2 Existing System Disadvantages

- Too many parameters .
- Slowing down the training speed .
- It is easy to over fitting problem.
- When the network is deeper, it means that the parameter space is larger, and the optimization problem becomes more difficult. Therefore, simply increasing the depth of the network causes more training errors.

6. METHODOLOGIES

6.1 MODULES NAME

- Dataset
- Importing the necessary libraries
- Retrieving the images and Splitting the dataset
- Building the model
- Apply the model and plot the graphs for accuracy and loss
- Accuracy on test set
- Saving the Trained Model

6.2 MODULES EXPLANATION:

6.2.1 Dataset:

In the first module, we developed the system to get the input dataset for the training and testing purpose. We have taken the dataset from Facial Expression.

6.2.2 Importing the necessary libraries:

We will be using Python language for this. First we will import the necessary libraries such as keras for building the main model, sklearn for splitting the training and test data, PIL for converting the images into array of numbers and other libraries such as pandas, numpy, matplotlib and tensorflow. Retrieving the images: We will retrieve the images and their labels. Then resize the images to (224,224) as all images should have same size for recognition. Then convert the images into numpy array.

6.2.3 Splitting the dataset:

Split the dataset into train and test. 80% train data and 20% test data. Convolutional Neural Networks The objectives behind the first module of the course 4 are:

- To understand the convolution operation
- To understand the pooling operation
- Remembering the vocabulary used in convolutional neural networks (padding, stride, filter, etc.)
- Building a convolutional neural network for multi-class classification in images.

6.2.4 Building the model:

For building the we will use sequential model from keras library. Then we will add the layers to make

convolutional neural network. In the first 2 Conv2D layers we have used 32 filters and the kernel size is (5,5). In the MaxPool2D layer we have kept pool size (2,2) which means it will select the maximum value of every 2 x 2 area of the image. By doing this dimensions of the image will reduce by factor of 2. In dropout layer we have kept dropout rate = 0.25 that means 25% of neurons are removed randomly. We apply these 3 layers again with some change in parameters. Then we apply flatten layer to convert 2-D data to 1-D vector. This layer is followed by dense layer, dropout layer and dense layer again. The last dense layer outputs 7 nodes as the Facial Expression Detection. This layer uses the softmax activation function which gives probability value and predicts which of the 7 options has the highest probability.

6.2.5 Apply the model and plot the graphs for accuracy and loss:

We will compile the model and apply it using fit function. The batch size will be 10. Then we will plot the graphs for accuracy and loss. We got average validation accuracy of 87.6% and average training accuracy of 98.3%. Accuracy on test set: We got an accuracy of 98.7% on test set.

6.2.6 Saving the Trained Model:

Once you're confident enough to take your trained and tested model into the production ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle. Make sure you have pickle installed in your environment. Next, let's import the module and dump the model into .pkl file.

7. PROPOSED METHODOLOGY

7.1 System Overview

The system follows a structured pipeline:

1. Image Acquisition – Facial images are captured via camera or sourced from a dataset.
2. Preprocessing – Images are converted to grayscale, resized to 224×224 pixels, and normalized.
3. Feature Extraction and Classification – The CNN model learns discriminative features and outputs class probabilities.
4. Decision Fusion – Softmax probabilities across seven emotion classes are fused to yield the final prediction.

7.2 Dataset

The model is trained and evaluated on a publicly available facial expression dataset from Kaggle containing labeled images across seven emotion categories: anger, disgust, fear, happiness, sadness, surprise, and neutral. The dataset is split 80/20 for training and testing.

7.3 Preprocessing

Preprocessing enhances data quality and model convergence:

- Grayscale Conversion – Reduces input dimensionality.
- Resizing – Standardizes spatial dimensions to 224×224.
- Normalization – Scales pixel values to [0, 1].
- Data Augmentation – Random horizontal flips, rotations, and brightness adjustments increase robustness.

7.4 CNN Architecture

The proposed architecture emphasizes parameter efficiency:

Layer Type	Filters / Units	Kernel / Pool Size	Activation
Conv2D	32	5×5	ReLU
Conv2D	32	5×5	ReLU
MaxPooling2D	–	2×2	–
Dropout (0.25)	–	–	–
Conv2D	64	3×3	ReLU
Conv2D	64	3×3	ReLU
MaxPooling2D	–	2×2	–
Dropout (0.25)	–	–	–
Flatten	–	–	–
Dense	256	–	ReLU
Dropout (0.5)	–	–	–
Dense (Output)	7	–	Softmax

Table -1: CNN Model Architecture Details

7.5. Design rationale:

- Dual convolutional layers before pooling capture richer local patterns before down-sampling.
- Max Pooling reduces spatial dimensions by a factor of two, retaining dominant features.
- Dropout mitigates overfitting by randomly deactivating neurons during training.
- Global flattening followed by a single dense layer replaces multiple fully connected layers, substantially reducing parameters.
- Softmax output provides probability distribution over seven emotion classes.

7.6 Training Configuration

- Optimizer: Adam with default learning rate (0.001)
- Loss Function: Categorical cross-entropy
- Batch Size: 10
- Epochs: Increased iteratively; accuracy gains observed up to convergence.

8. PROPOSED TECHNIQUE USED OR ALGORITHM USED:

Convolutional Neural Networks (CNN) for Real-Time Emotion Detection: The proposed system leverages Convolutional Neural Networks (CNN) for real-time facial emotion detection. CNNs are particularly effective for image classification tasks due to their ability to learn spatial hierarchies of features automatically. In this proposed algorithm, the system is trained on a dataset from Kaggle that contains images labeled with seven different facial expressions, such as happiness, sadness, anger, etc. Preprocessing is performed on the dataset to clean and organize the data, ensuring that it is ready for training. CNNs work by applying convolutional layers to the image data, which helps the network learn spatial features like edges, shapes, and textures in the images, essential for distinguishing different facial expressions. Improved CNN Architecture with Preprocessing for Enhanced Accuracy: The proposed CNN architecture involves several enhancements aimed at improving the overall accuracy of the facial emotion detection model. Preprocessing plays a crucial role in this approach, where the data is cleaned and redundant information is eliminated, helping the model focus on the most informative features. The training process involves multiple epochs, where the model is iteratively trained on the dataset. As the number of epochs increases, the model's accuracy improves, and the predictions become more reliable. The final trained model can classify facial emotions with high accuracy, even in real-time scenarios, making it suitable for practical application such as emotion detection in videos or live streams.

9. UML DIAGRAMS

9.1 USE CASE DIAGRAM



Fig 1 : Use case diagram

EXPLANATION: The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

9.2 CLASS DIAGRAM

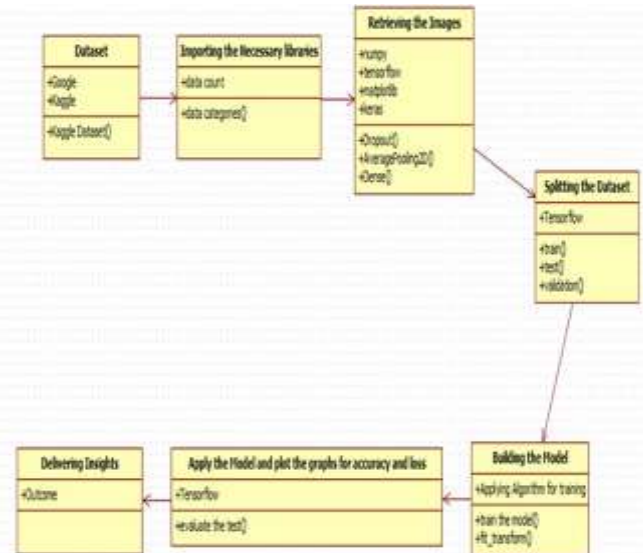


Fig 2 : Class diagram

EXPLANATION In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

9.3 OBJECT DIAGRAM

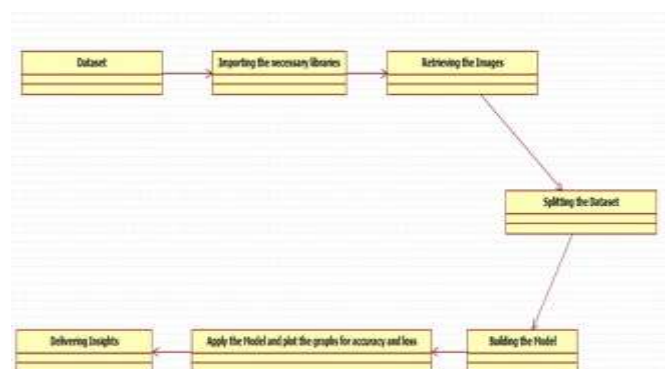


Fig 3 : Object diagram

EXPLANATION: In the above diagram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

9.4 STATE DIAGRAM

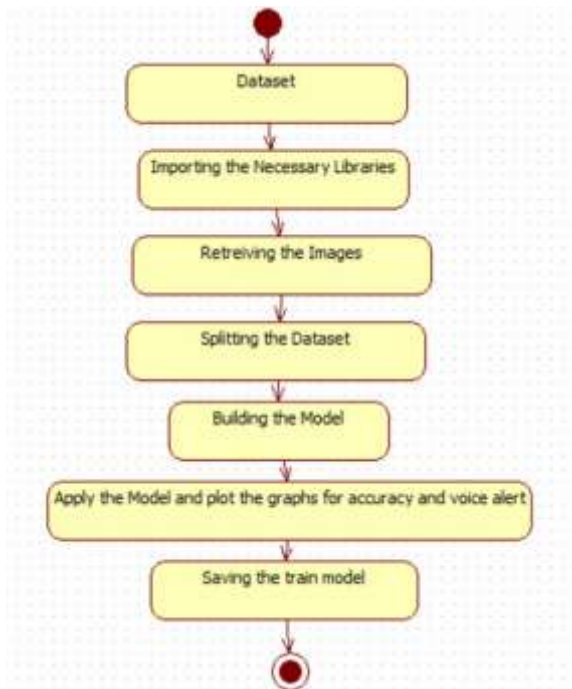


Fig 4 : State diagram

EXPLANATION: State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

9.5 ACTIVITY DIAGRAM

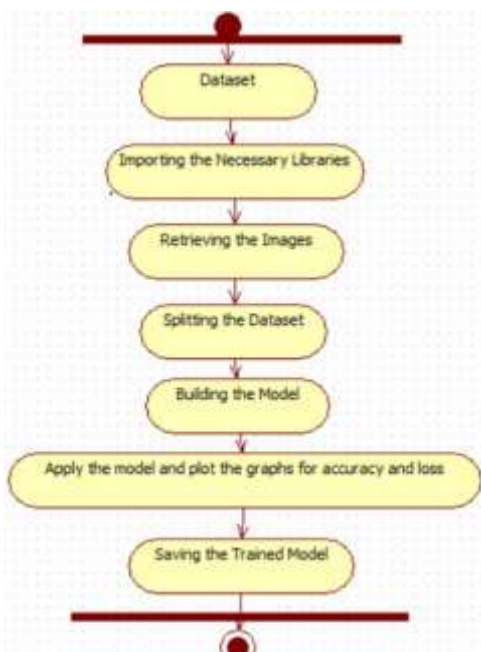


Fig 5 : Activity diagram

EXPLANATION: Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

9.6 SEQUENCE DIAGRAM

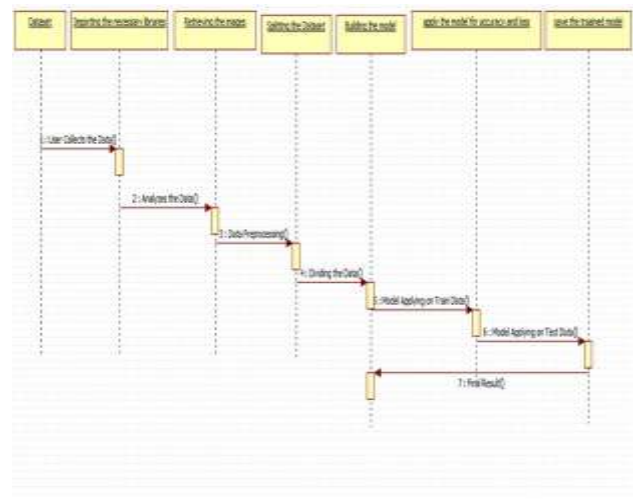


Fig 6 : Sequence diagram

EXPLANATION: A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

9.7 COLLABORATION DIAGRAM

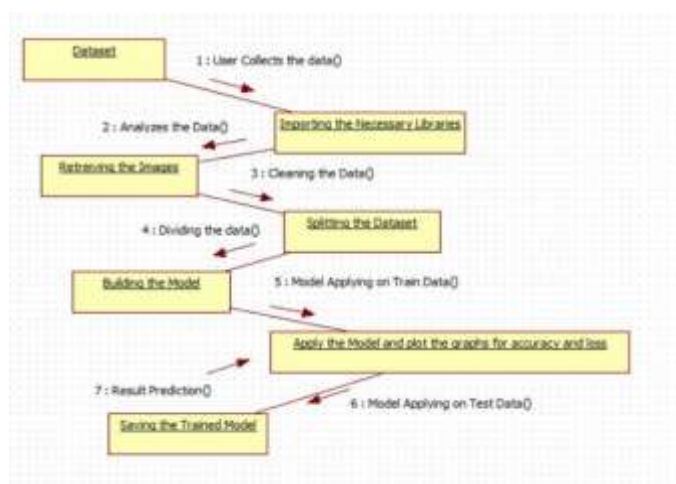


Fig 7 : Collaboration diagram

EXPLANATION: A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

9.8 COMPONENT DIAGRAM

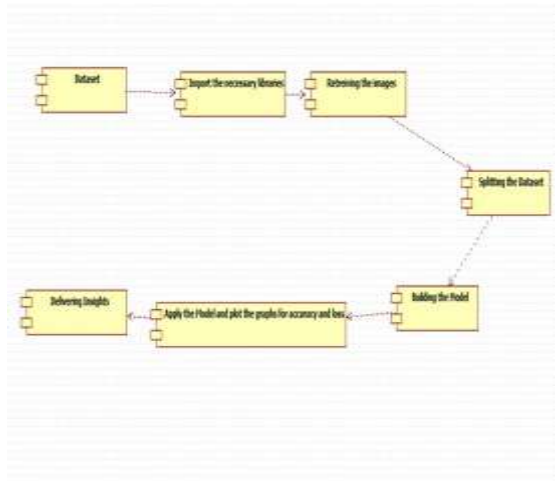


Fig 8 : Component diagram

EXPLANATION In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.

9.9 DATA FLOW DIAGRAM

Level 0

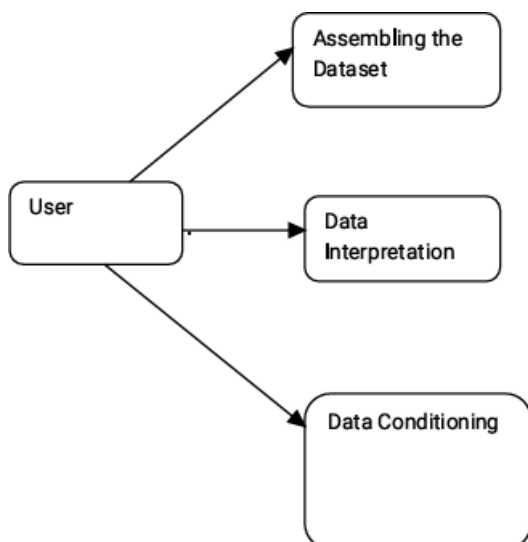


Fig 9 : Data flow diagram

Level 1

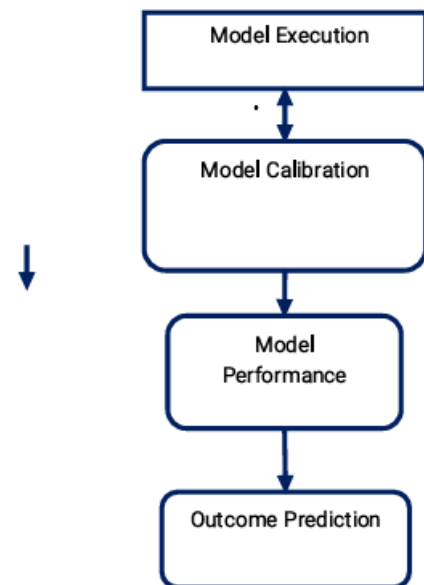


Fig 10 :Data flow diagram

EXPLANATION: Model Execution Model Calibration Model Performance Outcome Prediction Fig 4.9: Data Flow Diagrams A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

9.10 DEPLOYMENT DIAGRAM EXPLANATION:

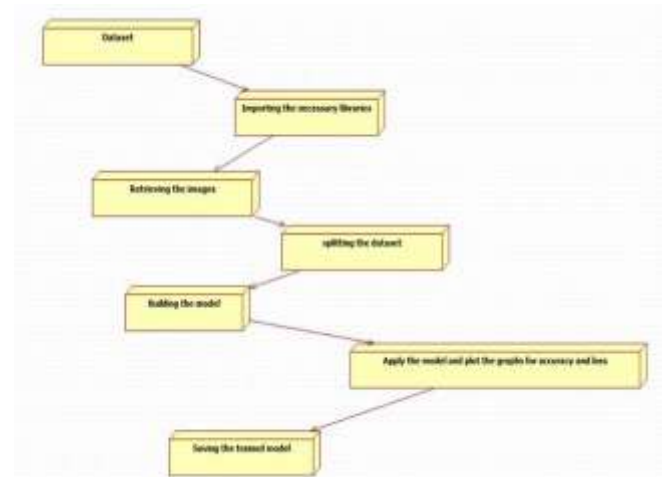


Fig 11: Deployment diagram

EXPLANATION: Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that are going to execute it.

10. EXPERIMENTAL RESULTS

10.1 Results

Metric	Value
Training Accuracy	98.3%
Validation Accuracy	87.6%
Test Accuracy	98.6%

Table -2: Model Performance Evaluation Results

The model achieves 98.6% accuracy on the test set, representing a notable improvement over baseline methods such as VGG-16-based FaceChannel and ROI-KNN approaches.

10.2 Comparison with Existing Methods

Method	Accuracy
VGG-16 FaceChannel (Barros et al.)	~93%
ROI-KNN CNN (Xiao et al.)	~94%
CPC Algorithm (Chang et al.)	~95%
Proposed CNN	98.6%

Table -3: Existing method Comparison

10.3 Discussion

The lightweight design reduces inference latency, making the model suitable for real-time applications on embedded devices and mobile platforms. Eliminating multiple fully connected layers decreases parameter count without significant accuracy loss. Dropout regularization and data augmentation effectively prevent overfitting despite the relatively small training set.

10. SYSTEM IMPLEMENTATION

The system is implemented in Python using Keras with a TensorFlow backend. Key libraries include NumPy, Pandas, Matplotlib, and scikit-learn. The trained model is serialized to an HDF5 (.h5) file for deployment. Real-time inference is demonstrated via webcam input, processing frames at interactive rates on consumer hardware.

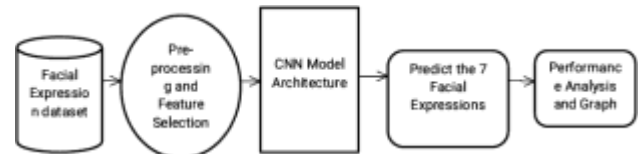


Fig -12: System Architecture

11. SOFTWARE TESTING

11.1 GENERAL

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

11.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

11.3 Types of Tests

11.3.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or

system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

11.3.2 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

11.3.3 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

11.3.4 Performance Test

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being sent to the system for to retrieve the results.

11.3.5 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or— one step up— software applications at the company level— interact without error.

11.3.6 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. Acceptance testing for Data Synchronization: ☐ The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node ☐ The Route add operation is done only when there is a Route request in need ☐ The Status of Nodes information is done automatically in the Cache Updation process.

12. FUTURE ENHANCEMENTS:

Although our model has achieved some results, there may be a lot of noise in the facial expressions captured in real life, such as the images with too strong or too dark lights, blurred images, most of the face is blocked, and other factors that are not conducive to detection. In order to solve this kind of problem, we need to continue our efforts. Although our model has achieved some results, there may be a lot of noise in the facial expressions captured in real life, such as the images with too strong or too dark lights, blurred images, most of the face is blocked, and other factors that are not conducive to

detection. In order to solve this kind of problem, we need to continue our efforts

13. HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system do and not how it should be implemented.

- PROCESSOR : DUAL CORE2DUOS
- RAM : 4GBDDR4
- HARDDISK : 500 GB

14. SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

- Operating System : Windows 10
- Platform : Spyder3
- Programming Language : Python
- Front End : Spyder3

15. FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, Firstly, the system is the first that achieves the standard notion of semantic security for data confidentiality in attribute based deduplication systems by resorting to the hybrid cloud architecture. 3.5

16. NON-FUNCTIONAL REQUIREMENTS

The major non-functional Requirements of the system are as follows

- **Usability:** The system is designed with completely automated process hence there is no or less user intervention.
- **Reliability:** The system is more reliable because of the qualities that are inherited from the chosen platform python. The code built by using python is more reliable. **Performance:** This system is developing in the high level languages and using the advanced back-end technologies it will give response to the end user on client system with in very less time.

- **Supportability:** The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is built into the system.
- **Implementation:** The system is implemented in web environment using Jupyter notebook software. The server is used as the intelligence server and windows 10 professional is used as the platform. Interface the user interface is based on Jupyter notebook provides server system

17. CONCLUSIONS

In this paper, our group proposes and designs a lightweight convolutional neural network for recognizing facial expressions. Our network model reduces the number of parameters in the convolutional layer by eliminating the fully connected layer. And our model has no obviously adverse effect on detection and classification. Our model obtains the good detection results by identifying images outside the dataset, which proves that the model designed in this paper is suitable for multi-classification of facial expressions. In general, we have realized a visual system that can be integrated on devices with low computing power to achieve facial expression classification and reduce a large number of parameters. After comparing with the models in recent years, the accuracy of our model is higher than theirs, and it has achieved good detection results in images outside the dataset from the experimental results.

18. ACKNOWLEDGEMENT

We express our sincere gratitude to our project guide for their valuable guidance, continuous support, and constructive suggestions throughout this research work on facial emotion recognition using Convolutional Neural Networks (CNN), which greatly helped us improve the quality of our work. We also thank our faculty members and institution for providing the necessary resources, facilities, and environment to successfully complete this project. We acknowledge the support of open source tools and libraries such as Python, TensorFlow, and Keras, as well as publicly available datasets that enabled efficient implementation and evaluation of the model. Finally, we extend our thanks to our team members and friends for their cooperation, effort, and support during all stages of this work, which played an important role in achieving the final results.

19. REFERENCES

1. P. Ekman and W. V. Friesen, "Constants across cultures in the face and emotion," *J. Personality Social Psychol.*, vol. 17, no. 2, pp. 124-129, 1971.

2. I. J. Goodfellow et al., "Challenges in representation learning: A report on three machine learning contests," *Neural Netw.*, vol. 64, pp. 59-63, Apr. 2015.
3. S. Lawrence, C. L. Giles, A. Chung Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98113, Jan. 1997.
4. H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Trans. Med. Imag.*, vol. 35, no. 5, pp. 12851298, May 2016.
5. T. Chang, G. Wen, Y. Hu, and J. Ma, "Facial expression recognition based on complexity perception classification algorithm," 2018, arXiv:1803.00185. [Online]. Available: <http://arxiv.org/abs/1803.00185>
6. M.-I. Georgescu, R. T. Ionescu, and M. Popescu, "Local learning with deep and handcrafted features for facial expression recognition," *IEEE Access*, vol. 7, pp. 6482764836, 2019.
7. C. Du and S. Gao, "Image segmentation-based multi-focus image fusion through multi-scale convolutional neural network," *IEEE Access*, vol. 5, pp. 1575015761, 2017.
8. M. Z. Uddin, M. M. Hassan, A. Almogren, A. Alamri, M. Alrubaiyan, and G. Fortino, "Facial expression recognition utilizing local direction-based robust features and deep belief network," *IEEE Access*, vol. 5, pp. 45254536, 2017.
9. M. Z. Uddin, W. Khaksar, and J. Torresen, "Facial expression recognition using salient features and convolutional neural network," *IEEE Access*, vol. 5, pp. 2614626161, 2017.
10. D. Amodei et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2016, pp. 173182.
11. C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2016, p. 1.
12. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 10971105.
13. M. R. Koujan, A. Akram, P. McCool, J. Westerfeld, D. Wilson, K. Dhaliwal, S. McLaughlin, and A. Perperidis, "Multi-class classification of pulmonary endomicroscopic images," in *Proc. IEEE 15th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2018, pp. 15741577.
14. O. Leonovych, M. R. Koujan, A. Akram, J. Westerfeld, D. Wilson, K. Dhaliwal, S. McLaughlin, and A. Perperidis, "Texture descriptors for classifying sparse, irregularly sampled optical endomicroscopy images," in *Proc. Annu. Conf. Med. Image Understand. Anal. Cham, Switzerland: Springer*, 2018, pp. 165176.
15. D. Hazarika, S. Gorantla, S. Poria, and R. Zimmermann, "Self-attentive feature-level fusion for

multimodal emotion detection," in Proc. IEEE Conf. Multimedia Inf. Process. Retr. (MIPR), Apr. 2018, pp. 196201.

16 K.-Y. Huang, C.-H. Wu, Q.-B. Hong, M.-H. Su, and Y.-H. Chen, "Speech emotion recognition using deep neural network considering verbal and nonverbal speech sounds," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), May 2019, pp. 58665870.

17 M. E. Kret, K. Roelofs, J. J. Stekelenburg, and B. de Gelder, "Emotional signals from faces, bodies and scenes influence observers' face expressions, Fixations and pupil-size," *Frontiers Human Neurosci.*, vol. 7, p. 810, Dec. 2013.

18 D. Kollias, P. Tzirakis, M. A. Nicolaou, A. Papaioannou, G. Zhao, B. Schuller, I. Kotsia, and S. Zafeiriou, "Deep affect prediction in the wild: Aff-wild database and challenge, deep architectures, and beyond," *Int. J. Comput. Vis.*, vol. 127, pp. 123, Jun. 2019.

19 D. Kollias, A. Schulc, E. Hajiyev, and S. Zafeiriou, "Analysing affective behavior in the first ABAW 2020 competition," 2020, arXiv:2001.11409. [Online]. Available: <http://arxiv.org/abs/2001.11409>

20 W. Y. Choi, K. Y. Song, and C.W. Lee, "Convolutional attention networks for multimodal emotion recognition from speech and text data," in Proc. Grand Challenge Workshop Hum. Multimodal Lang. (Challenge-HML), 2018, pp.

[21] E. Marinoiu, M. Zanr, V. Olaru, and C. Sminchisescu, "3D human sensing, action and emotion recognition in robot assisted therapy of children with autism," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 21582167. [22] Z. Du, S. Wu, D. Huang, W. Li, and Y. Wang, "Spatio-temporal encoder-decoder fully convolutional network for video-based dimensional emotion recognition," *IEEE Trans. Affect. Comput.*, early access, Sep. 10, 2019, doi: 10.1109/TAFFC.2019.2940224. [23] J. Yang, K. Wang, X. Peng, and Y. Qiao, "Deep recurrent multi-instance learning with spatio-temporal features for engagement intensity prediction," in Proc. 20th ACM Int. Conf. Multimodal Interact., Oct. 2018, pp. 594598. [24] P. Barros, N. Churamani, and A. Sciutti, "The FaceChannel: A light-weight deep neural network for facial expression recognition," in Proc. 15th IEEE Int. Conf. Autom. Face Gesture Recognit. (FG), Buenos Aires, AR, USA, Apr. 2020 pp. 449453. [25] M. Koujan, L. Alharbawee, G. Giannakakis, N. Pugeault, and A. Roussos, "Real-time facial expression recognition 'in the wild' by disentangling 3D expression from identity," in Proc. 15th IEEE Int. Conf. Autom. Face Gesture Recognit. (FG), Buenos Aires, AR, USA, May 2020 pp. 539546. [26] S. Xiao, P. Ting, and R. Fu-Ji, "Facial expression recognition using ROI-KNN deep convolutional neural networks," *Acta Automatica Sinica*, vol. 42, no. 6, pp. 883891, 2016. [27] M. Liu, S. Li, S. Shan, R. Wang, and X. Chen, "Deeply learning deformable facial action parts model for dynamic

expression analysis," in Proc. ACCV. Cham, Switzerland: Springer, 2014, pp. 143157. [28] G. Zhao, H. Yang, and M. Yu, "Expression recognition method based on a lightweight convolutional neural network," *IEEE Access*, vol. 8, pp. 3852838537, 2020. [29] A. F. Abate, P. Barra, S. Barra, C. Molinari, M. Nappi, and F. Narducci, "Clustering facial attributes: Narrowing the path from soft to hard biometrics," *IEEE Access*, vol. 8, pp. 90379045, 2020, doi: 10.1109/ACCESS.2019.2962010. [30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, <https://arxiv.org/abs/1409.1556> arXiv:1409.1556.

[Online]. Available: [31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 28182826. [32] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jul. 2016, pp. 12511258. [33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770778. [34] G. A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, pp. 19, Apr. 2017. [35] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 14991503, Oct. 2016.

BIOGRAPHIES



Mr. Pathipati Chandrasekhar is currently serving as an Assistant Professor in the Department of Computer Science and Engineering at Guru Nanak Institute of Technology (Autonomous), Hyderabad. He earned his B.Tech from SVVSN Engineering College, affiliated with Nagarjuna University, and completed his M.Tech in Computer Science with First Class and Distinction from JNTU Kakinada, Andhra Pradesh. With over eight years of professional teaching experience in various engineering colleges across Andhra Pradesh and Telangana, he has also worked for two years as a Cloud Consultant in Bangalore. He has published 11 research papers in national and international journals, attended 5 Faculty Development Programs (FDPs), and participated in 3 workshops. He is also the holder of 2 patents. His research interests include the Internet of Things (IoT), Artificial Intelligence, Quantum Computing, Deep Learning, and interdisciplinary applications.



Ms. Begari Nandini is currently pursuing B.Tech in the Department of Computer Science and Engineering at Guru Nanak Institute of Technology. She is currently in her 3rd year and has a keen interest in Artificial Intelligence and Machine Learning. She has earned certifications in Data Mining through NPTEL and Ethical Hacking. She has also received an offer letter recognizing her skills and participation. Her project experience includes SEO & Citation Agent developed during a hackathon, Smart Study Planner, and Web Blog projects. She is passionate about learning emerging technologies and enhancing her technical skills through research and practical applications.



Ms. Battala Lakshmi Sai Mahima is currently pursuing B.Tech in the Department of Computer Science and Engineering at Guru Nanak Institute of Technology. She is currently in her 3rd year and has a strong interest in Web Development and modern software technologies. She has developed skills in programming, responsive web application development, and problem solving through academic and project based learning. Her project experience includes web based applications and software development projects that focus on user friendly design and functionality. She is passionate about learning emerging technologies and improving her technical skills through research and practical implementation.



Ms. Vyshnavi is currently pursuing B.Tech in the Department of Computer Science and Engineering 3rd Year. at Guru Nanak Institute of Technology She is currently enhancing her technical knowledge in Java, MySQL, HTML, CSS, and DSA, with a focus on improving programming, problem-solving, and software development skills. She has earned an NPTEL certification in Programming in Java and demonstrating her commitment to continuous learning and technical growth.. Her project experience includes SEO & Citation Agent developed during a hackathon, She has worked on projects such as a Web Blog Project, She is interested in Frontend and Backend application development and She is also interested in creating impactful applications that solve real-world problems and contribute to meaningful technological advancements.