DISTORTED FACE RECONSTRUCTION USING 3D CNN

[1]S.Yoheswari, [2]Nirmal Kmar M, [3] Mohammed Thalha A, [4] Maathesh N
[1]Assistant Professor, Department of Computer Science and Engineering, K.L.N. College of Engineering, Pottapalayam, Tamil Nadu, India
[2,3,4]Student, Department of Computer Science and Engineering, K.L.N. College of Engineering, Pottapalayam, Tamil Nadu, India
[1] yoheswari1988@gmail.com

Abstract: The core objective of this project is to recognize and reconstruct distorted facial images, particularly in the context of accidents. This involves using deep learning techniques to analyze the features of a distorted face and regenerate it into a recognizable form. Deep learning models are well-suited for this task due to their ability to learn complex patterns and representations from data the input data consists of distorted facial images, typically obtained from MRI scans of accident victims. These images may contain various types of distortions such as swelling, bruising, or other injuries that affect facial appearance. By using MRI images, the project can focus on medical applications where accurate facial reconstruction is crucial for diagnosis or identification purposes.Two common pre-trained deep learning models, VGG19 and 3D CNN, are chosen for this project. VGG19 is a convolutional neural network (CNN) known for its effectiveness in image classification tasks, while 3D CNNs are capable of capturing spatial and temporal features from volumetric data like MRI scans. By leveraging these pre-trained models, the project can benefit from their learned representations and potentially achieve better reconstruction accuracy. The performance of the deep learning models is evaluated using metrics such as accuracy and error rate. Accuracy measures how well the models are able to reconstruct the facial features compared to the original images, while the error rate indicates the frequency of incorrect reconstructions. By quantifying these metrics, the project can assess the effectiveness of each algorithm in reconstructing distorted faces. The accuracy levels of VGG19 and 3D CNN are compared using the performance metrics. This comparison helps in identifying which model performs better in the task of facial reconstruction from distorted images. Visualizing the results in the form of a graph provides a clear and concise way to understand the comparative performance of the algorithms. The ultimate goal of this project is to develop a system that can accurately reconstruct distorted faces, which can be invaluable in identifying accident victims or assisting in medical treatments. By providing a reliable method for facial reconstruction, this technology can potentially save lives and improve outcomes for individuals involved in accidents.

Key words: Convolutional Neural Network (CNN), VGG19, Deep Learning, MRI Images, Local Binary Pattern

Corresponding Author: S.Yoheswari
Assistant Professor, Department of Computer Science and Engineering, K.L.N. College of Engineering, Tamil Nadu, India - 630612
Mail: yoheswari1988@gmail.com

Introduction:

The project helps to reconstruct the distorted face with a step-by-step process for the person who has undergone face distortion. In the first process of reconstruction, the distorted face is taken as input in the form of an MRI image. The distorted face is converted to MRI images because the MRI image has the exact detailed structures of every part of the face. The MRI scan uses the magnets and radio waves to produce the MRI images. The MRI is one of the powerful tools which helps to take images of the body parts which are underneath the skin. This method helps to find the distorted face with the help of MRI. The next step of this project is preprocessing in this stage the MRI image is converted into gray scaling for the easy identification of each part. The Gray scaling is helpful in differentiating each part by the brightest being white and the darkest being black.

Gray scaling is used in MRI images because it is used to emphasize the structure without any distorted color. The important reason why Grayscale is used is because it simplifies the algorithm and reduces the computational requirements. The brightness varies from 1- 100 between black and white. The lower level has a darker shade.The next process of this system is feature extraction. In this process the raw data are divided into manageable groups. In this process Local binary pattern is used in which the image is divided into pixel. The local binary pattern is used to capture the structured pattern ad the Gary scale image. Each divided pixel is compared with the other the nearby pixel to identify the value. The identified values are arranged in a specific order.The next step of this project is a data set. The data set is known as the collection of data. In this data set the data set is divided into a 70:30 ratio. The 70% data is the training data and the remaining 30% data is known as the testing data.

The data set is divided in this format in order to cross-check the data. The 30% testing data is used to train the 70% of the data. In the next step, the CNN algorithm is used. The CNN algorithm is the type of network architecture for deep learning algorithms that are especially used in face recognition. CNN comprises three layers, a convolutional layer, a pooling layer and a fully connected layer. the major operation takes place in the convolutional layer to extract facial features where there is a kernel or filter inside this layer traversing across the entire image fields to see whether a feature is present in the image. After multiple iterations, the filter or kernel sweeps the complete image and the dot product will be calculated in each iteration between the kernel and input pixels. the final output will be feature extracted and the image will be converted into numerical values, which allows to extraction of features or patterns relevant to it. The final step is performance metrics. The performance metrics are used to measure the activities and the accuracy. At this level, the accuracy and performance are tested. The final result of the classification is tested with the performance metrics.

Architecture of VGG-19:

The architecture of VGG-19 consists of five convolutional blocks, each of which is followed by a max-pooling layer. The convolutional blocks consist of two or three convolutional layers, each of which is followed by a ReLU activation function. The max-pooling layers reduce the dimensionality of the feature maps, which makes the network more efficient. The output of the fifth convolutional block is flattened into a one- dimensional vector, which is then passed to two fully connected layers. The first fully connected layer has 4096 neurons, and the second fully connected layer has 1000 neurons. The SoftMax activation function is applied to the output of the second fully connected layer, which produces a probability distribution over the 1000 object classes in the ImageNet dataset.

Architecture of 3D CNN:

The architecture of a 3D CNN is similar to that of a 2D CNN. It consists of a series of convolutional layers, pooling layers, and fully connected layers. The convolutional layers extract features from the input data, the pooling layers reduce the dimensionality of the feature maps, and the fully connected layers classify the input data. The main difference between 3D CNNs and 2D CNNs is the use of 3D kernels. A 3D kernel is a filter that is applied to a 3D volume of data. It is typically a small cube of voxels, and it is used to extract features from the data.

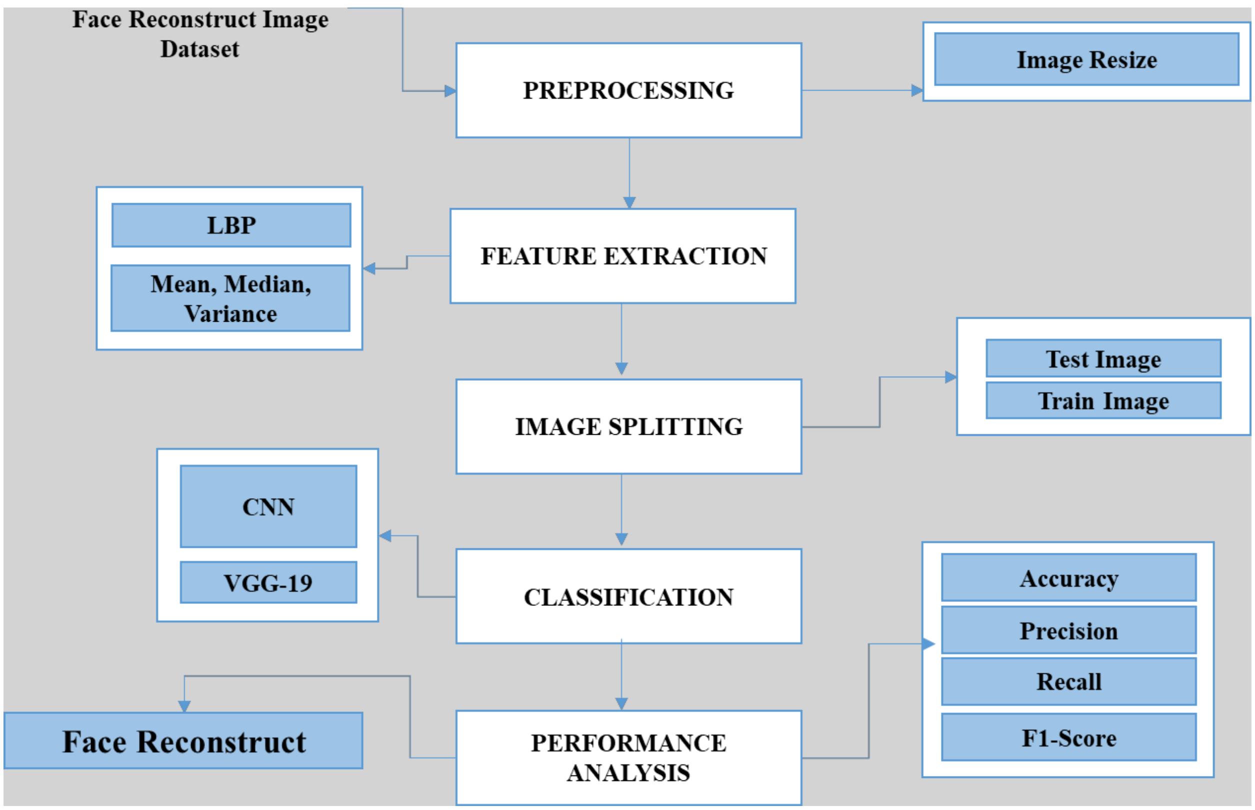

Fig. 1.1 Architecture diagram.

Input Image:

The dataset, Face Reconstruction MRI Image dataset is implemented as input. The dataset is taken from dataset repository. The input dataset is in the format '.png, '.jpg. In this step, we read or load the input image by using the imread () function. In our process, we are used the tkinter file dialogue box for selecting the input image.

Preprocessing:

In this process, we resize the image and convert the image into gray scale. To resize an image, resize () method can be called on it, passing in a two-integer tuple argument representing the width and height of the resized image. Resizing images is a common task in image processing, and there are many different ways to do it. When resizing images for documentation, it's often useful to maintain a consistent size for visual consistency. The function doesn't modify the used image; it instead returns another Image with the new dimensions. There are a number of different ways to convert an image to grayscale. The most common methods are: Average method: This method calculates the average of the red, green, and blue values of each pixel and replaces the original color values with the grayscale value. Luminance method: This method uses the luminance values of each pixel to calculate the grayscale value. The luminance of a pixel is a measure of its perceived brightness. Desaturation method: This method simply removes all of the color information from the image, leaving only the shades of gray.

Feature Extraction:

In this process, we extract the features from pre-processed image. Standard deviation is the spread of a group of numbers from the mean. The variance measures the average degree to which each point differs from the mean. Local Binary Pattern (LBP) is an effective texture descriptor for images which thresholds the neighboring pixels based on the value of the current pixel. LBP descriptors efficiently capture the local spatial patterns and the gray scale contrast in an image. Local binary pattern (LBP) is a feature extraction technique used in computer vision to describe the texture of an image. It is a simple and effective method that has been used in a variety of applications, including image classification, texture analysis, and face recognition. LBP works by comparing the intensity of a pixel to the intensities of its neighbors. If a neighbor's intensity is greater than or equal to the center pixel's intensity, then the corresponding bit in the LBP code is set to 1. Otherwise, the bit is set to 0. The order of the bits in the LBP code is determined by the order of the neighbors. The LBP code for a pixel is a binary number that represents the texture of the local neighborhood around that pixel. For example, if the LBP code for a pixel is 01010101, then this means that the intensities of the pixel's neighbors are arranged in the pattern.

Image Splitting:

During the machine learning process, data are needed so that learning can take place. In addition to the data required for training, test data are needed to e valuate the performance of the algorithm in order to see how well it works. In this process, we considered 70% of the input dataset to be the training data and the remaining 30% to be the testing data. Data splitting is the act of partitioning available data into two portions, usually for cross-validator purposes. Image splitting, also known as image segmentation, is the process of dividing an image into smaller, more manageable pieces. This can be done for a variety of reasons, such as to reduce file size, to improve image compression, or to make an image easier to analyze by computer vision algorithms. There are a number of different ways to split an image. The most common25 methods are: Spatial splitting: This method simply splits the image into a grid of smaller images. This is the most straightforward method, but it can be inefficient for images that have a lot of detail or that are not evenly sized. Color splitting: This method splits the image into separate channels for each color (red, green, and blue). This can be useful for images that have a lot of color variation. Texture splitting: This method splits the image into regions based on their texture. This can be useful for images that have different textures, such as a picture of a forest with trees, sky, and grass. Object splitting: This method splits the image into separate objects. This is the most complex method, but it is also the most accurate.

Classification:

In this process, we implement the deep learning algorithm such as VGG- 19 and CNN. VGG stands for Visual Geometry Group; it is a standard deep Convolutional Neural Network (CNN) architecture with multiple layers. VGG-19, also known as VGGNet-19, is a convolutional neural network (CNN) architecture developed by the Visual Geometry Group (VGG) at the University of Oxford. It is a 19-layer deep CNN that has been shown to achieve state-of-the-art results on a variety of image recognition tasks, including image classification, object detection, and image segmentation. A Convolutional Neural Network (CNN) is a type of deep learning algorithm that is particularly well-suited for image recognition and processing tasks. It is made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers. 3D convolutional neural networks (3D CNNs) are a type of CNN that is designed to process 3D data, such as videos or medical images. They are similar to 2D CNNs, but they have an additional dimension for depth. This allows them to capture spatial relationships between voxels (volume elements) in 3D data, which is important for tasks such as medical image analysis and video classification.

Performance Metrics:

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like, Accuracy of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.

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

Conculsion:

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system

phase according to the user needs. For the hard copy also output comes

Conclusion:

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

Reference:

1. Selvan, M. A. (2024). Deep Learning Techniques for Comprehensive Emotion Recognition and Behavioral Regulation.
2. Selvan, M. A. (2024). SVM-Enhanced Intrusion Detection System for Effective Cyber Attack Identification and Mitigation.
3. Selvan, M. A. (2024). IoT-Integrated Smart Home Technologies with Augmented Reality for Improved User Experience.
4. Selvan, M. A. (2024). Multipath Routing Optimization for Enhanced Load Balancing in Data-Heavy Networks.
5. Selvan, M. A. (2024). Transforming Consumer Behavior Analysis with Cutting-Edge Machine Learning.
6. FELIX, A. S. M. M. D., & KALAIVANAN, X. D. M. S. Averting Eavesdrop Intrusion in Industrial Wireless Sensor Networks.

7. Selvan, M. A. (2021). Robust Cyber Attack Detection with Support Vector Machines: Tackling Both Established and Novel Threats.
8. Selvan, M. A. (2023). INDUSTRY-SPECIFIC INTELLIGENT FIRE MANAGEMENT SYSTEM.
9. Selvan, M. A. (2023). FIRE MANAGEMENT SYSTEM FOR INDUTRIAL SAFETY APPLICATIONS.
10. Selvan, M. A. (2023). CONTAINMENT ZONE ALERTING APPLICATION A PROJECT BASED LEARNING REPORT.
11. Selvan, M. A. (2023). A PBL REPORT FOR CONTAINMENT ZONE ALERTING APPLICATION.
12. Reka, R., R. Karthick, R. Saravana Ram, and Gurkirpal Singh. "Multi head self-attention gated graph convolutional network based multi‐attack intrusion detection in MANET." Computers & Security 136 (2024): 103526.
13. Meenalochini, P., R. Karthick, and E. Sakthivel. "An Efficient Control Strategy for an Extended Switched Coupled Inductor Quasi-Z-Source Inverter for 3 Φ Grid Connected System." Journal of Circuits, Systems and Computers 32.11 (2023): 2450011.
14. Karthick, R., et al. "An optimal partitioning and floor planning for VLSI circuit design based on a hybrid bio-inspired whale optimization and adaptive bird swarm optimization (WO-ABSO) algorithm." Journal of Circuits, Systems and Computers 32.08 (2023): 2350273.
15. Rajagopal RK, Karthick R, Meenalochini P, Kalaichelvi T. Deep Convolutional Spiking Neural Network optimized with Arithmetic optimization algorithm for lung disease detection using chest X-ray images. Biomedical Signal Processing and Control. 2023 Jan 1;79:104197.
16. Karthick, R., and P. Meenalochini. "Implementation of data cache block (DCB) in shared processor using field-programmable gate array (FPGA)." Journal of the National Science Foundation of Sri Lanka 48.4 (2020).
17. Karthick, R., A. Senthilselvi, P. Meenalochini, and S. Senthil Pandi. "Design and analysis of linear phase finite impulse response filter using water strider optimization algorithm in FPGA." Circuits, Systems, and Signal Processing 41, no. 9 (2022): 5254-5282.
18. Karthick, R., and M. Sundararajan. "SPIDER-based out-of-order execution scheme for HtMPSOC." International Journal of Advanced Intelligence paradigms 19.1 (2021): 28-41.
19. Karthick, R., Dawood, M.S. & Meenalochini, P. Analysis of vital signs using remote photoplethysmography (RPPG). J Ambient Intell Human Comput 14, 16729–16736 (2023). https://doi.org/10.1007/s12652-023-04683-w
20. Elechi, P., & Onu, K. E. (2022). Unmanned Aerial Vehicle Cellular Communication Operating in Non-terrestrial Networks. In *Unmanned Aerial Vehicle Cellular Communications* (pp. 225-251). Cham: Springer International Publishing.
21. Prasad, B. V. V. S., Mandapati, S., Haritha, B., & Begum, M. J. (2020, August). Enhanced Security for the authentication of Digital Signature from the key generated by the

CSTRNG method. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)* (pp. 1088-1093). IEEE.

22. Khandelwal, A. R., Mutneja, L., Thakar, P., & Patil, P. (2019). Basics and Applications of Big Data.
23. Meena, S. B., Patil, P. R., Kandharkar, S. R., Hemalatha, N., Khade, A., Dixit, K. K., & Chinthamu, N. (2024). The Evolution Of Smart Grid Technologies: Integrating Renewable Energy Sources, Energy Storage, And Demand Response Systems For Efficient Energy Distribution. *Nanotechnology Perceptions*, 1098-1109.
24. Virmani, D., Ghori, M. A. S., Tyagi, N., Ambilwade, R. P., Patil, P. R., & Sharma, M. K. (2024, March). Machine Learning: The Driving Force Behind Intelligent Systems and Predictive Analytics. In *2024 International Conference on Trends in Quantum Computing and Emerging Business Technologies* (pp. 1-6). IEEE.
25. Kalluru, S. R., & Gurijala, P. K. R. Increasing Efficiency of Goods Receipt with Mobility Solutions.
26. Gurijala, P. K. R., & Kalluru, S. R. Enhancing Manufacturing Efficiency with Mobility Applications.
27. Gurijala, P. K. R., Kalluru, S. R., & Dave, R. Maximizing Procurement Efficiency through Purchase Requisitions Load Building.
28. Kalluru, S. R., & Gurijala, P. K. R. Improving Putaway Efficiency Through Innovative Solutions.
29. Lanka, S., Konjeti, P. C., & Pinto, C. A. (2024, June). A Review: Complete Analysis of the Cache Architecture for Better Performance. In *2024 Second International Conference on Inventive Computing and Informatics (ICICI)* (pp. 768-771). IEEE.