**Journal of Information Systems Engineering and Management**
2025, 10(11s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

**Research Article**

# Estimating Pain Intensity from Facial Expressions

Yogita Chavan[1], Dr. Chhaya Pawar[2]
[1]*Research Scholar, Mumbai, Maharashtra,*
*yogita84@gmail.com*
*Research Supervisor, Mumbai, Maharashtra*
*csp.cm.dmce@gmail.com*

**ARTICLE INFO**

Received: 14 Nov 2024
Revised: 26 Dec 2024
Accepted: 14 Jan 2025

**ABSTRACT**

Facial expressions are intricate, change over time, and are difficult to decipher, hence automated emotion recognition using facial expressions has been a popular research area in computer vision, which focuses on image processing and pattern recognition. The research community has access to many facial expression databases, which are crucial tools for analyzing a variety of face expression detection methods, but still the existing models are affected by over-fitting issues. In this research work, a framework using Hybrid optimization based deep CNN (HOA based deep CNN) has been developed that can recognize pain by facial expression and evaluate the pain's intensity. The proposed model uses UNBC-McMaster Shoulder Pain Expression Archive Database (UNBC) as the database. From the identified region of interest (ROI), feature extraction is done using the RESNET 101, facial activity descriptors and hybrid weighted facial activity descriptors. The collected features are processed using ensemble deep CNN classifiers. The optimization methods Adam, Stochastic Gradient Descent (SGD), Cat Swarm, and Grey Wolf (GWO) are used together to optimize the ensemble deep CNN. The hybrid optimization offers the best tuning, which substantially reduces the computing time and speeds up the convergence. The accuracy, sensitivity, and specificity achieved by the HOA-based deep CNN model based on the TP are 90.06%, 98.38%, and 99.35%, respectively. For the K-fold, the accuracy, sensitivity, and specificity achieved are 94.95%, 97.33%, and 99.04%, respectively.

**Keywords:** Pain intensity detection, deep CNN model, RESNET 101, hybrid optimization and facial activity descriptors.

## 1. Introduction:

Recent medical research has shown that facial expression is one of the effective indicators and reflects the health condition of a person. It has been clinically established, abnormal health conditions can cause aberrant and notable facial features and they are significantly different from normal facial expressions [9]. Pain is acknowledged as the fifth important indication of the health condition and is a clear reflection of injuries, diseases, and mental discomfort. Consequently, pain is a clear sign of a health problem. The expression on faces can often be characterized as a contemplative, spontaneous reaction to uncomfortable circumstances [10]. Thus, a thorough investigation of a patient's painful facial expressions might give medical professionals objective data for a precise disease diagnosis [7].

Reliable and accurate pain evaluation is a prerequisite for effective pain management. But pain is a complicated and highly subjective phenomenon [11], [12], and is frequently connected to unpleasant physical and psycho-physiological symptoms. Additionally, each person's perception of pain is distinct and differs from the other subject.[13][4]. The intricacy of assessing pain is further increased by this particular factor. Self-report is therefore regarded as the gold standard for measuring pain and has been proven to be helpful in offering insightful information for efficient pain treatment [14], [15][16-17].

While facial expression recognition has been the focus of recent advancements in automatic pain detection [18], research interest in pain estimates utilizing autonomic physiology assessed by skin conductance (SC) and heart rate (HR) is expanding [19]–[22]. These signals can be conveniently picked up and monitored in hospitals or even in a mobile ambulance , for instance, using wrist biosensors. The major goal of this research is to estimate the intensity of pain using skin conductance and heart rate data. Major autonomic alterations in the parameters are noted because of pain [23], which enhances sympathetic output. Sweating released due to pain enters into skin surface pores and impacts sympathetic excitatory efferent neurons innervate sweat glands [24], [25]. Electro dermal activity (EDA), which is altered by sweat release, causes an increase in skin conductance until the sweat is reabsorbed or evaporates [9].

A person's face, which is a rich source of actual information, represents their social interactions and health condition, like how they express their emotions and their pain [26]. The majority of multidimensional behavioral checklists and rating systems that are used to evaluate pain include a considerable portion of facial actions. The contraction of facial muscles during facial expression has an objective meaning according to the Facial Action Coding System (FACS) [6]. Automatic pain assessment could be accomplished by the examination of facial expressions. A person's health-related non-verbal cues can often be read from their face. It is possible to think of facial expression as a spontaneous and introspective response to traumatic situations. The Facial Action Coding System (FACS) is the foundation for the majority of facial expression studies. FACS is a tool for evaluating facial features objectively.

Deep learning algorithms are capable of identifying the presence of pain as well as the relative severity level by using pertinent information (such as a face image) gathered by a camera. The movement of the face muscles and its correlation with the PSPI scores can be used to infer this. The problems, however, still exist in facial image-based pain detection [6]. An unbalanced ambient light, distance and angle of the camera, and background can all have an impact on how facial images appear. Assessing pain levels is somewhat challenging due to external influences like the phenomenon of smiling in pain or variances in pain based on gender. A very few and limited databases containing labels for pain in facial images are available. As a result, it is challenging to find a full dataset of facial images with pain labels on each image. This research project aims to promote potential uses of pain detecting systems in the field of health informatics and to offer an efficient solution to overcome these issues.

The main aim of the research is to develop a HOA based deep CNN framework that can recognize pain by facial expression and evaluate the pain's intensity. The UNBC is the primary source of the data, which must be pre-processed to increase its quality. After preprocessing, ROI extraction is used to remove the unnecessary portions of the image and take out the key regions. Afterward, feature extraction is used to extract the characteristics that are important for identifying pain through the expressions of the subject's face. The process of feature extraction is carried out, the collected features are fed into an ensemble deep CNN classifier to identify the different levels of pain which is effectively tuned using the hybrid optimization, which substantially reduces the computing time and speeds up the convergence.

**Proposed Hybrid optimization based deep CNN model:**

In the proposed Deep CNN model, classifiers are enabled with hybrid optimization, which greatly aids in fine-tuning the hyper parameters. Adam, GWO, SGD, and CSO are combined to create the HOA model. The grey wolves search in a unique manner, that all the wolves search for the prey individually and once the prey is spotted the wolves will converge for the attacking of the prey. In attacking the prey, the selection of the careful step size plays an important role and this will be optimized by the enabling of the Adam update rule. Here, there is a need for an adaptive mechanism to identify whether the grey wolf should attack the prey or leave the prey based on their weight age and this will be performed using the characteristics obtained from SGD. Deep CNN is tuned using hybrid optimization and it efficiently recognizes the level of pain from the expression on the face, hence minimizing the over fitting issues.

The manuscript's Section 2 outlines recent studies, their techniques, and the difficulties, and Section 3 represents an illustration of an efficient pain intensity detection model. The proposed hybrid optimization is presented in Section 4, followed by a discussion of the experimental results in Section 5, and the conclusion is provided in Section 6.

## 2. Motivation:

Training a model on huge databases is challenging and time-consuming and over-fitting is another main challenge. To understand these challenges, a detailed review of available research work is carried out in the next section.

### 2.1 Literature review:

Ghazal Bargshady et al. [1] created a hybrid CNN-BiLSTM (EJH-CNN-BiLSTM) deep learning algorithm for the multi-classification of pain. A spatiotemporal Convolutional network was created by G Mohammad Tavakolian and Abdenour Hadid [2] for the purpose of estimating pain intensity from face videos. This model's results show that 3D deep architecture has a higher accuracy in estimating pain intensity, However, it was challenging to develop a model that uses both appearance and motion data to account for facial dynamics. A continuous pain monitoring technique was created by Mingzhe Jiang [3] with the classification of numerous physiological data, and it achieved great accuracy in the binary pain expression classification. Although these one-dimensional evaluation measures are thought to be effective in assessing acute pain, they necessitate interactive conversation between the patient and caregiver, which can be challenging for people with poor communication skills. In the process of creating a multimodal pain recognition system, Patrick Thiam et al. [4] established a number of classifier fusion algorithms that have been evaluated. This model performed better when estimating the severity of cold pain. But there were excessive over fitting problems with this model. This research used cutting-edge methods in a smart healthcare framework to assess pain using a sentiment analysis system that was created by Anay Ghosh et al. [5]. This system achieved greater accuracy with fewer training steps. However, the training can be rather imbalanced because there are a lot more background things than faces. A new algorithm for automatically identifying pain intensity using video frames was created by Ghazal Bargshady et al. [6]. The main limitations of this model are issues with exploding and vanishing gradients, as well as challenges with concurrent training and separation tasks. Nevertheless, it obtained good accuracy in a short amount of time and fewer epochs. A 3D deep architecture for dynamic facial video representation was created by Mohammad Tavakolian and Abdenour Hadid [7] which more effectively represents spontaneous face fluctuations. But in some instances, it was difficult for this model to determine whether a participant was in pain or not. Yikang Guo et al. experiment [8] on measuring the level of cold pain examined the significance of spatial-temporal information on facial expression based on cold pain and demonstrated good accuracy. However, it is very difficult to recognize expressions on the human face because faces of different people are presented in various poses that differ according to the patient's age.

**2.2 Challenges:**

- Face expression recognition requires the model to distinguish between subtle differences within each class (intra-class) and between different classes (inter-class). This poses a challenge as the distribution of facial expressions is highly complex and requires the balancing of complex 23 distribution of intra- and inter-class differences.
- RNNs, which can handle temporal dependencies, often suffer from the vanishing or exploding gradient problem, especially when dealing with long sequences.
- Training on large datasets can be time-consuming. Additionally, large datasets often contain a high degree of class imbalance, which can downgrade the model's performance, especially for underrepresented classes.
- Overfitting is a common problem, especially when the dataset is imbalanced. The model tends to memorize patterns from the majority class and performs poorly on the minority class.

## 3. Efficient pain intensity detection from facial expression based on Hybrid optimization based deep CNN model:

The main aim of the research is to develop a system that can assess the level of pain and evaluate pain based on facial expression. The UNBC is the initial source of the data, which is pre-processed to improve the quality. Following preprocessing, ROI extraction eliminates the unnecessary portions of the image and identifies the key regions. The characteristics that are crucial for recognizing pain from the subject's facial expressions are then extracted using feature extraction. The process of feature extraction is carried out using the RESNET 101, facial activity descriptors, and hybrid weighted facial activity descriptor. To differentiate between the different levels of pain, the collected features are fed into an ensemble deep CNN classifier. The hybridized optimization methods Adam, SGD, CSO, and GWO are used to optimize the ensemble deep CNN. In noisy situations, the Adam optimization is best suited for handling sparse gradients. SGD is a quick and efficient optimization method for figuring out the variables or function coefficients that lowers a cost function. The CSO and the GWO are also employed to reduce computational complexity because of their basic structures, lower storage needs, and reduced computational demands. The combination of all these optimization techniques results in producing an effective result. The ensemble approach also offers best tuning, which substantially reduces the computing time, speeds up convergence, and aids in producing the desired results.

Finally, the output generated from the deep CNN classifiers identifies the level of pain. The architecture of the proposed pain intensity detection model is shown in figure 1.

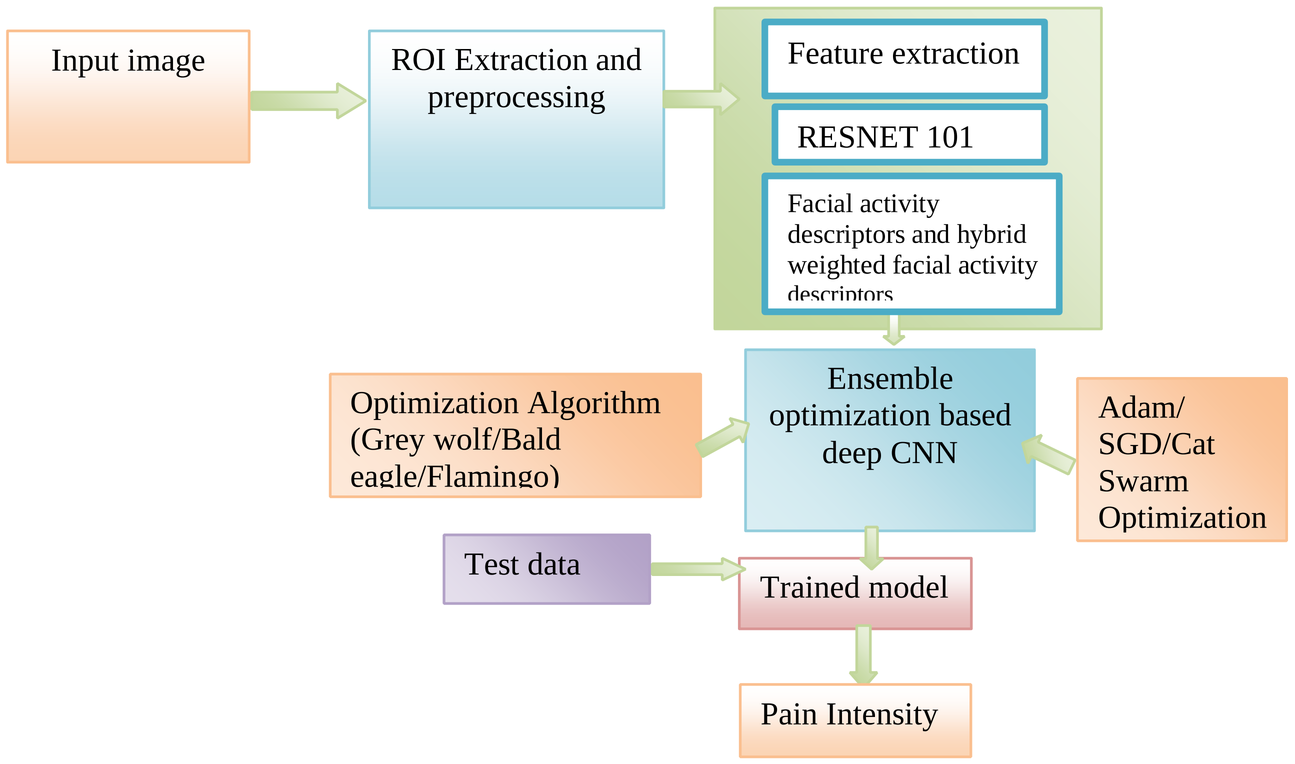

Input image
ROI Extraction and preprocessing
Feature extraction
RESNET 101
Facial activity descriptors and hybrid weighted facial activity descriptors
Optimization Algorithm (Grey wolf/Bald eagle/Flamingo)
Ensemble optimization based deep CNN
Adam/ SGD/Cat Swarm Optimization
Test data
Trained model
Pain Intensity

**Figure 1:** Architecture of the proposed pain intensity detection model

**3.1 Input:**

The UNBC is used as the input for the pain detection model, which is analytically represented as

$$B = \sum_{f=1}^{e} B_y \tag{1}$$

Where, $B$ is referred to as a database and $B_y$ denotes the first dataset images, with images ranging from 1 to $e$.

**3.2 Pre processing:**

Image preprocessing step improves the efficiency and effectiveness of model training and inference, especially when dealing with large image datasets. By compressing the size of large images, training times can be significantly shortened without compromising with the l performance, Preprocessing also enhances the quality of images by removing noise, adjusting contrast, and sharpening features, which ensures that the model learns from clean, high-quality data. Additionally, normalization of image pixel values helps create consistency across the dataset, allowing the model to converge more quickly. Moreover, preprocessing can remove unwanted distortions and artifacts, improving the model's ability to focus on relevant features rather than irrelevant distortions. Overall, preprocessing ensures that images are well-prepared for model training, leading to better performance and faster processing times.

ROI extraction is the process of locating and isolating particular portions or regions within an image that are of particular interest or importance for additional processing or analysis.

**3.3 Feature extraction:**

Feature extractor is a technique used in machine learning to improve accuracy. It increases the accuracy of the algorithm’s prediction by focusing on the most essential features and eliminating the unnecessary and unimportant ones.

### 3.3.1 RESNET 101:

The ResNet architecture is inspired by the VGG-19 model and builds upon the concept of deep convolutional neural networks (CNNs) by introducing shortcut connections. In a typical CNN, multiple layers are interconnected, each learning a different set of features at the end of every layer. In the case of ResNet, convolutional layers make up 33 of the total layers, and by using a stride of two, the model performs down-sampling directly. The final layers of the ResNet include a globally average pooling layer followed by a fully connected layer with SoftMax activation. As the removal of input features obtained from that layer, residual learning is easily understood. ResNet accomplishes this by using shortcut connections between each pair of the 33 filters to connect the input of one layer directly to the input $j^{th}$ of another layer $(j+t)^{th}$. A key feature of ResNet is its use of residual learning, where the input features from one layer are passed directly to the next layer through shortcut connections. This method helps mitigate the problem of vanishing gradients by enabling the network to learn residual mappings rather than the direct mapping itself. As the network trains, the weights are adjusted to emphasize the contribution of the layer directly beneath and reduce the influence of the one above. This structure makes ResNet easier to train compared to traditional deep CNNs and solves the problem of accuracy degradation that often arises in very deep networks. It is a modified version of the 50-layer Res Net called ResNet-101, which has 101 layers.

### 3.3.2 Facial activity descriptors:

The features that describe various aspects of face expressions and movements are referred to as facial activity descriptors. They are used to record and measure a variety of face activities, including emotions, gestures, actions, and facial muscle movements. These descriptors offer a structured description of face behavior, enabling analysis and comprehension of facial expressions in people. This section explains feature extraction relevant to pain.

**a) LBP:** LBP is a useful technique for feature extraction, carried out by encoding the pixel-level data that describes the local structure in texture images. In a grayscale image, the threshold value of the central pixel controls the values of the eight pixels that surround it for one particular pixel. To be more exact, the result will be set to one if the values of all neighboring pixels are greater than or equal to the values of the central pixels; otherwise, the result will be set to zero. The results are then averaged after being multiplied by weights with powers of two to get the LBP code for the center pixel. Circular adjacent pixels may be included in the LBP operator's scope. LBP's invariance to monotonic gray level changes brought on by changes in illumination is one of its key benefits. It achieves good performance in texture classification due to its straightforward calculation and strong discrimination. The following is a definition of the initial LBP operator:

$$LBP_{M,T} = \sum_{M=0}^{M-1} u(h_M - h_v)2^M$$

$$u(y) = \begin{cases} 1, & y \geq 0 \\ 0, & y < 0 \end{cases} \quad \textbf{(2)}$$

The center pixel gray values and its neighborhoods are represented as $h_v$ and $h_M$, total number of neighboring pixels is denoted as $M$, neighboring pixels radius is denoted as $T$, The operator is made insensitive to variations in the mean gray value by approximating the distribution with local neighborhood differences. LBP converts the relationship between the center pixel's and the neighboring pixels' gray levels into the binary numbers 0 and 1. It serves as a straightforward and effective local descriptor for texture. However, it lacks the local intensity difference and produces a huge different gray block with the same LBP value block.

**b) LTP:** LTP expands LBP to three-value codes in which gray levels in a zone of width $\pm g$ around $f_v$ are quantized to zero, to +1, to -1, and to zero. According to Eq. (3), the indicator $U(n)$ is replaced with a three-valued function and the binary LBP code is converted to a ternary LTP code. Since $g$ is a user-specified threshold in this case, LTP codes are less rigorously invariant to gray-level transformations but are still more noise-resistant.

$$U(n, f_v, g) = \begin{cases} 1, & n \geq f_v + g \\ 0 & |n - f_v| < g \\ -1 & n \leq f_v - g \end{cases} \quad \textbf{(3)}$$

**c) LDP:** LDP creates an eight bit binary code for each pixel in the image. The local region of the image of size 3 × 3 is convolved with Kirsch masks in eight directions to produce the eight bit pattern. The Kirsch masks are convolved for each 3 × 3 region with regard to the eight directions of the central pixel. Eight responses are received with the label $x_o, x_1, \ldots\ldots\ldots x_7$. The top $d$ responses are chosen from these and set to 1, while 0 is set for all other responses. Equation gives a mathematical definition of it.

$$T[g(u,v)] := (T_m = 1) \qquad if\ 0 \leq m \leq 7\ and\ x_m \geq \varphi \quad \textbf{(4)}$$

Where, $\psi = d^{th}(X)$ and $X = \{x_o, x_1, \ldots\ldots x_7\}$. For the entire image, the process is repeated, resulting in eight Kirsch masks being applied to each pixel. LDP-labeled image $M_X$ is the name given to the resulting image. The LDP-labeled image $M_X$ is segmented into regions, and a histogram is produced for each zone. The histograms from each region are combined to get the final histogram for the input image.

### 3.3.3 Hybrid weighted facial activity descriptor:

The term hybridization describes the process of merging numerous local feature descriptors to provide a more reliable and useful feature representation for a variety of computer vision tasks. In image analysis and recognition, the local feature descriptors LDP, LTP, and LBP are frequently utilized. These descriptors could perform better if they were combined, and it would also allow for the acquisition of more features of the image data. Each description is briefly described below, along with examples of how they can be combined:

**a) LDP:** LDP is used to determine the difference between the centre pixel and its central pixels. It is very helpful for capturing structural information in images and encoding local texture changes.

**b) LTP:** By adding a third value to the standard 0 and 1 range, LTP expands on LBP. The uniformity or diversity of the texture patterns as well as their presence or absence is encoded.

**c) LBP:** The local texture is encoded by LBP by comparing each pixel's intensity value to that of its neighbors. The presence or absence of specific intensity patterns is represented as a binary code.

Concatenating or combining the feature representations of LDP, LTP, and LBP is one method for hybridizing them. A more thorough representation that includes several texture qualities can be made by concatenating the features derived from each descriptor into a single feature vector. Combining LDP, LTP, and LBP can take advantage of each descriptor's advantages, capturing various facets of local texture and enhancing the overall discriminative power of the feature representation.

## 4. Ensemble deep CNN classifier:

The extracted features are fed forward to the deep CNN classifier to detect the pain intensity detection from the facial expression. Convolution, nonlinearity, and pooling stages which are the main stages in the feature learning process are combined with the fully connected stage in order to extract the features in depth. These phases are known as CONV, ReLU, POOL, and FC, respectively. ReLU denotes that the stage uses the ReLU function as a nonlinearity function. In the CNN model, several of these fundamental steps are sequentially stacked as layers, with the goal of automatically learning the deep characteristics from input. The Soft Max function will go on to employ the features that were extracted during the feature learning phase for the classification process. To provide the output in a probabilistic manner, it was applied to the network's final layer.

### a) Convolution layer:

The CNN model greatly benefits from convolution. It takes the form of sliding window 2D convolution with 3D input, which employs several filters of size $T \times T \times E$ to convolute with the 3D input of size B concurrently. One of the feature map's outputs is the convolution result from a single filter. Applying $G$ filters in a convolution layer, which is the entire output of the convolution stage, yields a stack of $G$ feature maps. In other words, the information in each convolution layer of the CNN model is viewed as the features in three dimensions.

**b) RELU layer:**

Since convolution is a linear process, the feature maps created by the convolution layer always undergo through a non-linear ReLU function to extract the non-linear property of the feature. By changing all of the input's negative numbers to zero while leaving the others as they were originally, it is extremely simple yet efficient.

$$\mathrm{Re}\,lu(i) = \begin{cases} 0, & i < 0 \\ i, & i >> 0 \end{cases} \tag{5}$$

**c) Pooling layer:**

Each feature map input was sub sampled by the CNN model's pooling layer. The feature map's width $(B)$ and height $C$ dimensions will decrease after the pooling stage. Sub sampling allows for the usage of different pooling sizes. When using a pooling size of 2×2, only one value out of four is chosen to represent the subsample. To choose the one sub sampling value from these four values, however, max pooling, average pooling, or any other pooling types might be used.

**d) Fully connected layer:**

A few Fully Connected (FC) layers are used in the output region of the majority of CNN models. The preceding layer's feature maps are structured as vector data for the FC layer's input and are connected to one another in a manner similar to a multi-layer perceptron (MLP) network.

**e) Soft max layer:**

In the process of learning features, a network is constructed by sequentially arranging layers of convolution, nonlinearity by ReLU function, and pooling. The normalization stage and dropout, which are also utilized in Alex Net, are a few of possible additional stages. In order to convert the network output into values in terms of probability for the classification process, the soft Max function is used. In (6), it serves a purpose.

$$s(D_j) = \frac{m^{Dj}}{\sum_{f=1}^{T} m^{Df}} \tag{6}$$

Where, the exponential value of $D_j$ is $m^{Dj}$, $s(D_j)$ is the soft maximum outcome of each $D_j$, output of each neuron $j$ is $D_j$, the vector component of $D$ is $T$, and the exponential output is $D_j$.

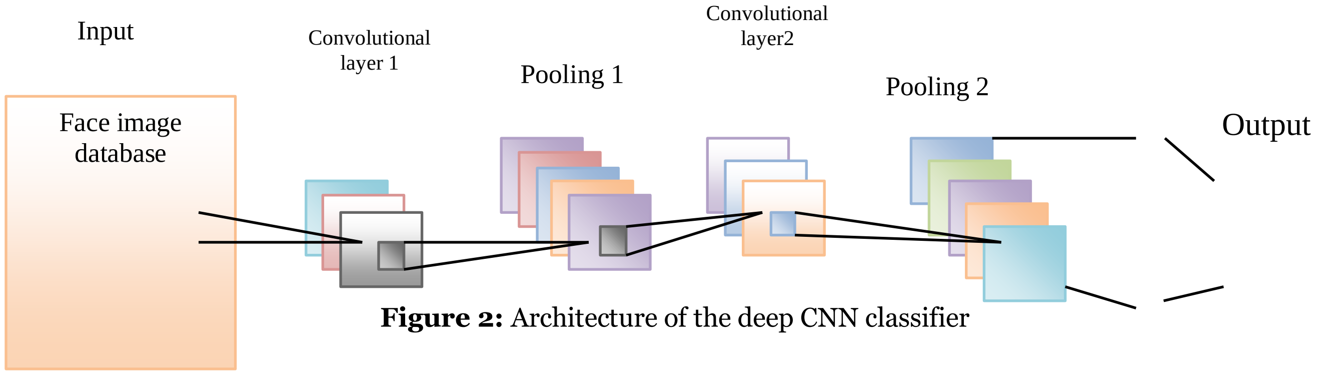


**Figure 2:** Architecture of the deep CNN classifier

## 4. Proposed Hybrid optimization algorithm:

Deep CNN classifiers are enabled with hybrid optimization, which greatly aids in fine-tuning the weight and bias of the classifier. Adam [34], GWO [35], SGD [36], and CSO [37] are typically combined to create the HOA model. The grey wolves search in a unique manner, that all the wolves search for the prey individually and once the prey is spotted the wolves will converge for the attacking of the prey. In attacking the prey, the selection of the careful step size plays an important role and this will be achieved by the enabling of the Adam update rule. Here, there is a need for an adaptive mechanism to identify whether the grey wolf should attack the prey or leave the prey based on their weight age and this will be performed using the characteristics obtained from SGD.

### *4.1 Initialization:*

In GWO, first choose the number of wolves you want to use before implementing the grey wolves to address the issues. Each wolf has a unique location, which is made up of $B$ dimensions, velocity values for each dimension, a fitness value that indicates how well the wolf has adapted to the fitness function. Since GWO keeps the top solution till the end of iterations, the best position in one of the wolf would be the final solution.

$$I = I_1 + \ldots\ldots\ldots\ldots\ldots I_n \tag{7}$$

$I$ denotes the individual wolves and $n$ denotes the total groups

### *4.2 Assigning ranks:*

In order to statistically explain the wolf's social hierarchy when creating GWO, we take the fittest solution into consideration as the alpha $a$. Therefore, beta $b$ and delta $d$, respectively, are used to denote the second and third-best options. We suppose that omega $x$ represents the remaining possible solutions.

### *4.3 Tracing mode:*

The tracing mode is the sub-model that simulates the grey wolf scenario while tracing some targets. When a wolf enters the tracing phase, it moves in all directions at its own velocity. Tracing mode's operation can be explained as:

### *4.4 Searching for the target:*

Step1: In accordance with equation (8), update the velocities for each dimension $(t_{m,e})$.

Step2: Verify that the speeds fall within the range of the maximum speed. Set the new velocity to the maximum value if it exceeds the range.

Step3: Update the $wolf_m$ location in accordance with equation (9).

**(8)**

$i_{best,e}$ denotes the wolf's position, which has the highest fitness value; $i_{m,e}$ represents the wolf's position. $v_1$ is a random number between $[0,1]$ and $b_1$ is a constant. $t_{m,e} = t_{m,e} + v_1 \times b_1 \times (i_{best,e} - i_{m,e}), \qquad where\ e = 1,2,\ldots\ldots\ldots U$

$$i_{m,e} = i_{m,e} + t_{m,e} \tag{9}$$

### *4.5 Encircling behavior:*

During a hunt, a pack of grey wolves encircles its prey. The subsequent equations are suggested as a mathematical model for encircling behavior:

$$\vec{E} = \left| \vec{D}.\vec{Y}_M(g) - \vec{Y}(g) \right| \tag{10}$$

$$\vec{Y}(g+1) = \vec{Y}_M - \vec{B}.\vec{E} \tag{11}$$

Where, $g$ indicates the current iteration, $\vec{B}$ and $\vec{D}$ represents the coefficient vectors, prey's position vector of the prey is denoted as $\vec{Y}_M$ , and the position vector of the wolf is denoted as $\vec{Y}$ .

### *4.6 Hunting behavior:*

Wolves possess the capacity to locate their prey and surround them. The alpha usually leads the hunt. There's a chance that the beta and delta will occasionally go hunting. In an abstract search space, the exact spot of the ideal (prey) is unknown. We assume that the alpha (best candidate solution), beta, and delta have superior knowledge of the location of prey, updating the positions to coincide with the locations of the top search agents in order to mathematically mimic the hunting behavior of grey wolves. In this regard, the formulas below are described as.

$$\vec{E}_a = \left|\vec{D}_1.\vec{Y}_a - \vec{Y}\right|,\ \vec{E}_a = \left|\vec{D}_2.\vec{Y}_b - \vec{Y}\right|,\ \vec{E}_a = \left|\vec{D}_3.\vec{Y}_c - \vec{Y}\right| \quad (12)$$

$$\vec{Y}_1 = \vec{Y}_a - \vec{B}_1.\left(\vec{E}_a\right),\ \vec{Y}_2 = \vec{Y}_b - \vec{B}_2.\left(\vec{E}_b\right),\ \vec{Y}_3 = \vec{Y}_c - \vec{B}_3.\left(\vec{E}_c\right) \quad (13)$$

$$\vec{Y}(g+1) = \frac{\vec{Y}_1 + \vec{Y}_2 + \vec{Y}_3}{3} \quad (14)$$

### *4.7 Attacking prey:*

As previously stated, the prey is attacked by the grey wolves when it stops moving to conclude the chase. To mathematically describe approaching the prey, we decrease the value of $\vec{b}$ . Also keep in mind that $\vec{b}$ limits $\vec{B}$ range of fluctuation. If you want, you may say that $\vec{B}$ is a random variable with a range of $[-2b, 2b]$, where $\vec{b}$ declines from 2 to 0 throughout the course of iterations. A search agent's next position can be anywhere between its current position and the location of the prey when random values of $\vec{B}$ are in the range $[-1,1]$. According to the research, $|B| < 1$ stimulates the wolves to kill their target.

The alpha, beta, and delta locations can now be used by the GWO algorithm's search agents to update their positions, which they can then use to begin an attack in the direction of the prey while employing the so far proposed operators. The GWO method is susceptible to local solution stagnation when used with these operators.

### *4.8 Searching for prey:*

The way that grey wolves search is distinct: each wolf looks for the prey on its own, and when it is found, all the wolves come together to attack the victim.

The grey wolves' vectors $\vec{B}$ and $\vec{D}$ are computed as follows:

$$\vec{B} = 2\,\vec{b}\,.\,\vec{j}_1 - \vec{b} \quad (15)$$

$$\vec{D} = 2.\,\vec{j}_2 \quad (16)$$

In this case, the components of $\vec{b}$ are linearly reduced over the course of iterations from 2 to 0, while the random vectors are $j_1, j_2$ which is in the range $[0,1]$.

A two-dimensional position vector and a few potential neighbors are shown to show the implications of Equations (10) and (11). A position of the wolf $(Y, Z)$ may change its position in accordance with the prey's position $(Y^*, Z^*)$. By varying the values of the $\vec{B}$ and $\vec{D}$ vectors, it is possible to travel to several locations relative to the current location around the best agent. For instance, setting $\vec{B} = (1,0)$ and $\vec{D} = (1,1)$ will lead to $(B^* - B, Z^*)$. Keep in mind that wolves can travel to any position using the random vectors $j_1$ *and* $j_2$. Equations (10) and (11) allow a grey wolf to update its location at any random point inside the area surrounding its prey.

In attacking the prey, the selection of the careful step size plays an important role and this will be achieved by the enabling of the adam update rule. In Adam update rule

Adam's update rule's thorough selection of step sizes is one of its key characteristics. Assuming $\in = 0$ represents the actual step taken in parameter space at time step $s$ is $\Delta_s = \alpha . \hat{g}_s / \sqrt{\hat{u}_s}$. In this example, $(1-\beta_1) > \sqrt{1-\beta_2}$, and $|\Delta_s| \le \alpha$ in all other cases, are the two upper constraints on the effective step size $|\Delta_s| \le \alpha.(1-\beta_1)/\sqrt{1-\beta^2}$. When a gradient has been zero at all time steps other than the present time step, the first situation only occurs in the most extreme case of sparsity, the effective step size will be smaller for the less sparse cases.

Here, there is a need for an adaptive mechanism to identify whether the grey wolf should attack the prey or leave the prey based on their weight age and this will be performed using the characteristics obtained from SGD.

$$B = 2b.j_1 - b \tag{17}$$

Replace $j_1$ with velocity of cat (8) from CSO hence

$$B = 2b.t_{m,e} + v_1 \times b_1 \times (i_{best,e} - i_{m,e}) - b \tag{18}$$

$$B = 0.5(2b, t - b)\alpha . \hat{g}_s / \sqrt{\hat{u}_s} + 0.5\, R(mJ) - \frac{\mu}{2} \hat{\nabla} \varepsilon (mJ) \tag{19}$$

In this approach, $R$ *and* $m$ stands for the weight vectors, and $\hat{\nabla} \varepsilon$ is the stochastic gradient of the time-averaged squared stochastic error. It is well known that, if convergence rate is expressed in terms of data time-samples, $J = 1$ produces the algorithm that converges the fastest. The speed of convergence, however, grows monotonically with $J$ if the convergence rate is expressed in terms of algorithm iterations, which are $J$ times less frequent than data time-samples.

**Algorithm1:** Pseudo code of hybrid optimization algorithm

| **S.NO** | **Pseudo code of the proposed HOA model** |
|---|---|
| **1** | Initialization: I |
| **2** | Searching for the target: |
| **3** | Determine the velocity of the wolf: $t_{m,e} = t_{m,e} + v_1 \times b_1 \times (i_{best,e} - i_{m,e}),\quad where\ e = 1,2,.........U$ |

| | |
|---|---|
| **4** | Update the position: $i_{m,e} = i_{m,e} + t_{m,e}$ |
| **5** | Encircling behaviour: $\vec{E} = \left\lvert \vec{D}.\vec{Y}_M(g) - \vec{Y}(g) \right\rvert$ |
| **6** | $\vec{Y}(g+1) = \vec{Y}_M - \vec{B}.\vec{E}$ |
| **7** | Hunting: $\vec{Y}(g+1) = \frac{\vec{Y}_1 + \vec{Y}_2 + \vec{Y}_3}{3}$ |
| **8** | Searching for the prey: $\vec{B} = 2\vec{b}.\vec{j}_1 - \vec{b}$ |
| **9** | $\vec{D} = 2.\vec{j}_2$ |
| **10** | Adam update phase: $\Delta_s = \alpha.\hat{g}_s / \sqrt{\hat{u}_s}$ |
| **11** | Weight age using the characteristics obtained from SGD. |
| **12** | $B = 2b.j_1 - b$ |
| **13** | Replace $j_1$ with velocity of cat (2) from CSO hence |
| **14** | $B = 2b.t_{m,e} + v_1 \times b_1 \times (i_{best,e} - i_{m,e}) - b$ |
| **15** | $B = (2b, t - b) del(s) + SGD$ |
| **16** | Termination: |

## 5. Result and discussion:

A model for detecting pain intensity is developed using HOA based deep CNN, and its efficacy is assessed with respect to other approaches.

### 5.1 Experimental set up:

The aim of assessing its performance and development, the HOA-based deep CNN model is implemented in Python on Windows 11 with 16GB RAM.

### 5.2 Dataset description:

#### 5.2.1 UBBC [26]:

The search for 129 participants, 63 men and 66 women who self-reported having a shoulder soreness issue used three physiotherapy clinics as well as advertisements placed on the McMaster University campus. A wide range of occupations made up the remaining third, which comprised the community's residents and one-fourth of students. Participants' shoulder pain was diagnosed with different conditions. More than half of the participants admitted to taking medicines.

**Experimental results:**

The analysis of the pain detection model, which was conducted using the developed HOA-based deep CNN, is expanded upon in this chapter. Sections 3a, 3b, and 3c show the original input image, whereas Sections 3d, 3e, and 3f show the LBP extracted image and Sections 3g, 3h, and 3i show the final output.

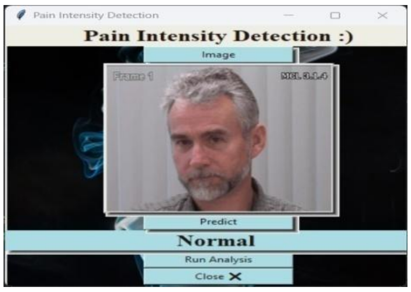
| | | | |
|---|---|---|---|
| Input image | | | |
| | a) | b) | c) |
| LBP image | | | |
| | d) | e) | f) |
| Pain intensity detection |  | | |
| | g) | | |

**Figure 3:** Result of an experiment utilizing a deep CNN based on HOA

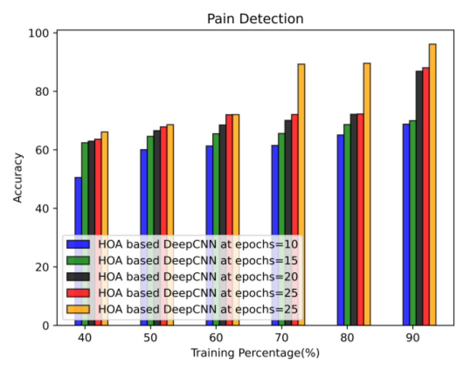
### 5.3 Performance analysis based on TP:

Figure 4 illustrates the metrics of the HOA-based deep CNN models for detecting the pain intensities.

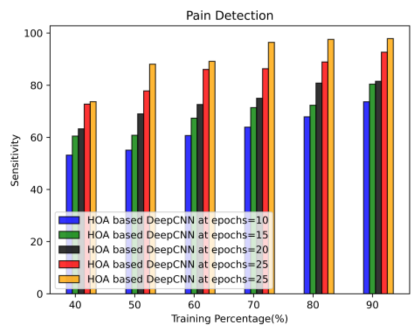
Figure 4a) shows that the HOA-based deep CNN technique achieves accuracy values during TP of 90 of 68.77%, 69.98%, 86.86%, 88.06%, and 96.14%.

Figure 4b) shows the outcomes of the deep CNN model based on the HOA. Their sensitivity levels are 73.65%, 80.42%, 81.52%, 92.67% and 97.90% respectively, at a 90% TP.

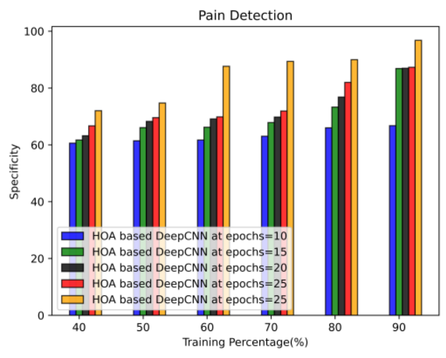
The results, which were subsequently obtained using the HOA-based deep CNN model and are shown in Figure 4c), had a specificity of 86.87%, 86.98%, 87.34%, and 96.83% with a TP of 90%.



**(a)**



**(b)**



**c)**

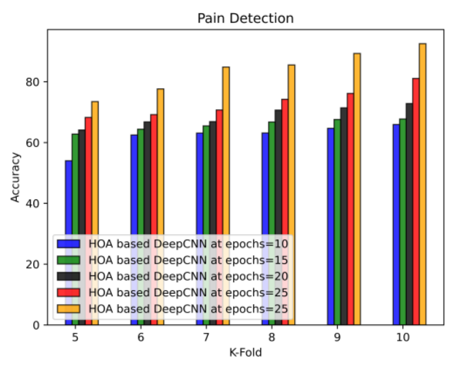
**Figure 4:** Performance analysis concerning TP a) accuracy, b) sensitivity and c) specificity

**5.4 Performance analysis based on K-fold:**

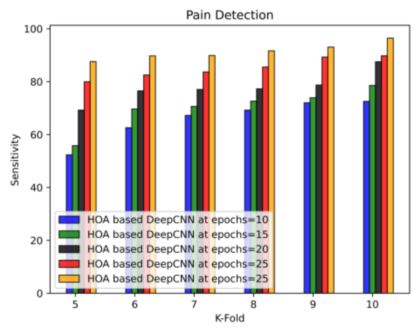
The HOA-based deep CNN model's effectiveness in detecting the pain intensities is depicts in Figure 5a). As part of the k-fold 10 the HOA-based deep CNN achieves accuracy values of 65.96%, 67.75%, 72.84%, 81.10%, and 92.54%.

The effectiveness of HOA-based deep CNN model in detecting the pain intensities is displayed in Figure 5b). The HOA-based deep CNN reaches values of 72.57%, 78.57%, 87.55%, 89.78% and 96.50% in terms of sensitivity.

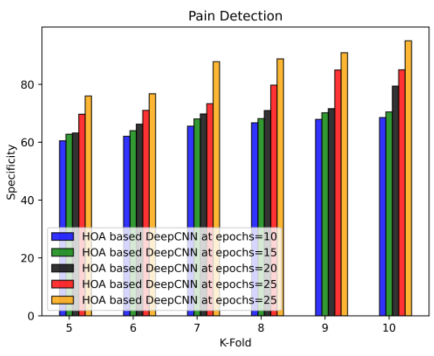
The effectiveness of HOA-based deep CNN in detecting the pain intensities is shown in Figure 5c). The HOA-based deep CNN attains specificity of 68.53%, 70.46%, 79.40%, 85.02%, and 95.06%.



**(a)**



**(b)**



c)

**Figure 5:** Performance analysis concerning k-fold a) accuracy, b) sensitivity and c) specificity

### 5.5 Comparative methods:

The performance of the proposed HOA-based deep CNN model is demonstrated through comparisons with SVM, MLP, CNN, BiLSTM, deep CNN, deep CNN based bald eagle search optimization (deep CNN-BES), and deep CNN based flamingo.

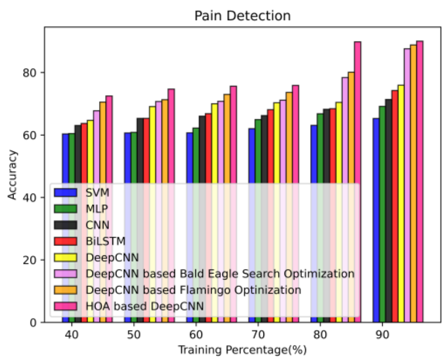
#### 5.5.1 Comparative analysis based on TP:

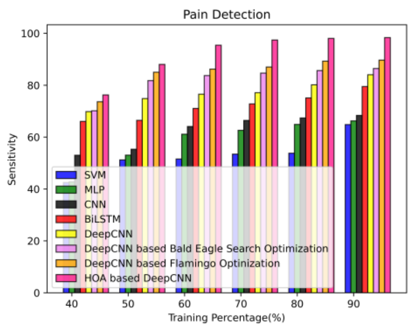
Figure 6) presents the metrics for the TP 90, which is utilized to contrast the effectiveness of the HOA-based deep CNN with that of the other comparison techniques.

The HOA-based deep CNN model's accuracy in determining pain intensity is seen in Figure 6a). The HOA-based deep CNN has a TP of 90 and 90.06% accuracy, which is 1.35% more accurate than the deep CNN-BES.

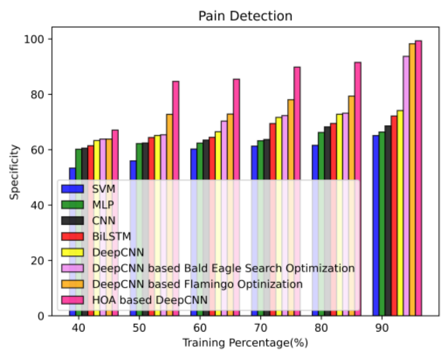
As demonstrated in Figure 6b), the HOA-based deep CNN model performs 8.86% better at detecting pain intensity than the deep CNN-BES and has a sensitivity of 98.38%.

As shown in Figure 6c), the HOA-based deep CNN model performs 1.06% better at detecting pain intensity than the deep CNN-BES and has a sensitivity of 99.35%.





(a) (b)



c)

**Figure 6:** Comparative analysis concerning TP a) accuracy, b) sensitivity and c) specificity

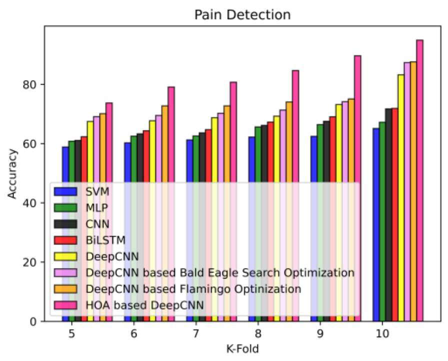
### 5.5.2 Comparative analysis based on K-fold:

Figure 7) presents the metrics for the K-fold 10, which is utilized to contrast the effectiveness of the HOA-based deep CNN with that of the other comparison techniques.

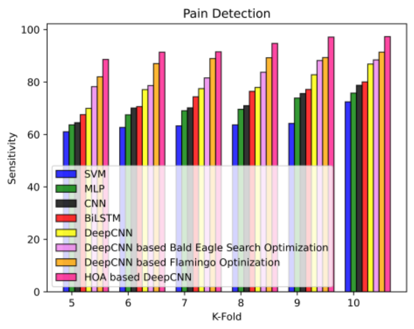
The HOA-based deep CNN model's accuracy in determining pain intensity is seen in Figure 7a). The HOA-based deep CNN achieves 94.95% accuracy during the k-fold 10, which is 7.71% more accurate than the deep CNN-BES.

As demonstrated in Figure 7b), the HOA-based deep CNN model performs 6.13% better at detecting pain intensity than the deep CNN-BES and has a sensitivity of 97.33%.

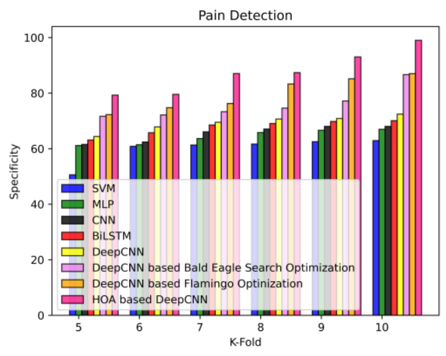
As shown in Figure 7c), the HOA-based deep CNN model performs 12.13% better at detecting pain intensity than the deep CNN-BES and has a sensitivity of 99.04%.



(a)



(b)



c)

**Figure 7:** Comparative analysis concerning k-fold a) accuracy, b) sensitivity and c) specificity

## 5.6 Comparative discussion:

The results of the models used to determine pain intensity are assessed in this section. SVM [27], MLP [28], CNN [29], BiLSTM [30], deep CNN [31], deep CNN based bald eagle search optimization [32], and deep CNN based flamingo [33] are shown in the table 1 and 2. In terms of the metrics, the HOA-based deep CNN model performs better than the other models.

**Table 1:** Comparative analysis based on TP

| | TP | | |
|---|---|---|---|
| Models | Accuracy | Sensitivity | Specificity |
| SVM | 65.30 | 64.82 | 65.11 |
| MLP | 69.17 | 66.27 | 66.39 |
| CNN | 71.39 | 68.37 | 68.61 |
| BiLSTM | 74.25 | 79.50 | 72.16 |
| deep CNN | 75.94 | 84.03 | 74.16 |
| deep CNN based bald eagle optimization | 87.63 | 86.45 | 93.74 |
| deep CNN based flamingo | 88.85 | 89.67 | 98.30 |
| HOA-based deep CNN | 90.06 | 98.38 | 99.35 |

**Table 2:** Comparative analysis based on K-fold

| | K-fold | | |
|---|---|---|---|
| Models | Accuracy | Sensitivity | Specificity |
| SVM | 65.12 | 72.42 | 62.92 |
| MLP | 67.23 | 75.77 | 66.97 |
| CNN | 71.72 | 78.76 | 68.06 |
| BiLSTM | 71.92 | 80.04 | 70.08 |
| deep CNN | 83.24 | 86.85 | 72.44 |
| deep CNN based bald eagle optimization | 87.40 | 88.44 | 86.69 |
| deep CNN based flamingo | 87.63 | 91.37 | 87.03 |
| HOA-based deep CNN | 94.95 | 97.33 | 99.04 |

## 6. Conclusion:

This research presents a Hybrid Optimization Algorithm (HOA)-based deep Convolutional Neural Network (CNN) model designed to recognize pain from facial expressions and assess its intensity. The UNBC dataset, which serves as the primary source of data, undergoes preprocessing to enhance its quality. Subsequently, a Region of Interest (ROI) extraction is performed to isolate relevant areas of the image, focusing on key features necessary for pain recognition. The next step involves feature extraction, utilizing the RESNET 101 architecture and weighted hybrid facial activity descriptors, along with other facial activity descriptors. These features are then inputted into an ensemble deep CNN classifier to identify various levels of pain.

To optimize the deep CNN model, hybrid optimization methods are applied. These methods provide optimal tuning, significantly reducing computation time and speeding up convergence. As a result, the HOA-based deep CNN model achieves impressive performance metrics. For the TP (test protocol), the accuracy, sensitivity, and specificity are 90.06%, 98.38%, and 99.35%, respectively. In the K-fold validation setup, these values improve to 94.95% accuracy, 97.33% sensitivity, and 99.04% specificity.

In summary, the model demonstrates high efficiency and effectiveness in recognizing pain and evaluating its intensity, with notable performance across both TP and K-fold test scenarios.

## References:

[1] Bargshady, Ghazal, Xujuan Zhou, Ravinesh C. Deo, Jeffrey Soar, Frank Whittaker, and Hua Wang. "Enhanced deep learning algorithm development to detect pain intensity from facial expression images." Expert Systems with Applications 149 (2020): 113305.

[2] Tavakolian, Mohammad, and Abdenour Hadid. "A spatiotemporal convolutional neural network for automatic pain intensity estimation from facial dynamics." International Journal of Computer Vision 127 (2019): 1413-1425.

[3] Jiang, Mingzhe, Riitta Mieronkoski, Elise Syrjälä, Arman Anzanpour, Virpi Terävä, Amir M. Rahmani, Sanna Salanterä, Riku Aantaa, Nora Hagelberg, and Pasi Liljeberg. "Acute pain intensity monitoring with the classification of multiple physiological parameters." Journal of clinical monitoring and computing 33 (2019): 493-507.

[4] Thiam, Patrick, Viktor Kessler, Mohammadreza Amirian, Peter Bellmann, Georg Layher, Yan Zhang, Maria Velana et al. "Multi-modal pain intensity recognition based on the senseemotion database." IEEE Transactions on Affective Computing 12, no. 3 (2019): 743-760.

[5] Ghosh, Anay, Saiyed Umer, Muhammad Khurram Khan, Ranjeet Kumar Rout, and Bibhas Chandra Dhara. "Smart sentiment analysis system for pain detection using cutting edge techniques in a smart healthcare framework." Cluster Computing 26, no. 1 (2023): 119-135.

[6] Bargshady, Ghazal, Xujuan Zhou, Ravinesh C. Deo, Jeffrey Soar, Frank Whittaker, and Hua Wang. "The modeling of human facial pain intensity based on Temporal Convolutional Networks trained with video frames in HSV color space." Applied Soft Computing 97 (2020): 106805.

[7] Tavakolian, Mohammad, and Abdenour Hadid. "Deep spatiotemporal representation of the face for automatic pain intensity estimation." In 2018 24th International Conference on Pattern Recognition (ICPR), pp. 350-354. IEEE, 2018.

[8] Guo, Yikang, Li Wang, Yan Xiao, and Yingzi Lin. "A personalized spatial-temporal cold pain intensity estimation model based on facial expression." IEEE Journal of Translational Engineering in Health and Medicine 9 (2021): 1-8.

[9] P. H. Tseng, I. G. M. Cameron, G. Pari, J. N. Reynolds, D. P. Munoz, and L. Itti, "High-throughput classification of clinical populations from natural viewing eye movements," Journal of Neurology, vol. 260, no. 1, pp. 275–284, 2013.

[10] K. D. Craig, K. M. Prkachin, and R. V. E. Grunau, Handbook of Pain Assessment. Guilford Press, 2011, ch. The Facial Expression of Pain.

[11] R. C. Coghill, J. G. McHaffie, and Y.-F. Yen, "Neural correlates of interindividual differences in the subjective experience of pain," Proceedings of the National Academy of Sciences of the United States of America, vol. 100, no. 14, pp. 8538–8542, 2003.

[12] C. S. Nielsen, A. Stubhaug, D. D. Price, O. Vassend, N. Czajkowski, and J. R. Harris, "Individual differences in pain sensitivity: genetic and environment contributions," Pain, vol. 136, no. 1, pp. 21–29, 2008.

[13] R. C. Coghill, "Individual differences in the subjective experience of pain: New insights into mechanisms and models," Headache, vol. 50, no. 9, pp. 1531–1535, 2010.

[14] E. B. Kim, H.-S. Han, J. H. Chung, B. R. Park, S.-N. Lim, K. H. Yim, Y. D. Shin, K. H. Lee, W.-J. Kim, and S. T. Kim, "The effectiveness of a self-reporting bedside pain assessment tool for oncology inpatients," Journal of Palliative Medicine, vol. 15, no. 11, pp. 1222– 1233, 2012.

[15] N. C. De Knegt, F. Lobbezoo, C. Schuengel, H. M. Evenhuis, and E. J. A. Scherder, "Self-reporting tool on pain in people with intellectual disabilities (STOP-ID!): a usability study," Augmentative and Alternative Communication, vol. 32, no. 1, pp. 1–11, 2016.

[16] G. A. Hawker, S. Mian, T. Kendzerska, and M. French, "Measures of adult pain: Visual Analog Scale for Pain (VAS Pain), Numeric Rating Scale for Pain (NRS Pain), McGill Pain Questionnaire (MPQ), Short-Form McGill

Pain Questionnaire (SF-MPQ), Chronic Pain Grade Scale (CPGS), Short Form-36 Bodily Pain Scale (SF36 BPS), and Measure of Intermittent and Constant Osteoarthritis Pain (ICOAP),” Arthritis Care and Research, vol. 63, no. S11, pp. S240–S252, 2011.

[17] C. Eckard, C. Asbury, B. Bolduc, C. Camerlengo, J. Gotthardt, L. Healy, L. Waialar, C. Zeigler, J. Childers, and J. Horzempa, “The integration of technology into treatment programs to aid in the reduction of chronic pain,” Journal of Pain Management & Medecine, vol. 2, no. 3, p. 118, 2016.

[18] D. Lopez-Martinez, O. Rudovic, and R. Picard, “Personalized Automatic Estimation of Self-Reported Pain Intensity from Facial Expressions,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Hawaii, USA: IEEE, 72017, pp. 2318–2327.

[19] D. Lopez-Martinez and R. Picard, “Multi-task Neural Networks for Personalized Pain Recognition from Physiological Signals,” in Seventh International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW), San Antonio, TX, 2017.

[20] M. Kachele, M. Amirian, P. Thiam, P. Werner, S. Walter, G. Palm, and ¨ F. Schwenker, “Adaptive confidence learning for the personalization of pain intensity estimation systems,” Evolving Systems, vol. 8, no. 1, pp. 71–83, 3 2017.

[21] S. Gruss, R. Treister, P. Werner, H. C. Traue, S. Crawcour, A. Andrade, and S. Walter, “Pain Intensity Recognition Rates via Biopotential Feature Patterns with Support Vector Machines,” PLOS ONE, vol. 10, no. 10, 10 2015.

[22] S. Walter, S. Gruss, K. Limbrecht-Ecklundt, H. C. Traue, P. Werner, A. Al-Hamadi, N. Diniz, G. M. da Silva, and A. O.

[23] Andrade, “Automatic pain quantification using autonomic parameters,” Psychology and Neuroscience, vol. 7, no. 3, pp. 363–380, 2014.

[24] E. E. Benarroch, “Pain-autonomic interactions: A selective review,” Clinical Autonomic Research, vol. 11, no. 6, pp. 343–349, 12 2001.

[25] A.-A. Dube, M. Duquette, M. Roy, F. Lepore, G. Duncan, and ´ P. Rainville, “Brain activity associated with the electrodermal reactivity to acute heat pain,” NeuroImage, vol. 45, no. 1, pp. 169–180, 3 2009.

[26] A.B. Ashraf, et al., The painful face–pain expression recognition using active appearance models, Image Vis. Comput. 27 (12) (2009) 1788–1796, http://dx.doi.org/10.1016/j.imavis.2009.05.007.

[27] Schuldt, Christian, Ivan Laptev, and Barbara Caputo. "Recognizing human actions: a local SVM approach." In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., vol. 3, pp. 32-36. IEEE, 2004.

[28] Taud, Hind, and J. F. Mas. "Multilayer perceptron (MLP)." Geomatic approaches for modeling land change scenarios (2018): 451-455.

[29] Zhang, Kai, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. "Learning deep CNN denoiser prior for image restoration." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3929-3938. 2017.

[30] Siami-Namini, Sima, Neda Tavakoli, and Akbar Siami Namin. "The performance of LSTM and BiLSTM in forecasting time series." In 2019 IEEE International conference on big data (Big Data), pp. 3285-3292. IEEE, 2019.

[31] Shin, Hoo-Chang, Holger R. Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M. Summers. "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning." IEEE transactions on medical imaging 35, no. 5 (2016): 1285-1298.

[32] Sayed, Gehad Ismail, Mona M. Soliman, and Aboul Ella Hassanien. "A novel melanoma prediction model for imbalanced data using optimized SqueezeNet by bald eagle search optimization." Computers in biology and medicine 136 (2021): 104712.

[33]Mandawkar, Umakant, and Tausif Diwan. "Alzheimer disease classification using tawny flamingo based deep convolutional neural networks via federated learning." The Imaging Science Journal 70, no. 7 (2022): 459-472.

[34] Reddi, Sashank J., Satyen Kale, and Sanjiv Kumar. "On the convergence of adam and beyond." arXiv preprint arXiv:1904.09237 (2019).

[35] Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Andrew Lewis. "Grey wolf optimizer." Advances in engineering software 69 (2014): 46-61.

[36] Amari, Shun-ichi. "Backpropagation and stochastic gradient descent method." Neurocomputing 5, no. 4-5 (1993): 185-196.

[37 ]Chu, Shu-Chuan, Pei-Wei Tsai, and Jeng-Shyang Pan. "Cat swarm optimization." In PRICAI 2006: Trends in Artificial Intelligence: 9th Pacific Rim International Conference on Artificial Intelligence Guilin, China, August 7-11, 2006 Proceedings 9, pp. 854-858. Springer Berlin Heidelberg, 2006.