CHAPTER 3
BRAIN TUMOR DETECTION AND CLASSIFICATION

Image dataset are bifurcated into two dataset; Training dataset and Validation dataset. Generation of Feature Matrix using Training dataset and Feature Matrix of Query image are performed with different processing stages, which are explained as a flowchart. Pre-processing, Segmentation, Feature Extraction and Classification stages are performed with a different techniques mathematical model is developed for proposed method.

3.1 PROCESS FOR THE PROPOSED SYSTEM

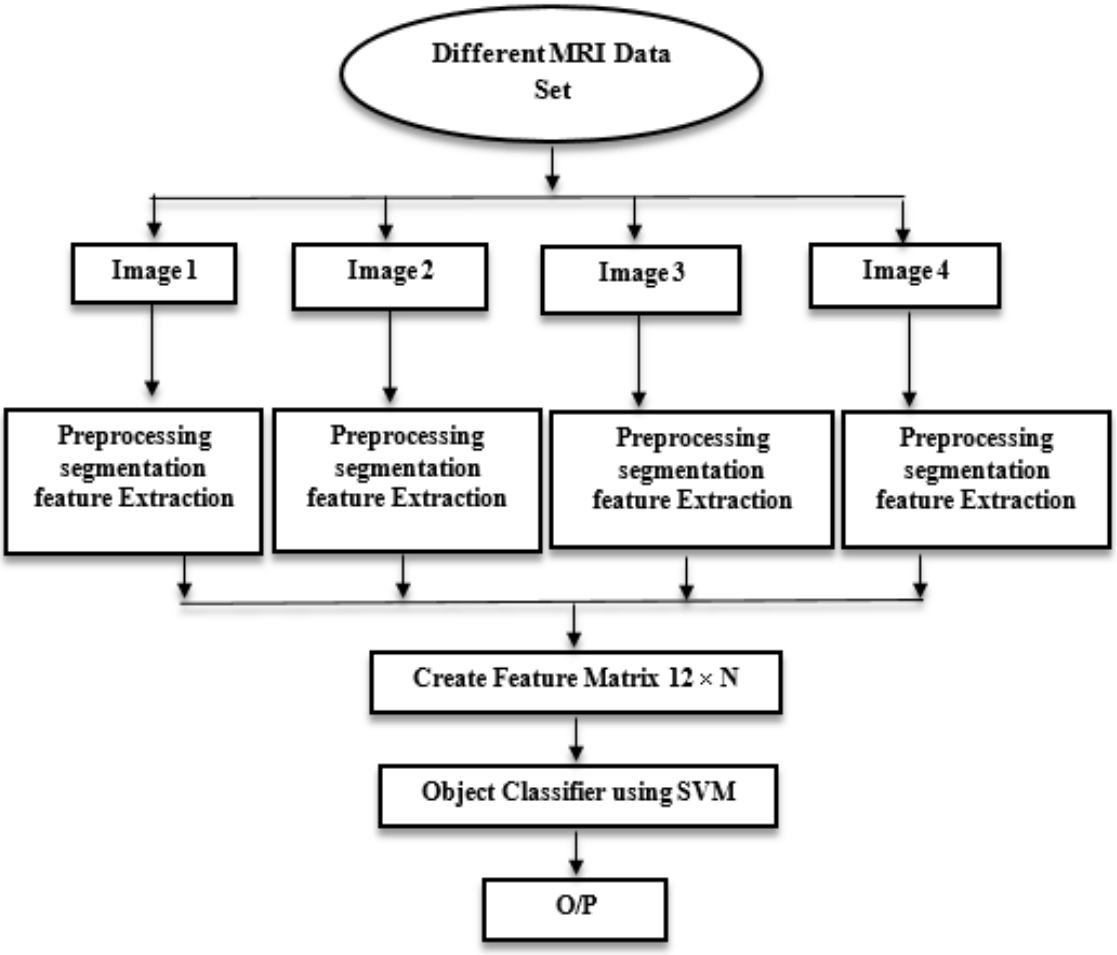

Figure 3.1: Generation of Feature Matrix using Training Dataset

Figure 3.1 shows the training flowchart depicts the stages required to train an object classifier utilising a Support Vector Machine model. Beginning the procedure is the collection of distinct datasets containing images labelled Image 1, Image 2, Image 3, and Image 4.

The initial phase of the training procedure is pre-processing, which prepares the images for further analysis. This includes noise removal. After pre-processing, segmentation is undertaken to distinguish the objects of interest from the background. Following segmentation, pertinent features from the segmented objects are extracted using feature extraction. These characteristics may include shape descriptors, texture patterns, depending on the classification task's specific requirements. Once the features have been extracted, a feature matrix is constructed to symbolise the objects in a structured format that is appropriate for input to the SVM model. Using the feature matrix, the SVM model is then trained to understand the patterns and characteristics of the objects. The SVM is a supervised learning algorithm that seeks to identify the optimal hyperplane for classifying objects into their respective categories. The trained object classifier is the output of the SVM model; it can classify new objects into their respective classifications based on the learned patterns. The training protocol includes pre-processing, segmentation, and feature extraction of various image datasets. The extracted features are then utilised to generate a feature matrix, which is then used to train an SVM model. The result of the training process is an object classifier that can effectively classify new objects based on previously learned patterns.

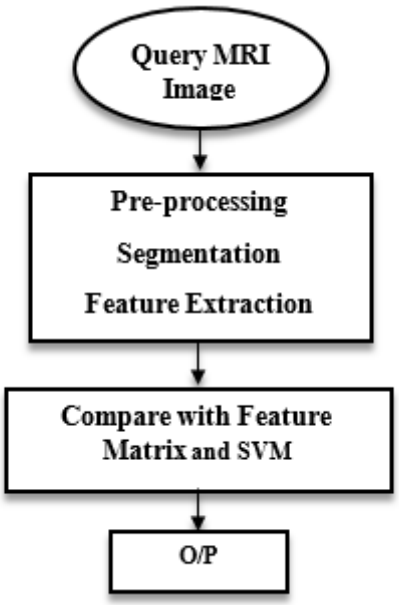

Figure 3.2: Generation of Feature Subspace using Query Image

Figure 3.2 shows the "Flowchart of Query" is a graphical depiction of the stages required to process an image query. It begins with the query image as input and then proceeds through multiple stages of pre-processing, segmentation, feature extraction, comparison, and output generation. The initial step is the pre-processing phase, which prepares the query image for further analysis. This may involve noise reduction in order to improve the image's quality and eliminate unnecessary data. Next, the procedure of image segmentation occurs. In this phase, the query image is divided into meaningful regions or objects based on their visual characteristics. Segmentation assists in isolating distinct image elements, which can facilitate the extraction of relevant features for comparison.

After segmentation, discriminative features are extracted from the segmented regions using feature extraction techniques. These features capture the distinguishing features of the objects in the query image. This may include texture and shape, as well as any other pertinent characteristics. Once the features have been extracted, they are compared to a pre-built Feature Matrix, which functions as a database of features from a set of known images or objects. This matrix contains feature vectors that represent numerous categories of objects or images. The objective is to locate the most similar features or objects in the Feature Matrix that closely match the extracted features of the query image.

SVM algorithm is used to perform this comparison. SVM classifies the query image using the extracted features and the stored knowledge in the Feature Matrix. Each object or image category is assigned a similarity score or probability of match. The output is then generated based on the comparison outcomes. It could be a ranked catalogue of objects or image categories with their respective similarity scores or probabilities. This output identifies the closest parallels to the query image and provides pertinent information or suggestions based on visual similarity. The "Flowchart of Query" summarises the sequential stages involved in processing a query image, including pre-processing, segmentation, feature extraction, comparison with a Feature Matrix using SVM, and output generation based on the results of the comparison. This method facilitates effective and efficient image retrieval and object recognition duties.

3.2 IMAGE DATASET

MRI is a medical imaging technique that uses a strong magnetic field and radio waves to generate detailed images of the body's internal structures. The process needs an MRI scanner, which is a huge tube that contains a massive circular magnet. This magnet creates a magnetic

field that aligns the protons of hydrogen atoms in the body. The protons are then exposed to radio waves, causing the protons to rotate. When the radio waves are turned off, the protons relax and realign themselves, emitting radio waves in the recovery process that can be sensed by the machine to develop an image. In the case of brain imaging, different types of MRI sequences can be utilized to visualize various aspects of brain anatomy and pathology.

Following are descriptions of some common MRI sequences used in brain imaging, along with a sample brain image for each sequence:

T1-WEIGHTED IMAGING:

T1-weighted images provide good anatomical detail and are often used as a baseline reference in brain imaging. In T1-weighted images, cerebrospinal fluid appears dark, while gray matter and white matter have intermediate and bright intensities, respectively.

T2-WEIGHTED IMAGING:

T2-weighted images are sensitive to changes in water content and are useful for detecting abnormalities such as edema, inflammation, or fluid-filled spaces. In T2-weighted images, cerebrospinal fluid appears bright, while gray matter and white matter appear darker.

FLUID-ATTENUATED INVERSION RECOVERY:

FLAIR imaging suppresses the signal from cerebrospinal fluid, allowing better visualization of lesions that may be obscured on T2-weighted images. It is particularly useful in detecting lesions associated with demyelinating diseases, such as multiple sclerosis.

DIFFUSION-WEIGHTED IMAGING:

DWI measures the random motion of water molecules in the brain, providing information about the microstructural integrity of tissues. It is highly sensitive to acute ischemic stroke; as restricted diffusion can indicate areas of reduced blood flow.

GRADIENT-ECHO IMAGING:

Gradient-echo imaging is sensitive to magnetic susceptibility differences and is commonly used to detect hemorrhages, microbleeds, and other iron-related changes in the brain. It provides excellent visualization of blood products.

Each sequence provides unique information about different aspects of brain structure and pathology, aiding in the diagnosis and management of various neurological conditions. It's

important to note that the provided sample images are for illustrative purposes only and do not represent real patient data.

3.3 PRE-PROCESSING OF THE FILTERING ALGORITHMS

In the Pre-processing stage, Median Filter, Wiener Filter, Anisotropic Filter and Non Local Means Filters are described.

3.3.1 MEDIAN FILTER

MSE, RMSE, and PSNR values from the Median Filter and Wiener Filter outputs on brain MRI images were compared in a study. The outcome demonstrates that the Median Filter outperforms the competition. In terms of PSNR, MSE, and RMSE values, Median Filter images have superior pixel quality than Wiener Filter images. To help with proper diagnosis, noise must be removed, and image contrast must be increased. This study is a development of the earlier study [29], yet there are many details that are missing. Therefore, the goal of the current research is to use a hybrid method to preserve the intricate information included in the image.

The windowed hybrid median filter is a nonlinear class windowed filter that effectively reduces impulse noise while maintaining edges. One has better corner-preserving features than the median filter hybrid's simple form. The fundamental principle of the filter is to take the median value obtained by applying the median approach to each element of the signal (picture) many times with a different window shape. The original pixel value and the median of these two medians make up the result. The noise levels were changed from 10% to 90%, and RMSE and PSNR values were determined at every level of noise. When the RMSE and PSNR values of all the filters are compared, it can be inferred that the adaptive median filter outperforms the other filters. Filters are put to the test at various levels and kind of noise. As a result, adaptive median filter has been employed to suppress noise in the technique[29].

Equation of Median Filter as follows:

$$Q(i,j) = \text{median}\{I(s,t)\},\ \text{where}\ (s,t) \in M_{ij}\} \qquad \ldots (3.1)$$

where Q is output image, I is the input image and M_{ij} (window/mask).

The above equation Q (i, j) = median {I(s,t)}, where (s,t) ϵ Mij, represents a process of calculating the pixel values of the output image Q(i,j) based on the median value of a window/mask Mij in the input image I. The input image I is being analyzed with a sliding window/mask Mij of a certain size, which is centered at pixel (i,j) in the output image Q. The

median value of all the pixel intensities within the window/mask is calculated and assigned to the corresponding pixel (i,j) in the output image Q.

The median is a statistical measure that gives the central value of a dataset and is less sensitive to extreme values or outliers than other measures such as mean or mode. By using median instead of mean, the output image is less likely to be affected by outliers in the input image, and the output is smoother and more robust to noise. the equation Q(i,j) = median {I(s,t)}, where (s,t) ϵ Mij, is a common image processing operation known as median filtering, which is often used for noise reduction and image enhancement.

3.3.2 WIENER FILTER

Among the most important steps in increasing the quality of a picture for visual perception is to remove noise. Filtering away undesirable signals is a good way to accomplish successful de-noising. among all the noises, speckle noise provides the highest efficiency for the median, average, and wiener filters. In compared to various filters, the median and wiener filters require the shortest duration to execute. In general, salt and pepper noise produce the highest outcomes for average filter. Due to dual picture collection from clinical modalities and picture transfer from modality to workplace in the primary computing device, scans are frequently influenced by interference. Because this noise frequently degrades the overall clarity of the source pictures, picture noise removal is generally used in clinical picture analysis to speed up diagnosis. In this paper, researchers use the Wiener filter to offer a reasonably simple and effective technique for removing Gaussian disturbances from clinical pictures. Whenever it relates to noise removal or picture denoising, the Wiener filter is the best choice. As element of the picture processing, it examines both the deterioration value and noise. The practical findings show that the Wiener filter is resilient and preserves information. Due of its efficiency and quickness, the Wiener filter is widely used. It is considered simple since it calculates a collection of ideal filter weights that decrease the noise level of an incoming information using a system of linear equations. The accuracy, accessibility, and durability of clinical picture de-noising are demonstrated in the following picture. Because Wiener filtering has the benefits of being simple to calculate and having a great noise impact, it is generally utilized. Several effective de-noising techniques are centered on the Wiener filtration concept, which aims to recover the actual picture while achieving the lowest mean errors[29].

Consider degraded input image $x(m, n)$, the Discrete Fourier Transform of $x(m, n)$ is $X(u, v)$. The estimated value of the input image is calculated by multiplying result of $X(u, v)$ with the Wiener filter $W(u, v)$:

$$\hat{S}(u, v) = W(u, v)\, X(u, v) \qquad \ldots (3.2)$$

The Wiener filter is defined as,

$$W(u, v) = \frac{H^*(u,v)}{|H(u,v)|^2 + \frac{P_n\ (u,v)}{P_S\ (u,v)}} \qquad \ldots (3.3)$$

$H(u, v)$ = Fourier transform of the point spread function (PSF)

$P(u, v)$ = Power spectrum of the signal process, generated by the application of the Fourier transform of the signal autocorrelation

$P_n\ (u, v)$ = Power spectrum of the Gaussian noise process, generated by the application of the Fourier transform of the noise autocorrelation

The inverse Fourier Transform of $\hat{S}(u, v)$, got the output (restored image). The Equation (3.2) represents a method for restoring a degraded image. The degraded input image is represented by $x(m,\ n)$ and its Discrete Fourier Transform is represented by $X(u, v)$. The estimated value of the input image is obtained by multiplying $X(u, v)$with the Wiener filter $W(u, v)$. The Wiener filter is calculated using the Fourier transform of the point spread function (PSF), the power spectrum of the signal process, and the power spectrum of the Gaussian noise process. The PSF describes the blurring effect of the imaging system on the input image. The power spectrum of the signal process represents the frequency distribution of the signal in the image, and the power spectrum of the Gaussian noise process represents the frequency distribution of the noise in the image. The Wiener filter, the degraded input image is restored, and the output image is obtained by taking the inverse Fourier transform of the product of the Wiener filter and the Discrete Fourier Transform of the degraded input image. This method is commonly used in image processing to restore degraded images that have been corrupted by noise or other forms of distortion.

3.3.3 ANISOTROPIC FILTER

A pre-processing is used in an edge-preserving diffusion technique that has been presented. Pre-processing and diffusion are the two key steps of the suggested methodology. The pre-processing pixel is replaced by the test image pixel that has been pre-filtered through a

Gaussian filter if the difference value between the pixels of the corrupted picture and the pixels of the smoothed image using a Gaussian filter is greater than the threshold the pixel is displayed unchanged if the difference is below the threshold T. To maintain the edge boundaries and detail information during the diffusion process, a semi-adaptive threshold function in diffusion coefficient is used. This technique improves the capacity to remove noise when the noise level is high and gets around the problem of the sharp edges blurring in photos. It is clear that the suggested strategy outperforms more traditional methods in terms of effectiveness. According to the testing results, the suggested approach increases SSIM by 5% and PSNR by 30%. The restored images show that this strategy can successfully enhanced edge retention and noise removal in the context of anisotropic diffusion filtering [117].

|∇I|| represents the magnitude of gradient, $g||\nabla I||$ indicates the edge stopping and σ is scale value, div is the divergence operator, ∇ and Δ are the gradient and Laplace operator. The notations like N, E, W, S describes North, East, West and South & ∇is neighbour differences.

$$I_t = div(c(x,y,t).\nabla \mathrm{I}) \qquad \ldots (3.4)$$

$$c(x,y,t) = g(\|\nabla \mathrm{I}(\mathrm{x},\mathrm{y},\mathrm{t})\|) \qquad \ldots (3.5)$$

Where, div is the divergence operator, ∇ is the gradient. The notations like N, E, W, S describes North, East, West and South & ∇is neighbour differences.

The Equation (3.4) represents an image processing operation for computing the temporal derivative of an image. I is the input image and ∇I represents its gradient. The symbol div() represents the divergence operator, which is applied to the product of a scalar function c(x,y,t) and the gradient of the image ∇I. The function c(x,y,t) is defined as g(‖∇I(x,y,t)‖), where g() is a scalar function that computes the magnitude of the gradient ∇I at each pixel location (x,y,t).

$$I_{i,j}^{t+1} = I_{i.j}^{t} + \lambda\,[c_N.\nabla_N I + c_S\nabla_S I + c_E\nabla_E I + c_W\nabla_W I]_{i.j}^{t} \qquad \ldots (3.6)$$

The symbol ∇ (not to be confused with ∇, which we use for the gradient operator) indicates nearest-neighbor differences:

$$\nabla_N I_{i,j} = I_{i-1,j} - I_{i,j} \qquad \ldots (3.7)$$

$$\nabla_S I_{i,j} = I_{i+1,j} - I_{i,j} \qquad \ldots (3.8)$$

$$\nabla_E I_{i,j} = I_{i,j+1} - I_{i,j} \qquad \ldots (3.9)$$

$$\nabla_W I_{i,j} = I_{i,j-1} - I_{i,j} \quad \ldots (3.10)$$

$$c_{Si.j}^t = g(\nabla_N I_{i.j}^t) \quad \ldots (3.11)$$

$$c_{Ni.j}^t = g(\nabla_S I_{i.j}^t) \quad \ldots (3.12)$$

$$c_{Ei.j}^t = g(\nabla_E I_{i.j}^t) \quad \ldots (3.13)$$

$$c_{Wi.j}^t = g(\nabla_W I_{i.j}^t) \quad \ldots (3.14)$$

The symbols ∇_N, ∇_S, ∇_E, and ∇_W represent the north, south, east, and west gradient components of the image I, respectively. These gradient components are computed by subtracting the intensity values of neighbouring pixels in the corresponding direction from the intensity value of the central pixel (i,j). The functions $c_{Si.j}^t$, $c_{Ni.j}^t$, $c_{Ei.j}^t$ *and* $c_{Wi.j}^t$ represent the weights assigned to the north, south, east, and west gradient components, respectively. These weights are computed using the scalar function g (), which takes as input the magnitude of the corresponding gradient component at the pixel location (i,j) and returns a scalar value.

For High contrast edges over low contrast ones,

Leads to Gaussian blurring different functions were used for $g(\nabla I)$ giving perceptually similar results. The images in this paper were obtained using g,

$$g(\nabla I) = e^{\left(\frac{\|\nabla I\|}{K}\right)^2} \quad \ldots (3.15)$$

For wide regions over smaller ones,

$$g(\nabla I) = \frac{1}{1+\left(\frac{\|\nabla I\|}{K}\right)^2} \quad \ldots (3.16)$$

K is the constant and $0 \leq \lambda \leq \frac{1}{4}$ for the numerical scheme to be stable, N, S, E, W are the mnemonic subscripts for North, South, East and West. The scale-spaces generated by these two functions are different: the first privileges high-contrast edges over low-contrast ones, the second privileges wide regions over smaller ones described by Canny[117].: a histogram of the absolute values of the gradient throughout the image was computed. The constant K was fixed either by hand at some fixed value or using the "noise estimator" and K was set equal to the 90% value of its integral at every iteration.

3.3.4 NON LOCAL MEANS FILTER

On brain MRI images, a study tries to identify non-linear noise reduction methods and noise models. The Non-Local Means filtration method was employed. The outcome demonstrates that Non-Local Mean Filters offer improved SSIM, PSNR, and MSE values for Gaussian denomination MRI pictures, however the mean application execution time for NLMF is too long. The goal of future development on NLMF is to speed up calculation. A non-local maximum likelihood method utilizing the Kolmogorov-Smirnov similarity test has been reported [111]. They took into account symmetric Rician noise that was roughly distributed using a Gaussian distribution with a mean and standard deviation of zero.

In a distinct noisy picture, n = {n(i) | i ∈ I}, the projected result $NL(n)(x_i)$ is quantified as a weighted mean of entire pixel of scan,

$$NL(n)(x_i) = \sum_{j \in I} q(x_i, x_j)\, n(x_j) \qquad \ldots (3.17)$$

In the Equation's (3.17), x_i and x_j are variables that denote set or space elements or coordinates. The functions $n(x_i)$ and $n(x_j)$ assign a value to each element x_i and x_j, respectively. $q(x_i, x_j)$ is a function that calculates the weight or influence of the relationship between the elements x_i and x_j. The precise form of this function depends on the current situation or context. I is a collection of indices or elements j over which the summation is conducted. The symbol "∈" signifies that j is a member of the set I. The value of $NL(n)$at a specific point x_i is determined by adding the product of the weight $q(x_i, x_j)$ and the value of the function $n(x_j)$ for each element x_j in the set I. This suggests that the value at x_i is dependent on the values at other sites x_j, as weighted by the influence function q. The Equation (3.17) represents an application of a linear operator or transformation to the function $n(x)$, yielding the new function $NL(n)(x_i)$. This equation's precise interpretation and meaning depend on the context in which it is used, such as physics, signal processing, or network analysis.

In the Equation (3.18), $q(x_i, x_j)$ indicates the weight assigned to $n(x_j)$ in attempt to recreate the pixel x_i and computed as:

$$q(x_i, x_j) = \frac{1}{Z_i}\, e^{\left(- \frac{\left\| n(I)_i - n(I)_j \right\|^2_{2,a}}{h^2} \right)} \qquad \ldots (3.18)$$

The Equation (3.18) represents a function of weight or influence between the constituents x_i and x_j. x_i and x_j are variables that denote set or space elements or coordinates. The functions $n(I)_j$ and $n(I)_j$ assign a value to each element x_i and x_j, respectively. The subscript "I" indicates that these functions are set or context-dependent I. $\left\|n(I)_i - n(I)_j\right\|_{2,\,a}^2$ is the Euclidean distance or norm between vectors $n(I)_i$and $n(I)_j$. The subscript "2" indicates the Euclidean norm, and "a" represents a distance calculation parameter. h is a parameter that affects the influence function's spread or breadth or filtration level that regulates the degradation of the logarithmic value and hence the degradation of the values as a measure of Euclidean ranges. It determines how much neighbouring elements influence one another. e represents the natural logarithm base, which is approximately equal to 2.7182. Z_i is a unique normalization factor for x_i. It assures that the sum of the weights for a fixed $x_i$is 1. The Equation (3.18) computes the weight and influence between elements x_i and x_j based on their Euclidean distance in the $n(I)$ space. The influence is modelled using a Gaussian distribution in which the distance is multiplied by h^2 and then divided by its exponent. This assures that elements that are closest to xi have a greater weight than those that are further away. The normalization factor Zi assures that the sum of the weights for a constant x_i is 1. The precise interpretation and significance of this equation depend on its application context. It is frequently encountered in disciplines such as machine learning, where it is utilized for tasks such as density estimation, clustering, and defining similarity measures between data points. The choice of parameters and the exact form of $n(I)$ would be unique to the problem at hand. The value of the filtering parameter writes h = k σ. The value of k decreases as the size of the patch increases. For larger sizes, the distance of two pure noise patches concentrates more around $2\sigma^2$ and therefore a smaller value of k can be used for filtering.

3.4 SEGMENTATION USING HYBRID METHOD

In the Segmentation Stage, Multi-thresholding Cuckoo Search Algorithm using different objective functions; like, Otsu, Kapur Entropy, Tsallis Etropy and Proposed method are described. Brain tumor localization and segmentation from MRI are hard and important tasks for several applications in the field of medical analysis. As each brain imaging modality gives unique and key details related to each part of the tumor, many recent approaches used four modalities T1, T2, and FLAIR. Although many of them obtained a promising segmentation result on the BRATS 2018 dataset, they suffer from a complex structure that needs more time to train and test. The manual segmentation and analysis of structural MRI images of brain

tumors is an arduous and time-consuming task which, thus far, can only be accomplished by professional neuroradiologists. Therefore, an automatic and robust brain tumor segmentation will have a significant impact on brain tumor diagnosis and treatment.

3.4.1 CUCKOO SEARCH ALGORITHM

The Cuckoo Search Algorithm is a metaheuristic optimization algorithm that imitates the behaviour of cuckoo birds in laying their eggs in other birds' nests. It is used to solve various optimization problems by searching for the best solution through a process of generating new solutions and eliminating poor ones. The algorithm uses a combination of randomization and local search to efficiently explore the search space and converge towards the optimal solution. The key feature of the algorithm is the use of a random walk called the Levy Flight, which helps the algorithm to efficiently explore the search space.

The CS optimization algorithm is generally based on the following three principles[113]:

1. Each cuckoo bird lays one egg at a time and randomly places its egg in a host bird's nest.
2. The best nests containing high-quality eggs are carried over to the next generations.
3. The number of available host nests is fixed. The host bird discovers foreign eggs with a probability pα, and the range of pα is from 0 to 1. Note that the best nests are selected for further calculations.

The host bird can either throw the egg away or abandon the nest, and build a completely new nest. For simplicity, this last assumption can be approximated by the fraction pa of the n nests are replaced by new nests (with new random solutions). For a maximization problem, the quality or fitness of a solution can simply be proportional to the value of the objective function. Other forms of fitness can be defined in a similar way to the fitness function in genetic algorithms. For simplicity, we can use the following simple representations that each egg in a nest represents a solution, and a cuckoo egg represent a new solution, the aim is to use the new and potentially better solutions (cuckoos) to replace a not-so-good solution in the nests. Of course, this algorithm can be extended to the more complicated case where each nest has multiple eggs representing a set of solutions. For this present work, we will use the simplest approach where each nest has only a single egg

Based on these three principles, the CS process can be summarized as follows: While generating new solution x_i^{t+1} for cuckoo i, a Lévy light is performed[21]:

$$x_i^{t+1} = x_i^t + \alpha_0 (x_i^t - x_{best}) \oplus Levy(\lambda) \quad \ldots (3.19)$$

Where α_0 is the step scaling factor, $\alpha_0 > 0$, and x_{best} represents the current optimal solution (Objective Function), $\oplus$ Element-wise multiplication. The above equation is essentially the stochastic equation for random walk. In general, a random walk is a Markov chain whose next status/location only depends on the current location (the first term in the above equation) and the transition probability (the second term). The product $\oplus$ means entry wise multiplications. This entry wise product is similar to those used in PSO, but here the random walk via L´evy flight is more efficient in exploring the search space as its step length is much longer in the long run. Levy flights are drawn from a Levy distribution, which can be defined by:

$$Levy(\lambda) \sim u = t^{-\lambda}, (1 < \lambda \leq 3) \quad \ldots (3.20)$$

which has an infinite variance with an infinite mean. Here the steps essentially form a random walk process with a power law step-length distribution with a heavy tail. Some of the new solutions should be generated by L´evy walk around the best solution obtained so far, this will speed up the local search. However, a substantial fraction of the new solutions should be generated by far field randomization and whose locations should be far enough from the current best solution, this will make sure the system will not be trapped in a local optimum. From a quick look, it seems that there is some similarity between CS and hill-climbing in combination with some large scale randomization. But there are some significant differences. Firstly, CS is a population-based algorithm, in a way similar to GA and PSO, but it uses some sort of elitism and/or selection similar to that used in harmony search. Secondly, the randomization is more efficient as the step length is heavy tailed, and any large step is possible. Thirdly, the number of parameters to be tuned is less than GA and PSO, and thus it is potentially more generic to adapt to a wider class of optimization problems. In addition, each nest can represent a set of solutions, CS can thus be extended to the type of meta-population algorithm.

3.4.2 LÉVY FLIGHT MODELLING

Lévy flight modelling is a mathematical framework used to describe a specific type of random motion or behavior known as Lévy flights[21]. Lévy flights are characterized by long jumps or displacements between consecutive positions, with the distances of these jumps following a heavy-tailed probability distribution. There are two steps in the implementation of random numbers with Levy flight. The first stage is choosing the random flying direction, and the second stage considers creating the steps that will follow the

selected Levy distribution. A uniform distribution is used to choose the random direction. The following is a definition of the Levy step size: $Levy(\beta) = u/|v|^{\frac{1}{\beta}}$ represents the probability density function known as the Lévy distribution. Let's break the equation down into its component parts: The shape of the distribution is determined by the parameter. Typically, the value is positive. The distribution is scaled by the constant u. |v| signifies v's absolute value, which is a random variable. The Lévy distribution characterizes the probability distribution of random variables with heavy tails, in which the tails diminish more slowly than in a Gaussian (normal) distribution. The Lévy distribution is characterized by the parameter, which regulates the decay rate of the tails. When is less than 1, the distribution has a heavier tail, which indicates a greater likelihood of large values. As approaches 0, the distribution's tails become heavier and it becomes more prone to extreme values. As approaches positive infinity, the distribution approaches a Gaussian distribution.

$$Levy(\beta) = \frac{u}{|v|^{\frac{1}{\beta}}} \qquad \text{where, } \lambda - 1 = \beta \qquad \ldots (3.21)$$

where u and v are drawn from normal distributions. This implies that: The equation describes the presumed normal distributions of the random variables u and v. u is a random variable whose distribution is normal with mean 0 and variance σ_u^2. v is a random variable with a normal distribution that has a mean of 0 and a variance of σ_v^2, which is equal to 1.

$$u \sim N(0, \sigma_u^2),\ v \sim N(0, \sigma_v^2),\ \sigma_u = \left\{ \frac{\Gamma(1+\beta)\sin(\frac{\pi\beta}{2})}{\Gamma\left(\frac{1+\beta}{2}\right)\beta\, 2^{\frac{(\beta-1)}{2}}} \right\}^{\frac{1}{2}}, \sigma_v = 1 \qquad \ldots (3.22)$$

Where Γ correspond standard gamma function. The gamma function extends the factorial function to values that are not integers. The gamma function is applied to (1+ β) and ((1+ β)/2) in this equation. u's variance is computed using the sine function and various other mathematical operations.

3.4.3 FLOW CHART OF THE CUCKOO SEARCH ALGORITHM

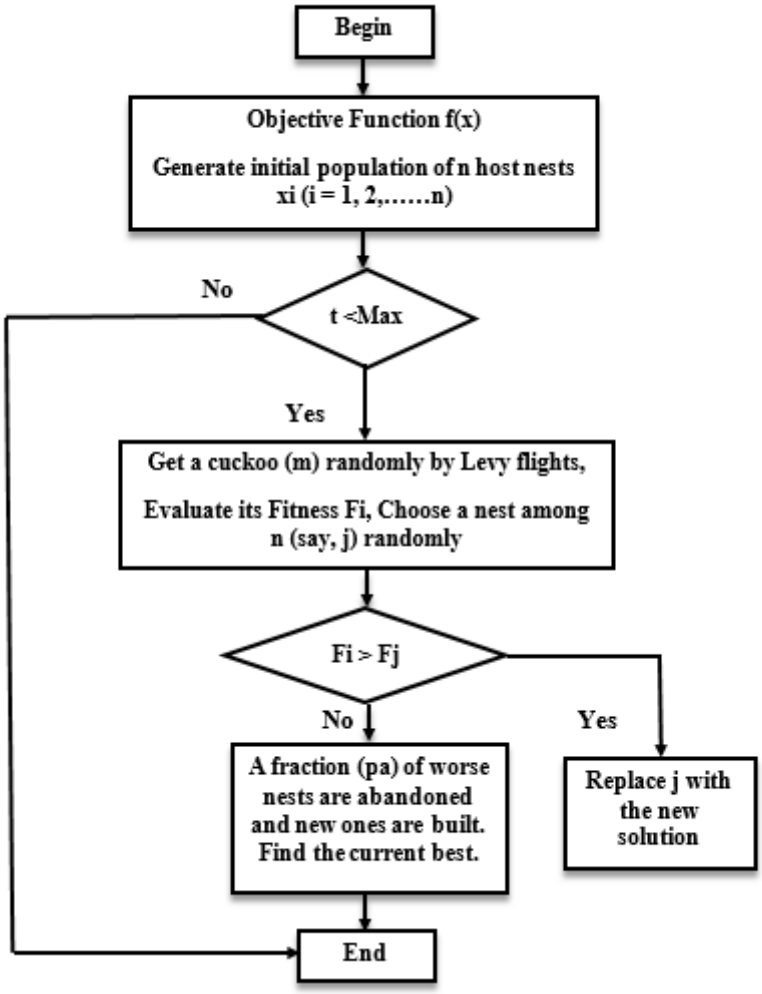

Figure 3.3: Flowchart of Cuckoo Search Algorithm [21]

The supplied description describes "Cuckoo Search," a metaheuristic optimisation algorithm inspired by cuckoo avian behaviour. Figure 3.3 shows the flowchart of the CSA. The objective of the algorithm is to locate the optimal solution for a given objective function, denoted by f(x). The procedure begins with the generation of an initial population of host nests, denoted by xi (i = 1, 2,...,n), where n is the number of nests. These structures serve as prospective optimisation problem solutions. The algorithm then enters a loop that proceeds until a specified condition, denoted by t Max, is satisfied. A cuckoo (m) is chosen at random within each iteration using Levy flights, which simulate the flying behaviour of cuckoos. The objective function is used to evaluate the fitness of the selected cuckoo, Fi.

Next, a nursery (j) from the population is selected at random. If the fitness of the selected cuckoo (Fi) is greater than the fitness of the selected nest (Fj), a superior solution has been

identified. In this instance, a portion (pa) of the worst nests are abandoned and new nests are constructed in their place. The goal of the algorithm is to enhance the quality of the colonies over time. Throughout the iterations, the algorithm maintains a record of the finest solution discovered to date. If a new solution replaces one of the nests (j), the algorithm updates the population by substituting the new solution for the old solution. This procedure is repeated until the loop condition is met, at which point the algorithm terminates. The ultimate objective of this method is to discover the optimal solution for the objective function by iteratively enhancing the quality of the nests through cuckoo selection, nest replacement, and abandonment of poorer nests.

The description of the CSA flowchart is: Define the objective function that measures the quality or suitability of a solution, f(x). This function evaluates the performance of a solution in tackling an optimization problem. Create an initial population of n nests xi (i=1,2,.....n): Create a nest-representative initial population of potential solutions. Each nest is designated with the value 'xi', where 'n' represents the number of nesting. t < Max Establish a termination condition to regulate the utmost number of iterations or the algorithm's halting criteria. 't' represents the current iteration, whereas 'Max' represents the utmost number of permitted iterations. Receive a random cuckoo (m) on Levy flights: Randomly select a cuckoo using the Levy flight technique. Levy flights are a form of random walk that imitates the behavior of cuckoo birds when foraging. Evaluate its Fitness Fi: Evaluate the fitness (quality) of the cuckoo solution using the objective function. This phase evaluates the performance of the cuckoo in tackling the optimization problem. Choose a brood at random from n (let's say j): Select at random one brood (answer) from the initial population. Compare the fitness of the cuckoo solution (Fi) to that of the randomly selected nest (Fj). If the fitness of the cuckoo is greater than that of the brood, proceed to the next step. Otherwise, a percentage (pa) of poorer colonies are abandoned and replaced with new ones. Find the prevailing finest. Substitute the new solution for j: If the fitness of the cuckoo is greater than that of the nest, a portion (pa) of the worst nests are abandoned, or discarded. The abandoned colonies are then replaced with new ones. In addition, the present optimal solution is identified and substituted for the arbitrarily selected nest (j). Determine if the termination condition has been met. End the algorithm if the utmost number of iterations is reached or if the halting criteria are met. Otherwise, advance to the next iteration by incrementing 't' and return to choose a brood at random from n (let's say j).

3.4.4 ADVANTAGES OF THE CUCKOO SEARCH ALGORITHM

The Cuckoo Search Algorithm is a metaheuristic optimization algorithm inspired by the behavior of cuckoo birds in laying eggs. While CSA has shown promise in solving various optimization problems, it has several advantages.

Convergence Speed: One of the main challenges of CSA is its convergence speed. In some cases, the algorithm may require a large number of iterations to converge to an optimal solution. This slow convergence can be problematic, especially when dealing with time-critical applications or problems with a large search space.

Exploration and Exploitation Balance: CSA needs to strike a balance between exploration and exploitation of the search space. Exploration involves exploring new regions to avoid getting trapped in local optima, while exploitation focuses on exploiting promising regions to refine the solutions. Finding the right balance between these two aspects can be challenging, as an excessive focus on exploration may lead to slow convergence, while an excessive focus on exploitation may result in premature convergence to suboptimal solutions.

Parameter Selection: CSA relies on various parameters that influence its behavior, such as the number of cuckoos, the probability of egg laying, and the random walk step size. Selecting appropriate parameter values for a given problem can be non-trivial and often requires extensive experimentation and tuning. Inadequate parameter selection may lead to poor performance or even failure of the algorithm to find optimal solutions.

Handling Constraints: Many real-world optimization problems involve constraints that need to be satisfied. Incorporating constraints into CSA can be challenging, as the algorithm's natural behavior may not guarantee constraint satisfaction. Ensuring that the generated solutions adhere to the problem constraints adds complexity to the algorithm and may require additional modifications or penalty functions.

Scalability: The scalability of CSA is another challenge, particularly when dealing with large-scale optimization problems. As the problem size increases, the algorithm's performance can deteriorate due to increased computational complexity and a higher risk of

getting trapped in local optima. Efficient strategies and techniques need to be developed to enhance the scalability of CSA for complex real-world problems.

3.4.5 OBJECTIVE FUNCTIONS

Objective functions, also known as fitness functions or cost functions, play a crucial role in optimization problems. They define the goal or objective that an optimization algorithm seeks to achieve. The objective function quantifies the quality or desirability of a potential solution in relation to the problem at hand.

3.4.5.1 OTSU

Otsu's method is a popular multi-thresholding optimization method that aims to minimize the intra-class variance between pixels within the same class (i.e., brain tissue). The method calculates the optimal thresholds by iteratively selecting the threshold that maximizes the between-class variance. The threshold values are chosen such that they minimize the sum of the intra-class variances of the segmented regions. Otsu's method is simple, fast, and does not require prior knowledge of the image statistics. The Otsu algorithm's main goal is to optimise between-class variance by choosing an appropriate threshold value, p_i is the probability of the pixel intensity value to be i where i ranges from 0 to 255 and L is the total number of distinct intensity levels in the gray scale image.

$$x_{best} = Arg\ max\{\sigma_B^2(t)\} \quad \ldots (3.23)$$

$$\sigma_B^2 = \sigma_0^2 + \sigma_1^2 + \sigma_2^2 + \cdots \sigma_r^2 \quad \ldots (3.24)$$

$$\sigma_0^2 = \omega_0(\mu_0 - \mu_T)^2 \quad \ldots (3.25)$$

$$\sigma_1^2 = \omega_1(\mu_1 - \mu_T)^2 \quad \ldots (3.26)$$

$$\sigma_r^2 = \omega_r(\mu_r - \mu_T)^2 \quad \ldots (3.27)$$

$$\omega_0 = \text{weight} = \frac{\sum_{i=0}^{t_1-1} p_i}{\sum_{i=0}^{L-1} p_i} \quad \ldots (3.28)$$

$$\omega_1 = \text{weight} = \frac{\sum_{i=t_1}^{t_2-1} p_i}{\sum_{i=0}^{L-1} p_i} \quad \ldots (3.29)$$

$$\omega_r = \text{weight} = \frac{\sum_{i=t_r}^{L-1} p_i}{\sum_{i=0}^{L-1} p_i} \quad \ldots (3.30)$$

$$\mu_0 = \text{mean} = \frac{\sum_{i=0}^{t_1-1} i.p_i}{\sum_{i=0}^{t_1-1} p_i} \quad \ldots (3.31)$$

$$\mu_1 = \text{mean} = \frac{\sum_{i=t_1}^{t_2-1} i.p_i}{\sum_{i=t_1}^{t_2-1} p_i} \quad \ldots (3.32)$$

$$\mu_r = \text{mean} = \frac{\sum_{i=t_r}^{L-1} i.p_i}{\sum_{i=t_r}^{L-1} p_i} \quad \ldots (3.33)$$

$$\mu_T = \text{mean} = \frac{\sum_{i=0}^{L-1} i.p_i}{\sum_{i=0}^{L-1} p_i} \quad \ldots (3.34)$$

3.4.5.2 KAPUR ENTROPY

Kapur's method is another multithresholding optimization method that is based on the principle of maximizing the entropy of the segmented image. The method calculates the optimal thresholds by minimizing the conditional entropy of the segmented regions, given the gray-level intensities of the pixels. The method is computationally efficient and can be used to segment images with a large number of intensity levels. The primary objective of Kapur's entropy method is to identify the probability distribution that maximizes entropy (uncertainty) while satisfying a given set of constraints or information. When the available data are insufficient to determine the exact probability distribution, but can provide certain constraints or partial information about the distribution, the method is frequently employed.

$$H_k = -\sum_{i=0}^{L-1} p_i \ ln\ p_i \quad \ldots (3.35)$$

In the Equation (3.35), H_k represents the k-th entropy, which is a measure of the uncertainty or information content of a probability distribution. Entropy quantifies the average quantity of information necessary to describe or foretell a random event. Depending on the formulation or requirements, multiple entropy metrics, such as Shannon entropy (k=1) may be used. This is the probability of the i-th event or outcome in a probability distribution. The probability p_i should be nonnegative and add up to 1. i = 0 to L-1, where L is the total number of potential outcomes or events. This symbol represents the natural logarithm function, which is used to compute the logarithm of the probability p_i. Typically, the natural logarithm is represented as ln(x) or $\log_e(x)$, where e is the natural logarithm's base, which is approximately equal to 2.71828. $\sum_{i=0}^{L-1}$ represents the summation operator, which adds the

individual elements for i between 0 and L-1. The sum of all conceivable events or outcomes in the probability distribution is computed.

Let p_i is the probability of the pixel intensity value to be i where i ranges from 0 to 255 and L is the total number of distinct intensity levels in the gray scale image.

$$x_{best} = Arg\ max\{H_T(t)\} \quad \ldots (3.36)$$

The Equation (3.36) represents the selection of the optimal solution x_{best}based on the function's maximal value. Let's break down the equation's components: x_{best} : This is the variable that will contain the optimal solution. It is the optimal or most desirable candidate among a group of alternatives. This abbreviation stands for "argument of the maximum." It indicates the argument or input value that maximizes the performance of a given function. $H_T(t)$: This represents the set of solutions from which the optimal solution is chosen. The function $H_T(t)$ is a performance metric or objective function that evaluates the quality or suitability of each solution t. The precise definition and calculation of $H_T(t)$ are context- and problem-dependent. The related entropies are as follows when the histogram is divided into different sections ($H_0, H_1, \ldots. H_r$) by the threshold t. $H_0, H_1, \ldots. H_r$ added together form the objective function $H_T(t)$,

$$H_T = H_0 + H_1 + H_2 \ldots + H_r \quad \ldots (3.37)$$

$$H_0 = -\sum_{i=0}^{t_1-1} \frac{p_i}{w(0)} ln \frac{p_i}{w(0)} \quad \ldots (3.38) \qquad w(0) = \sum_{i=0}^{t_1-1} p_i \quad \ldots (3.39)$$

$$H_1 = -\sum_{i=t_1}^{t_2-1} \frac{p_i}{w(1)} ln \frac{p_i}{w(1)} \quad \ldots (3.40) \qquad w(1) = \sum_{i=t_1}^{t_2-1} p_i \quad \ldots (3.41)$$

$$H_2 = -\sum_{i=t_2}^{t_3-1} \frac{p_i}{w(1)} ln \frac{p_i}{w(2)} \quad \ldots (3.42) \qquad w(2) = \sum_{i=t_2}^{t_3-1} p_i \quad \ldots (3.43)$$

$$H_r = -\sum_{i=t_r}^{L-1} \frac{p_i}{w(r)} ln \frac{p_i}{w(r)} \quad \ldots (3.44) \qquad w(r) = \sum_{i=t_m}^{L-1} p_i \quad \ldots (3.45)$$

3.4.5.3 TSALLIS ENTROPY

The Tsallis entropy-based method is a multi-thresholding optimization method that is based on the principle of maximizing the generalized entropy of the segmented image. The method uses a non-extensive entropy measure known as the Tsallis entropy to calculate the optimal

thresholds. The method is robust to noise and can be used to segment images with non-Gaussian intensity distributions. In image processing, Shannon entropy exhibits the property of extensivity and is additive. Entropy was first introduced in thermodynamics to characterise physical systems with a great deal of microstate diversity. Furthermore, it is assumed that the system's microstates are independent of one another for entropy's extensibility. The extensibility, however, might no longer be valid for some systems with long-distance interactions, long-term memory, and fractal-like structures. To explain these systems, Tsallis introduces a non-extensive entropy[92] that is represented as: Multifractal concepts and structures are quickly acquiring importance in many active areas (e.g.,non-linear dynamical sys terns, growth models, commensurate /incommensurate structures). This is due to their utility as well as to their elegance. Within this framework, the quantity which is normally scaled is p_i^q, where p_i is the probability associated to an event and q any real number. We postulate for the entropy

$$S_{TS} = \frac{1-\sum_{i=1}^{L} p_i^q}{q-1} \qquad \ldots (3.46)$$

In equation (3.46), q is a real value that indicates the system's nonextensivity. Shannon entropy replaces Tsallis entropy in the q → 1 limit, restoring the system's extensibility. The information theory was additionally clarified by the non-extensive generalisation of entropy. Tsallis entropy in image segmentation exhibits potential superiority and adaptability in a broader scope of image class[120]. Concavity: Let us extend here a property already mentioned, namely that q > 0 (q < 0) implies that the extremum of S is a maximum (minimum). The ideal threshold x_{best} is produced by maximizing S_T, and it is denoted by:

$$x_{best} = Arg\ max\{S_T(t)\} \qquad \ldots (3.47)$$

In the Equation (3.47), x_{best} denotes the variable we wish to determine, which will be the argument that maximizes the expression enclosed by curly brackets. Within the brackets, $S_T(t)$ represents a function that is evaluated at time T and is dependent on the variable t. This function $S_T(t)$ returns a score or quantity corresponding to the input value t at time T.

To determine x_{best}, we evaluate the function $S_T(t)$ for various values of t and determine the t value that produces the highest score. The $Arg\ max$ notation identifies the argument that

maximizes the expression contained within the curly brackets. The total entropy of the image is written as follows:

$$S_T = S_0 + S_1 + S_2 \ldots + S_r + (1-q).(S_0.S_1.S_2 \ldots .S_r) \quad \ldots (3.48)$$

Equation (3.48) is a mathematical expression that calculates the value of S_T, which is the sum of multiple terms. $(S_0.S_1.S_2 \ldots .S_r)$ represent individual variables in the expression. Each term contributes to the S_T total. The terms $(S_0.S_1.S_2 \ldots .S_r)$ are added together, signifying their addition. The ellipsis (...) indicates that the series may continue beyond S_r. Multiply (1-q) by the sum of all terms $(S_0.S_1.S_2 \ldots .S_r)$. The symbol (.) represents multiplication, and the expression S_0 is the product of zero. $(S_0.S_1.S_2 \ldots .S_r)$ represents the sum of these elements. Lastly, the complete expression (1-q). The expression $(S_0.S_1.S_2 \ldots .S_r)$ is added to the sum of the terms $(S_0.S_1.S_2 \ldots .S_r)$. To calculate S_T, one must first determine the value of each individual term $(S_0.S_1.S_2 \ldots .S_r)$. Then, you would determine S_0, the product of these variables. $(S_1.S_2 \ldots .S_r)$. Multiply the product by (1-q) next. Finally, the value of S_T is obtained by adding the sum of the individual terms to the product of the multiplication. In the Equation (3.48) third part on the right side displays the pseudo-additivity of Tsallis entropy.

$$S_0 = \frac{1-\sum_{i=0}^{t_1-1}\left(\frac{p_i}{w(0)}\right)^q}{q-1} \quad \ldots (3.49) \qquad w(0) = \sum_{i=0}^{t_1-1} p_i \quad \ldots (3.50)$$

$$S_1 = \frac{1-\sum_{i=t_1}^{t_2-1}\left(\frac{p_i}{w(1)}\right)^q}{q-1} \quad \ldots (3.51) \qquad w(1) = \sum_{i=t_1}^{t_2-1} p_i \quad \ldots (3.52)$$

$$S_r = \frac{1-\sum_{i=t_r}^{L-1}\left(\frac{p_i}{w(r)}\right)^q}{q-1} \quad \ldots (3.53) \qquad w(r) = \sum_{i=t_m}^{L-1} p_i \quad \ldots (3.54)$$

Where, $w(0), w(1), \ldots, w(r)$ are the cumulative probability of the different class in the Tsallis entropy algorithm. $S_0, S_1, \ldots . S_r$ are the entropy of the different class. The optimal result of Equation (3.47) depends on the nonextensive parameter q, which describes the strength of internal correlation of the image. In other words, for an arbitrary two pixels in the image, their gray-level values may have long-range correlations. More specifically, for an image containing several objects, the pixels of objects will exhibit similar gray-level values, even though they are not adjacent to each other. It is possible to measure this kind of long-range correlation by nonextensive entropy[120] and this idea inspired a new algorithm that is discussed below. Since the parameter q is an additional index that can tune

the optimal threshold, it is of great importance to determine the exact value of q for a given image. Recently, Abdiel and coauthors introduce a methodology to evaluate the nonextensive parameter q of an image. Based on the information theory, the generalized redundancy of an image that presents nonextensive properties can be expressed as[115].

3.4.5.4 PROPOSED METHOD

The biggest limitation with Otsu is its assumption of binary classes: It partitions the grayscale histogram to two classes. However, in real-world segmentation problems we most generally deal with images having more than two class of segments. As per the literature survey CS using Kapur's entropy gives higher PSNR values than using Otsu technique. Tsallis entropy is non-extensive, which means that if two identical systems combine, the entropy of combined system is not equal to summation of entropy of its subsystems. Compare to CS using Kapur's entropy method Tsallis entropy gives higher PSNR values and other parameters. The non-extensive entropy algorithm is suitable for describing the long-range correlations within an image. However, like other entropy-based algorithms, it is still very sensitive to the perturbation of signals, so the scope of its application is limited. By comparison, the Otsu algorithm is stable but not accurate for small target extraction. Therefore, it is possible to combine the advantages of the two and develop a new algorithm with a more general scope of application. We proposed new method, combined Otsu and Tsallis entropy as an objective function to find the x_{best}[12].

The nonextensive entropy algorithm is suitable for describing the long-range correlations within an image. However, like other entropy-based algorithms, it is still very sensitive to the perturbation of signals, so the scope of its application is limited. By comparison, the Otsu algorithm is stable but not accurate for small target extraction. Therefore, it is possible to combine the advantages of the two and develop a new algorithm with a more general scope of application. It is worth mentioning that the nonextensive parameter q in Tsallis entropy is now determined by information redundancy and cannot be tuned arbitrarily.
Based on Equations (3.24) and (3.48), a new objective function can be written as:

$$\mu(t) = S_T - (\sigma_W^2)^{1-q} \qquad \ldots (3.55)$$

In order to retain the concavity of Tsallis entropy, q > 0 should be satisfied[112]. Alternatively, q < 1 is called superextensivity, which will increase the total entropy of the system in comparison with the extensive case (q = 1) [121]. In practice, almost all categories of images exhibit the property of superextensivity[14]. Therefore, the proper range of the no extensive

parameter can be 0 < q < 1. From Equations (3.23) and (3.47), we can see that both of the two algorithms are aimed to maximize the objective functions. The aim of Equation (3.55) is to maximize the objective function, i.e.,

$$t^* = Arg\ max\{\mu(t)\} \qquad \ldots (3.56)$$

The optimal threshold is obtained from Equation (3.56) with the above mentioned range of q. For a synthetic image having a bimodal histogram distribution, as shown in Figure 3.4, the profile of each peak is the normalized q-Gaussian distribution function[119]. From Equations (3.23) and (3.47), we can see that both the Otsu algorithm and the Tsallis entropy algorithm indicate the valley gray-level between the two peaks as the optimal threshold, which exactly coincides with the result of Equation (3.56). For other natural pictures that have an arbitrary histogram distribution, there is no evidence that the result of Equation (3.23) coincides with that of Equation (3.47), whereas Equation (3.55) shows a trade-off between them and Equation (3.56) may yield a proper suggestion. For the histogram of Figure 3.4, it should be noted that the magnitude difference between S_T and σ_W^2 is very large. As shown in Figure 3.5, both of them are functions of threshold t. However, the values of the Tsallis entropy algorithm are totally suppressed by those of the Otsu algorithm for any possible threshold t. Therefore, it is unsuitable to combine S_T and σ_W^2 directly.

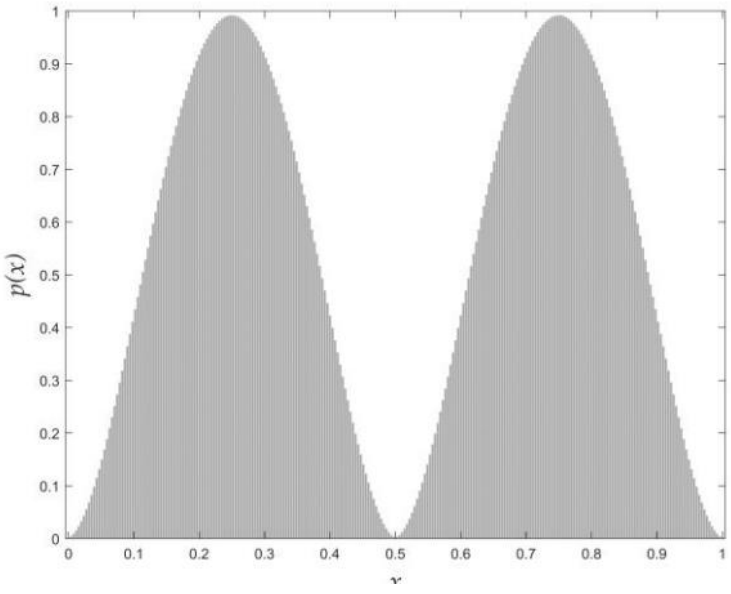

Figure 3.4: Normalized histogram distribution

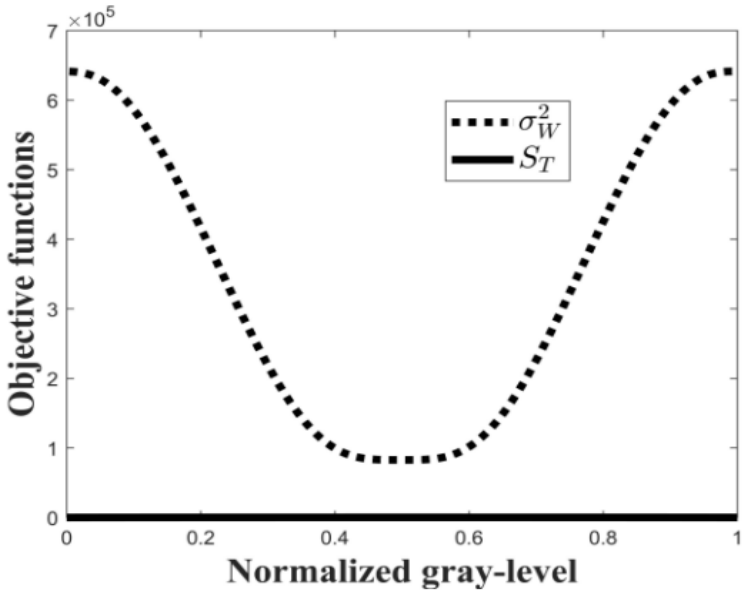

Figure 3.5: Objective functions of the Otsu and Tsallis algorithms

In order to avoid the impact of the magnitude difference, the q-exponential function[112] can be adopted to revise the magnitude of σ_W^2. By definition, Tsallis entropy with a continuous probability distribution function can be expressed as:

$$S_{TS} = \frac{1-\int_0^1 p(x)^q\, dx}{q-1} \qquad \ldots (3.57)$$

where p(x) represents the probability density of the normalized gray-level value. For a system presenting nonextensive q-entropy, the corresponding probability distribution can be written as the q-Gaussian function[119],

$$p(x) = \frac{1}{Z_q}\left[1-(1-q)\frac{x^2}{\sigma^2}\right]^{\frac{1}{1-q}} \qquad \ldots (3.58)$$

The equation itself comprises of a mathematical expression circumscribed by square brackets, multiplied by $1/Z_q$ and raised to the power of $1/(1-q)$. The expression enclosed by square brackets, $1-(1-q)\frac{x^2}{\sigma^2}$, represents the distribution's kernel. It consists of the variable x, parameter q, and standard deviation. It describes the variation in probability density as a function of x.The entire expression, $1-(1-q)\frac{x^2}{\sigma^2}$, is elevated to the fractional exponent power of $1/(1-q)$. This exponentiation modifies the structure of the kernel function, affecting the distribution's tails and overall shape. The complete expression is finally divided by $Z_q$the normalization constant. This division assures that the integral of the PDF over its entire domain

equals 1, thereby validating it as a probability distribution. σ^2 is the variance of x and Z_q is the partition function to keep the probability normalization condition, i.e.,

$$Z_q = \int_0^1 \left[1-(1-q)\frac{x^2}{\sigma^2}\right]^{\frac{1}{1-q}} dx = \frac{\sigma\sqrt{\pi}}{2\sqrt{1-q}} \frac{\Gamma\left(1+\frac{1}{1-q}\right)}{\Gamma\left(\frac{3}{2}+\frac{1}{1-q}\right)} \quad \ldots (3.59)$$

The specified equation determines the value of the normalization constant Z_q. Let's deconstruct the equation: Z_q represents the constant of normalization. The expression $\int_0^1 \left[1-(1-q)\frac{x^2}{\sigma^2}\right]^{\frac{1}{1-q}} dx$ is the integral of the function $\left[1-(1-q)\frac{x^2}{\sigma^2}\right]^{\frac{1}{1-q}}$within the range [0, 1]. The integrand is $1/(1-q)$, which represents a function dependent on the variable x and involving the parameters q and q modifies the form or behaviour of the integrand. The parameter represents the distribution's standard deviation. It quantifies the variability or distribution of the integrand. The integral is computed over the interval [0, 1], which signifies that the integration is conducted from x=0 to x=1. The expression for Z_q is located on the right side of the equation and represents the result of the integral. The expression $2\sqrt{1-q}$ represents a fraction that includes the parameters and q. The gamma functions evaluated at $1+1/(1-q)$ and $3/2+1/(1-q)$ respectively. Γ(k) is the Gamma function and will reduce to factorial (k – 1)! If k is an integer. Substituting p(x) into Equation (3.57) yields:

$$S_{TB} = \frac{1-\int_0^1 \frac{1}{z_q^q}\left[1-(1-q)\frac{x^2}{\sigma^2}\right]^{\frac{q}{1-q}} dx}{q-1} = \frac{1-\xi(\sigma^2)^{\frac{1-q}{2}}}{q-1} \quad \ldots (3.60)$$

where:

$$\xi = \left[\frac{\pi}{4(1-q)}\right]^{\frac{1-q}{2}} \left[\frac{\Gamma\left(\frac{3}{2}+\frac{1}{1-q}\right)}{\Gamma\left(1+\frac{1}{1-q}\right)}\right]^q \frac{\Gamma\left(\frac{1}{1-q}\right)}{\Gamma\left(\frac{3-q}{2(1-q)}\right)} \quad \ldots (3.61)$$

is the integration constant for a given value of q. If p_a and p_b are two identical q-Gaussian distribution functions, according to the nonextensivity of Tsallis entropy, the total entropy can be written as:

$$S_T = S_0 + S_1 + S_2 \ldots + S_r + (1-q).(S_0.S_1.S_2 \ldots .S_r) \quad \ldots (3.62)$$

$$S_T = \frac{1 - \xi_0(\sigma_0^2)^{\frac{1-q}{2}}}{q-1} + \frac{1 - \xi_1(\sigma_1^2)^{\frac{1-q}{2}}}{q-1} + \ldots\ldots + \frac{1 - \xi_r(\sigma_r^2)^{\frac{1-q}{2}}}{q-1} +$$

$$(1-q)\left[\frac{1-\xi_0(\sigma_0^2)^{\frac{1-q}{2}}}{q-1}\frac{1-\xi_1(\sigma_1^2)^{\frac{1-q}{2}}}{q-1}\ldots\frac{1-\xi_r(\sigma_r^2)^{\frac{1-q}{2}}}{q-1}\right] \quad \ldots (3.63)$$

The equation you supplied describes the relationship between the values of S_T when $S_0 + S_1 + S_2 \ldots + S_r$ is given as an argument to the function versus when 0, 1, 2, … and r are passed separately. Let's deconstruct the equation: S_0 returns the value of $S_T$when the argument 0 is provided as its argument, S_1 returns the value of S_T when the argument 1 is provided as its argument, similarly for S_r. S_T is equal to the sum of $S_0 + S_1 + S_2 \ldots + S_r$, as well as the product of (1-q) and $S_0 + S_1 + S_2 \ldots + S_r$. (1-q) is a parameter that modifies the relationship among the S_T values. σ_0^2, σ_1^2, …, σ_r^2.represent the respective standard deviations associated with the values 0, 1, ….,r. $(\sigma_0^2)^{\frac{1-q}{2}}$ denotes the standard deviation of 0 to the power of ((1-q)/2). $(\sigma_1^2)^{\frac{1-q}{2}}$ denotes the standard deviation of 1 to the power of ((1-q)/2). Similarly, $(\sigma_r^2)^{\frac{1-q}{2}}$ denotes the standard deviation of r to the power of ((1-q)/2). Each term's denominators (q-1) assure the correct scaling of the equation.

Substituting $\sigma_0^2 = \sigma_1^2 = \ldots\ldots = \sigma_r^2 = \sigma_W^2$ into Equation (3.63) yields:

$$S_T = \frac{\xi_0\xi_1\ldots.\xi_r(\sigma_W^2)^{1-q}-1}{1-q} \quad \ldots (3.64)$$

Therefore, the magnitude of $(\sigma_W^2)^{1-q}$ is comparable with S_T at the proper range of q, and the rationality of Equation (3.55) is shown.

3.5 FEATURE EXTRACTION USING DISCRETE WAVELET TRANSFORM

In the Feature Extraction of the Brain MRI Image, Discrete Wavelet Transform is used. The DWT is an efficient and useful tool for signal and image processing applications. The growing "success" is due to the achievements reached in the field of mathematics, to its multiresolution processing capabilities, and also to the wide range of filters that can be provided. These features allow the DWT to be tailored to suit a wide range of applications. The DWT is a mathematical technique used to decompose a signal or image into a set of sub-bands, each representing a different frequency range. It is widely used in signal and image processing, compression, and feature extraction. DWT works by applying a series of filters to the input signal or image, which separate it into low-frequency and high-frequency components. These components can then be

further decomposed into sub-bands, creating a tree-like structure known as a wavelet decomposition. In image processing, DWT is often used in conjunction with the Gray-Level Co-occurrence Matrix technique. GLCM is a statistical method used to extract texture features from an image. By analysing the spatial relationships between pixels with similar gray levels, GLCM can extract information about the texture, such as roughness or smoothness. DWT can be used to pre-process an image before applying GLCM, which can enhance the texture features and improve their accuracy. DWT can also be used to extract texture features directly, without the need for GLCM.

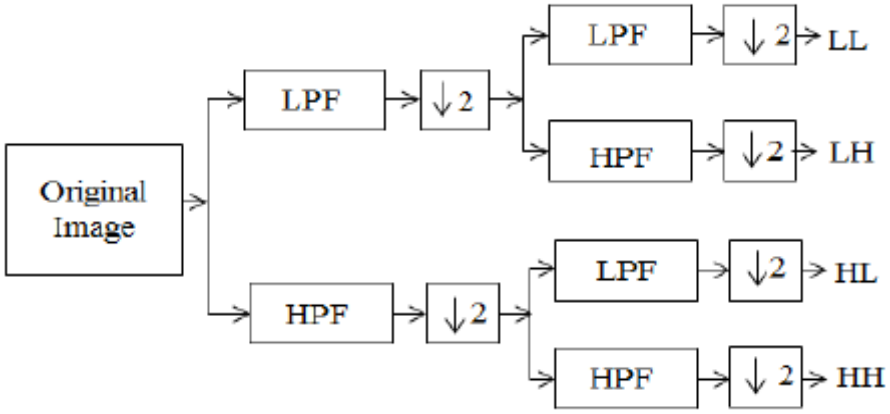

Figure 3.6: Two level Discrete Wavelet Transform [12]

The DWT is a mathematical algorithm used to decompose a signal into wavelet coefficients. The equation for the DWT can be expressed as follows: Let x[n] be a discrete-time signal of length N, and h[n] and g[n] be two finite-length filter coefficients of length L. Then, the DWT of x[n] can be expressed as:

For the first level decomposition:

$$c_{1,k} = \sum_{n=0}^{N-1} h[n]\, x[2k-n]\,,\; d_{1,k} = \sum_{n=0}^{N-1} g[n]\, x[2k-n]$$

$$where\ k = 0, 1, \ldots, \frac{N}{2} - 1 \qquad \ldots (3.65)$$

$c_{1,k}$ and $d_{1,k}$ are the wavelet coefficients of the first level decomposition.

For the second level decomposition

$$c_{2,k} = \sum_{n=0}^{N-1} h[n]\, c_{1,2k-n}\,,\; d_{2,k} = \sum_{n=0}^{N-1} g[n]\, c_{1,2k-n}$$

$$where\ k = 0, 1, \ldots, \frac{N}{4} - 1 \qquad \ldots (3.66)$$

and $c_{2,k}$ and $d_{2,k}$ are the wavelet coefficients of the second level decomposition.

The process can be repeated for higher level of decomposition. Second level of decomposition is used in this research. In the Equation (3.66), c and d represent the approximation and detail coefficients, respectively. The h and g filters are typically chosen as the Daubechies wavelet filters, but other wavelet filters can also be used. In the DWT, LL, LH, HL, and HH refer to the approximation, horizontal detail, vertical detail, and diagonal detail coefficients, respectively, that are obtained after applying the wavelet transform to an image or signal. These coefficients can be computed using filter banks, where the LL coefficients represent the low-frequency approximation and the remaining coefficients represent the high-frequency details in different directions.

Let x(n) be the input signal or image, and h(n) and g(n) be the low-pass and high-pass filter coefficients, respectively, used in the wavelet transform. The DWT equations for obtaining the LL, LH, HL, and HH coefficients can be written as follows:

$$LL(n) = x * h * h\ (downsampling\ by\ 2) \qquad \ldots (3.67)$$

$$LH(n) = x * h * g\ (downsampling\ by\ 2) \qquad \ldots (3.68)$$

$$HL(n) = x * g * h\ (downsampling\ by\ 2) \qquad \ldots (3.69)$$

$$HH(n) = x * g * g\ (downsampling\ by\ 2) \qquad \ldots (3.70)$$

where "*" denotes convolution and down sampling by 2 reduces the size of the coefficients by half. For Equations (3.67) to (3.70) assume periodic boundary conditions, where the input signal is periodically extended to avoid boundary effects. Different boundary conditions or extension modes can be used in practice, depending on the application and the properties of the signal or image.

3.6 FEATURE CLASSIFICATION USING SUPPORT VECTOR MACHINE

In the Feature Classification of the Brain MRI Image, Support Vector Machine is described. Support Vector Machine is one of best machine learning algorithms, which was proposed in 1990's and used mostly for pattern recognition. This has also been applied to many pattern classification problems such as image recognition, speech recognition, text categorization, face detection and faulty card detection, etc. Pattern recognition aims to classify data based on either a priori knowledge or statistical information extracted from raw data, which is a powerful tool in data separation in many disciplines. SVM is a supervised type of machine learning. algorithm in which, given a set of training examples, each marked as belonging to one of the many categories, an SVM training algorithm builds a model that predicts the category of the new example. SVM has the greater ability to generalize the problem, which is the goal in statistical learning. The statistical learning theory provides an outline for studying the problem of gaining knowledge, making predictions, making decisions from a set of data[30].

SVM is a powerful and widely-used machine learning algorithm for classification and regression analysis. SVM is particularly useful when dealing with complex and high-dimensional data because it can find the best possible decision boundary that separates the data points into different classes. SVM was first introduced in the 1990s by Vapnik and his colleagues and has since then become a popular choice for various applications in the fields of computer vision, natural language processing, and bioinformatics, among others. The basic idea behind SVM is to transform the input data into a higher-dimensional space, where it becomes easier to find a linear or nonlinear decision boundary that separates the different classes. This decision boundary is defined by a hyperplane that maximizes the margin between the two closest points from different classes. The points that lie on the margin are called support vectors, hence the name Support Vector Machines. One of the advantages of SVM is that it can handle both linear and nonlinear data by using different types of kernels. SVM can also be used for binary classification and multiclass classification problems and can even handle imbalanced datasets. Another advantage of SVM is that it is less prone to overfitting compared to other algorithms like decision trees and neural networks.

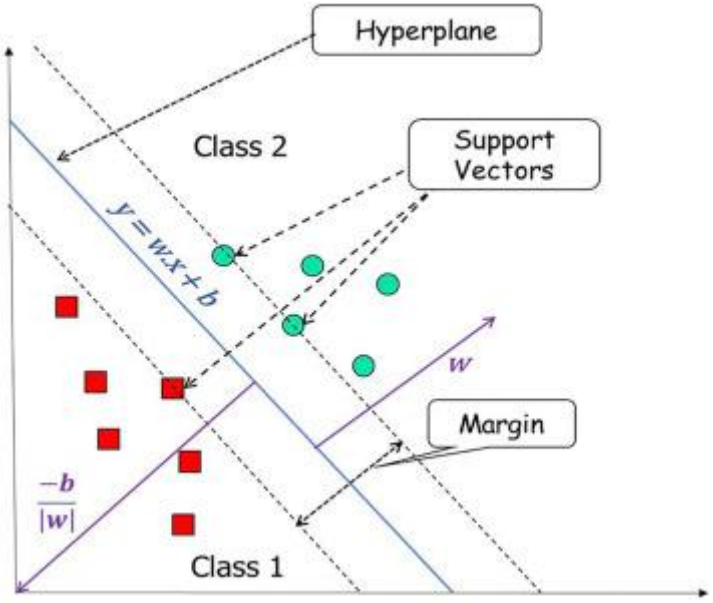

Figure 3.7: Support Vector Machine[87]

The equation for a Support Vector Machine depends on the type of problem it is solving. There are two main types of SVM: linear SVM and non-linear SVM.

For Linear SVM:

In the case of linearly separable data, the equation for a linear SVM can be written as:

$$f(x) = sign(w^T x + b) \quad \dots (3.71)$$

where w is the weight vector perpendicular to the hyperplane, b is the bias term, x is the input vector, and sign is the sign function that returns -1 for negative values and 1 for positive values. The goal of the SVM is to find the weight vector w and the bias term b that maximize the margin between the two classes of data while correctly classifying all the training examples.

For Non-linear SVM:

For non-linearly separable data, a kernel function is used to map the input vectors into a higher-dimensional space where the classes can be separated by a hyperplane. The equation for a non-linear SVM is then:

$$f(x) = sign(\sum \alpha_i \, y_i \, K(x_i, x) + b) \quad \dots (3.72)$$

where $\sum \alpha_i \, y_i \, K(x_i, x)$ is the dot product of the weight vector (which is now a linear combination of the support vectors) and the kernel function evaluated at the training examples and the input vector x, and b is the bias term. The values α_i and y_i represent the Lagrange

multipliers and the class labels of the support vectors, respectively. The kernel function K(xi, x) can be any function that measures the similarity between two vectors, such as the radial basis function kernel or the polynomial kernel. The goal of the SVM is again to find the values of αi and b that maximize the margin while correctly classifying all the training examples.

3.7 CONCLUSION

Different processing stages with Filtering techniques, Segmentation, Feature Extraction and Classification are developed using mathematical model for proposed method. For image dataset Magnetic Resonance Imaging Brain Images is used for the research work. In Pre-processing of the Brain Images stage, different filters Wiener filter, Anisotropic filter, Median filter, Non-Local Means filter and Combined filters are described. In Segmentation of the MRI Brain Image stage, Multi-thresholding Cuckoo Search Algorithm using different objective functions are used. In Feature extraction of the Brain Images stage, Discrete Wavelet Transform is described. In Feature Classification of the Brain Images stage, Support Vector Machine is described.