A Project Report
On
Confidence factor weighted Gaussian function induced parallel fuzzy rank level fusion for inference and its application to face recognition
And
Its further modification using Type-2 fuzzy logic
With
Python Implementations and Results

Under the guidance of

Prof. Jamuna Kanta Sing

Department of Computer Science & Engineering

Jadavpur University, Kolkata -700023

Submitted On

12 June, 2021

By

Dibyajyoti Dhar, 001710501067

Final Year Undergraduate Student

Department of Computer Science & Engineering

Jadavpur University, Kolkata -700023

Sing et al(2020)[1] proposes a novel approach for inference using fuzzy rank-level fusion and explores its application to face recognition using multiple biometric representations, more specifically a new scheme for generating fuzzy ranks induced by a Gaussian function based on the confidence of a classifier. In contrast to the conventional ranking, this fuzzy ranking reflects some associations among the outputs (confidence factors) of a classifier. These fuzzy ranks, produced by various representations of a face image, are fused weighted by the corresponding confidence factors of the classifier to calculate the final ranks while recognizing a face. In different practical purposes, where multiple traits of a person are unavailable, this method is proved to be highly effective. The experimental results using different feature vectors of a face image employing different classifiers testify that this method can significantly improve recognition accuracy as compared to those from individual feature vectors and as well as some commonly used rank-level fusion methods.

Confidence factor weighted Gaussian function induced parallel fuzzy rank-level fusion (CFGaussFRLF) method

Below are the procedures of the inference system through confidence factor weighted Gaussian density function induced fuzzy rank-level fusion (CFGaussFRLF) method. The steps include 1) to generate fuzzy ranks based on the outputs (confidence factors) of a classifier and 2) to sum the confidence factor weighted fused ranks from several classifiers, trained by different feature vectors, to get the final ranking for classification and recognition. This inference procedure addresses the issues of conventional ranking, where ranking is done without utilizing the distribution (closeness) of classifier's outputs; leading to improper inference. This study has designed the inference system for face recognition. Three different types of sensors have been used here for feature extraction from a facial image. These feature vectors are propagated in parallel to an identical classifier for generation of a list of k classes based on top k ranking. Unlike the conventional way, where the rankings are represented by natural numbers, here these k classes are ranked by taking the complement of a Gaussian density function, resulting in fuzzy ranks in the range [0, 1]. The fuzzy ranks are generated based on the outputs of the classifier, which are treated as confidence factor (CF) of the classifier with respect to the test pattern under consideration. Unlike conventional ranking, fuzzy ranking actually indicates a number between 0 to 1, where 0 indicates non-existence in the fuzzy set, 1 indicates complete existence in the fuzzy set and other membership function within the range denotes partial inclusion.

Afterwards, the fuzzy ranks are combined, corresponding to a class, from the different classifiers. The same operation is performed for the confidence factors. If a class does not appear in the top k ranks for a classifier, penalty is imposed both for its fuzzy rank and confidence factor. It may be noted that the lower value of fuzzy rank sum indicates the possibility of being the winner; whereas the same is contrary in the case of confidence factor. To make the confidence factor sum in line with fuzzy rank sum, it is normalized and its complement is taken for further processing. Finally, with respect to a class, the fuzzy rank sum and confidence factor sum are fused by means of multiplication operation. The class having the lowest fused value is said to be the winner and is assigned as the class of the test pattern under consideration.

The CFGaussFRLF methodology has been compared with (i) generalized two-dimensional FLD (G-2DFLD) method[2], (ii) fuzzy generalized two dimensional LDA (FG-2DLDA) method[3] and (iii) modular local binary patterns (LBP)[4] method. For the classifiers, (i) radial basis function (RBF) neural network, (ii) support vector machines (SVM) and (iii) most frequently used k-nearest neighbor (k-NN), are used.

CFGaussFRLF method Mathematical Formulation

- Let $\mathbf{X}$ is a face image.
- There are $\mathbf{H}$ different feature vectors corresponding to each $\mathbf{X}$, suppose $(\mathbf{L_1}, \mathbf{L_2}, \ldots, \mathbf{L_H})$
- $\mathbf{L}$ is the input to the classifier

Now, suppose when $\mathbf{L_1}$ is an input, Classifier generates fuzzy ranks $(\mathbf{R_1}^{\mathbf{L1}}, \mathbf{R_2}^{\mathbf{L1}}, \ldots, \mathbf{R_C}^{\mathbf{L1}})$ and confidence factors $(\mathbf{CF_1}^{\mathbf{L1}}, \mathbf{CF_2}^{\mathbf{L1}}, \ldots, \mathbf{CF_C}^{\mathbf{L1}})$, where $\mathbf{C}$ is the total number of distinct classes. For example, $\mathbf{R_i^{Lj}}$ and $\mathbf{CF_i^{Lj}}$ represent the fuzzy rank and confidence factor of the ith class while classifying the input feature vector, $\mathbf{L_j}$

- The confidence factors are normalized and satisfy the following condition

$$\sum_{c=1}^{C} CF_c^{L_i} = 1;\ \forall\ i,\ i = 1,\ 2,\ \ ...,\ H$$

- The fuzzy rank for a class, **c** using $\mathbf{L_i}$ feature vector is generated by the complement of a Gaussian density function, as defined follows:

$$R_c^{L_i} = \left(1 - \ e^{-\frac{(CF_c^{L_i} - 1.0)^2}{2 \times 1.0}}\right),\ i = 1,\ 2,\ \ ...,\ H;\ c = 1,\ 2,\ \ ...,\ C$$

It may be noted that $\mathbf{R_c^{Li}}$ lies within [0, 1] and lowest value (0 in ideal case) is said to be the winner, just opposite to possession of rank 1.

- Suppose, $\mathbf{K^{Li}}$ represents the set of top (first) k ranked classes generated by the classifier while using the $\mathbf{L_i}$ feature vector. Further, it may be noted that entries of $\mathbf{K^{Li}}$ and $\mathbf{K^{Lj}}$, $\mathbf{(i \neq j)}$ may differ as they are generated by two different feature vectors. The rank sum $\mathbf{RS_c}$ and complement of confidence factor sum $\mathbf{CFS_c}$ corresponding to a class **c** is generated as follows:

$$RS_c = \sum_{i=1}^{H} \begin{cases} R_c^{L_i} = R_c^{L_i}, & if\ R_c^{L_i}\ \in\ K^{L_i} \\ R_c^{L_i} = P_c^{R}, & otherwise \end{cases}$$

$$CFS_c = 1 - \frac{1}{H} \sum_{i=1}^{H} \begin{cases} CF_c^{L_i} = CF_c^{L_i}, & if\ R_c^{L_i}\ \in\ K^{L_i} \\ CF_c^{L_i} = P_c^{CF}, & otherwise \end{cases}$$

where $\mathbf{P_c^R}$ and $\mathbf{P_c^{CF}}$ are penalties, which are imposed to class c if it does not come into the top k ranks. These penalties actually negate the possibility of class **c** to become an unlikely winner.

- The fusion of $\mathbf{RS_c}$ and $\mathbf{CFS_c}$ are realized by means of the multiplication operation to generate the final score, which is used for final ranking, as defined follows:

  $\mathbf{FS_c = RS_c \times CFS_c}$

- Finally, that class is selected, which has the least (minimum) final score and assign it as the class of the input image X, as defined follows:

$$class\ (\mathbf{X}) = \arg \min_{c=1,\ 2,\ ...,\ C} \{FS_c\}$$

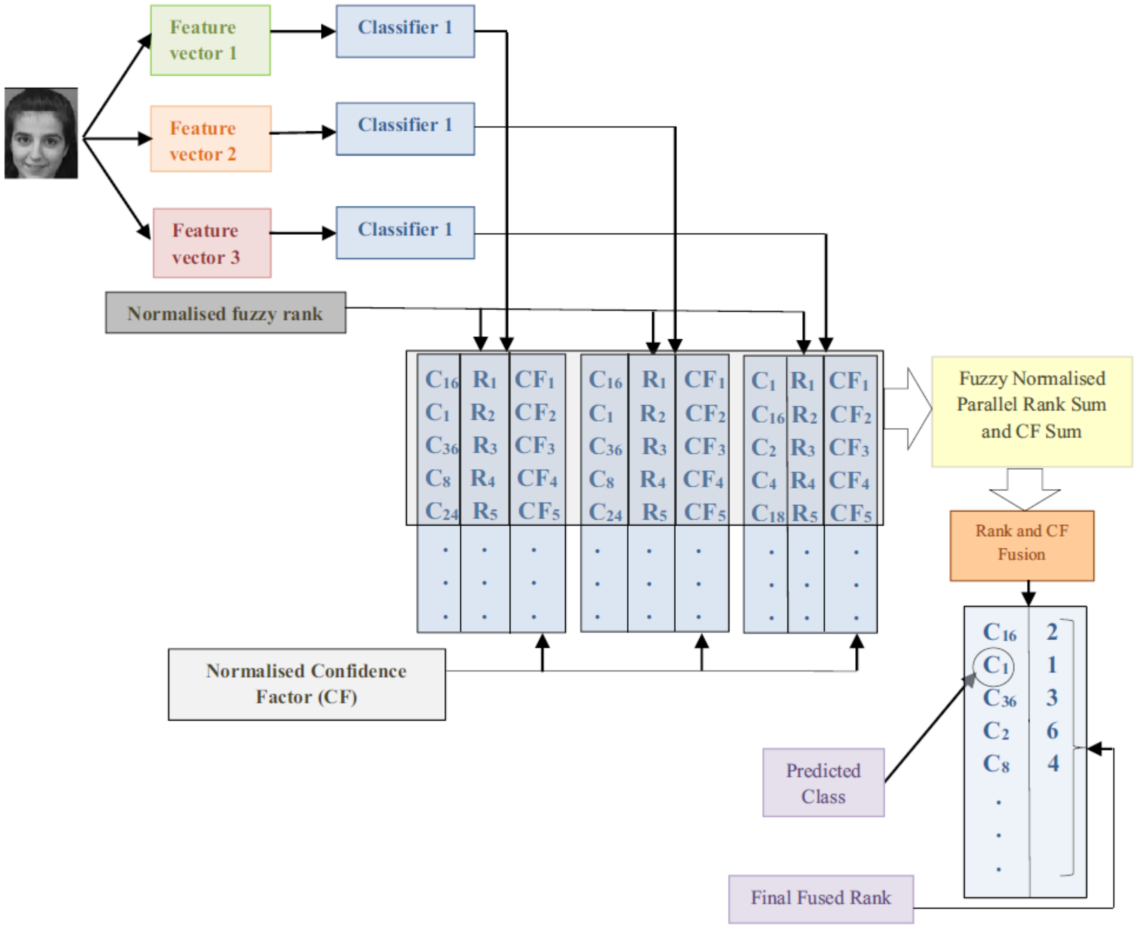

Fig 1. Work flow of the CFGaussFRLF method in the form of a schematic diagram

Experimental Result

AT&T dataset result : CFGaussFRLF method gives the best recognition rate

Comparative performance of different methods on the AT&T face database.

Classifier	Method	Average recognition rate (standard deviation)				
		$s = 3$	$s = 4$	$s = 5$	$s = 6$	$s = 7$
k-NN	G-2DFLD	92.66 (2.62)	95.82 (1.70)	97.45 (0.87)	98.44 (0.87)	98.34 (1.11)
	FG-2DLDA	93.33 (2.56)	95.94 (1.62)	97.53 (0.90)	98.56 (0.86)	98.38 (1.10)
	Modular LBP	93.12 (1.99)	95.76 (1.74)	97.50 (0.83)	98.53 (0.84)	98.29 (1.06)
	Highest rank	93.19 (1.92)	95.91 (1.50)	97.53 (0.83)	98.50 (0.85)	98.37 (0.83)
	Modified highest rank	93.31 (2.57)	95.94 (1.41)	97.85 (1.01)	98.59 (0.88)	98.46 (0.56)
	Borda count	93.27 (2.61)	95.89 (1.63)	97.75 (0.91)	98.62 (0.79)	98.37 (0.83)
	Weighted Borda count	93.40 (2.53)	96.00 (1.47)	98.03 (0.90)	98.78 (0.69)	98.54 (0.66)
	Nonlinear weighted rank fusion	94.03 (2.05)	96.14 (1.38)	98.28(0.60)	98.92 (0.66)	98.62 (0.68)
	CFGaussFRLF	**94.16 (1.97)**	**96.29 (1.41)**	**98.33 (0.61)**	**99.16 (0.55)**	**98.79 (0.51)**
RBF neural networks	G-2DFLD	92.82 (2.67)	95.94 (1.21)	97.73 (0.87)	98.66 (0.96)	98.42 (1.11)
	FG-2DLDA	93.54 (2.62)	96.03 (1.59)	97.85 (1.01)	98.75 (0.93)	98.54 1.04
	Modular LBP	93.43 (2.21)	95.85 (1.71)	97.83 (0.92)	98.56 (0.86)	98.50 (1.07)
	Highest rank	93.50 (2.50)	96.00 (1.51)	97.85 (1.01)	98.66 (0.80)	98.33 (0.81)
	Modified highest rank	93.62 (2.33)	96.06 (1.42)	98.08 (0.88)	98.75 (0.93)	98.62 (0.68)
	Borda count	93.58 (2.23)	96.03 (1.64)	98.00 (1.04)	98.65 (0.80)	98.54 (0.66)
	Weighted Borda count	93.66 (2.22)	96.09 (1.63)	98.25 (1.01)	98.91 (0.67)	98.92 (0.48)
	Nonlinear weighted rank fusion	94.46 (1.75)	96.49 (1.38)	98.50 (0.63)	99.17 (0.56)	99.04 (0.62)
	CFGaussFRLF	**94.68 (1.59)**	**96.55 (1.43)**	**98.58 (0.63)**	**99.35 (0.59)**	**99.13 (0.69)**
SVM	G-2DFLD	93.03 (2.51)	95.99 (1.23)	97.68 (0.91)	98.72 (0.92)	98.50 (1.07)
	FG-2DLDA	93.62 (2.61)	96.08 (1.47)	98.00 (1.03)	98.88 (0.63)	98.62 (0.68)
	Modular LBP	93.43 (2.21)	95.88 (1.68)	97.85 (0.95)	98.69 (0.79)	98.58 (0.61)
	Highest rank	93.67 (2.30)	96.06 (1.46)	98.00 (1.04)	98.81 (0.91)	98.42 (0.85)
	Modified highest rank	93.79 (2.15)	96.10 (1.43)	98.05 (1.05)	98.91 (0.81)	98.71 (0.69)
	Borda count method	93.75 (2.13)	96.15 (1.55)	98.08 (0.88)	98.84 (0.71)	98.62 (0.68)
	Weighted Borda count	93.83 (2.06)	96.25 (1.29)	98.28 (0.83)	99.03 (0.48)	98.83 (0.65)
	Nonlinear weighted rank fusion	94.63 (1.81)	96.66 (1.47)	98.68 (0.59)	99.34 (0.57)	99.04 (0.73)
	CFGaussFRLF	**94.73 (1.85)**	**96.73 (1.22)**	**98.73 (0.62)**	**99.41 (0.63)**	**99.25 (0.71)**

UMIST database : CFGaussFRLF method gives the best recognition rate

Comparative performance of different methods on the UMIST face database.

Classifier	Method	Average recognition rate (standard deviation)			
		$s = 4$	$s = 6$	$s = 8$	$s = 10$
k-NN	G-2DFLD	85.81 (3.12)	91.89 (2.03)	95.27 (1.17)	96.73 (1.24)
	FG-2DLDA	85.92 (3.07)	92.16 (1.37)	95.31 (1.15)	96.94 (1.05)
	Modular LBP	86.16 (2.98)	92.27 (1.57)	95.21 (1.22)	96.90 (1.18)
	Highest rank	86.37 (2.85)	92.36 (1.47)	95.62 (1.23)	97.17 (1.22)
	Modified highest rank	86.45 (2.86)	92.62 (1.53)	95.85 (1.12)	97.31 (1.25)
	Borda count method	86.61 (2.86)	92.88 (1.20)	95.93 (0.85)	97.45 (1.27)
	Weighted Borda count	86.94 (2.85)	93.18 (1.37)	96.25 (0.81)	97.55 (1.21)
	Nonlinear weighted rank fusion	87.27 (3.03)	93.32 (1.41)	96.55 (0.85)	97.79 (0.98)
	CFGaussFRLF	**87.44 (3.04)**	**93.49 (1.23)**	**96.66 (0.94)**	**97.84 (0.95)**
RBF neural networks	G-2DFLD	86.05 (3.08)	92.09 (2.13)	95.43 (1.28)	96.88 (1.16)
	FG-2DLDA	86.30 (3.00)	92.42 (1.49)	95.58 (1.17)	97.05 (1.09)
	Modular LBP	86.43 (3.03)	92.23 (1.59)	95.51 (1.13)	97.03 (0.98)
	Highest rank	86.56 (2.89)	92.93 (1.15)	95.78 (1.11)	97.29 (1.21)
	Modified highest rank	86.67 (2.87)	93.17 (0.98)	96.03 (0.95)	97.51 (1.22)
	Borda count	86.84 (2.77)	93.10 (0.88)	96.13 (0.73)	97.73 (1.15)
	Weighted Borda count	87.19 (2.78)	93.39 (0.93)	96.39 (0.90)	97.81 (0.88)
	Nonlinear weighted rank fusion	87.52 (3.01)	93.56 (0.99)	96.87 (1.07)	98.14 (0.73)
	CFGaussFRLF	**87.66 (3.02)**	**93.65 (0.91)**	**96.98 (0.98)**	**98.25 (0.86)**
SVM	G-2DFLD	86.22 (3.09)	92.28 (2.06)	95.54 (1.29)	96.92 (1.16)
	FG-2DLDA	86.82 (2.83)	92.59 (1.51)	95.70 (1.11)	97.14 (0.95)
	Modular LBP	86.77 (2.79)	92.50 (1.51)	95.62 (1.17)	97.01 (1.01)
	Highest rank	86.95 (2.67)	92.95 (1.41)	95.90 (1.11)	97.42 (1.05)
	Modified highest rank	87.06 (2.68)	93.08 (1.05)	96.14 (0.93)	97.55 (1.13)
	Borda count method	87.14 (2.83)	93.24 (0.90)	96.22 (0.69)	97.65 (1.11)
	Weighted Borda count	87.35 (2.82)	93.41 (0.41)	96.47 (0.89)	97.89 (0.92)
	Nonlinear weighted rank fusion	87.61 (3.05)	93.60 (0.96)	96.91 (1.10)	98.28 (0.84)
	CFGaussFRLF	**87.72 (3.00)**	**93.70 (0.84)**	**97.04 (0.95)**	**98.36 (0.81)**

Moreover, FERET and AR face databases demonstrate that the CFGaussFRLF method improves the performances in comparison with the individuals and also some of the state-of-the-art rank-level fusion methods. And on the analysis of statistical significance of the classifiers, it is vivid that SVM produces superior results for CFGaussFRLF method in comparison to k-NN, and RBF neural networks.

Modification of CFGaussFRLF Method using Type 2 Fuzzy Logic

Implementation 1:

- The fuzzy rank for a class, $\mathbf{c}$ using $\mathbf{L_i}$ feature vector is generated by the complement of a Gaussian density function, as defined follows, like the previous approach

$$R_c^{L_i} = \left(1 - e^{-\frac{(CF_c^{L_i} - 1.0)^2}{2 \times 1.0}}\right), \ i = 1, \ 2, \ \ldots, H; c = 1, \ 2, \ \ldots, C$$

It may be noted that $\mathbf{R_c^{Li}}$ lies within [0, 1] and lowest value (0 in ideal case) is said to be the winner, just opposite to possession of rank 1.

At this stage, we will actually apply the Type 2 fuzzy logic.

For $\mathbf{R_c^{Li}}$, $\mathrm{i} = 1, 2, \ldots, \mathrm{H}$, we have to plot the values in the $\mathbf{t}$ vs $\mathbf{R_c^{Li}}$ graph,
where $\mathbf{t = 1 + (i\text{-}1)/(H - 1)}$

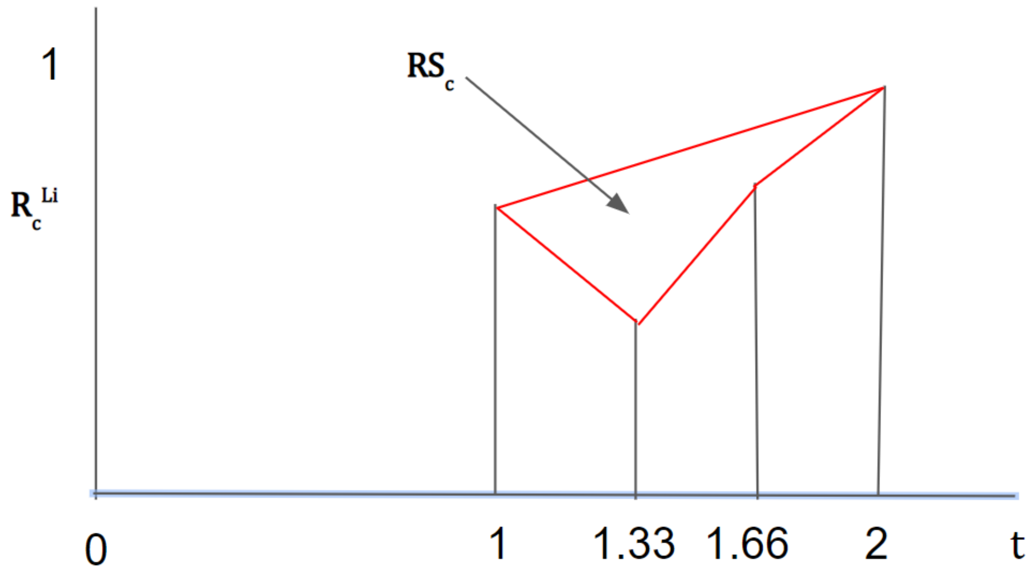

Fig 2. Calculation of $\mathbf{RS_c}$ as an area of the bounded polygon formed by all $\mathbf{(t, R_c^{Li})}$ coordinates

And $\mathbf{RS_c}$ will be the area of the bounded polygon generated by all the $\mathbf{H}$ points having coordinates $\mathbf{(t, R_c^{Li})}$ for all $\mathrm{i} = 1, 2, \ldots, \mathrm{H}$. As shown in the example in Fig. 2, for $\mathrm{H}=4$, rank sum for class $\mathbf{c}$ can be calculated. Basically we will here using a normalized form of fuzzy set Type 2 as for $\mathrm{i} = 2, 3$; $\mathbf{RS_c}$ has no definite value.

- The fusion of $\mathbf{RS_c}$ and $\mathbf{CFS_c}$ are realized by means of the multiplication operation to generate the final score, which is used for final ranking, as defined follows:

$\mathbf{FS_c = RS_c \times CFS_c}$

- Finally, that class is selected, which has the least (minimum) final score and assign it as the class of the input image $\mathbf{X}$, as defined follows:

$$class\ (\boldsymbol{X}) = \arg \min_{c=1,\ 2,\ \dots,\ C} \{FS_c\}$$

Why will it improve the accuracy?

If R_c^{Li} does not belong to the top k fuzzy rank list, for CFGaussFRLF method, we are actually imposing a hard-coded penalty P_c^R in the summation for calculation of RS_c. But, using fuzzy logic Type 2, no assumed penalty is required as the area of the bounded polygon will increase significantly, which will auto-penalize the image for being in that class c, as FS_c value will increase. If we compare Fig 3 with Fig 2, it will be clear.

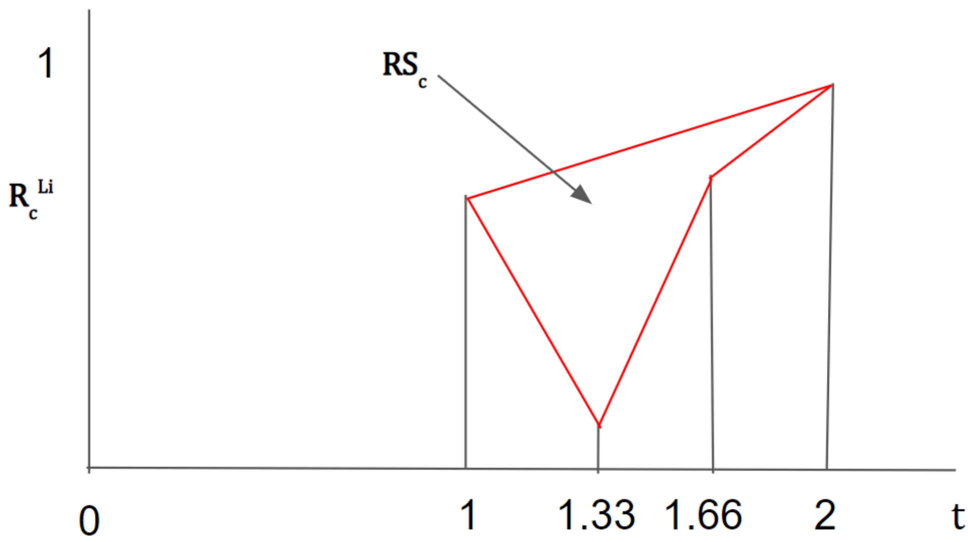

Fig 3. Consider $k = 3$, $H = 4$, Here $\mathbf{RS_c}$ value is increasing independent of hard-coded penalty.

Implementation 2:

To make it improve further, we have considered sorting the R_c^{Li} in order to increase the accuracy. If the gap between the lowest value of R_c^{Li} and the highest value of R_c^{Li} is high, it will increase the area significantly in this case, which will better prevent the image from belonging to class **c**.
From an example, it will be more clear.
Suppose, we have the following coordinates to determine the $\mathbf{RS_c}$ area.

x	1	1.33	1.66	2
y = R_c^{Li}	0.60	0.15	0.65	0.75

After sorting only the y values, we will get the coordinates in the following form.

x	1	1.33	1.66	2
y = $\mathbf{R_c^{Li}}$	0.15	0.60	0.65	0.75

After calculating the $\mathbf{RS_c}$ area, the later steps will be the same as that of **Implementation 1**.

Implementation 3:

To obtain the maximum accuracy in the above algorithm, we consider to sort the $\mathbf{R_c^{Li}}$ values in zigzag order. The more the fluctuation in the zigzag-ordered-sorted array, the more the possibility of the increment of the area, the more would be the auto-penalization for the image from belonging to class **c**.

Suppose, we have the following coordinates to determine the $\mathbf{RS_c}$ area.

x	1	1.33	1.66	2
y = $\mathbf{R_c^{Li}}$	0.60	0.15	0.65	0.75

After sorting only the y values in zigzag order, we will get the coordinates in the following form.

x	1	1.33	1.66	2
y = $\mathbf{R_c^{Li}}$	0.75	0.15	0.65	0.60

After calculating the $\mathbf{RS_c}$ area, the later steps will be the same as that of **Implementation 1**.

Experimental Result of the Algorithm on IRIS dataset

The experiments are done upon the standard iris dataset. (https://github.com/djdhar/CFGaussFRLF/blob/main/iris.csv)

For each experiment, the dataset is randomly split into 2 equally sized parts. a) train b) test The train dataset consists of 25 data points of 'setosa' class, 25 data points of 'versicolor' class, 25 data points of 'virginica' class. Same for the test dataset.

The detailed results of three experiments are the following.

Random Set 1

Test Set : *https://github.com/djdhar/CFGaussFRLF/blob/main/iris_split/testset_1.csv*

Train Set : *https://github.com/djdhar/CFGaussFRLF/blob/main/iris_split/trainset_1.csv*

Experiment and Implementation (Source Code) :
https://github.com/djdhar/CFGaussFRLF/blob/main/Train_Test_Split_of_CFGaussFRLF_Set_1.ipynb
Result :

Type 1 fuzzy Logic **(Penalty required)**		Type 2 fuzzy Logic **(Penalty not required)**					
CFGaussFRLF algorithm :		**Implementation 1:**		**Implementation 2: (Considering R_c^{Li} values sorted in order to calculate RS_c as area)**		**Implementation 3: (Considering R_c^{Li} values zigzag-ordered sorted in order to calculate RS_c as area)**	
True Detection	71	True Detection	55	True Detection	58	True Detection	64
False Detection	4	False Detection	20	False Detection	17	False Detection	11

Random Set 2
Test Set : https://github.com/djdhar/CFGaussFRLF/blob/main/iris_split/testset_2.csv
Train Set : https://github.com/djdhar/CFGaussFRLF/blob/main/iris_split/trainset_2.csv
Experiment and Implementation (Source Code) :
https://github.com/djdhar/CFGaussFRLF/blob/main/Train_Test_Split_of_CFGaussFRLF_Set_2.ipynb

Result :

Type 1 fuzzy Logic **(Penalty required)**	Type 2 fuzzy Logic **(Penalty not required)**

CFGaussFRLF algorithm :		**Implementation 1:**		**Implementation 2: (Considering R_c^{Li} values sorted in order to calculate RS_c as area)**		**Implementation 3: (Considering R_c^{Li} values zigzag-ordered sorted in order to calculate RS_c as area)**	
True Detection	72	True Detection	49	True Detection	53	True Detection	61
False Detection	3	False Detection	26	False Detection	22	False Detection	14

Random Set 3
Test Set : *https://github.com/djdhar/CFGaussFRLF/blob/main/iris_split/testset_3.csv*
Train Set : *https://github.com/djdhar/CFGaussFRLF/blob/main/iris_split/trainset_3.csv*
Experiment and Implementation (Source Code) :
https://github.com/djdhar/CFGaussFRLF/blob/main/Train_Test_Split_of_CFGaussFRLF_Set_3.ipynb

Result :

Type 1 fuzzy Logic **(Penalty required)**		Type 2 fuzzy Logic **(Penalty not required)**					
CFGaussFRLF algorithm :		**Implementation 1:**		**Implementation 2: (Considering R_c^{Li} values sorted in order to calculate RS_c as area)**		**Implementation 3: (Considering R_c^{Li} values zigzag-ordered sorted in order to calculate RS_c as area)**	
True Detection	69	True Detection	49	True Detection	52	True Detection	61
False Detection	6	False Detection	26	False Detection	23	False Detection	14

Conclusion:

1) Using type-2 fuzzy logic, imposing of the penalty is irrelevant. Experimental results clearly show that with or without imposing the penalty, results obtained are exactly the same.
2) Sorting the $\mathbf{R}_c^{Li}$ values improves the accuracy, zigzag order sorting improves the accuracy to a greater extent. Auto-penalization due to area fluctuation is an important factor.
3) Increasing the number of features (four in iris dataset) will certainly better the accuracy as it will enjoy a large scope for auto-penalization.

References:

1. Jamuna Kanta Sing, Aniruddha Dey, Manas Ghosh, Confidence factor weighted Gaussian function induced parallel fuzzy rank-level fusion for inference and its application to face recognition, Information Fusion, Volume 47, 2019, Pages 60-71, ISSN 1566-2535
2. S. Chowdhury, J.K. Sing, D.K. Basu, M. Nasipuri, Face recognition by generalized two-dimensional FLD method and multi-class support vector machines, Appl. Soft Comput. 11 (7) (2011) 4282–4292.
3. A. Dey, J.K. Sing, Face recognition by Fuzzy generalized 2DFLD method and RBF neural network classifier, IEEE Computational intelligence: Theories, Application and Future Direction, 2015, pp. 1–6.
4. D. Huang, C. Shan, M. Ardabilian, Y. Wang, L. Chen, Local binary patterns and its application to facial image analysis: A survey, IEEE Trans. Syst. Man Cybern. Part C 41 (6) (2011) 765–781.